Testing Framework

As the site is currently using PHP, the PHPUnit framework was chosen to test aspects of various PHP files.

Test Cases Developed

1. **TestException** - This test case verifies whether the *about.php* file correctly includes a specific HTML snippet. This is done via reading the contents of the file and ensuring that the file has the line of *<?php include('about_snippet')>*. If the snippet is found, it sends a success message, otherwise will lead to a failure and send a test error message.
2. **TestCurrentYearDisplayedInAbout** - This test case checks if the *about.php* file is displaying the correct current year. This is done through searching for the current year output and ensuring that the year is accurately matching with the displayed output.

Basis of CI/CD

To set up the pipeline, a .gitlab-ci.yml file is needed with the proper configuration. This file consists of jobs that will run, setup in different stages. The stages chosen so far are testing, building, and deploying. Different docker images are used to run the jobs specified, currently using Debian and a more specific phpunit image for testing. Scripts provide what will be done throughout the pipeline.

Continuous Integration

A ‘testing website’ stage has been set up that will utilize an image that contains the phpunit framework. The script for this stage will run the test files for the website.

GitLab CI/CD Configuration:

Create a .gitlab-ci.yml file in the root of your GitLab repository.

Configure the CI/CD pipeline stages. Use scp or another method to copy your website files to the deployment server.

Include a step to echo a link or update a configuration file with the deployment link.

yaml

Copy code

Stages:

```
  - deploy

deploy:

  stage: deploy

  script:

        - scp -r /path/to/your/app/* user@your-server:/path/to/destination

        - echo "Deployed at: http://your-server-link"
```

Continuous Deployment

Using webhooks

One way of getting deployment to work is by using webhooks. Webhooks are basically a way to tell a system that something has happened from another system. In our case, a webhook will see that a push has been made to a branch designated for deployment, and communicate that with the production server. From there the production server will do what it needs to do to get the changes and deploy it.

To set up a webhook, this can be done in GitLab by creating a webhook, setting up the URL that will trigger the change, and the actual trigger. On the server end, the webhook package would need to be installed, and a configuration file to let the webhook know what to do. A deployment script is needed, which will run when the trigger is activated.

Also the link can be added into the