ICT171   
Introduction to Server Environments and Architectures **Student Name :  
Muhammad Ali  
  
Student ID:  
35362113**

[Company name] | [Company address]

2025

**IP ADDRESS : 13.239.237.8  
DOMAIN NAME : foodone.xyz**

DOCUMENTATION OF SERVER  
   
 **Section : 1**  
 **Launching an AWS EC2 Instance**  
**1)** Create an AWS account on AWS Educate using credentials.  
  
**2)** After logging in, proceed to the EC2 service in the AWS Management Console.  
  
**3)** Click "Launch Instance" to initiate the creation of a virtual server and you have to determine the **geographical region** in which you’d like your server to be hosted.  
  
A screenshot of a computer

AI-generated content may be incorrect.  
  
**4)** I chose the operating system. My own choice is Ubuntu, and so I used Ubuntu Server 24.04 LTS (Long Term Support), a stable and popular option. I used the 64-bit x86 architecture, which covers most situations and has excellent support.  
  
A screenshot of a computer

AI-generated content may be incorrect.  
A screenshot of a computer

AI-generated content may be incorrect.  
  
**5)** I have selected t2.micro instance. It comes under the free tier of AWS, so it is ideal in cases of low usage or first time users.  
  
**6)** I created a new key pair .pem that I downloaded. The file is required to connecting and access the instance later from the terminal.  
  
**7)** In network settings, i enable the following options:  
 Allow SSH  
 Allow HTTP traffic from the internet.  
 Allow HTTPS traffic from the internet.  
  
and also I can go with the default options and click "Launch" to start instance and after some minutes, the new instance show in my dashboard  
  
A screenshot of a computer

AI-generated content may be incorrect. **Section : 2**  
  
 **Assigning Elastic IP**  
To maintain the public IP address of the EC2 instance the same even after restart, we create and allocate an Elastic IP (EIP). Otherwise, there would be a new public IPv4 address each time the instance is restarted, which would disrupt the availability of the web site.  
  
Navigate to AWS EC2 Dashboard go Elastic IPs.  
Click "Allocate Elastic IP address" and confirm allocation.   
  
A screenshot of a computer

AI-generated content may be incorrect.

**Section : 3**  
**Connect Ubuntu To EC2 Server  
  
1)** Go to your AWS Management Console EC2 Dashboard first.  
  
**2)** Locate your running instance, right-click it, and select "Connect."  
  
**3)** This will lead you to the "Connect to instance" page. From there, click the SSH client tab.  
  
**4)** In the SSH client example, look in the section and see the command copy that command.  
A screenshot of a computer program

AI-generated content may be incorrect.  
  
  
  
**5)** Open the computer's command prompt and navigate into the directory in which your .pem key resides (the directory in which you saved it after you created your EC2 instance).  
  
**6)** Paste the copied command into the Command Prompt  
  
  
**7)** If asked to enter or confirm permissions, simply respond "yes" to proceed.  
  
**8)** When you complete these steps, you will have successfully accessed your Ubuntu server through your EC2 instance.  
  
A computer screen with white text

AI-generated content may be incorrect.

**Now Installing Apache Webserver**The following command must be executed in order to install the Apache web server:

“**sudo apt install apache2**”  
  
Type in the above command and hit Enter. The system will ask you to proceed. Enter "yes" to proceed. After the installation is complete, Apache will be successfully installed.  
To make sure it is in working order, take your public IP address and paste it into the internet browser. Provided everything is properly set up, you will see the Apache default welcome screen, which means your website is live.  
  
A screenshot of a computer

AI-generated content may be incorrect.  
  
  
  
  
 **Section : 4  
  
Install PHP RUNTIME & MYSQL Data Base Server**

**1)** The PHP runtime and PHP MySQL connection must first be installed to do this, use the command below.  
  
“**sudo apt install php libapache2-mod-php php-mysql**”  
  
**2)** After that, we must install the MySQL server database, which we do by using

**“sudo apt install mysql-server”**  
  
**3)** After thatFirst login into MySQL as a root user then use this command to access the MySQL Prompt.  
 **“sudo mysql -u root -p”  
  
3)** Change authentication plugin So that WordPress can authenticate correctly, update the root user to utilize the native password plugin:   
  
**“ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql\_native\_password BY 'Yourpassword';”  
my password : pak@123  
  
4)** Create a new MySQL user for WordPress for using this command  
  
**CREATE USER 'm\_ali'@'localhost' IDENTIFIED BY 'pak@123';  
  
5)** Create a new database for WordPress All WordPress customizations and content will be kept in this database.  
  
**“CREATE DATABASE m;”  
  
6)** Grant privileges to WordPress user Allocate full access to the “m\_ali” on “m" database.

**“GRANT ALL PRIVILEGES ON m.\* TO 'm\_ali'@'localhost';”**

**Section : 5  
  
Deploying Web Server On WordPress**To install WordPress, visit the temporary server directory initially and download the newest package from the official WordPress website using wget. Upon completion, extract the archieve using tar and move the resulting directory WordPress to Apache's document root directory, i.e., /var/www/html. The opening of the WordPress setup page using the server IP with the /wordpress directory is the initial step in initiating setup. During the process, as a backup in case an error occurs in writing to the wp-config.php file, WordPress supplied config code can be manually copied to the new wp-config.php created in the directory WordPress. It resolves the permission issue and allows installation to proceed smoothly.  
Below is the all commands of connecting to webserver.  
  
**1)** “sudo apt update”

**2)** “sudo apt install apache2”

**3)** “sudo apt install php libapache2-mod-php php-mysql”

**4)** “sudo apt install mysql-server”

**5)** “sudo mysql -u root”

**6)** “CREATE DATABASE m\_ali;”

**7)** “CREATE USER 'm\_ali'@'localhost' IDENTIFIED BY 'pak@123';”

**8)** “GRANT ALL PRIVILEGES ON m\_ali.\* TO 'm\_ali'@'localhost';”

**9)** “FLUSH PRIVILEGES;”

**10)** “EXIT;”

**11)** “cd /tmp”

**12)** “wget https://wordpress.org/latest.vip”

**13)** “tar -xvf latest.zip”

**14)** “sudo mv wordpress/ /var/www/html/”

**15)** “cd /var/www/html/wordpress”

**16)** “sudo cp wp-config-sample.php wp-config.php”

**17)** “sudo nano wp-config.php”

**18)** “sudo nano 000-default.conf”

**19)** “sudo systemctl restart apache2”

**Section : 6  
  
1) Connect a Namecheap Domain to a WordPress Website Hosted by Apache.**In the process of installing the full WordPress website onto the Apache hosted server, I had linked the Namecheap purchased domain with my server using the technique of setting DNS records then adjusting Apache virtual host configurations.  
To buy a domain, you can use different websites. I used Namecheap myself, and you can use it too. You just need to look for your desired domain name in it and purchase it.Below is the step by step process.  
  
**2) Domain Purchase and DNS Configuration.**I begin with the custom domain (foodone.xyz) from Namecheap. To redirect the domain to divert traffic to my server with public IP address 13.239.237.8, I accessed the Advanced DNS tab in the domain dashboard from Namecheap and configured the A Records as.  
**A Record (Host @)** Points for root domain (foodone.xyz) to server IP 13.239.237.8.  
**A Record (Host www)** Points sub domain (www.foodone.xyz) to the same IP.  
  
I then configured the Apache web server in my Ubuntu instance to recognize and host the WordPress website using my domain. This was done by modifying the default virtual host setup with the command.  
  
**“sudo nano /etc/apache2/sites-available/000-default.conf”**

After that I done configuration inside the <VirtualHost \*:80> block.  
  
**“ServerAdmin webmaster@localhost”**

**“DocumentRoot /var/www/html/wordpress”**

**“ServerName foodone.xyz”**

**“ServerAlias www.foodone.xyz”**

**“ErrorLog ${APACHE\_LOG\_DIR}/error.log”**

**“CustomLog ${APACHE\_LOG\_DIR}/access.log combined”**After these configurations restart apache below is the command   
  
**“sudo nano /var/www/html/wordpress/wp-config.php”**After that went to the WordPress directory and edited the wp-config.php.  
  
**“sudo nano /var/www/html/wordpress/wp-config.php”**

In that command i manually added the following two lines to indicate the domain for WordPress in addition to configuring the database.  
  
**“define('WP\_HOME', 'http://foodone.xyz');”**

**“define('WP\_SITEURL', 'http://foodone.xyz');”**after these configurations wait for approximately 10–15 minutes for the propagation of the DNS, then checked the setup using the URL <http://foodone.xyz>

[**http://foodone.xyz/wp-admin**](http://foodone.xyz/wp-admin)WordPress initially redirected to the server's IP due to cached settings, but manual adjustment inwp-config.php resolved the issue and implement correct domain based URL.

**Section : 7  
  
1) Enabling HTTPS with a Free SSL Certificate Using Certbot.**Having already Namecheap domain (foodone.xyz) to the WordPress website hosted in an Apache server, I proceeded to encrypt the website with an SSL certificate. HTTPS activation ensures all the data transferred from the server to users is encrypted, thereby increasing security as well as the users' trust.  
  
**2) Installing Certbot for Apache**To obtain and install a free SSL certificate Let's Encrypt, I used a tool called Certbot. First, I attempt to install Certbot.  
  
**“sudo apt install certbot python3-certbot-apache”**This command first produced an error message indicating that there was not a candidate for installation. I changed the package repository to fix this.  
  
**“sudo apt-get update”**After that Certbot installed successfully.  
  
**3) Generating and Installing the SSL Certificate.**  
Once installed, I executed the following command to begin the SSL setup for my Apache server.  
 **“sudo certbot - -apache”**

In the execution of this process.  
  
I have to provide my email address in order to receive renewal notices.  
  
accepted the terms of service.  
  
Then, using the Apache setup, Certbot automatically identified my domain names (foodone.xyz and www.foodone.xyz).  
  
After obtaining SSL certificate, Certbot modified my Apache settings to automatically reroute all HTTP traffic to HTTPS.

**4) Verifying HTTPS Functionality**I completed the procedure, then browsed to https://foodone.xyz in the browser. The website now opened using an HTTPS connection with an SSL certificate from Let’s Encrypt. There is a padlock icon in the browser’s address bar, which means the connection is encrypted.  
  
A screenshot of a certificate

AI-generated content may be incorrect.

**Refrences**.  
  
https://www.youtube.com/watch?v=8Uofkq718n8

**Github.**<https://github.com/ranaali231/ICT171_Assignment-2>  
  
  
**Video Link:**https://echo360.net.au/media/5b64b346-b251-4255-9435-266bdf12a1a1/public