Deploying a static site with netlify

If you want to share a site that doesn't have a backend and provides the same information to all visitors, then deploying a static site would be perfect. As you know, the most efficient way to deploy these static sites is through Netlify.

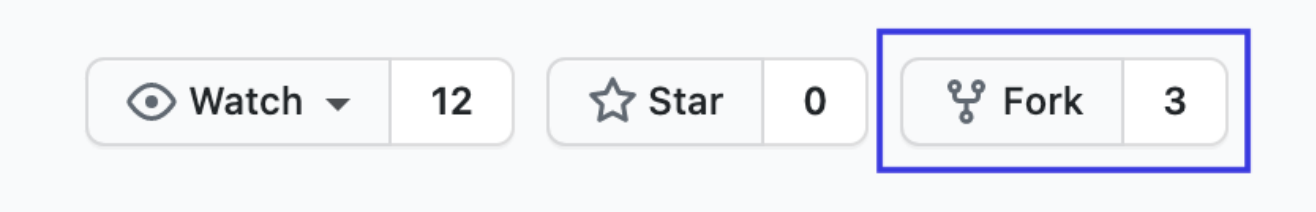
To do so, you'll need to have the following:

- A Netlify account, which can be created step-by-step in "Getting Started with Netlify".
- A GitHub account that you'll be using to fork an existing repository. If you don't know how to fork a repository, check out our course on GitHub.

Forking the repository

For the purpose of this article, we'll be using Codecademy's "deploying-a-static-site-with-netlify-sample" repository. You can use your own repository if you wish, but you'll have to modify the following steps accordingly.

To fork the repository to your personal GitHub account, click the **Fork** button on the original repository's page:



You should then have your own copy of the "deploying-a-static-site-with-netlify-sample" repository, which you'll soon deploy.

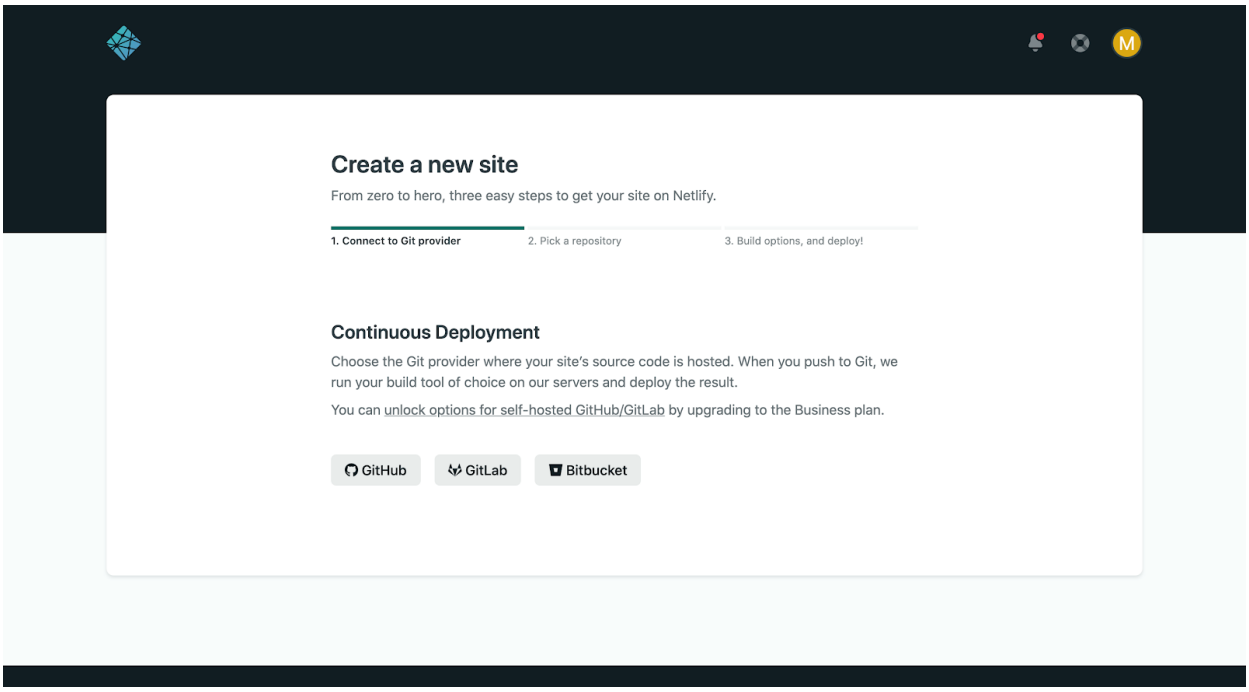
Using Netlify

Let's get started with deploying the GitHub repository — it only takes a few simple steps! First, make sure you're logged in to your Netlify account and then go to your dashboard.

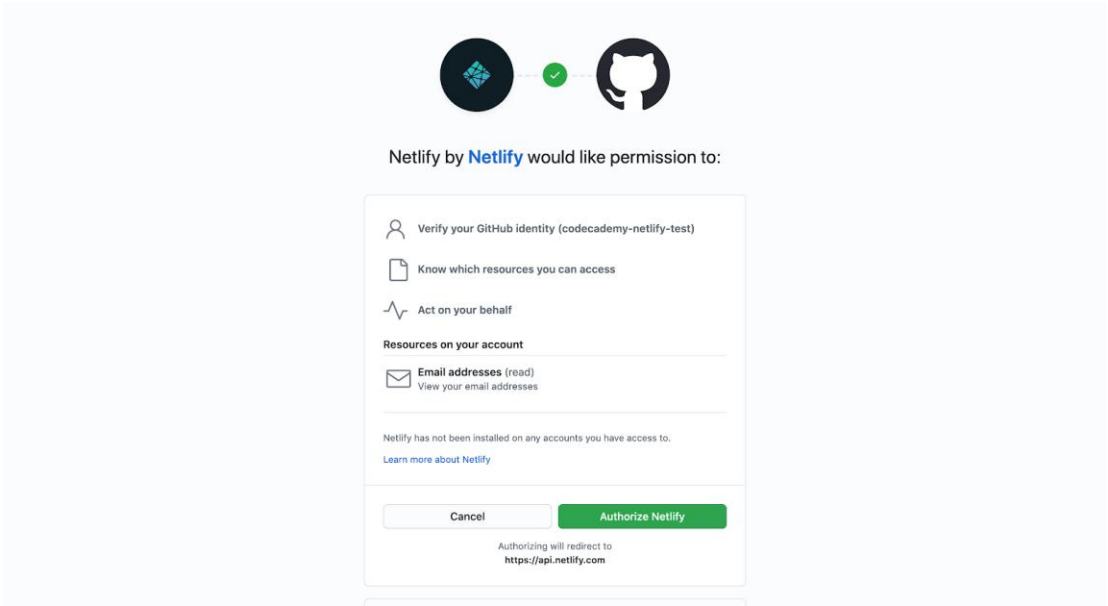
- https://app.netlify.com (hit 'log in with email' → Hotmail account → ?98net)

Creating a site from Git

In your Netlify dashboard, click **New site from Git** (not shown) to start the deployment process. You'll be directed to a page (shown below) that shows the first step, which is connecting to your Git provider:



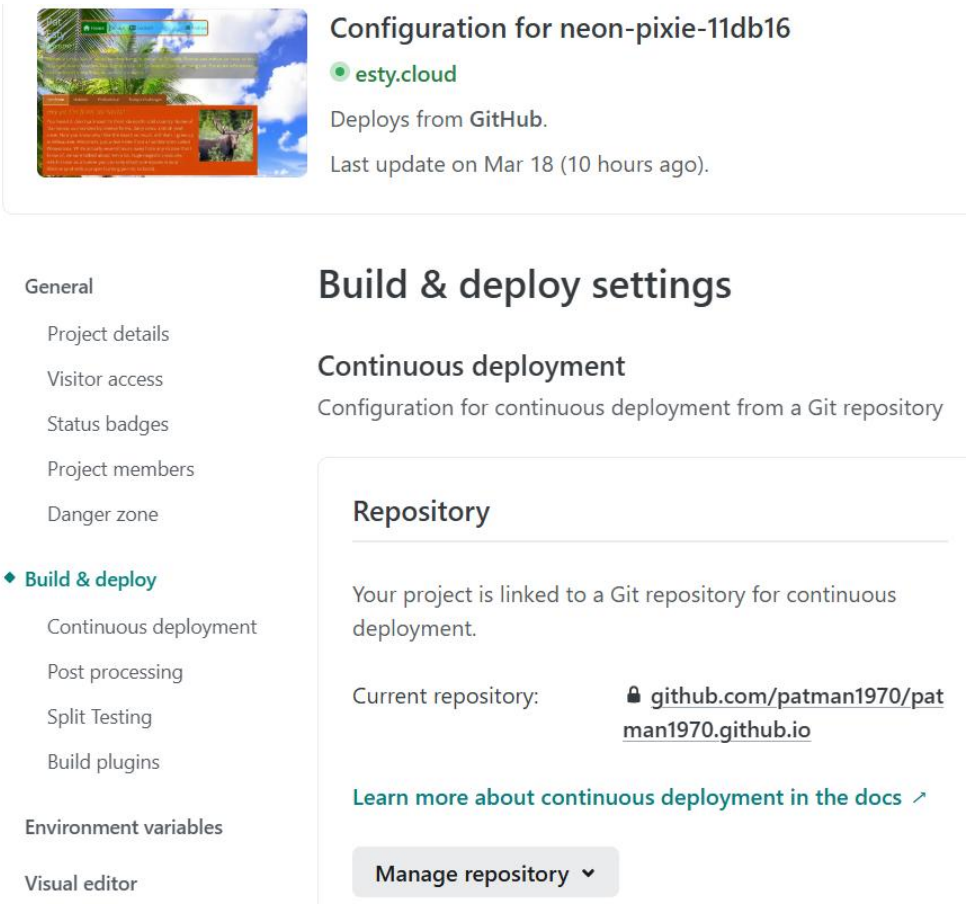
You should select the **GitHub** option because that is where your forked repository is hosted. However, **GitLab** and **Bitbucket** are available if you want to use those for other projects.



Notes:

1) If you didn't create your Netlify account using GitHub, you'll need to provide connection permissions by clicking **Authorize Netlify**. Doing so will eventually allow Netlify to connect with the GitHub repository that you want to deploy.

2) You also need to install Netlify to your GitHub account by clicking **Install**. Note that you may have to enter your GitHub password again. If you successfully connected your GitHub account to Netlify, you should see your GitHub username and forked "deploying-a-static-site-with-netlify-sample" repository under "Continuous Deployment: GitHub App" -- choose the "**deploying-a-static-site-with-netlify-sample**" repository to link it with Netlify.
3) In the next step, check that the "Owner" dropdown is set to the correct team (your Netlify account name in this case) and "Branch to deploy" is set to **main**. Scrolling down, you'll see options for "**Basic build settings**" (or "**Build & deploy settings**"):



4) Regarding the "Build command" and "Publish directory", note the following:

- Because this is an HTML, CSS, and JavaScript build, there is no need for *build commands*, which are instructions that Netlify uses to build certain types of sites like those from frontend frameworks and static site generators.
- Since no build commands are used for this site, you also don't need a *publish directory*, which would contain deploy-ready HTML files and assets generated by the build.

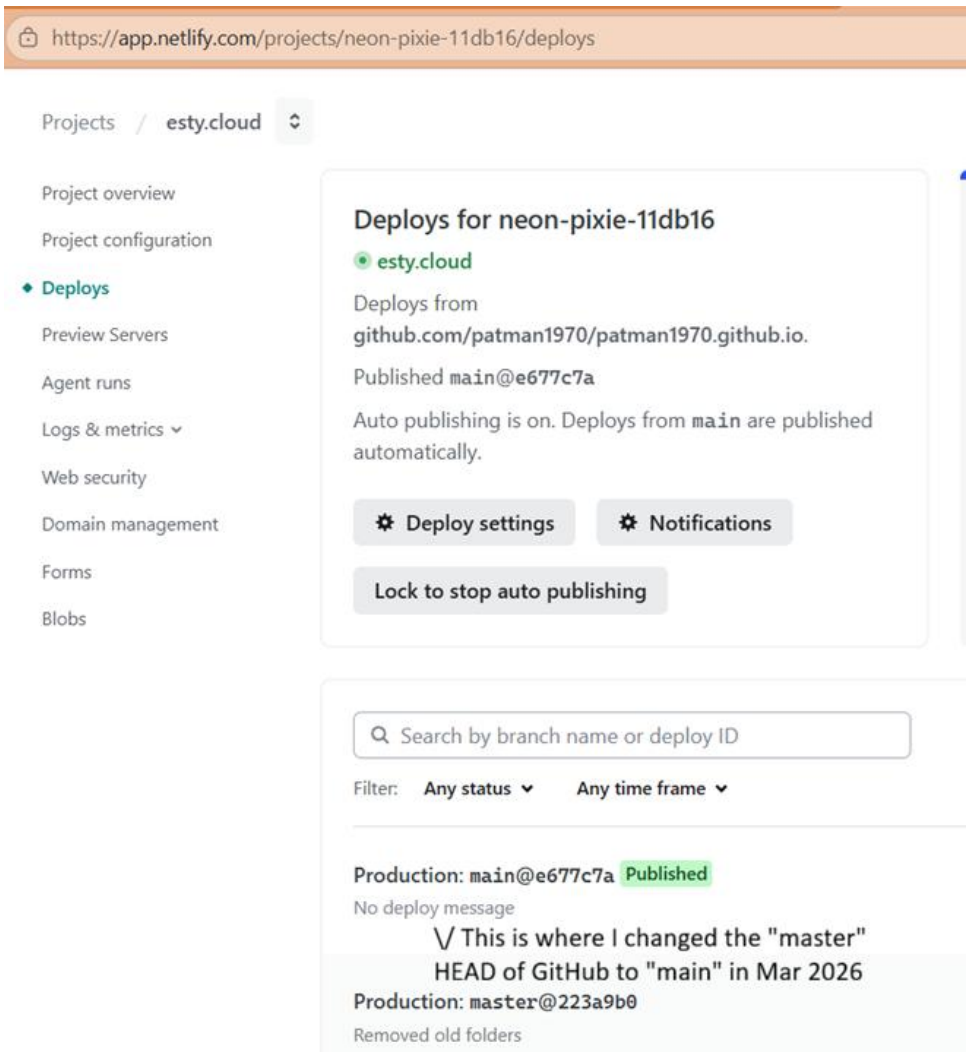
5) You're now ready to deploy, so click **Deploy site**! You'll then be directed to the site's dashboard. The build status should continue to show "Site deploy in progress" and

"Deploying your site" until the site has been successfully deployed. At that point, the status will change to "Your site is deployed" and you'll see a link to your deployed site: Click the final link, and you'll be directed to your deployed site:

6) Notice that the page's background color changes every time you click the 👏 (clap) button!

Production deploys

You can check the ongoing log that contains in-depth information about the site's build status, which is especially useful for debugging any errors that may arise. To see these logs, scroll down to the "Production deploys" section of the **Site overview** tab:



Analyzing the logs

In the logs, you should specifically look for "Published" or "Failed" statuses:

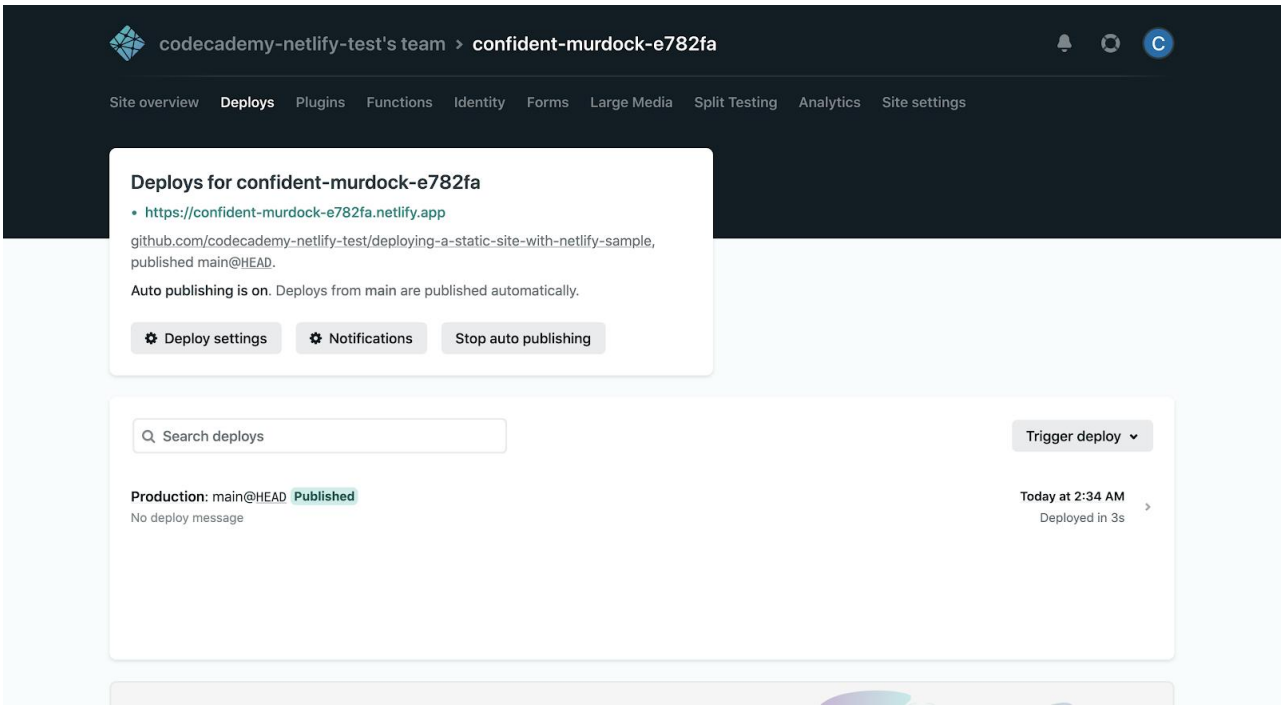
- A “Published” status means that your site has been successfully deployed.
- A “Failed” status means that something went wrong in the deployment process.

Since every project is unique, the best way to fix a “Failed” deployment is to either google the error or browse through Netlify’s documentation. With that being said, you may find the following resources from Netlify to be helpful:

- Build troubleshooting tips
- Netlify Community forum
- Frequently encountered problems during builds guide

Making changes to the site

Next, select the **Deploys** tab located at the top of the site dashboard:



Notice that “Auto publishing is on”, meaning that your site automatically deploys each time you push a commit on GitHub (continuous deployment). Although not recommended, you can turn this setting off by clicking **Stop auto publishing**.

To make changes to your site, update your forked GitHub repository. Here’s a simple example for changing the site’s heading (note that these steps are specific to the GitHub website interface):

1. Navigate back to your forked “deploying-a-static-site-with-netlify-sample” repository and make an edit to the `<h1>` text (e.g., you could change the text inside the `<h1>` element from “Welcome to your website!” to “Welcome to Jane’s website!”).
2. Commit your changes and wait a few moments for Netlify to update the build.
3. Like before, you can monitor the build status through the site’s dashboard in your Netlify account.
4. After the build is complete, you should notice a difference in your published site’s heading — it’s that easy to make changes to your Netlify site!

Site settings

Let’s now explore the **Site settings** tab, also at the top of the site dashboard.

Changing the site name

Notice that the name of your site is randomly generated by Netlify. To rename your site to something more appropriate, click the **Change site name** button located in the “Site information” panel. From there, you can **Save** a new unique name, which also determines the site’s default URL, like so:

![Screenshot of changing the site’s name]

Build settings

Next, click the **Build & deploy** tab on the left to see even more settings that you can manipulate:

![Screenshot of the “Build & deploy” section of the “Site settings” tab]

You’ll first notice a panel for “Build settings”. As touched upon before, many frontend frameworks like React and static site generators like Jekyll require special build commands that tell Netlify how to build your specific site. Build commands weren’t used in this case because it’s only a JavaScript, HTML, and CSS site without any frontend frameworks or static site generators.

Deploy notifications

While still in the **Build & deploy** tab on the left, jump down to the **Deploy notifications** section:

![Screenshot of the “Deploy notifications” section of the “Site settings” tab]

Here you can configure deploy notifications, which can inform you or external services about your site’s deploy activity. For example, the various types of Git notifications can do things like add a comment to your GitHub pull requests indicating the status of the associated deploy.

Note that some types of notifications are only available with paid Netlify plans.

Wrapping up

Congrats! You've just deployed your very own static site using Netlify, and you now know how to share your websites with the world. Specifically, you learned about:

- The use-case of static sites.
- Deploying a static site through Netlify's GitHub integration.
- Analyzing the site logs and debugging "Failed" deployments.
- Using auto publishing to make changes to the site.
- Various site settings like build commands and deploy notifications.

Most importantly, notice how many configurations Netlify handled throughout the process — deploying an entire site only took a few clicks!

Although you've used Netlify to deploy a static site, there's so much more that it can do! Features like pre-built forms, in-depth analytics, and built-in user authentication can take your websites even further.

Article written by The Codecademy Team
The Codecademy Team, composed of experienced educators and tech experts, is dedicated to making tech skills accessible to all. We empower learners worldwide with expert-reviewed content that develops and enhances the technical skills needed to advance and succeed in their careers. **Meet the full team**