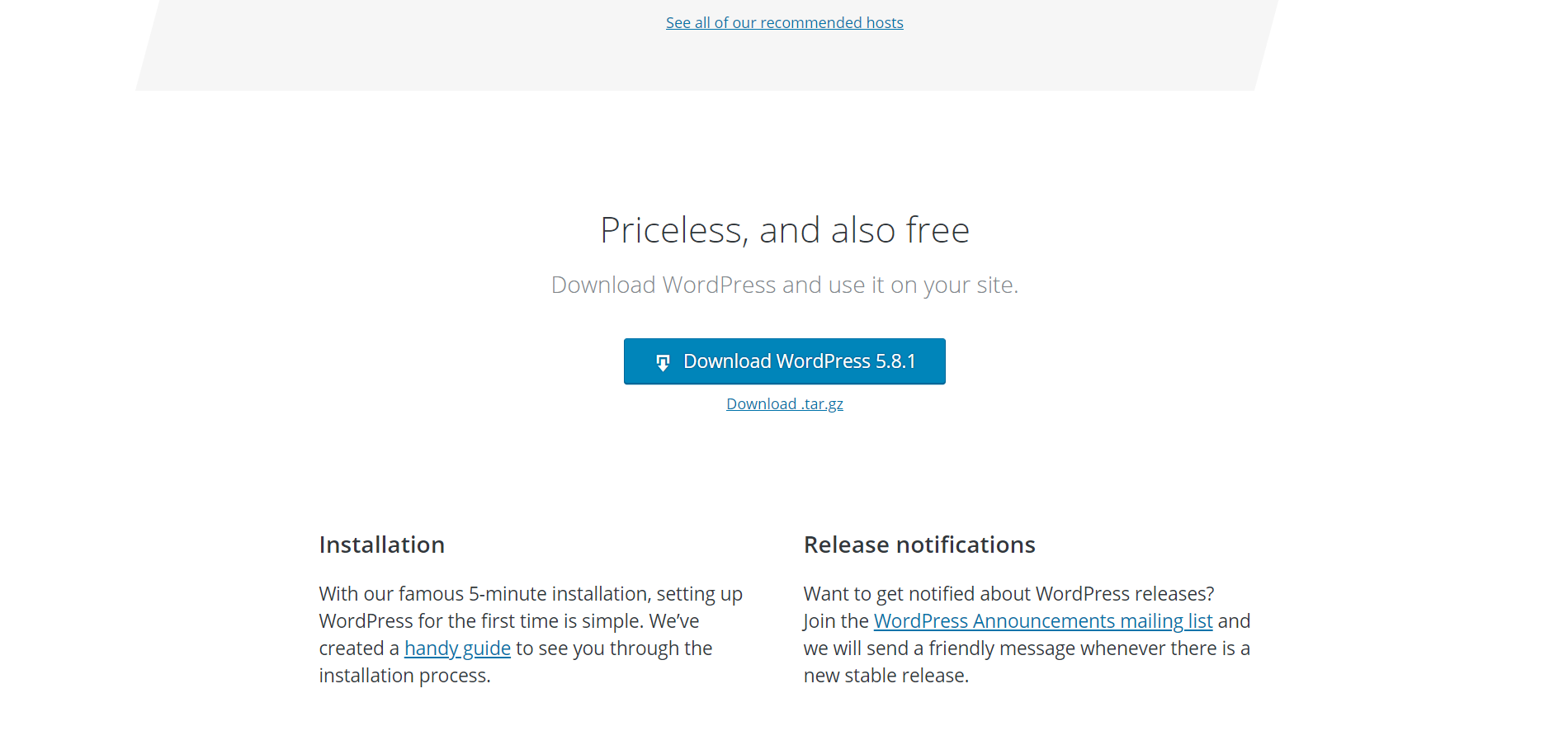
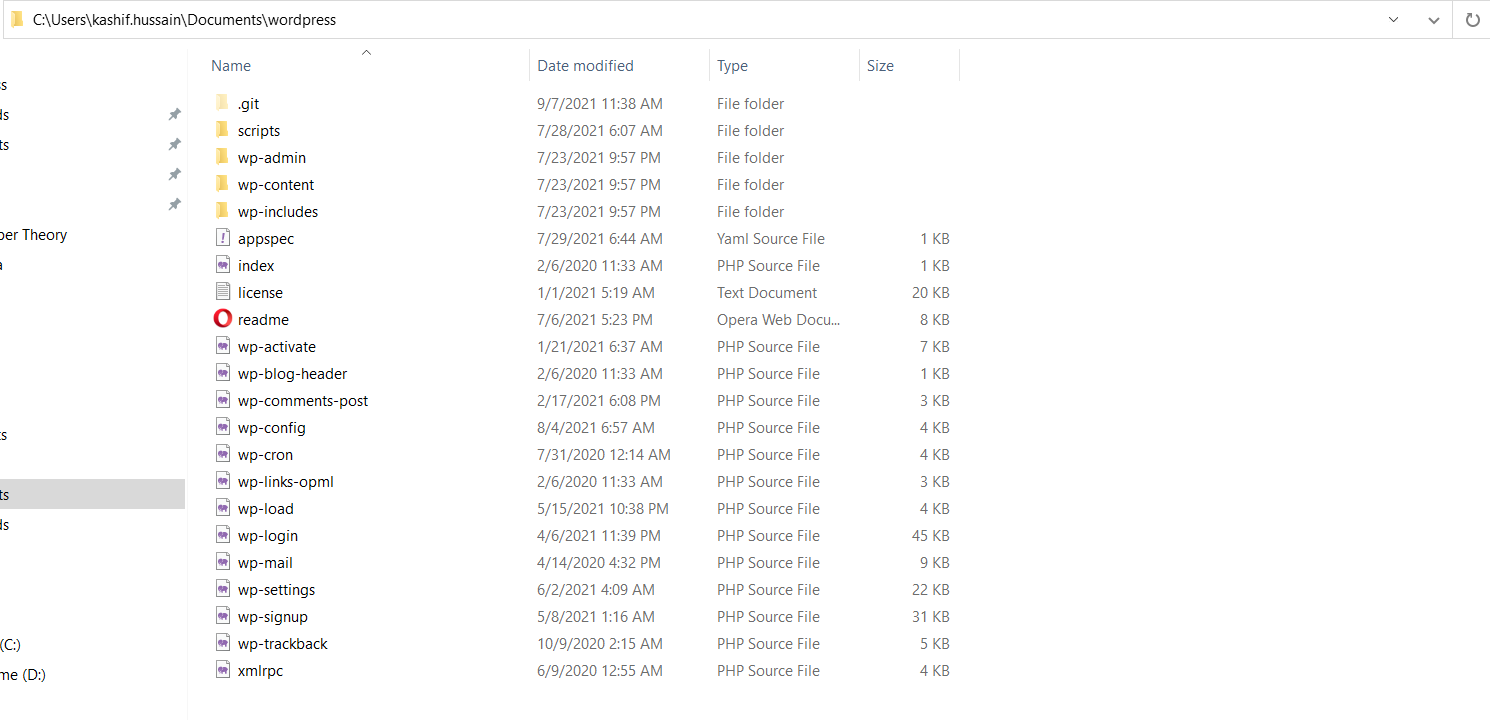
**Automate WordPress Using AWS CodePipeline**

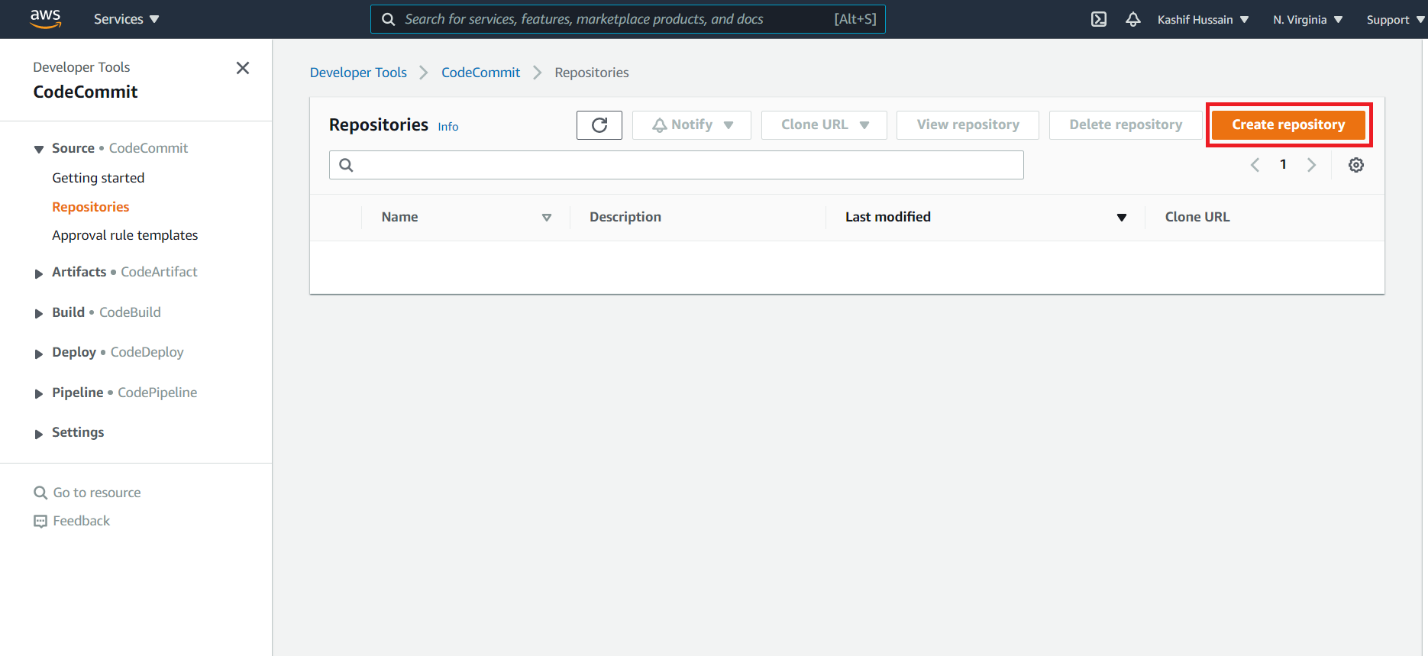
This document is regarding on how to deploy WordPress application in AWS EC2 instance and commit changes in it using CodeDeploy and CodePipeline. Following step by step process is shown on how to deploy WordPress on AWS EC2 instance.

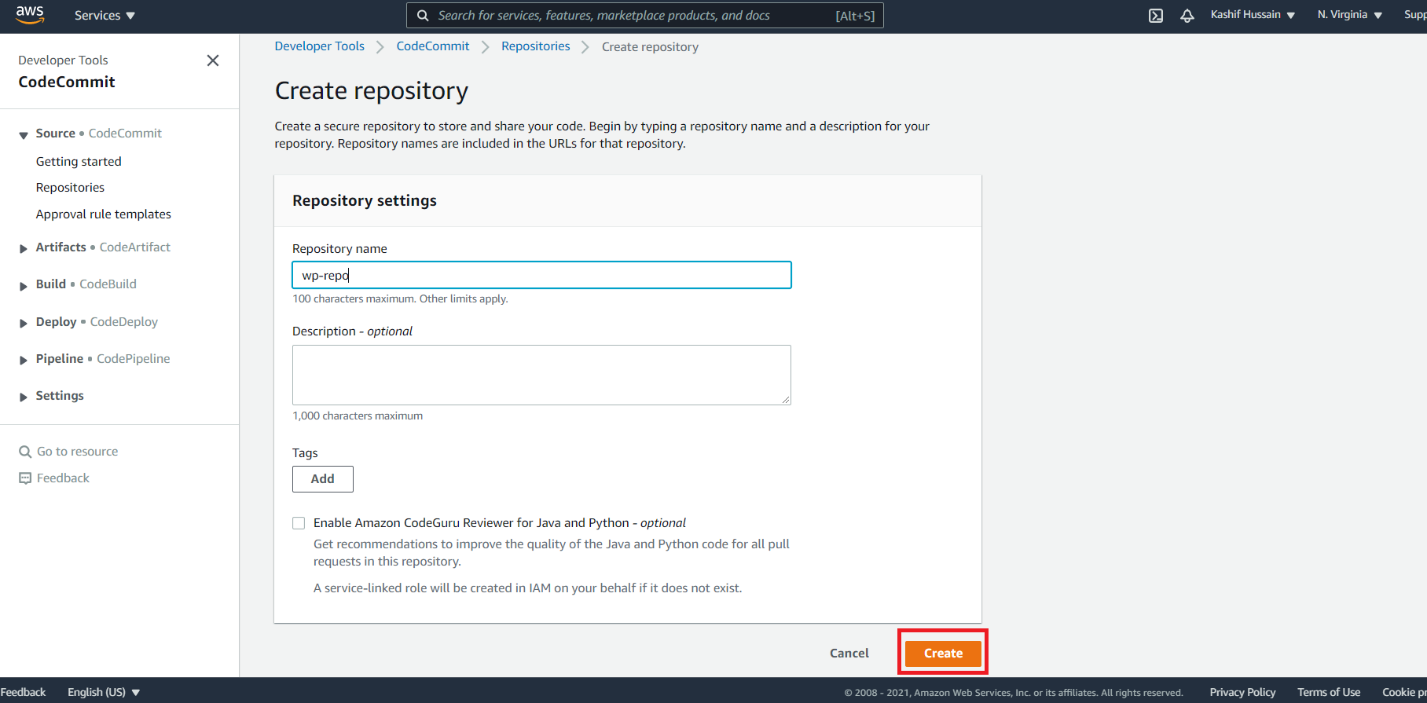
* Download WordPress application from URL: <https://wordpress.org/download/> in your local host (i.e., in my case I have downloaded it at C:\Users\kashif.hussain\Documents\wordpress in my windows system)





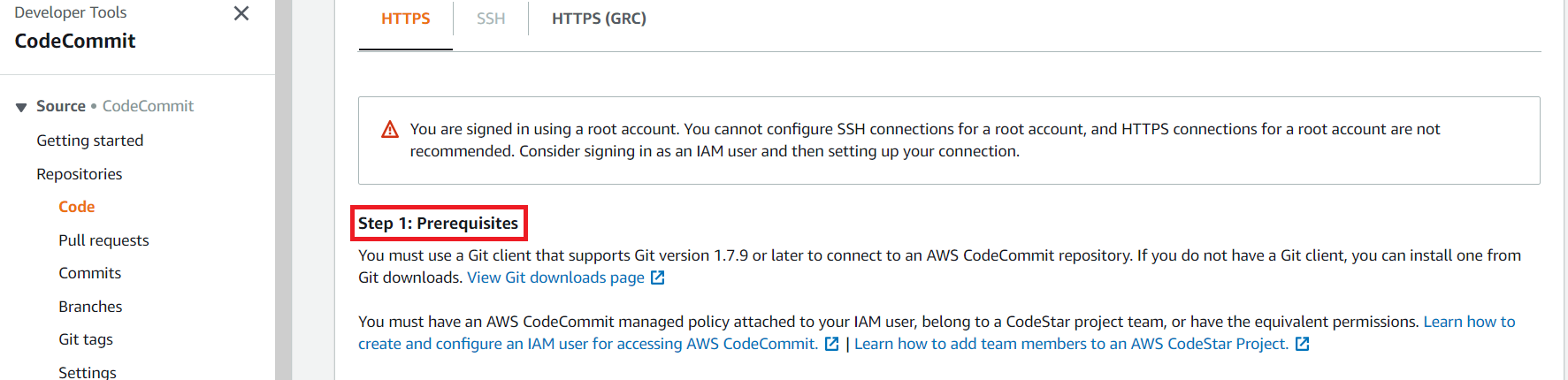
* Next step is to create a cloud-based repository which is an AWS CodeCommit.





* In order to clone the WordPress codes into the CodeCommit repository from local host, you must first install and create some pre-requisites, which are:

1. **Git client** that supports Git version 1.7.9 or later to connect to an AWS CodeCommit repository.
2. **AWS CodeCommit managed policy attached to your IAM user** so that credentials for Clone URL (SSH/HTTPS) can be generated.



* Now, in IAM service, **create a User and attach a CodeCommit policy** so that cloning credentials can be generated and IAM user can connect to CodeCommit repository.

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

* After Creating User, navigate into the **Security Credentials tab**, and look into the **“HTTPS Git Credentials for AWS CodeCommit”** section, and click on **Generate Credentials tab**, where you will get Username & Password for cloning the Code Commit Repository.

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

* Now open CodeCommit repository which was created and select **Clone HTTPS**, where you will get URL for cloning the repository.

|  |
| --- |
|  |

* Next, open the CLI in your local host (which I have used VS Code), navigate into the directory where WordPress Code is and run following commands,

|  |
| --- |
| > git remote add origin <https://git-codecommit.us-east-1.amazonaws.com/v1/repos/wp-rep> |

**Note:** After entering the above command, you have to provide Username & Password, where we have generated when creating User in IAM Service.

|  |
| --- |
| > git status  > git add.  > git commit -m “first Commit”  > git push |

* After entering the above commands, all our code has been pushed to CodeCommit repository.

|  |
| --- |
|  |

* Next, we will **create an IAM Service Role for “CodeDeploy“**.

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

* We will again create a Role, this time it will be an [**AmazonS3FullAccess**](https://console.aws.amazon.com/iam/home#/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAmazonS3FullAccess)Role and we will attach it with our Targeted AWS EC2 Instance so that it can access to the Amazon S3 buckets that contain the CodeDeploy agent.

|  |
| --- |
|  |

|  |
| --- |
|  |

* Now, launch AWS EC2 instance (here, I am using Ubuntu: 18.04.5) and set up an environment for WordPress by installing APACHE server, PhP and MySQL Database.

|  |
| --- |
| **APACHE Server Status** |

|  |
| --- |
| **MySql Database Status:** |

|  |
| --- |
| **PhP Status:** |

* Next, **create a CodeDeploy Service** since We have already created IAM Service Role for CodeDeploy, we can proceed to create a service.

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

* Now, create **Deployment Group**,

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

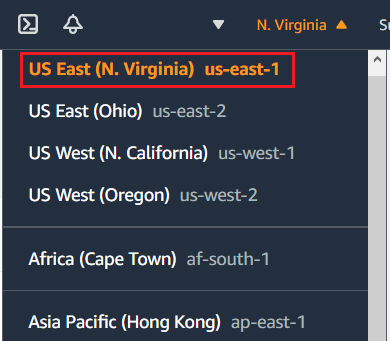
* Next, **install CodeDeploy agent** in our targeted AWS EC2 instance from S3 artifact along with its pre-requisites, for that, enter the following command.

|  |
| --- |
| $ sudo apt-get install ruby  $ sudo apt-get install wget |

Make sure you are at **/home/ubuntu** directory.

|  |
| --- |
| $ wget https://aws-codedeploy-**us-east-1**.s3.amazonaws.com/latest/install |

The above highlighted text determines from which region you are pulling the artifact. Since currently deploying in N.Virginia, use its respected region code i.e.;



|  |
| --- |
| $ chmod +x ./install  $ sudo ./install auto |

After entering the above command, check the status of CodeDeploy agent, which is as follows,

|  |
| --- |
|  |

* After successfully installing CodeDeploy Agent, now, we will work on **appspec.yml file**, which is an important aspect for CodeDeploy Service to run. One of its important feature is to mentioned the source file and its destination directory which instruct CodeDeploy to where the file needs to be deployed on targeted environment (here, it is AWS EC2 instance). Below is the screenshot,



**Note:**

1. Since we want all files from repository to be placed in specified destination field, therefore, we will enter “ **/** “ sign in the source field.
2. **appspec.yaml** file should be placed in the directory where all our code is also placed.

* Now, will work on **CodePipeline Service**, an integral part to fetch our codes from CodeCommit repository to targeted environment through CodeDeploy. To create a Pipeline, follow below options,

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

* Skip the Build Stage

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

* After building Pipeline, CodePipiline Service will process all the code in the CodeCommit Repository

|  |
| --- |
|  |

* After successful **Source** process, it will move towards **Deploy** Process, where all the codes will be deployed in the AWS EC2 instance.

|  |
| --- |
|  |

* When both process is successful, it means code is deployed. To test it, **use Public IPv4 address** of the designated AWS EC2 instance and open it on Web Browser.

|  |
| --- |
|  |

* Now to test if pipeline is automating the changes, we will alter/change some text in the installation page code (i.e., **install.php** file) and push it to repository.

|  |
| --- |
|  |

* After committing changes in the code, we can see CodePipeline automates those changes in the deployed server as well.

**Before changes**

|  |
| --- |
|  |

**After Changes**

|  |
| --- |
|  |