How to Host a Static Website on GitHub Pages

A step-by-step guide using the GitHub website

Mandy Hathaway · mandyhathaway.com

OVERVIEW

This guide walks you through hosting a static website on GitHub Pages using only the GitHub website. No software installation or command line experience required. By the time you are done, your site will have a real public URL and you will know how to update it whenever you need to.

GitHub Pages is a free hosting service built into GitHub. It is designed for static sites, meaning sites built from HTML, CSS, and JavaScript files without a backend server or database. It is a practical, professional option for portfolios, documentation, and personal projects.

What You Will Need

- A GitHub account. Creating one is free at github.com.
- A completed website folder ready to upload. Your site should already be built and working locally, with all HTML, CSS, image, and other files in place. The main page must be named index.html.
- A browser. No other software is required.

Estimated Time

10 to 15 minutes from start to a live site.

What You Will Have When You Are Done

A live website at a URL in the format https://yourusername.github.io/. If you connect a custom domain (covered in the optional section at the end of this guide), that URL can be replaced with a domain of your choosing.

SECTION 1: WHAT IS GITHUB

GitHub is an online platform for storing and managing code. It uses a version control system called Git, which tracks every change you make to your files over time. Think of it like a detailed save history for your project, where you can see exactly what changed, when, and why.

For the purposes of this guide, you do not need to understand Git in depth. The GitHub website handles everything through a point and click interface, and this guide covers every step you need.

Key Terms

Repository

A repository (usually shortened to repo) is the folder where your project lives on GitHub. It stores all your files and their full change history. Each project you host will have its own repository.

Commit

A commit is a saved snapshot of your files at a particular moment. When you upload files or make changes through the GitHub website, you commit them. GitHub records exactly what changed so you can always go back if something goes wrong.

Branch

A branch is a parallel version of your repository. GitHub Pages publishes from a specific branch. By default, most repositories start with a branch called main, and that is the one you will use in this guide.

GitHub Pages

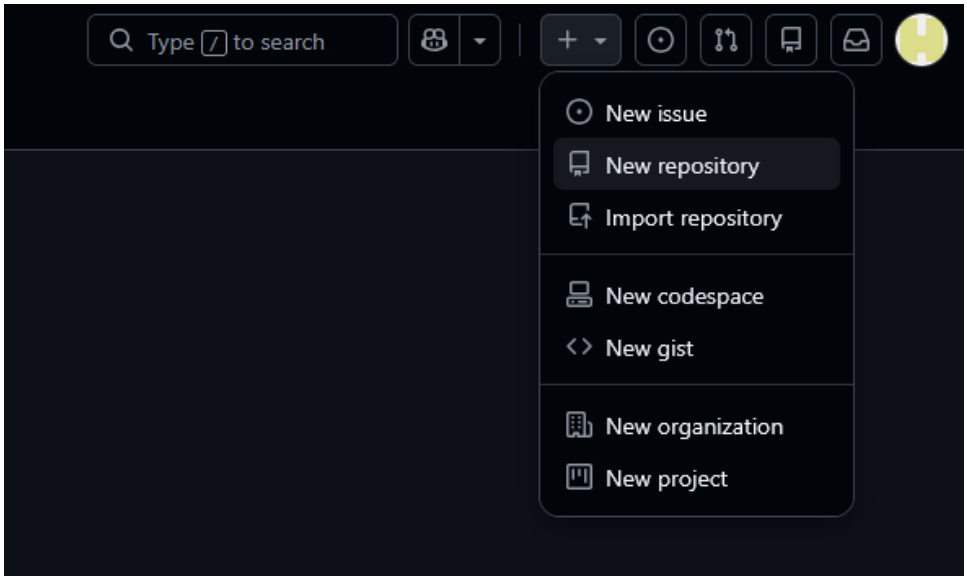
GitHub Pages is a feature built into GitHub that takes the files in your repository and serves them as a live website. It rebuilds automatically every time you commit new changes.

> **Tip**
> If some of these terms are unfamiliar right now, that is fine. They will make more sense as you work through the steps.

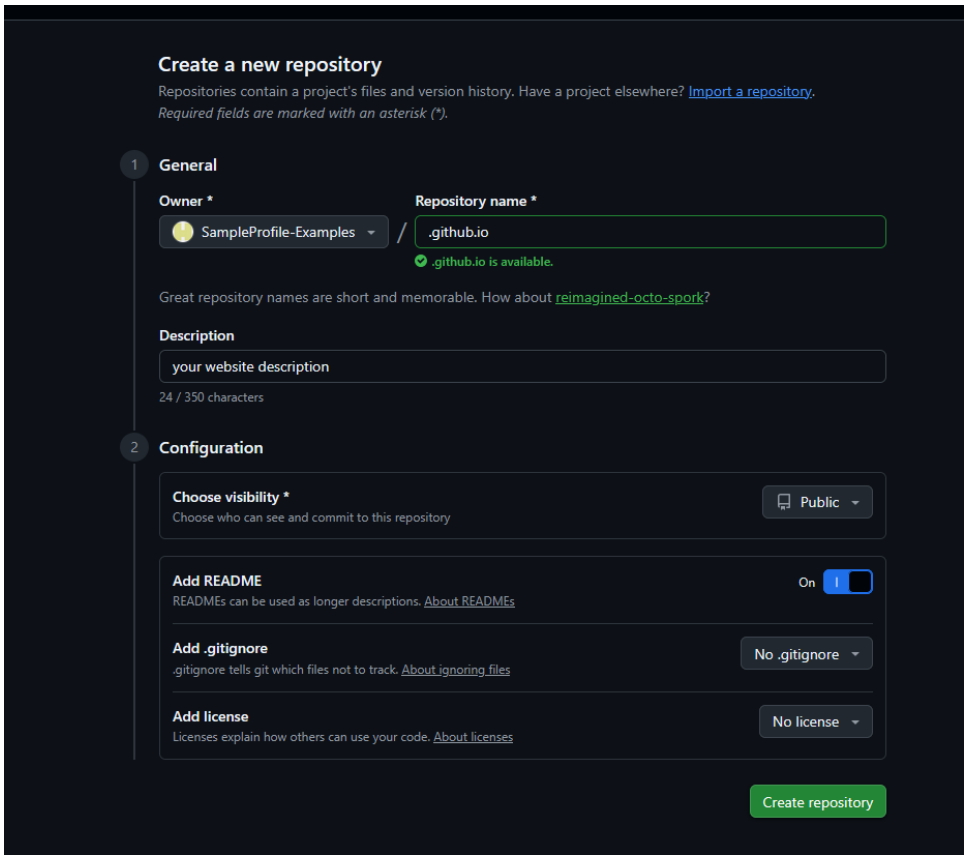
SECTION 2: CREATING YOUR REPOSITORY

Your repository is where your site files will live on GitHub. This section walks you through creating one.

1. Go to **github.com** and sign in to your account.
2. Click the **+** button in the toolbar at the top of any GitHub page, then select **New repository** from the dropdown. You can also go directly to github.com/new.



3. In the **Repository name** field, enter your GitHub username followed by .github.io. For example: `mandyhathaway.github.io`. This exact format is required to publish your site at the clean root URL.
4. Set the repository to **Public**. GitHub Pages on a free account requires the repository to be public.
5. Check the box labeled **Add a README file**. This initializes the repository so it is ready to receive files right away.
6. Click **Create repository**.



Issue: The repository name shows a red warning.

Solution: Repository names can only contain letters, numbers, hyphens, and underscores. Make sure you are using the format yourusername.github.io with no spaces or special characters.

Issue: You do not see the New button on the GitHub dashboard.

Solution: Make sure you are signed in. Look for the + button in the top toolbar. It is visible on every page when you are signed in.

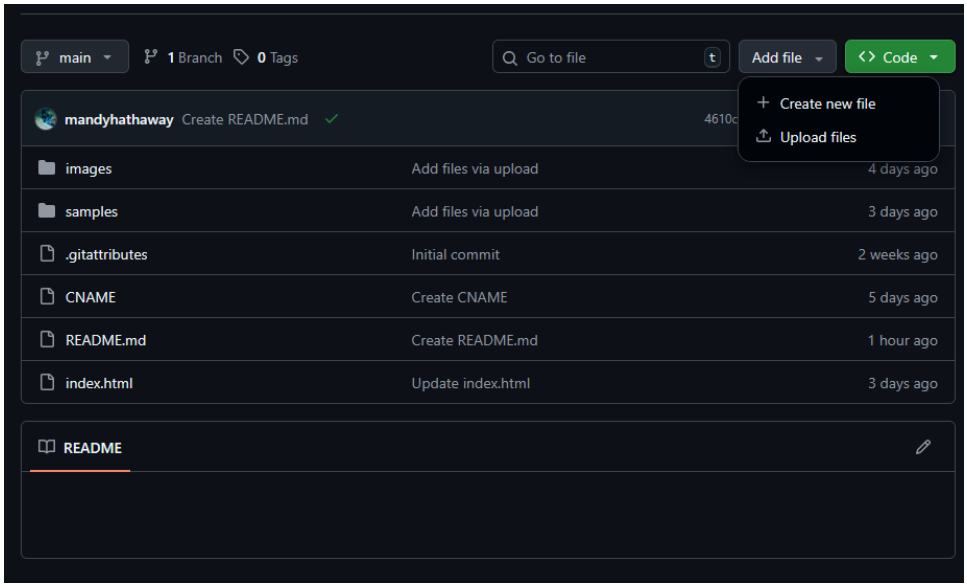
SECTION 3: UPLOADING YOUR SITE FILES

With your repository created, the next step is uploading your site files. Your main page must be a file named index.html in the root of the repository. GitHub Pages will look for this file first when someone visits your site URL.

Important

Double-check that your main HTML file is named index.html exactly, with a lowercase i. If it is named Index.html or index.HTML, GitHub Pages will not find it and your site will show a 404 error.

1. Open your repository on github.com. You should see the repository page with your README file listed.
2. Click **Add file** and then select **Upload files** from the dropdown.

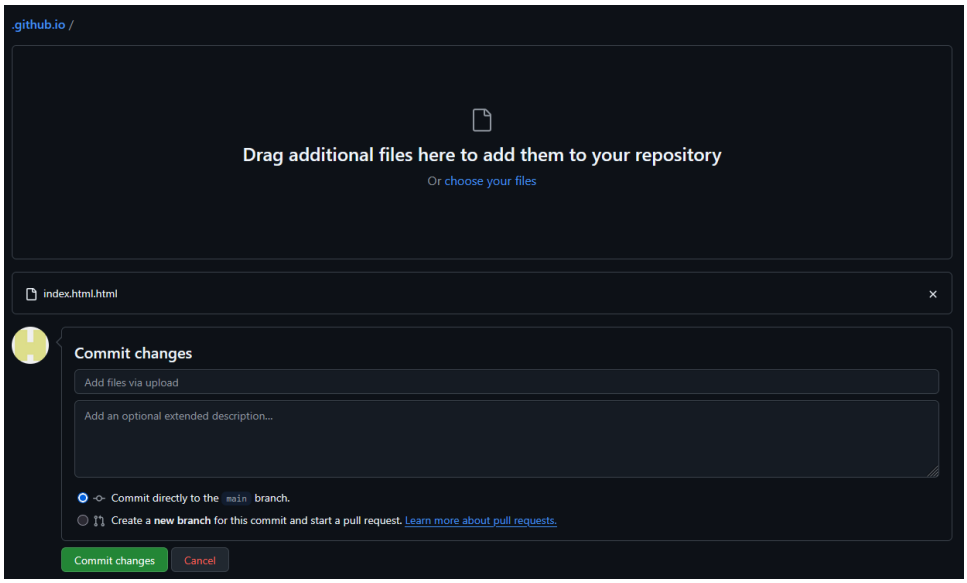


3. Drag your site files and folders into the upload area, or click Choose your files to browse for them. You can upload an entire folder at once by dragging it directly into the upload area.

Important

If your file was named index.html and it appears in the upload list as index.html.html, Windows added a hidden .html extension to a file that already had one. To fix this, open File Explorer, click View, then check File name extensions to make extensions visible. Rename the file to index.html and upload it again.

4. Scroll down to the Commit changes section. Add a short description in the first field if you like, such as "Add initial site files." This becomes the commit message and helps you identify the change later.
5. Click **Commit changes**.



Issue: The upload area does not appear or the page seems stuck.

Solution: Try refreshing the page and starting again. If the issue persists, try a different browser.

Issue: Some files are missing after upload.

Solution: The GitHub website has a file size limit of 25MB per file and a limit of 100 files per upload. For larger projects, break the upload into smaller batches and upload subfolders separately.

Issue: A folder did not upload correctly and its contents are missing.

Solution: The GitHub website does not always handle deeply nested folders reliably. Navigate into your repository, click Add file, and upload the contents of that folder separately.

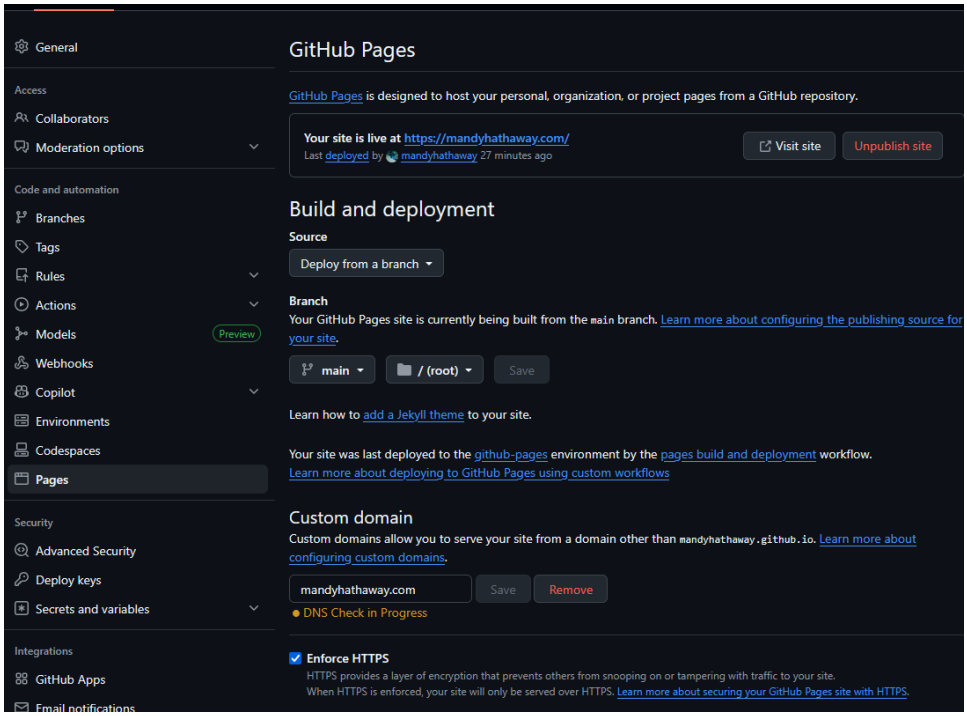
SECTION 4: ENABLING GITHUB PAGES

Your files are now on GitHub. The last step before your site goes live is telling GitHub Pages which branch to publish from.

1. Go to your repository on github.com.
2. Click the **Settings** tab near the top of the repository page.
3. In the left sidebar, click **Pages**. It is listed under the Code and automation section.

4. Under **Branch**, click the dropdown that currently says None and select **main**.
5. Leave the folder set to **/ (root)** unless your index.html file is inside a subfolder.
6. Click **Save**.

GitHub will take a minute or two to build and deploy your site. Refresh the Pages settings page after a minute and you will see a message that your site is live, along with your URL.



Tip

Bookmark your Pages settings page. It is the quickest way to check your deployment status and find your URL whenever you need it.

Issue: The Branch dropdown only shows None and there is no main option.

Solution: Your repository may not have been initialized with a README. Go back to your repository, click Add file, then Create new file, name it README.md, add any content, and commit it. The main branch will then appear in the Pages settings.

Issue: The site shows a 404 error after deployment.

Solution: The most common cause is a missing or misnamed index.html file. Go back to your repository and confirm that index.html exists in the root folder, not inside any subfolder.

Issue: The Pages settings page says the site failed to build.

Solution: Click the link next to the error message to see the build log. A failed build usually means a syntax error in an HTML or CSS file, or a broken file path. Fix the issue, commit the corrected file, and GitHub Pages will attempt to rebuild automatically.

SECTION 5: VERIFYING YOUR SITE

Once GitHub Pages reports that your site is live, open a browser and go to your URL:

```
https://yourusername.github.io/
```

Replace yourusername with your actual GitHub username. The URL is case-sensitive.

If your site loads correctly, you are done. If it does not, the troubleshooting items in Section 4 cover the most common causes.

> **Tip**
> GitHub Pages can take up to 10 minutes to publish for the first time. If your URL returns a 404 right after enabling Pages, wait a few minutes and try again before troubleshooting.

Updating Your Site

Every time you want to update your site, the process is straightforward. Navigate to the file you want to change in your repository, click the pencil icon to edit it, make your changes, and click Commit changes. For uploading new or replacement files, use the same Add file process from Section 3.

GitHub Pages picks up the new version and republishes automatically, usually within a minute or two.

> **Tip**
> Whether you are publishing for the first time or pushing an update, it can take up to 10 minutes for changes to appear on your live site. If things do not look right immediately after committing, give it a few minutes and refresh before assuming something went wrong.

Issue: An edited file is not showing the updated version on the live site.

Solution: Your browser may be serving a cached version of the page. Try a hard refresh by pressing Ctrl + Shift + R, or open the site in a private browsing window to see the current live version.

Issue: You edited the wrong file and want to revert the change.

Solution: Go to the file in your repository, click History to see previous versions, find the commit before your change, and use the revert option. Alternatively, edit the file again and recommit the corrected version.

SECTION 6: CONNECTING A CUSTOM DOMAIN (OPTIONAL)

By default, your site lives at a github.io URL. If you have purchased a custom domain name, such as mandyhathaway.com, you can connect it to your GitHub Pages site so it loads at your own domain instead.

> **Important**
> This section requires access to your domain registrar, which is the service where you purchased your domain. The exact steps for updating DNS records vary by registrar. Common options include Namecheap, GoDaddy, Google Domains, and Cloudflare.

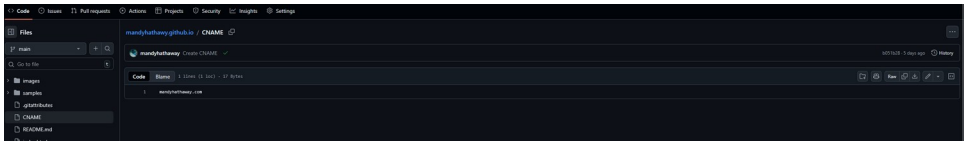
Step 1: Add a CNAME File to Your Repository

A CNAME file tells GitHub Pages which custom domain to respond to.

1. In your repository on github.com, click **Add file** and then **Create new file**.
2. Name the file `CNAME` in all capital letters with no file extension.
3. In the file contents area, type your domain name on a single line with no http:// or www prefix. For example:

```
mandyhathaway.com
```

4. Click **Commit changes**.



> **Issue:** GitHub Pages ignores the CNAME file and still shows the github.io URL.
> **Solution:** Make sure the filename is CNAME in all capital letters with no extension. A file named cname or CNAME.txt will not be recognized.

Step 2: Add Both Sets of DNS Records

Log in to your domain registrar and find the DNS settings for your domain. You will need to add both sets of records below — the A records for your apex domain and the CNAME record for www. Adding both ensures your site loads correctly whether visitors type www or not.

> **Important**
> Before adding the new records, check whether your registrar already has any A records or CNAME records for your domain. Existing A records pointing to a different host will conflict with the GitHub IP addresses and prevent your site from loading. Delete any conflicting records before adding the ones below. If you are unsure whether a record is safe to delete, check your registrar documentation or contact their support. Some records, like MX records for email, should be left alone.

For an apex domain (for example, mandyhathaway.com)

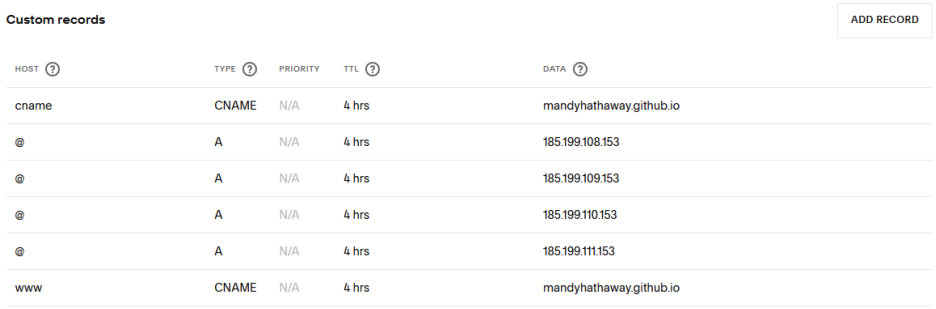
Add four A records pointing to the following GitHub IP addresses:

```
185.199.108.153
185.199.109.153
185.199.110.153
185.199.111.153
```

For a www subdomain (for example, www.mandyhathaway.com)
Add a CNAME record with the name www pointing to your github.io address:

```
yourusername.github.io
```

> **Tip**
> Most registrars let you add both an apex domain and a www subdomain. Adding both means your site will load correctly whether visitors include www or not.

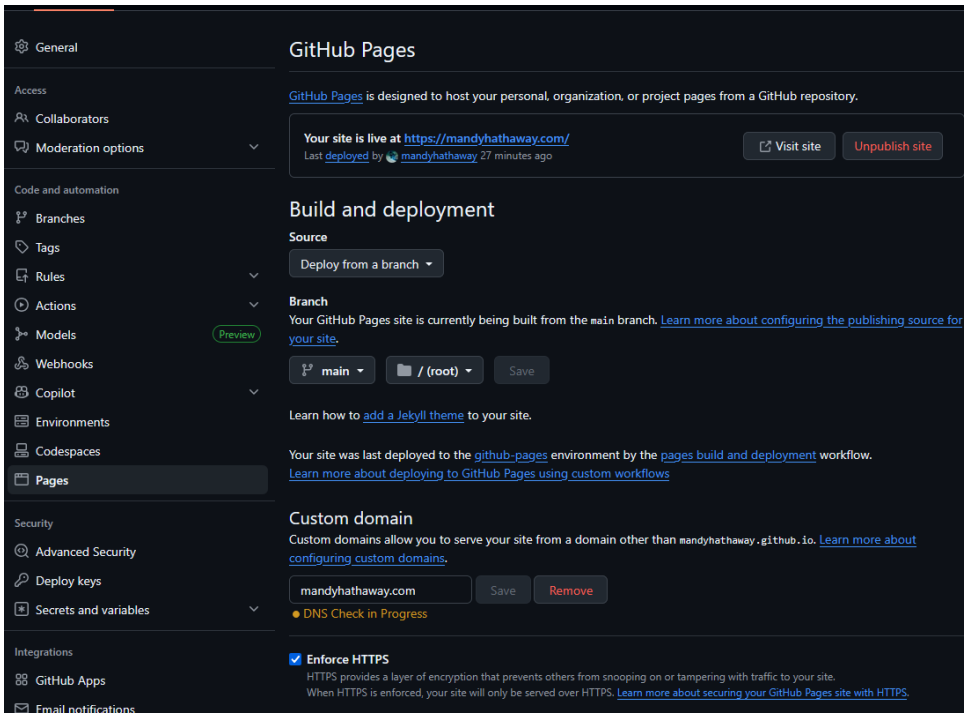


> **Note**
> Your registrar may automatically add a cname host entry when you save your custom domain in GitHub Pages settings. This is normal and you can leave it in place. The records you need to add manually are the four A records pointing to the GitHub IP addresses and the www CNAME record pointing to yourusername.github.io. The exact layout of your DNS panel will look different depending on which registrar you use, but the record types and values are the same regardless.

> **Issue:** You are not sure where to find DNS settings in your registrar.
> **Solution:** Look for a section called DNS Management, DNS Settings, or Advanced DNS in your registrar control panel. If you cannot find it, search your registrar name plus "DNS settings" for instructions specific to that provider.

Step 3: Add the Custom Domain in GitHub Pages Settings

1. Go to your repository **Settings**, then **Pages**.
2. Under **Custom domain**, type your domain name and click **Save**.
3. Check the box labeled **Enforce HTTPS** once it becomes available. This may take a few minutes after saving your domain.



DNS changes can take anywhere from a few minutes to 48 hours to propagate fully. If your domain does not load immediately, wait a few hours and try again before troubleshooting.

Issue: GitHub Pages shows a domain verification error after entering your custom domain.

Solution: This means your DNS records have not fully propagated yet, or there is a typo in your CNAME file or DNS settings. Double-check that the domain in your CNAME file exactly matches what you entered in the Pages settings, and verify your DNS records at your registrar.

Issue: The site loads at the custom domain but shows a security warning about the certificate.

Solution: Make sure Enforce HTTPS is checked in your Pages settings. If the option is greyed out, GitHub is still provisioning your SSL certificate. Wait 15 to 30 minutes and check again.

Issue: The apex domain works but www does not, or vice versa.

Solution: Check that you have added both the four A records for the apex domain and the CNAME record for the www subdomain in your DNS settings. Both sets are required for the site to load at both addresses.

QUICK REFERENCE

Your Site URL

```
https://yourusername.github.io/index.html
```

GitHub Pages will also load your site at the root URL without the filename. Both addresses point to the same page:

```
https://yourusername.github.io/
```

Useful Links

- GitHub Pages documentation: docs.github.com/pages
- Check DNS propagation: dnschecker.org
- GitHub file size and upload limits: docs.github.com/repositories/working-with-files/managing-large-files

File Requirements

- Main page must be named index.html
- The repository must be named yourusername.github.io exactly
- All file and folder names are case-sensitive
- Files must be in the root of the repository unless you configure a subfolder in Pages settings
- Maximum individual file size via the website upload: 25MB
- Maximum files per upload batch: 100

Update Workflow

To edit an existing file: open the file in your repository, click the pencil icon, make your changes, and click Commit changes.

To add or replace files: click Add file, then Upload files, drag in your updated files, and click Commit changes.

GitHub Pages rebuilds automatically after every commit, usually within one to two minutes. Allow up to 10 minutes for changes to appear on the live site.