

Project 1 NGINX Deployment Slides

Joshua Sparks, Northern Kentucky University

Prepared for: Prof. Samuel Cho, ASE 230, Fall 2025

Overview:

The goal of this part of the project was to deploy the REST APIs with **NGINX** in accordance to the following constraints:

- Deployment steps/tutorial documented in Marp slide (and converted into PDFs)
- full points if marp file is made with screen captures, no partial points
- A screen capture of your NGINX server working should be included
- Saved in `presentation/` directory

Steps

1. Locate the nginx.conf file, this can be done by executing `brew info nginx`. Example location:

```
/opt/homebrew/etc/nginx
```

2. CD to the correct file path. Example command:

```
cd /opt/homebrew/etc/nginx
```

3. Perform `nano nginx.conf` to edit the file. Example command:

```
nano nginx.conf
```

Screen Captures:

```
[joshsparks@MacBookAir ~ % brew info nginx
==> nginx: stable 1.29.2 (bottled), HEAD
HTTP(S) server and reverse proxy, and IMAP/POP3 proxy server
https://nginx.org/
Installed
/opt/homebrew/Cellar/nginx/1.29.2 (27 files, 2.5MB) *
  Poured from bottle using the formulae.brew.sh API on 2025-10-17 at 15:38:41
From: https://github.com/Homebrew/homebrew-core/blob/HEAD/Formula/n/nginx.rb
License: BSD-2-Clause
==> Dependencies
Required: openssl@3 ✓, pcre2 ✓
==> Options
--HEAD
    Install HEAD version
==> Caveats
Docroot is: /opt/homebrew/var/www

The default port has been set in /opt/homebrew/etc/nginx/nginx.conf to 8080 so t
hat
nginx can run without sudo.
```

1. nginx will load all files in /opt/homebrew/etc/nginx/servers/.

```
2. [joshsparks@MacBookAir ~ % cd /opt/homebrew/etc/nginx
[joshsparks@MacBookAir nginx %
```

```
3. [joshsparks@MacBookAir nginx % nano nginx.conf
```

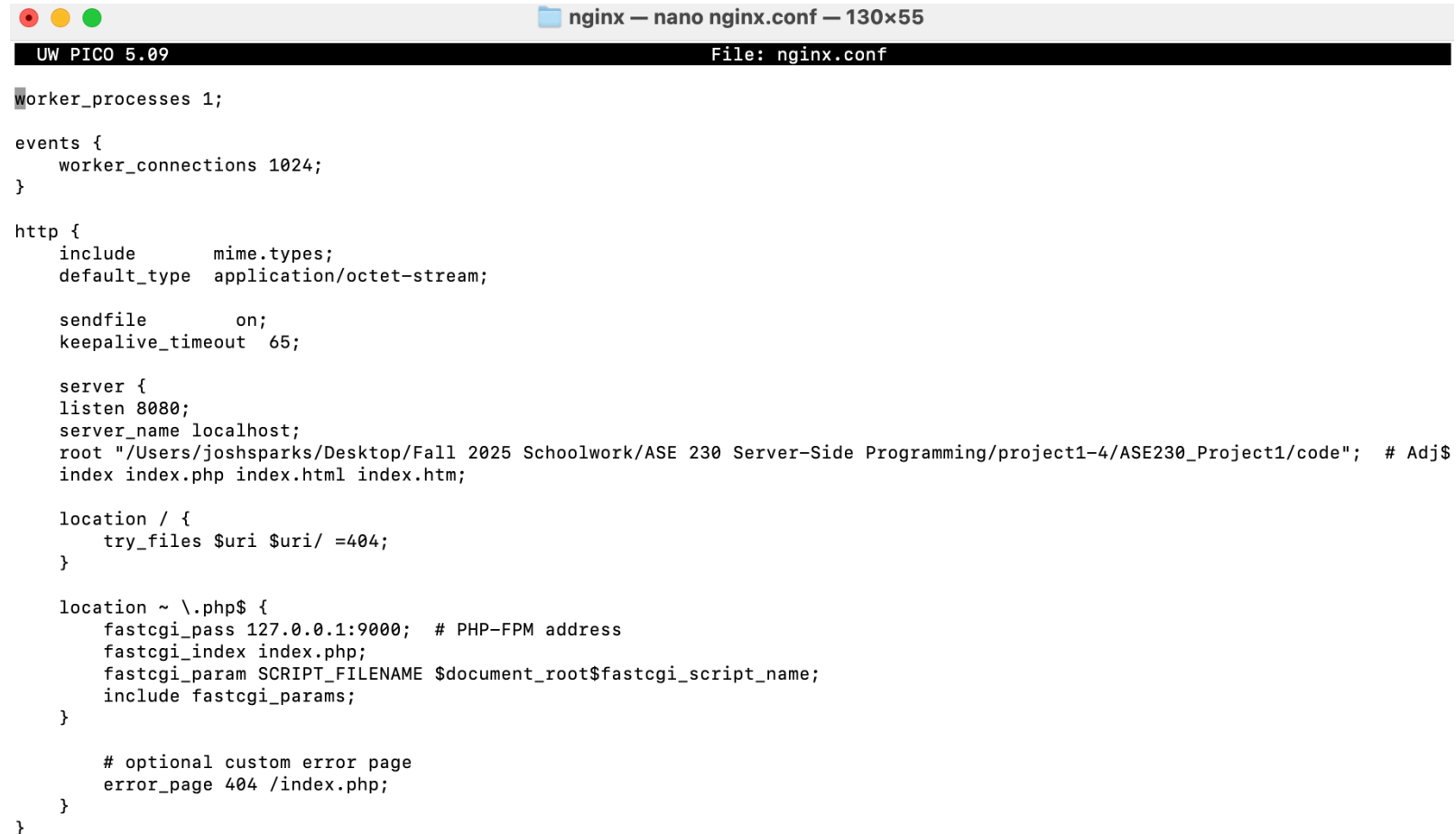
4. Replace the server block with the following:

```
server {
    listen 8080;
    server_name localhost;
    root "/Users/joshsparks/Desktop/Fall 2025 Schoolwork/ASE 230 Server-Side Programming/project1-4/ASE230_Project1/code"; # Adjust path for your system
    index index.php index.html index.htm;

    location / {
        try_files $uri $uri/ =404;
    }

    location ~ \.php$ {
        fastcgi_pass 127.0.0.1:9000; # PHP-FPM address
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        include fastcgi_params;
    }
}
```

Screen Capture:



The screenshot shows a terminal window with a title bar that reads "nginx — nano nginx.conf — 130x55". The terminal has a black background with white text. The top status bar shows "UW PICO 5.09" on the left and "File: nginx.conf" on the right. The main content area displays the configuration for the nginx.conf file. The configuration includes settings for worker processes, events, http, and a server block. The server block is configured to listen on port 8080, serve files from a specific directory, and use PHP-FPM for .php files. A custom error page is also defined for 404 errors.

```
worker_processes 1;

events {
    worker_connections 1024;
}

http {
    include      mime.types;
    default_type application/octet-stream;

    sendfile      on;
    keepalive_timeout 65;

    server {
        listen 8080;
        server_name localhost;
        root "/Users/joshsparks/Desktop/Fall 2025 Schoolwork/ASE 230 Server-Side Programming/project1-4/ASE230_Project1/code"; # Adj$
        index index.php index.html index.htm;

        location / {
            try_files $uri $uri/ =404;
        }

        location ~ \.php$ {
            fastcgi_pass 127.0.0.1:9000; # PHP-FPM address
            fastcgi_index index.php;
            fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
            include fastcgi_params;
        }

        # optional custom error page
        error_page 404 /index.php;
    }
}
```

5. Restart NGINX with the following command:

```
nginx -s reload
```

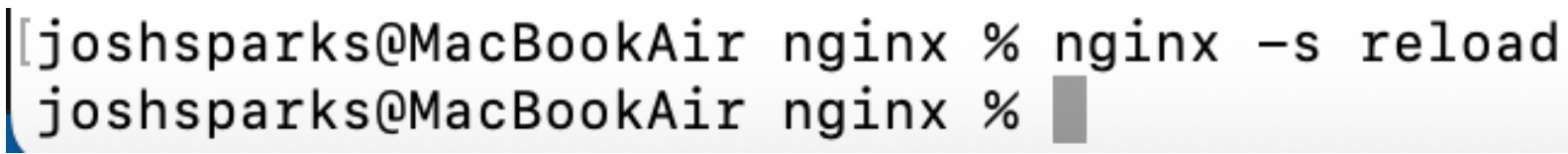
6. If php is not running, run the following command:

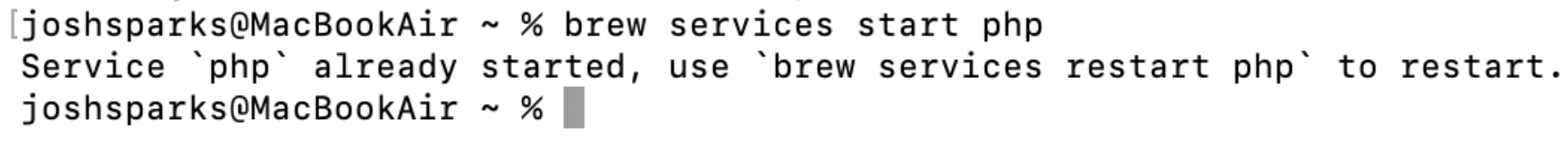
```
brew services start php
```

7. To test NGINX is working properly, go to the following address on your web browser:

```
http://localhost:8080/your_file_path/file_you_want_to_test
```

Screen Captures:

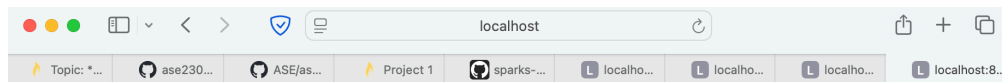
5. A terminal window showing the command `nginx -s reload` being executed. The prompt is `joshsparks@MacBookAir nginx %`.

6. A terminal window showing the command `brew services start php` being executed. The output is `Service `php` already started, use `brew services restart php` to restart.` The prompt is `joshsparks@MacBookAir ~ %`.

My first web address test:

http://localhost:8080/HTML_JavaScript_Tests/issues_api_tests.html

Screen Capture:



Test issues.php APIs:

Show All Issues

Click the button above to test the API...

Issue ID:

Submit

Enter Issue ID above and click submit to test the API...

Issue ID:

Issue Name:

Submit

Enter Issue ID and Issue Name above and click submit to test the API...

Issue ID:

Issue Name:

Board ID:

Issue Status:

Submit

Enter old/updated Issue ID, old/updated Issue Name, old/updated Board ID, and old/updated Issue Status above and click submit to test the API...

Issue ID:

Issue Status:

Submit

Enter Issue ID and updated Issue Status above and click submit to test the API...

Issue ID:

Issue Comment:

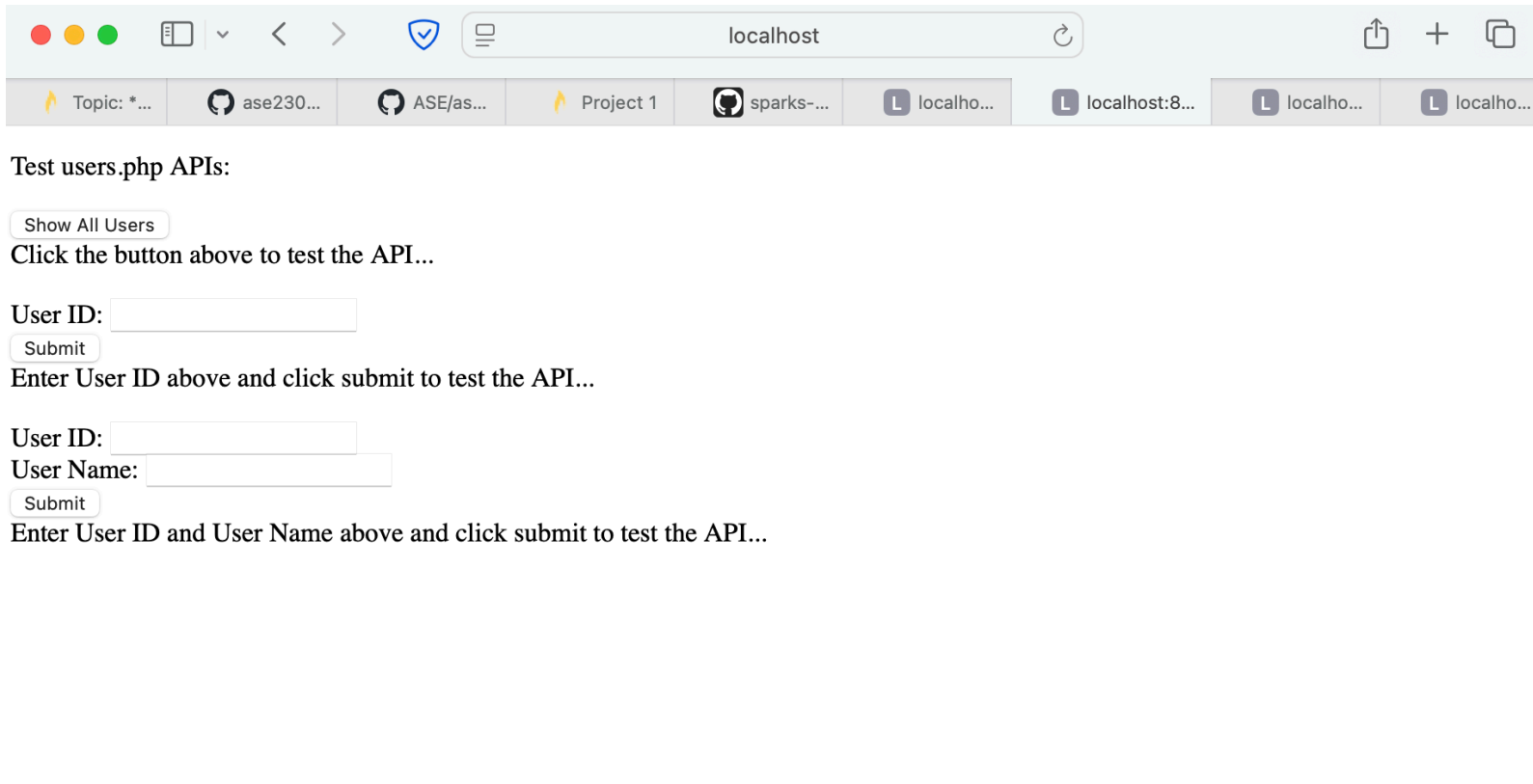
Submit

Enter Issue ID and comment above and click submit to test the API...

My second web address test:

http://localhost:8080/HTML_JavaScript_Tests/users_api_tests.html

Screen Capture:



The screenshot shows a web browser window with the address bar set to 'localhost'. The browser's tab bar contains several tabs, including 'Topic: *...', 'ase230...', 'ASE/as...', 'Project 1', 'sparks-...', and several 'localhost' tabs. The main content area of the browser displays a web page titled 'Test users.php APIs:'. The page contains two identical form sections. Each section starts with a 'Show All Users' button, followed by the instruction 'Click the button above to test the API...'. The first form section has a 'User ID:' label and an input field, followed by a 'Submit' button and the instruction 'Enter User ID above and click submit to test the API...'. The second form section has a 'User ID:' label and an input field, a 'User Name:' label and an input field, followed by a 'Submit' button and the instruction 'Enter User ID and User Name above and click submit to test the API...'.

Test users.php APIs:

Show All Users

Click the button above to test the API...

User ID:

Submit

Enter User ID above and click submit to test the API...

User ID:

User Name:

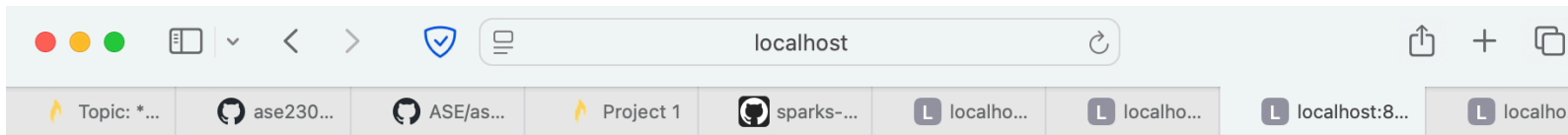
Submit

Enter User ID and User Name above and click submit to test the API...

My third web address test:

http://localhost:8080/HTML_JavaScript_Tests/boards_api_tests.html

Screen Capture:



Test boards.php APIs:

Show All Boards

Click the button above to test the API...

Board ID:

Submit

Enter Board ID above and click submit to test the API...

Board ID:

Board Name:

Submit

Enter Board ID and Board Name above and click submit to test the API...

References:

Setting up NGINX For PHP.md, pages 15, 16, 21, 25, 26