# Assignment 3 part 1

The goal of this assignment is to setup a Bash script that generates a static `index.html` file that contains some system information. The script will be configured to run automatically every day at 05:00 using a `systemd` service and timer. The HTML document created by this script will be served with an `nginx` web server that will run on your Arch Linux droplet along with a firewall setup using `ufw` to help secure your server.

This will help you to develop your skills in and understanding of:

- scripting
- general system administration (user and permission management)
- working with `systemd` timers and services
- working with a web server `nginx`
- and configuring a simple firewall using `ufw`

This assignment builds on concepts covered in class and the flipped material, including system administration, `systemd`, scripting, and Nginx configuration. Not all steps are detailed explicitly; you are expected to apply your understanding and previous experience to complete the tasks.

You can clone the `generate_index` script from this sourcehut repository.

## task 1

Create a system user `webgen` with a home directory at `/var/lib/webgen` and a login shell appropriate for a non-login user. Ensure the user has ownership of the home directory all of its sub-directories and files.

What is the benefit of creating a system user for this task rather than using a regular user or root?

new system users home directory structure with files.

```
/var/lib/webgen/
├── bin/
│   └── generate_index
└── HTML/
    └── index.html
```

## task 2

Create a service file `generate-index.service` that will run the `generate_index` script provided above. This service should specify the `webgen` user and group in a directive. It should also not run until *after* the network-online target is active using the *optional dependency* directive.

Create a timer `generate-index.timer` that will run the service everyday at 05:00

*How will you verify that the timer is active and that the service runs successfully?" "What commands can you run to check logs and to confirm the service's execution?*

## task 3

Modify the main `nginx.conf` file to ensure the server runs as the `webgen` user. Create a separate server block file that configures Nginx to serve the `index.html` file on port 80. Use the Arch Wiki as a reference for nginx and server block configuration.

Why is it important to use a separate server block file instead of modifying the main `nginx.conf` file directly?

How can you check the status of the nginx services and test your nginx configuration?

## task 4

When your server is configured and you are serving the HTML document with your system information you are ready to set up a firewall.

Install `ufw` and configure `ufw` to:

- allow ssh and http from anywhere (in practice we would probably only allow ssh from our home or work ip)
- enable ssh rate limiting

How can you check the status of your firewall?

## task 5

With everything configured visit your droplets ip address in the browser and take a screenshot of the system information page. The screenshot should clearly show your ip address and the system information page.

You are also going to submit your ip address so that I can see that your server is working.

## general

Can you you think of ways to enhance the `generate_index` script to include additional system information or error handling?

Even if your final configuration doesn't work perfectly, submit your attempts and include notes on what you tried and where you encountered challenges.

You may refer to class materials, the Arch Wiki, and man pages, but your solutions should be your own.

Make use of the git command line tool, don't just upload files using the web console.

## deliverables

Work in a new remote git repository. Commit your work as you make changes to files. Your git repository should include:

- The script that generates your HTML document
- Your service file
- Your timer file
- Your nginx.conf file
- Your server block file
- Your success, it works screenshot
- a README.md
    - Include necessary instructions to setup a new server.
    - Make sure that information in your readme addresses questions in the tasks. Don't be too literal, you don't need to include the questions.
    - Where should files go, what commands need to be run... This doesn't need to be a tutorial, just provide the necessary information to set everything up on a new server.
    - ==alternative:==
        - Write a script the sets up everything and copies the files to correct location.
        - Your readme would only need to contain instructions on running this script.

**Submit:**

- a link to your remote repository
- and a link to your ip address. When I look at the page it should show the current date and other system information below. Your number of installed packages might be a little different.

## rubric

| criteria | excellent | good | satisfactory | incomplete |
|---|---|---|---|---|
| System user creation and setup | 5 | 4 | 3 | 0 |
| service and timer setup | 10 | 8 | 6 | 0 |
| nginx configuration | 5 | 4 | 3 | 0 |
| nginx server block | 10 | 8 | 6 | 0 |
| firewall setup | 5 | 4 | 3 | 0 |
| git | 5 | 4 | 3 | 0 |
| success shot and working server | 5 | | | 0 |