Sr. Software Engineer (SSE)

ABSTRACT

This is one of the subject from my personal notes series named “Coding-With-Arqam” that I am developing from the start of my professional development career.

Subject

Server Configuration

SERVER CONFIGURATION

--> Digital Ocean:

-> First we need to create a droplet on digital ocean. Its very simple UI based task (no command needed).

-> We can select price of 5$ while creating droplet.

-> Types:

-> Docker:

-> Virtual machine.

-> It doesn't requires any dependency. There is no need to install setup, set environment, etc.

-> Fast as compared to AWS, etc. // Arqam Personal observation

-> container: That contains dependencies.

-> Process Sequence: Docker file > image -> container

-> images:

-> base image.

-> Image that we create ourselves from docker file.

-> Commands:

-> Show Images of the dockers: docker images

-> Create Image: docker build -t imageName dockerFilePathInLocalMachine / docker build -t imageName .

Note: dockerFilePathInLocalMachine will be a single dot when the doc file is in the same path

-> Create/Run Container:

-> docker run -d -p 80:80 imageName

-> docker run -d -p TargetPort:LocalHostPort imageName

-> docker run -it -d -p TargetPort:LocalHostPort imageName // "it" flag will forcefully run the container. not a good to use

-> Test docker is running or not:

-> docker container ps

-> docker container ps -a // shows container with logs. like active and stoped containers

-> logs: docker container logs containerName / docker container logs containerId / docker container logs containerName 2>anyErrorFile.txt

-> cat docker / cat anyOtherFile

-> curl -i localhost:3000 // Command used to run the project on cmd like hiting on url on browser.

-> Directly using PM2:

->

->

--> AWS:

-> Amazon Web Services.

-> Services:

-> Infrastructure as a service (IaaS)

-> Platform as a service (PaaS)

-> Packaged software as a service (SaaS)

-> IAM:

-> Identity Access Management.

-> IAM allows you to manage users and their level of access to the aws console.

-> It is used to set users, permissions and roles. It allows you to grant access to the different parts of the aws platform.

-> IAM Identities:

-> IAM Users

-> IAM Groups

-> IAM Roles

-> Creating IAM Roles:

-> In the navigation pane of the console, click "Roles" => "Create Role" => "Create Role".

-> Prerequisite:

-> AWS account.

-> Terminal with ssh support.

-> PEM file generated by AWS.

-> IP address of EC2 instance.

-> Babun installed on your windows.

-> Deployment with CodeDeploy(from aws web) (for auto deploy):

-> Step 1: Create a Key Pair

-> Click "Key Pairs" within "NETWORK & SECURITY" tag.

-> Click "Create Key Pair".

-> Give key name and then click "create" button.

-> Step 2: Enter the CodeDeploy Console

-> Click the home icon on the upper left corner of the AWS Management Console.

Find CodeDeploy under Developer Tools and click to open the AWS CodeDeploy Console.

-> In the AWS CodeDeploy Console, click Get Started Now.
-> Select Sample Deployment and click Next.
-> Step 3: Launch a Virtual Machine (Amazon EC2 instances)
-> Click the home icon on the upper left corner of the AWS Management Console.
Find CodeDeploy under Developer Tools and click to open the AWS CodeDeploy Console.
->
-> Deployment commands:
-> ssh ec2-user@ip-address -i PEMfile (e.g: ssh ec2-user@35.160.185.49 -i selftuts.pem) // this will login
-> sudo su // this will move to the root user
-> Now Intalling node:
-> curl --silent https://rpm.nodesource.com/setup_6.x | sudo bash - // This will download the RPM
-> yum install nodejs
-> exit // exit from the root user
-> Now Installing Project from github:
-> mkdir workspace
-> cd workspace
-> ls // nothing will show here because of no project
-> git clone githubRepoUrl
-> ls // will show the project
-> npm install
-> vim app/lib/database.js // if you need to edit the database.js file
-> node app.js // start the app
-> Exposing port (e.g 3000, 5000) over the internet:
-> Step 1: Login to aws
-> Step 2: go to Ec2
-> Step 3: go to Instances (just to view you instance name)
-> Step 4: Security Group
-> Step 5: check your instance
-> Step 6: edit inbound rule (on the buttom of the screen)
-> Step 7: Add rule
-> Step 8: Protocol: TCP, Port Range: 5000 (or other like 3000), Source: anywhere
-> Step 9: Save
-> Step 10: Hit the url on browser. It will be working fine.
-> Storage Services:
-> AWS S3:
-> S3 stands for Simple Storage Service.
-> It is Object-based storage, i.e., you can store the images, word files, pdf files, etc.
-> The files which are stored in S3 can be from 0 Bytes to 5 TB.
-> If you create a bucket, URL look like:
-> https://Region_Name/Bucket_Name (e.g: https://s3-eu-west-1.amazonaws.com/mybucket
-> Terms:
-> Buckets:
-> Container used for storing the objects.
-> A bucket is like a folder that stores the objects.
-> A bucket name should be unique.
-> A bucket name should start with the lowercase letter, must not contain any invalid characters.
-> It should be 3 to 63 characters long.
-> Objects:
-> Entities which are stored in an S3 bucket.
-> Keys:
-> A key is a unique identifier for an object.
-> Every object in a bucket is associated with one key.
-> Regions:
-> You can choose a geographical region in which you want to store the buckets that you have created.
-> Data Consistency Model
-> Creating an S3 Bucket:
-> Sign in to the AWS Management console.
-> Move to the S3 services.
-> To create an S3 bucket, click on the "Create bucket".
-> Enter the bucket name which should look like DNS address, and it should be resolvable.
-> Click on the "Create" button.
-> Uploading Data on the bucket:
-> Now, click on the "javatpointbucket" to upload a file in this bucket.
-> Click on the "Upload" button to add the files to your bucket.
-> Click on the "Add files" button and then select the file from the system.
-> Click on the "upload" button.
-> Making access as public:
-> Now, Move to the properties of the object "jtp.jpg" and click on the object URL to run the file appearing on the right side of the screen
-> On clicking the object URL, the screen will open with "Access Denied" message.
-> To overcome from the above problems, we need to set the permissions of a bucket, i.e., "javatpointbucket" and unchecked all of them.
-> Save these permissions.

-> Enter "confirm" in a textbox, then click on the "confirm" button.
-> Click on the "Actions" dropdown and then click on the "Make public".
-> Now, click on the Object URL of an object to run the file (It will open the file/image now).

-> Versioning:
-> Versioning is a means of keeping the multiple forms of an object in the same S3 bucket.
-> Versioning can be used to retrieve, preserve and restore every version of an object in S3 bucket.
-> Example:
-> Bucket consists of two objects with the same key but with different version ID's such as photo.jpg (version ID is 11) and photo.jpg (version ID is 12).

-> EC2(Backbone of AWS):
-> Stands for Amazon Elastic Compute Cloud.
-> Web service that provides resizable compute capacity in the cloud.
-> EC2 is a service that allows business subscribers to run application programs in the computing environment.
-> EC2 instance:
-> Virtual server in EC2 for running applications on AWS infrastructure.
-> EBS:
-> Elastic Block Store.
-> EC2 is a virtual server in a cloud while EBS is a virtual disk in a cloud.
-> Amazon EBS allows you to create storage volumes and attach them to the EC2 instances.
-> Creating an EC2 instance:
-> Sign in to the AWS Management Console.
-> Click on the EC2 service.
-> Click on the Launch Instance button to create a new instance.
-> Now, we have different Amazon Machine Images. We will be using Amazon Linux AMI 2018.03.0 (HVM) as it has
built-in tools such as java, python, ruby, perl, and especially AWS command line tools.
-> Choose an Instance Type, and then click on the Next. Suppose I choose a t2.micro as an instance type.
-> Setup the price and no of instances in the next (config) tab.
-> Now, add the EBS volume and attach it to the EC2 instance. Root is the default EBS volume. Click on the Next.
-> Now, Add the Tags and then click on the Next.
-> Configure Security Group.
-> Review an EC2 instance that you have just configured, and then click on the Launch button.
-> Create a new key pair and enter the name of the key pair. Download the Key pair.
-> Click on the Launch Instances button.
-> To use an EC2 instance in Windows, you need to install both Putty and PuttyKeyGen.
-> Download the Putty and PuttyKeyGen.
-> Remaining on this link: https://www.javatpoint.com/aws-ec2-creating-an-instance

-> YUM vs RPM:
-> YellowDog Updater Modified => RedHat Package Manager
-> Package manager => Package container that includes information on what dependencies are needed by the package and build instructions.
-> rpm -q chorome // equals to double click on chrome in windows. -q is for query. -l for list.

--> FireBase:
-> NodeJS Project Deployment:
-> Sign up on firebase
-> Create a project using web.
-> Now use CMD and follow below sequence of commands:
-> Enter into the project directory.
-> npm i -g --save
-> npm install -g firebase-tool
-> firebase login // you will get already loggedin message because you have been loggedin on.
-> firebase init

-> Angular Project Deployment:
->
-> MEAN Project Deployment:
->

--> MySQL Deployment:
-> https://www.freemysqlhosting.net/

--> Heroku:
-> Points:
-> I have deployed db to "freemysqlhosting" and then used it.
-> heroku cli downloaded and then installed on the machine.
-> There will be specific port against that web url.

-> Node + Angular Project Deployment:
-> heroku login
-> heroku git:clone -a cascoode
-> cd cascoode

// Below 3 commands will continue to use when you need to commit.
-> git add .
-> git commit -am "make it better"
-> git push heroku master
-> Note: If you want to show angular project on 3000 port, you have to do the following steps in loopback.
-> ng build --prod / ng build
-> copy files and place in to client folder.
-> give client folder path in the middleware.json
-> "files": {
"loopback#static": {
"params": "$!../client"
}
}
-> comment or del root.js in boot folder.

--> PuTTY:
-> PuTTY is a free and open-source terminal emulator, serial console and network file transfer application.
-> It supports several network protocols, including SCP, SSH, Telnet, rlogin, and raw socket connection.
-> It can also connect to a serial port.
->

--> PM2:
-> npm install -g pm2@latest

--> Cross Env PM2:
->
->

--> Notes:
-> chmod 400 mykey.pem

Reference Links

- *https://opensource.com/resources/what-docker (DOCKER)*
- *http://pm2.keymetrics.io/docs/usage/quick-start/*
- *https://nodejs.org/de/docs/guides/nodejs-docker-webapp/ (DOCKER)*
- *https://stackoverflow.com/questions/26029982/docker-container-exits-immediately/26061378*
- *https://www.freecodecamp.org/news/how-to-deploy-a-node-js-application-to-amazon-web-services-using-docker-81c2a2d7225b/*
- *https://www.javatpoint.com/aws-tutorial (AWS)*