# Development Server

## Server Setup/Installs

### Server Configurations

- Added firewall rules:
  - sudo ufw allow 80
  - sudo ufw allow 443
  - sudo ufw reload
- Adjusted /etc/ssh/sshd_config
  - AllowTcpForwarding yes
  - GatewayPorts yes
  - sudo systemctl restart sshd
- Generate SSH key for github and  add to ssh-agent

```
ssh-keygen -t ed25519 -C "your_email@example.com"
# saved to /home/YOU/.ssh/id_github
eval $(ssh-agent -s)
ssh-add ~/.ssh/id_github
```

- ** Create application user that has access to /var/www resources ** Unimplemented

### github-ssh

*Custom shell command to check for github ssh key, generate one if not, start the ssh agent, and add the key to the agent*

- Created `github-ssh` executable and moved it to /usr/local/bin

## libreoffice

*Used in headless mode to convert files to pdf*

Install:

```
sudo apt install libreoffice
```

Command:

```
libreoffice --headless --convert-to pdf $1 --outdir $2
# $1 is input file
# $2 is outdir location
```

## nvm (Node Version Manager)

*Used to install multiple versions of node and switch between versions*

Install:

```
# Download and install nvm:
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.40.1/install.sh | bash

# in lieu of restarting the shell
\. "$HOME/.nvm/nvm.sh"

# Download and install Node.js:
nvm install 22

# Verify the Node.js version:
node -v

# Should print "v22.14.0".
nvm current # Should print "v22.14.0".
```

```

# Verify npm version:
npm -v # Should print "10.9.2".
```

## pnpm (Node package manager similar to npm)

*Used as a package manager for monorepo environments*

Install:

```
curl -fsSL https://get.pnpm.io/install.sh | sh -
```

## nginx

*Used as a reverse proxy to serve multiple applications from the same domain*

Install:

```
sudo apt install nginx
```

Configurations:

Listens on port 443 for https connections

## pm2 (Node process manager)

*Used to manage node related processes, restart servers, and monitor*

Install:

```
npm install pm2@latest -g
```

Commands:

```
pm2 startup # Generate a startup script to handle server reboots
```

```

pm2 start [desired applications] # Start the applications

pm2 save # Save the applications in the startup script to resurrect on system reboot
```

## nest (NestJS cli)

*Used to scaffold and interact with NestJS server applications*

Install:

```
npm install -g @nestjs/cli
```

## Docker

```
sudo install -m 0755 -d /etc/apt/keyrings

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg

echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt install docker-ce docker-ce-cli containerd.io

sudo usermod -aG docker $USER
```

## puppeteer

```
npm i -g puppeteer

```

Create App Armor profile to allow Puppeteer to launch Chromium browser

```
# /etc/apparmor.d/chrome-dev-builds

abi <abi/4.0>,
include <tunables/global>

profile chrome /home/neonetda.org/mweitzenhoffer/.cache/puppeteer/chrome/**/chrome-linux64/chrome flags=(unconfined) {
  userns,

  include if exists <local/chrome>
}

```

## App Directory

*Based on best practices for storing and serving applications on a Linux system, all application codebases are installed in the /srv directory and websites are served from the /var/www*

### /var/www

- /website-main --> Main website
- /website-admin --> Admin dashboard
- /wordpress-lms --> LearnDash WordPress install

### /srv

- /website-api --> Api and database logic to serve content to Website
- /file-manager --> Service to upload files to the system and serve them
- /shared-files --> Files uploaded to the system to be served to end users