**Lab Steps**

Task 1: Sign in to AWS Management Console

1. Click on the  button, and you will get redirected to AWS Console in a new browser tab.
2. On the AWS sign-in page,
   * Leave the Account ID as default. Never edit/remove the 12 digit Account ID present in the AWS Console. otherwise, you cannot proceed with the lab.
   * Now copy your **User Name** and **Password** in the Lab Console to the **IAM Username and Password** in AWS Console and click on the **Sign in** button.
3. Once Signed In to the AWS Management Console, Make the default AWS Region as **US East (N. Virginia) us-east-1.**

**Note:**If you face any issues, please go through [**FAQs and Troubleshooting for Labs**](https://www.whizlabs.com/labs/support-document/faqs-and-troubleshooting).

Task 2: Launching an EC2 Instance

1. Make sure you are in the **US East (N. Virginia) us-east-1** Region.
2. Navigate to menu in the top, then click on **EC2** in the **Compute** section.
3. Click on **Instances** from the left side bar and then click on 
4. Name : Enter ***MyEC2Instance***
5. **For Amazon Machine Image (AMI):** Search for **Amazon Linux 2 AMI** in the search box and click on the **select** button.  
   Graphical user interface, text, application, email

   Description automatically generated
6. **For Instance Type:** select ***t2.micro***

***Graphical user interface, application

Description automatically generated***

1. **For Key pair:**Select **Create a new key pair**Button
   1. Key pair name: **MyEC2Key**
   2. Key pair type:**RSA**
   3. Private key file format: **.pem**
2. Select **Create key pair** Button.
3. In Network Settings Click on **Edit**:

* Auto-assign public IP: **Enable**
* Select **Create new Security group**
* Security group name : Enter **CodeDeploy-SG**
* Description: **Security group for CodeDeploy**
  + To add **SSH**,
    1. Choose Type: 
    2. Source: Custom (Allow specific IP address) or Anywhere (From ALL IP addresses accessible).
  + For **HTTP,**
    1. Click on **Add security group rule**
    2. Choose Type: **HTTP**
    3. Source: 

1. Click on 

* IAM Instance profile: Select **task\_68\_profile\_<RANDOM\_NUMBER>**

1. Keep Rest thing Default and Click on **Launch Instance** Button.
2. Select **View all Instances** to View Instance you Created
3. **Launch Status:** Your instance is now launching, Click on the instance ID and wait for complete initialization of instance state till status change to **Running**
4. Note down the sample IPv4 Public IP Address of the EC2 instance. A sample is shown in the screenshot below.

Graphical user interface, text, application, email

Description automatically generated

Task 3: SSH into EC2 Instance

* Please follow the steps in [SSH into EC2 Instance](https://www.whizlabs.com/labs/support-document/ssh-into-ec-instance).

Task 4: Install AWS CLI, Git version control, and CodeDeploy Agent

1. Switch to root user:

sudo su

1. Now run the updates using the following command:
   * yum -y update
2. Install the Git and AWS CLI:
   * yum install awscli git -y
3. Install the CodeDeploy agent:
   * Install the prerequisite i.e. ruby and wget
     + yum install ruby wget -y
   * Download the CodeDeploy agent's installer
     + wget https://aws-codedeploy-us-east-1.s3.us-east-1.amazonaws.com/latest/install
   * Change the permission of the installer
     + chmod +x ./install
   * Install the latest version of the CodeDeploy agent
     + sudo ./install auto
   * To check that the service is running, run the following command:
     + service codedeploy-agent status
   * **Note**: If the output is **"The AWS CodeDeploy agent is running as PID 785"** or any other PID. It means codedeploy agent is running fine.
   * If the output says, **"error: No AWS CodeDeploy agent running",** then start the agent by running the below command and check the status again.
   * Start the agent:
     + service codedeploy-agent start
   * check the status again:
     + service codedeploy-agent status

Task 5: Clone the application and copy application files from S3 Bucket

1. Clone the git repository
   * git clone https://github.com/WordPress/WordPress.git /tmp/WordPress
2. To get the S3 Bucket name, run the following command:
   * aws s3 ls
   * **Note: Copy the bucket name starting with whizlabs followed by a number**
3. Copy the script files to /tmp/WordPress/scripts folder
   * **Syntax:**

aws s3 cp s3://{Bucket-name}/ /tmp/WordPress --recursive

* + **Example: aws s3 cp s3://whizlabs/ /tmp/WordPress --recursive**

**A screen shot of a computer

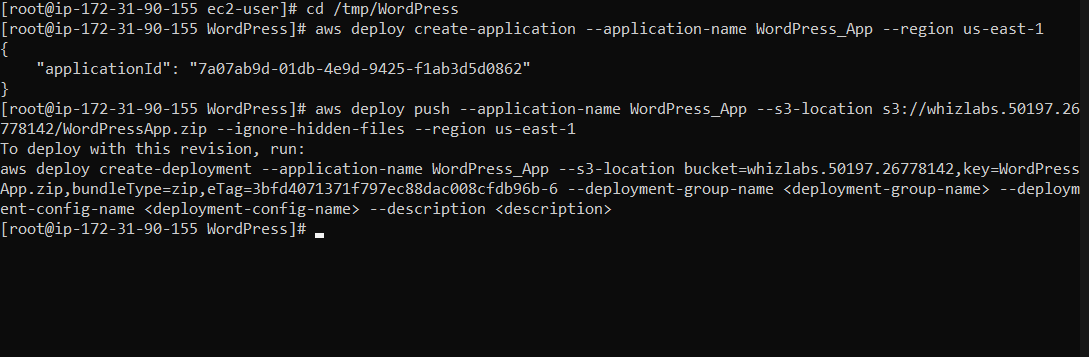
Description automatically generated with low confidence**

Task 6: Create a CodeDeploy application and push the bundled application to S3 Bucket

1. Switch to the application's repository:
   * cd /tmp/WordPress
2. Run the AWS's create application command:
   * aws deploy create-application --application-name WordPress\_App --region us-east-1
3. To bundle the application into a single zip file and push them to the bucket.
   * **Syntax:**

aws deploy push --application-name WordPress\_App --s3-location s3://{Bucket-name}/WordPressApp.zip --ignore-hidden-files --region us-east-1

* + **Example: aws deploy push --application-name WordPress\_App --s3-location s3://whizlabs/WordPressApp.zip --ignore-hidden-files --region us-east-1**
  + **Note: Ignore the output, WordPressApp.zip is now created and stored in the S3 bucket.**

****

Task 7: Create a Deployment group

1. Navigate to **CodeDeploy** by clicking on the  menu at the top, then click on **CodeDeploy** in the **Developer Tools** section.
2. Click on the Application, present on the left sidebar below Deployments under the CodeDeploy option.
3. Created application is present there, click on the Application name, **WordPress\_App**.
4. To create a deployment group, Click on the A picture containing logo

   Description automatically generated

* Deployment group name: **Whiz\_WordPress\_DepGroup**
* Select the **service role** from the dropdown

Graphical user interface, text, application, Word

Description automatically generated

* Deployment type: **In-place deployment**
* Environment configuration: Select **Amazon EC2 Instances**
  + In Tag Group 1, use the following tags to select the EC2 Instance for deployment:  
    **Note: The value should be the same as the tag defined while creation of EC2 Instance.**
    - Key: **Name**
    - Value: **MyEC2Instance**  
        
      Graphical user interface, text, application, email

      Description automatically generated
  + Matching instances will show the record as **1 unique matched instance**.
* Agent configuration with AWS Systems Manager: **Never**
* Deployment settings: Select **CodeDeployDefault.AllAtOnce**
* Load balancer: **Uncheck**

1. Click on the A picture containing logo

   Description automatically generated
2. A deployment group is now created.  
   A picture containing text

   Description automatically generated

Task 8: Create a deployment and deploy the application on the instance

1. To create a deployment, click on the A picture containing logo

   Description automatically generated
2. Deployment settings:

* Keep all the options as default.
* Revision location: Click on the textbox to **select the application's S3 URI**

**Graphical user interface, text, application, email

Description automatically generated**

* Revision type: **.zip**
* Keep all the options as default.

1. Click on the A picture containing logo

   Description automatically generated
2. Deployment is created and started to install the application on the instance.
3. Deployment status shows succeed if everything is done correctly.  
   Graphical user interface, text, application

   Description automatically generated

Task 9: Test the application using EC2 Instance Public IPv4 DNS

1. Navigate to **EC2** by clicking on the  menu in the top, then click on **EC2** in the **Compute** section.
2. On the dashboard, click on the Instances (running).
3. To view the metadata of EC2 Instance, click on the Instance ID for the instance named MyEC2Instance.
4. Copy the Public IPv4 DNS of the Instance.  
   Graphical user interface, text, application, email

   Description automatically generated
5. Open the new tab in the browser and paste the public IP, then put **/WordPress/** after the Public IPv4 DNS.
   * Link: **http://ec2-18-205-155-113.compute-1.amazonaws.com/WordPress/**
6. The application is ready to do the setup.  
   Graphical user interface, text, application

   Description automatically generated

Task 10: Setup the WordPress site

1. Click on the **Let's go!** button.
2. Fill in the below details:
   * Database name: **test**
   * User Name: **root**
   * Password: Keep it blank
   * Database host: **localhost**
   * Table Prefix: **wp\_**
3. Click on the **Submit** button.

Graphical user interface, application

Description automatically generated

1. On the very next page, To start the WordPress application, Click on the **Run the installation**button.
2. On the Welcome page, fill in the details about your WordPress application.
   * Site title: **WordPress**
   * Username: **root**
   * Password: copy the password present
   * Your email: **someone@example.com**
3. Click on the **Install WordPress**button.

Task 11: Access the WordPress site

1. Enter the credentials to login
   * Username or Email Address: **someone@example.com**
   * Password: paste the copied password
2. After login it will be redirected to the admin page, here you can customize your WordPress application.
3. To see the homepage of the sample application, remove /wp-admin/ from the URL and hit enter.

Graphical user interface, text, application, Teams

Description automatically generated

1. Or, Paste the Public IPv4 DNS of the EC2 instance and put /WordPress/ at the end of the URL.  
   Graphical user interface

   Description automatically generated with medium confidence

Task 12: Validation of the lab

1. Once the lab steps are completed, please click on the A picture containing text

   Description automatically generated button on the right side panel.
2. This will validate the resources in the AWS account and displays whether you have completed this lab successfully or not.
3. Sample output :

Table

Description automatically generated

Task 13: Delete the resources created

1: Delete the CodeDeploy application

1. Navigate to **CodeDeploy** by clicking on the  menu at the top, then click on **CodeDeploy** in the **Developer Tools** section.
2. Click on the **Application**, present on the left sidebar below **Deployments** under the **CodeDeploy** option.
3. Created application is present there, click on the Application name, WordPress\_App.
4. To delete the application, click on the **Delete application** button.
5. In the confirmation pop-up, type **delete** to complete the deletion of the application.

2: Terminate the EC2 instance

1. Navigate to **EC2** by clicking on the  menu in the top, then click on **EC2** in the **Compute** section.
2. Click on the **Instances**, present on the left sidebar below **Instances**.
3. To terminate the EC2 instance, perform the following task:
   * Select the EC2 Instance
   * Click on the Instance state
   * Choose Terminate instance
4. Confirm the termination of EC2 Instance, by clicking on the **Terminate** button.

**Completion and Conclusion**

1. You have successfully created and launched Amazon EC2 Instance.
2. You have successfully logged into the EC2 instance by SSH.
3. You have successfully installed AWS CLI, Git, and CodeDeploy Agent.
4. You have successfully cloned the WordPress application and copied the application files from S3 Bucket.
5. You have successfully created a CodeDeploy application and pushed the Bundled application to the S3 bucket.
6. You have successfully created a deployment group, deployment and deployed the application on the instance.
7. You have successfully tested the application using Public IPv4 DNS of the EC2 Instance.
8. You have successfully done a setup of the WordPress site and accessed the homepage of the WordPress application.

**End Lab**