



Full LAMP Stack Platinum Elite

Usage instructions:

1. Launch the product via 1-click. **Please wait until** the instance passes all status checks and is running. You can connect using your Amazon private key and 'ubuntu' login via your SSH client.

To update software, use: **sudo apt update && sudo apt upgrade -y**

2. The server is enabled and provisioned with the following software:

- **Apache 2:** A robust, open-source web server that delivers web content to users.
- **MySQL:** A popular relational database management system for storing and managing data.
- **PHP:** A server-side scripting language widely used for web development.
- **phpMyAdmin:** A web-based tool for managing MySQL databases through a graphical interface.
- **JAVA:** A versatile, high-level programming language used for building cross-platform applications.
- **Nginx:** A high-performance web server and reverse proxy, often used for load balancing and serving static content.
- **Git:** A distributed version control system for tracking changes in source code during software development.
- **Composer:** A dependency management tool for PHP, allowing easy management of libraries and packages.
- **Node.js & NPM:** Node.js is a JavaScript runtime for server-side applications; NPM is its package manager.
- **Docker:** A platform for developing, shipping, and running applications in containers.
- **Docker-Compose:** A tool for defining and running multi-container Docker applications.
- **Redis:** An in-memory data structure store used for caching, messaging, and real-time data processing.
- **Supervisor:** A process control system for monitoring and managing long-running processes.
- **Visual Studio Code Server:** A browser-based version of Visual Studio Code for remote development.
- **Jenkins:** An open-source automation server used for continuous integration and continuous delivery (CI/CD).

Help & Info: **Starter Commands**

- **MySQL** (port 3306)

Starting Commands:

- Check status: **sudo systemctl status mysql**
- To restart: **sudo systemctl restart mysql**
- Login to MySQL: **sudo mysql -u root -p**

Press **"Enter"** when asked for password

- **Nginx**

Starting Commands:

- Check status: **sudo systemctl status nginx**
- Check configuration: **sudo nginx -t**
- Restart Nginx: **sudo systemctl restart nginx**

User Steps:

- Edit your server block configuration in: **/etc/nginx/sites-available/default** to customize your web server settings.
- After making changes, test the configuration with **nginx -t** and then restart Nginx.

- **Git** (port 9418)

Starting Commands:

- Check version: **git --version**
- Configure Git: **git config --global user.name "Your Name"**
- Clone a repository: **git clone <repository-url>**

User Steps:

- Set up your global user name and email for Git commits.
- Clone a repository and start working on your codebase.

- **Composer**

Starting Commands:

- Check version: **composer --version**

User Steps:

- Navigate to your project directory and run `composer install` to set up the project dependencies.
- Use `composer require <package-name>` to add new packages to your project.

- **Node.js & NPM (port 8085)**

Starting Commands:

- Check versions: **`node -v` and `npm -v`**
- Initialize a project: **`npm init`**
- Install a package: **`npm install <package-name>`**

- **Docker & Docker-Compose**

Starting Commands:

- Check docker status: **`sudo systemctl status docker`**
- Start Docker: **`sudo systemctl start docker`**
- Check Docker version: **`docker -v`** or
`sudo docker run hello-world`
- Start containers with Compose: **`sudo docker-compose up -d`**

- **Redis**

Starting Commands:

- Check Redis: **`sudo systemctl status redis-server`**
- Start Redis: **`sudo systemctl start redis-server`**
- Access Redis CLI:
`redis-cli`
 - SET and GET: Store and retrieve a value:
`SET mykey "Hello, Redis!"`
`GET mykey`
 - The command `GET mykey` should return "Hello, Redis!".
`Exit`
- Configuration files located at:
`sudo nano /etc/redis/redis.conf`
`sudo systemctl restart redis-server`

- **Supervisor**

Starting Commands:

- Check status: **sudo systemctl status supervisor**
- Start Supervisor: **sudo systemctl start supervisor**
- Add a new process: Create a configuration file in: **/etc/supervisor/conf.d/**

- **Visual Studio Code Server** (port 8088)

Starting Commands:

- Check status: **sudo systemctl status code-server**
- Start Code Server: **sudo systemctl start code-server**

- Change the Nginx “server_name” to your Instance Public IP address:

sudo nano /etc/nginx/sites-available/code-server

```
GNU nano 7.2
server {
    listen 8088;
    server_name 18.212.100.162;

    client_header_buffer_size 16k;
    large_client_header_buffers 4 32k;

    location / {
        proxy_pass http://127.0.0.1:8087/;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
```

Exit & Save

- Restart Code-server:

sudo nginx -t

sudo systemctl reload nginx

sudo systemctl restart code-server

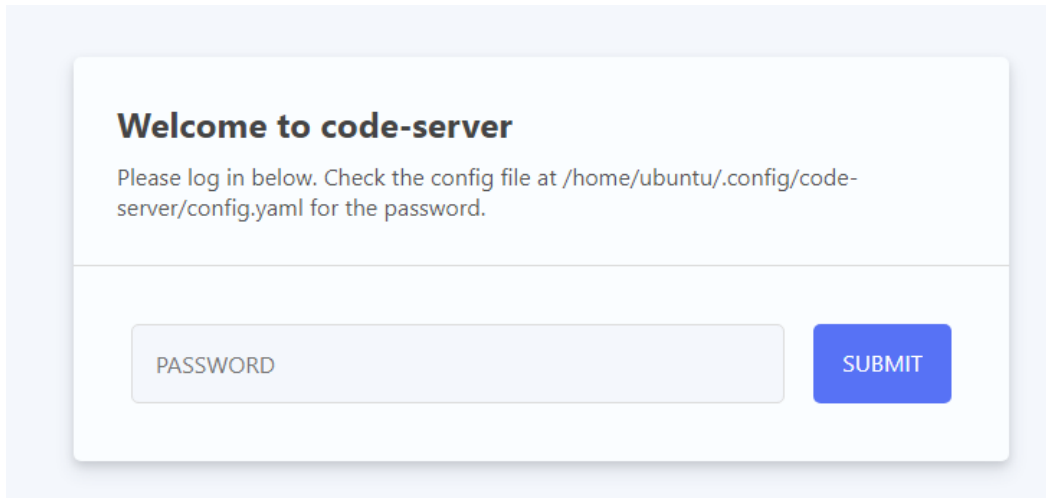
Web Interface password is located at:

cd /home/ubuntu/.config/code-server

sudo nano config.yaml

- Access in browser: Visit `http://your_server_ip:8088`

Ex: `http://363.653.34:8088`



Note: You need to close (X) out any error messages to reload page if you are using a small instance size.

- If you need to reboot server: **sudo reboot**

- **phpMyAdmin (port 8083)**

Starting Commands:

Create your unique password, use the following steps:

sudo mysql -u root -p

At the MySQL prompt, run the following commands to create your password:

ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'your_new_password_here';

FLUSH PRIVILEGES;

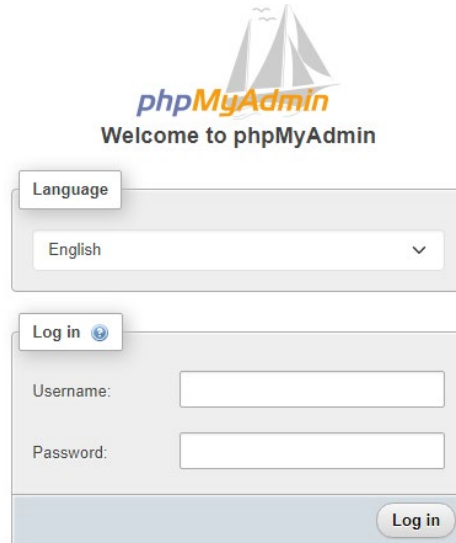
EXIT;

Log into phpMyAdmin:

Now you can log into phpMyAdmin using the username **root** and the new password you set.

- Access phpMyAdmin: Visit: http://your_server_ip:8083

Ex: <http://567.352.2:8083>



The image shows the phpMyAdmin login page. At the top, there is a logo with a sailboat and the text "phpMyAdmin" and "Welcome to phpMyAdmin". Below this, there is a "Language" dropdown menu set to "English". Underneath is a "Log in" button with a key icon. Below the button are two input fields: "Username:" and "Password:". At the bottom right, there is a "Log in" button.

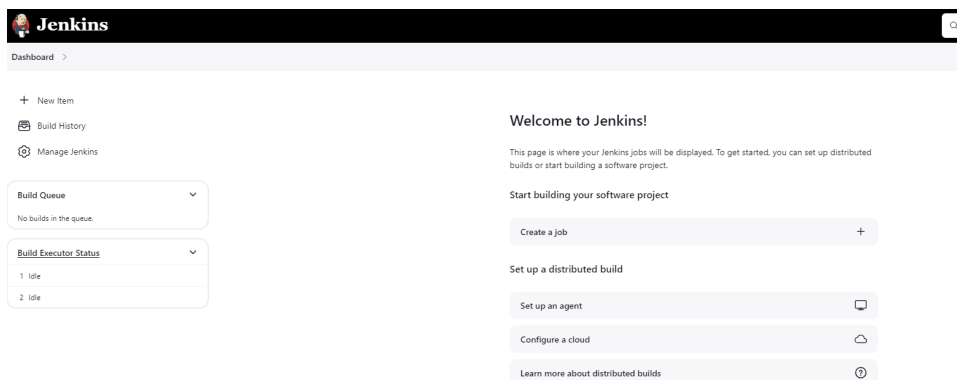
- **Jenkins** (port 8081)

Starting Commands:

- Check Status: **`sudo systemctl status jenkins`**
- **Access Jenkins: Visit** http://your_server_ip:8081

Ex: <http://567.352.2:8081>

You should now be able to log into Jenkins without requiring a password.



Create or Reset the Admin User:

- Go to: Manage Jenkins > Credentials.....

Re-enable Security:

After resetting the password, re-enable security by editing the config.xml file again and setting

sudo nano /var/lib/jenkins/config.xml

Restart Jenkins to apply the changes:

sudo systemctl restart jenkins

Configuration files found at:

sudo nano /etc/default/jenkins

- **Apache 2** (port 8086)

Starting Commands:

- Start Apache: **sudo systemctl start apache2**
- Check status: **sudo systemctl status apache2**
- Restart Apache: **sudo systemctl restart apache2**

User Steps:

- To test if Apache is running, visit your server's IP Public address in a web browser. You should see the default Apache welcome page.

Ex: http://23.396.653:8086

- To serve a new website, place your HTML files in: /var/www/html/

AWS Data

- Data Encryption Configuration: This solution does not encrypt data within the running instance.
- User Credentials are stored: /root/.ssh/authorized_keys & /home/ubuntu/.ssh/authorized_keys
- Monitor the health:
 - Navigate to your Amazon EC2 console and verify that you're in the correct region.
 - Choose Instance and select your launched instance.
 - Select the server to display your metadata page and choose the Status checks tab at the bottom of the page to review if your status checks passed or failed.

Extra Information: (Optional)

Allocate Elastic IP

To ensure that your instance **keeps its IP during restarts** that might happen, configure an Elastic IP. From the EC2 console:

1. Select ELASTIC IPs.
2. Click on the ALLOCATE ELASTIC IP ADDRESS.
3. Select the default (Amazon pool of IPv4 addresses) and click on ALLOCATE.
4. From the ACTIONS pull down, select ASSOCIATE ELASTIC IP ADDRESS.
5. In the box that comes up, note down the Elastic IP Address, which will be needed when you configure your DNS.
6. In the search box under INSTANCE, click and find your INSTANCE ID and then click ASSOCIATE.
7. Your instance now has an elastic IP associated with it.
8. For additional help: <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/elastic-ip-addresses-eip.html>