SALES

Hackathon

2023

AUTHORS

Andrea Narbot
Bhavana Srinivas
Seán Mc Ternan

TABLE OF CONTENTS

INTRODUCTION 1

Chapter 1: GETTING YOU SET UP AND READY! 2

NETLIFY 2

GITHUB 3

CONTENTFUL 5

Chapter 2: DEPLOYMENT 13

Chapter 3: CONTENTFUL FEATURES 18

SETTING UP YOUR BUILD HOOKS 19

UPDATING YOUR CONTENT 21

EDITING EXISTING CONTENT 21

ADDING NEW CONTENT 23

Chapter 4: NETLIFY FEATURES 25

BRANCH DEPLOYS 26

COLLABORATIVE DEPLOY PREVIEWS 29

ROLLBACKS 33

SITE TRANSFER 35

Chapter 5: BONUS CHALLENGES 37

INSTALL LIGHTHOUSE PLUGIN 37

TURN ON ANALYTICS 38

PASSWORD PROTECTION 38

SUBDOMAINS / HP EDGE 38

CONCLUSIONS 39

GLOSSARY 40

INTRODUCTION

Welcome to the Sales Hackathon! A fun challenge that is designed to put you in the driving seat of a web project that closely resembles a real-world scenario, where a user deploys a composable website!

We'll be using Netlify (CI/CD*), NextJS (Framework*), GitHub (Code Repository/Repo), Contentful (CMS*). This exercise will show you hands-on how Netlify is the all-in-one infrastructure that allows developers to build, host, and scale their web applications.

The following combination of tools really helps to highlight the advantage of the Jamstack and why composable architecture is so powerful. Each tool is best in class, and when combined, it speeds up development for teams and empowers individual users like yourselves to develop, deploy and scale fast.

Here are two examples of the project:

https://seans-experience.netlify.app/
https://narbota.netlify.app/

By the end of this, you will have gone through all of the steps a team would take to get a site onto Netlify and deliver it to the world - You'll also finish up with a nice portfolio site showcasing your work experience that you can take away and continue to develop once you've finished the hackathon.

Thanks again for taking part, and Happy Hacking!

Solutions Engineering Team

CHAPTER 1

GETTING YOU SET UP AND READY!

Before we get started, we're going to need to sign up for the services that we'll be using:

Netlify - Hosting/CI-CD

GitHub - Code Editing

Contentful - Content Management System

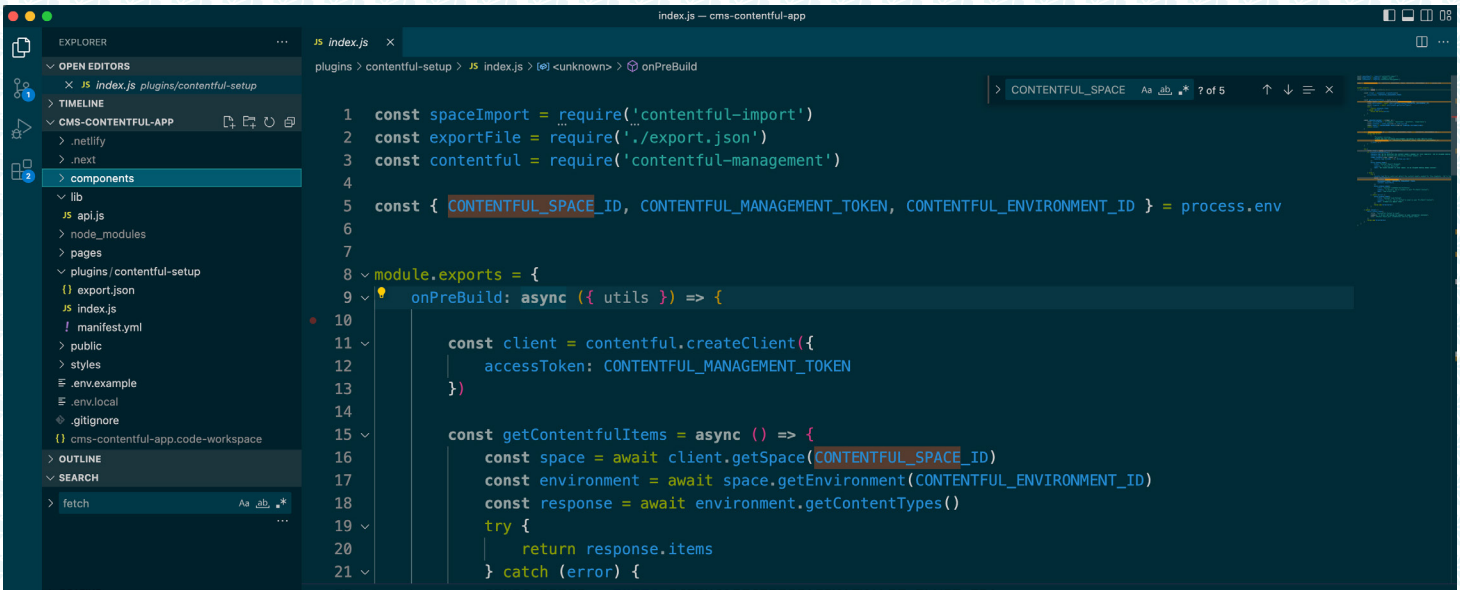
netlify

You should have received an invite to the Sales Hackathon team in Netlify in your company email. Please accept the invitation to be able to have access to the Sales Hackathon team. Please contact Andrea Narbot on slack if you did not receive an invitation. You will use your own personal team (account) in Netlify and you'll also need access to the team so you can transfer your site there at the end.

Next, is the walk-through to create accounts in GitHub and Contentful. If you happen to have either or both, then skip to the steps where we go through copying the required API Keys* for the account.

GitHub

GitHub is a platform for developers to host and share their code, track changes, and collaborate with others. It allows developers to work together on projects and makes it easier to share and contribute code. When you see people coding on their computers in a window that looks like this:



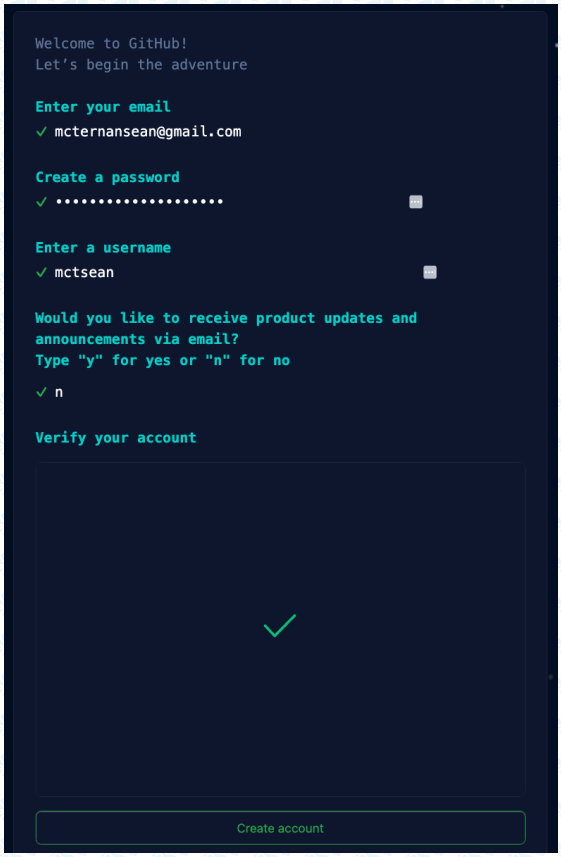
They are usually pushing this code from their local computer to a remote location, or repository. In this case, GitHub. Other remote repository vendors you may have heard of are BitBucket and GitLab. They all do essentially the same thing - Store project code remotely, so many people can work on the same project.

Why is it so important?

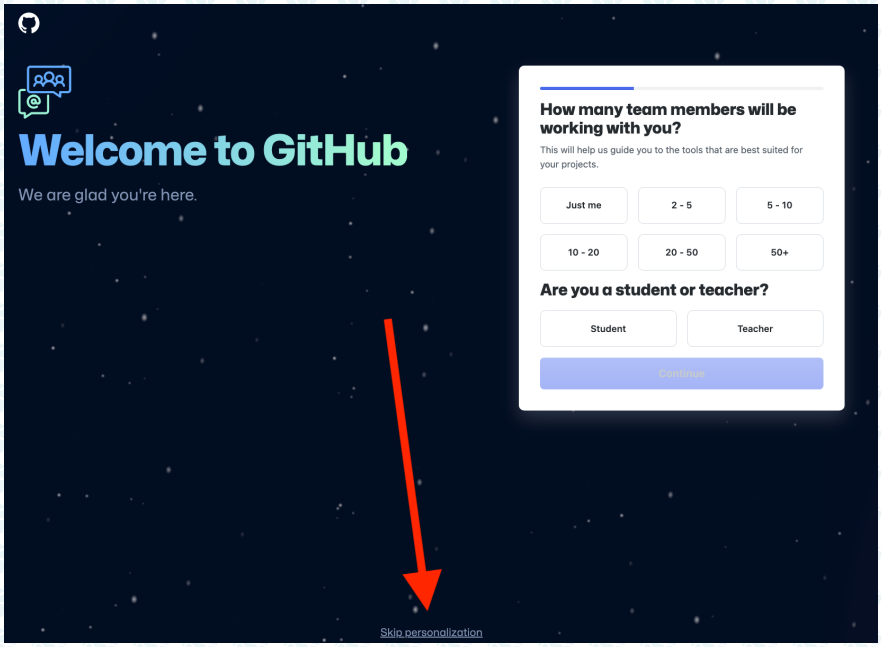
Without a remote git vendor like GitHub, it would be very hard to share your code and its step-by-step changes with your team. Having it live remotely allows everyone to see the change history not only from themselves but from the entire team.

Visit the following URL and Signup for a GitHub Account:

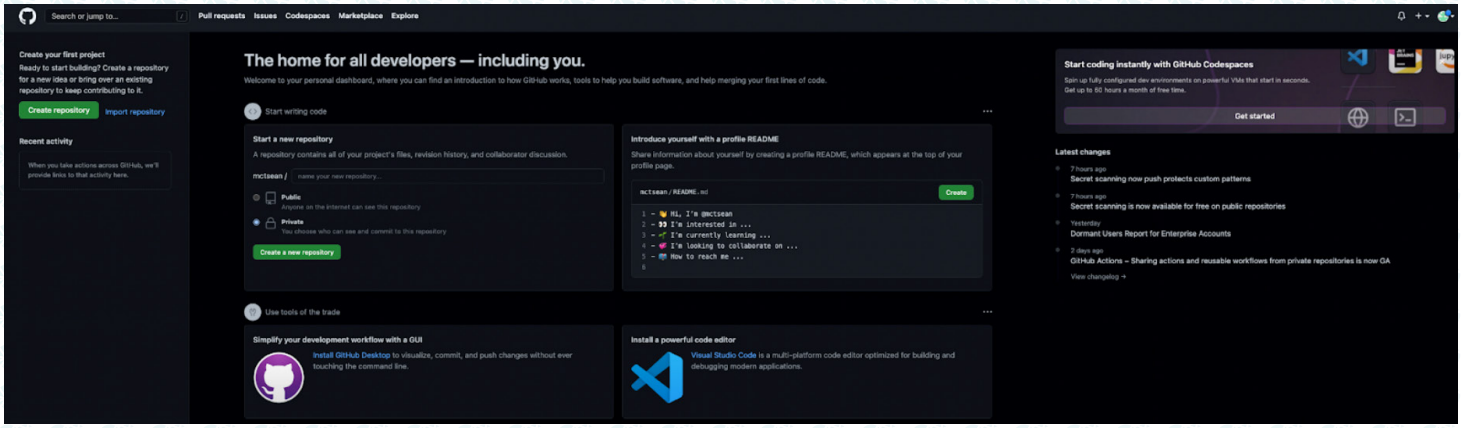
https://github.com/signup?source=login



Enter your details and solve a simple puzzle, then select "Create account". You'll be directed to a new page where you can paste the number that has been sent to your email. It's recommended that you select the 'skip personalization' option on the next window to avoid unnecessary setup steps.



Congratulations! You've just created a GitHub account. You'll now have a great conversation starter when you're at your next social engagement.



Contentful is a cloud-based content management system (CMS) that helps organizations create, manage, and deliver digital content across multiple channels. When we say content here, we mean the very reason people might visit your website - Blogs, images, recipes, or in our case, your professional experience. It's all content! Contentful provides a central repository for storing and organizing content, as well as tools for creating and publishing that content to various platforms. For our purposes, we'll be publishing to a website, but Contentful can get connected to any digital location with an internet connection. Contentful makes it easy for content creators to collaborate, manage workflows, and ensure that their content is consistent and up-to-date.

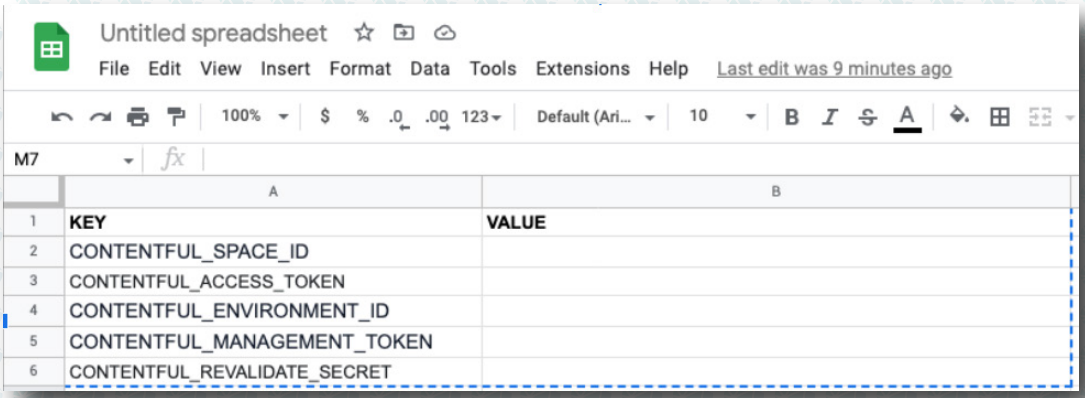
Other CMS vendors in this space are Sanity, Strapi, Drupal, and WordPress. You may be confused because you've heard that Drupal and WordPress are also website-building tools that we don't support. We don't, however, we can support them in a headless* mode. This means that we can grab content from them without the need for them to generate a website or build the website. So why would people want to use them as website builders? Mainly because they've been around a long time and people are slow to change :) The best way to reach people like that is to send them a fax explaining that Netlify is a much better option.

Why is it so important?

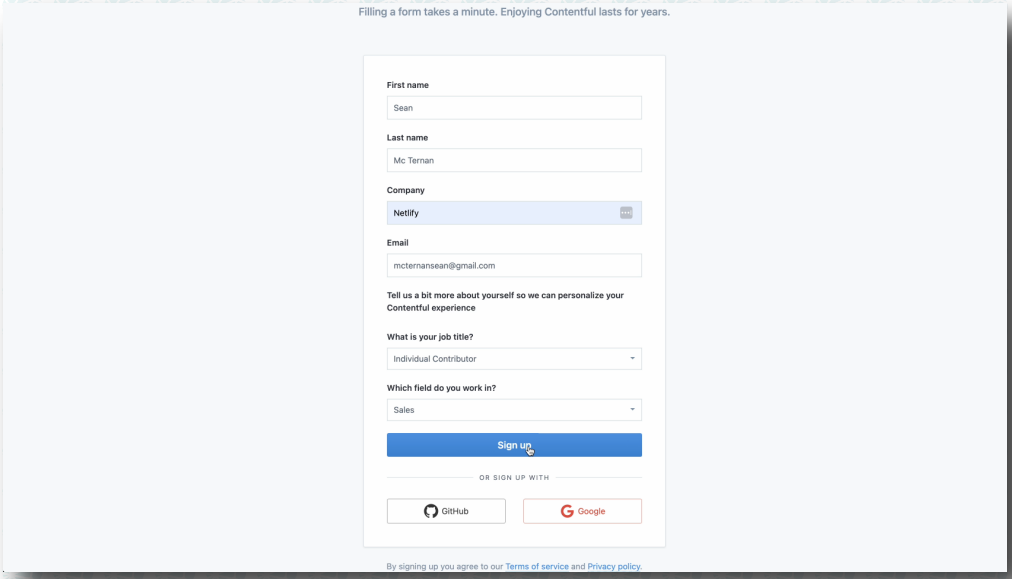
Like many aspects of the Jamstack, Contentful focused on a single area that teams needed help with and made a great tool for that exact thing! Before, people would need to use built-in editors within their site to deploy individual pages or pieces of content. This leads to very slow releases of new content and often the teams would need to pull in DevOps* to push their new content to the site - Not anymore! Using a modern headless CMS like Contentful to release new content, Contentful will let any service that is consuming that content (like Netlify) know that there has been an update and something needs to be done, in the case of this project - rebuild your site with the fresh content!

Visit the following link and sign up for a contentful account:

https://www.contentful.com/sign-up/

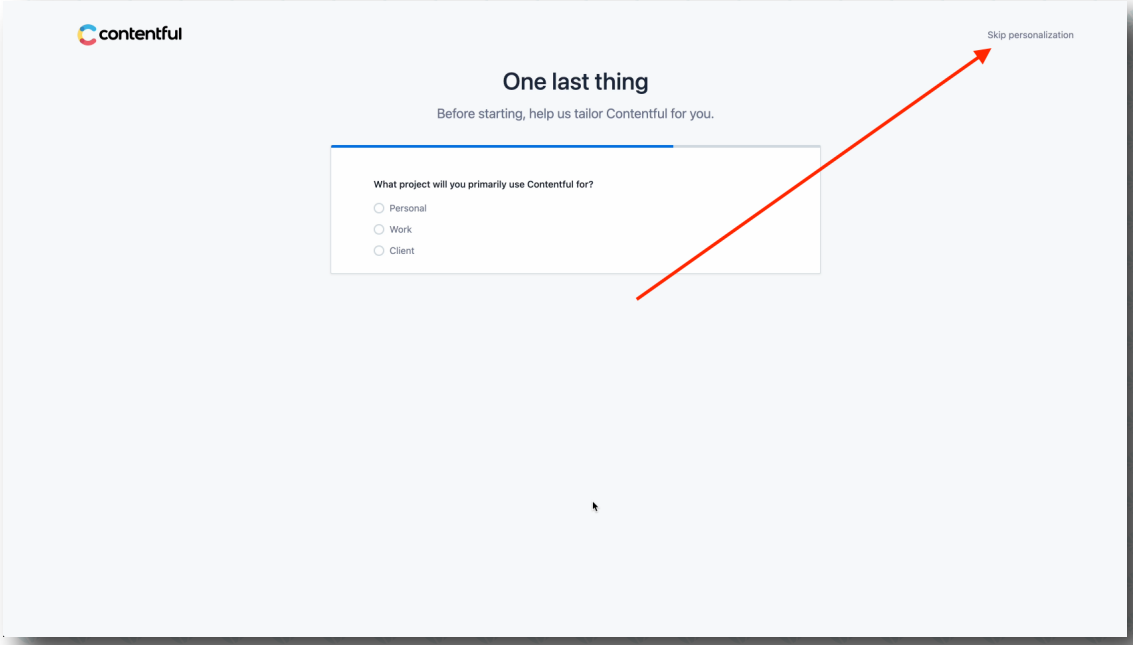


The process is much faster if you select the Signup with GitHub/Google option, but feel free to enter your details manually.



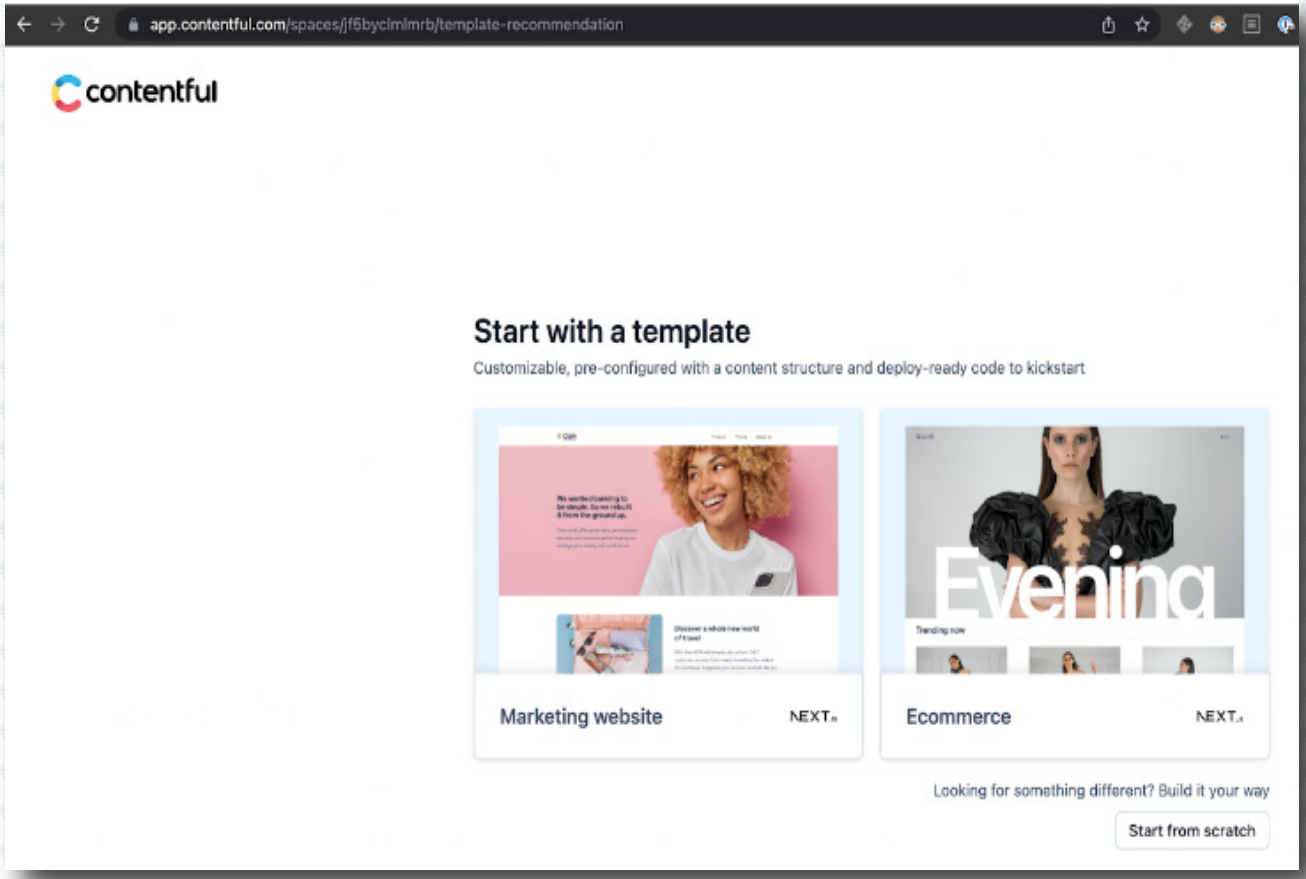
Enter the details that are requested from the Contentful signup flow, and select Sign Up.

Note: *If your window looks different than either of those below, simply jump directly to your email and hit the confirmation link.*

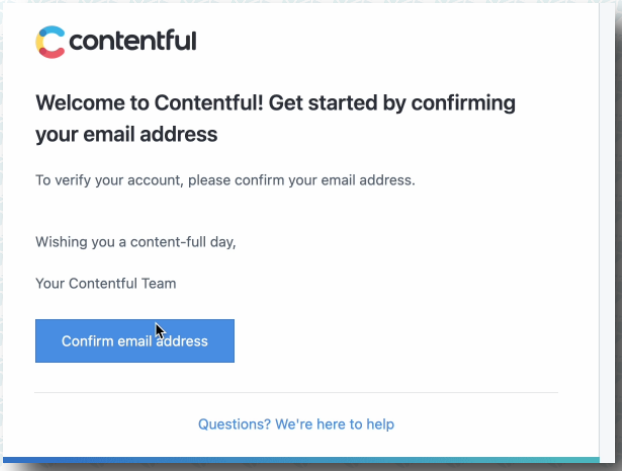


It's recommended that you select the 'skip personalization' option on the next window to avoid unnecessary setup steps.

If you see the image below - select 'Start from scratch'



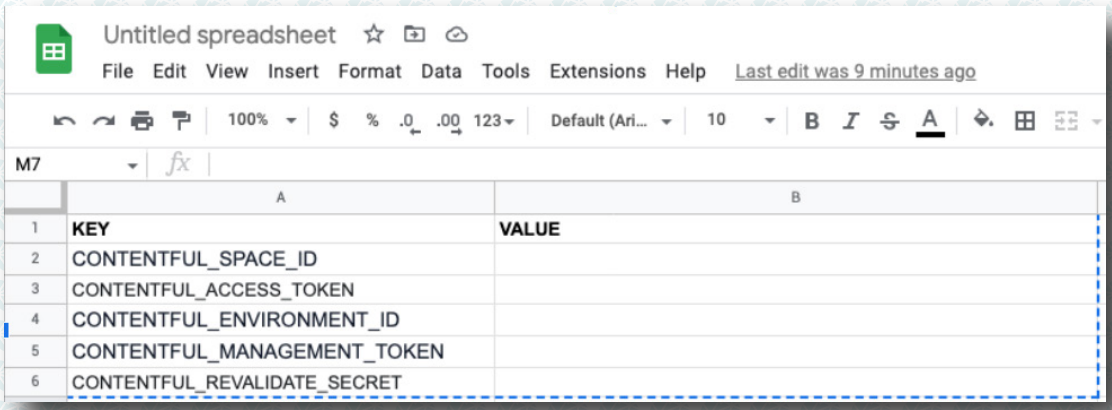
Make sure to head over to your email provider and confirm the email address you gave during the signup. This will ensure your account is saved.



CONTENTFUL_SPACE_ID
CONTENTFUL_ACCESS_TOKEN
CONTENTFUL_ENVIRONMENT_ID
CONTENTFUL_MANAGEMENT_TOKEN
CONTENTFUL_REVALIDATE_SECRET

You don't absolutely need to take the next steps, but it will make Chapter 3 much easier if you do. Go ahead and make a blank Google Sheet and paste the following value into the first row with the header **KEY**, and make the second row called **VALUE**.

It should look like this:



As we go along, we're going to paste the **VALUES** into their corresponding **KEY** row. This will make sense later on and it should make things easier for you and help prevent pasting wrong values into the template you're deploying.

What are these values for?

Picture when you login to any online service, like your email - The provider needs a way to verify that you are who you say you are. The values above do something similar for Contentful. Rather than a real user logging into Contentful with a username and password, they ask for keys and IDs from services like Netlify to ensure they're connecting the right service to the right account and its content. Keeping with the theme of an email provider this is how the values above might be thought of:

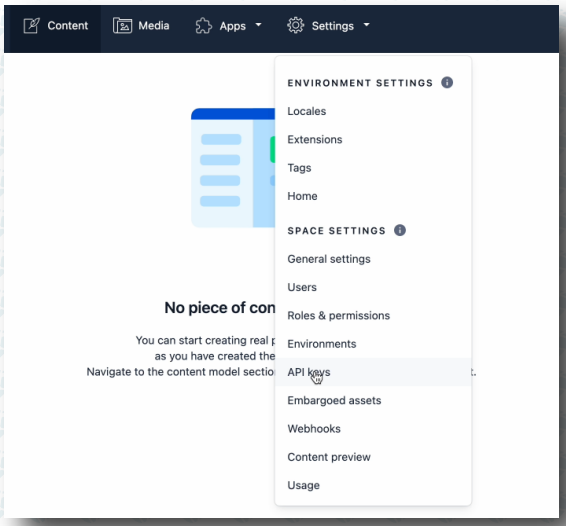
CONTENTFUL_SPACE_ID = The Username
CONTENTFUL_ACCESS_TOKEN = The Password
CONTENTFUL_ENVIRONMENT_ID = The specific account you would like to access after you log in

The tokens below don't really fit with the theme of email, but we'll do our best to explain what they are:

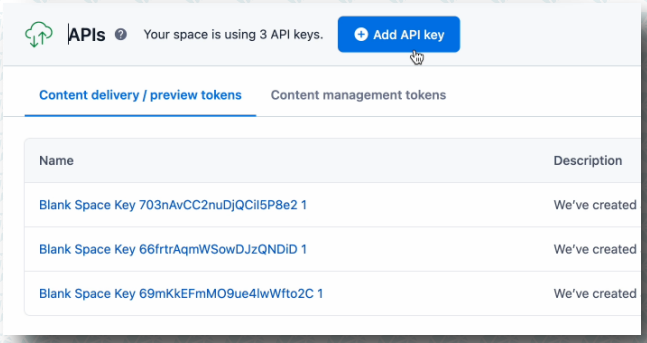
CONTENTFUL_MANAGEMENT_TOKEN
This is a special token that allows you to write content to Contentful, rather than the token above which allows you to read content from Contentful. Breaking up permission like this helps teams limit who can do what and prevents the nightmare scenario where someone breaks your account!

CONTENTFUL_REVALIDATE_SECRET
There is no simple way to describe this without going deep into NextJs, the framework we are using - But in short, this helps our framework to determine if any content has been updated since a visitor to our website has last asked for it.

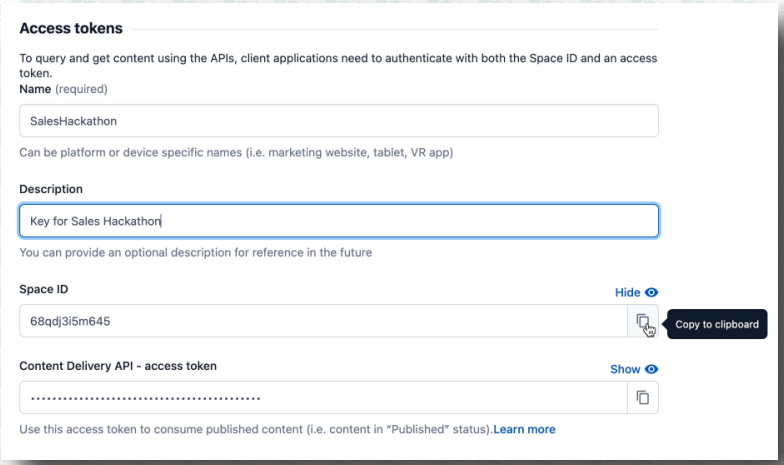
Back over on Contentful,
Navigate to **Setting** > **API Keys**



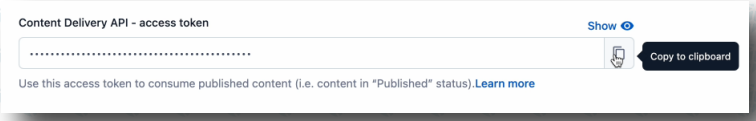
Select **Add API Key**



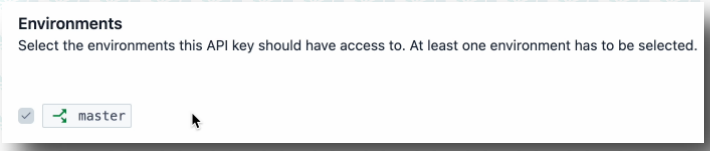
Give the Key a Name, and a Description. Next copy the *Space ID* and paste it into the **VALUE** column for the CONTENTFUL_SPACE_ID **KEY** row on your spreadsheet.



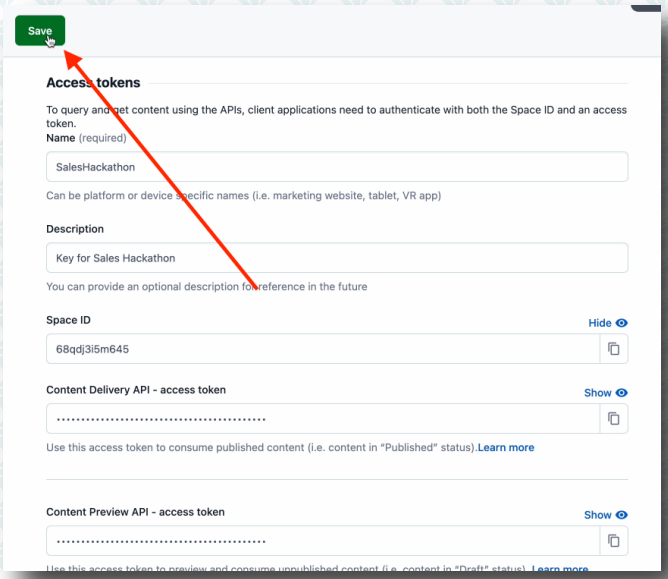
Do the same for the *Content Delivery API - access token*, we'll paste this into the **VALUE** for the CONTENTFUL_ACCESS_TOKEN **KEY**.



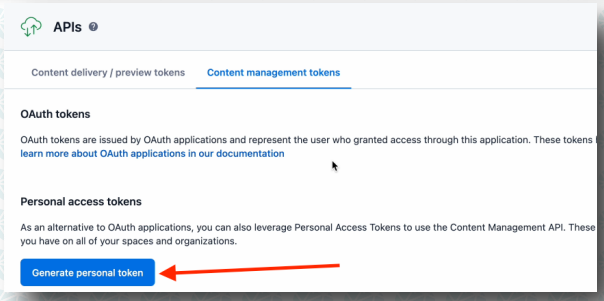
Scroll down to the bottom of the page, copy the name of your environment (It's most likely going to be master), and paste its **VALUE** into the CONTENTFUL_ENVIRONMENT_ID **VALUE** row. Or if it's labeled master, then you just write *master*.



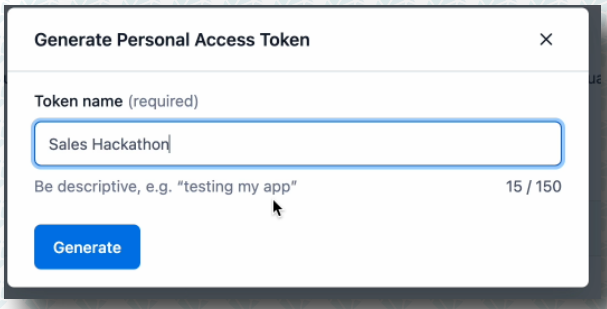
Scroll back to the top and click the ‘**Save**’ button.



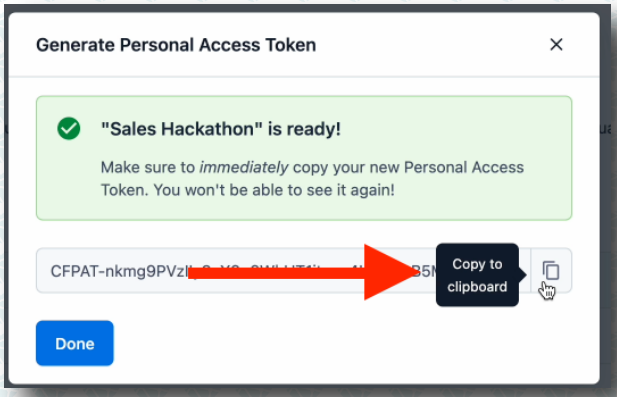
Next, navigate to ***Setting > API Keys***, click on the *Content management tokens* tab and select **Generate personal token**.



Go ahead and give it a name and hit *Generate*.



Copy the value and paste it into the CONTENT-FUL_MANAGEMENT_TOKEN **VALUE** area on the spreadsheet.



Finally, go over to your spreadsheet and paste any **VALUE** you want into the CONTENT-FUL_REVALIDATE_SECRET for example: 90ebab4482e8169523418f9268

Whew! You did it. We promise this will make sense later, but now you are set up for what you'll accomplish in chapter 2 and beyond.

When you're finished all of the above steps, your sheet should look something like this:

KEY	VALUE
CONTENTFUL_SPACE_ID	wtffzmo9n8yk
CONTENTFUL_ACCESS_TOKEN	7kjHqF98Jo5aYC5bAiWxxxxxxcwP-H_gG46YOaVO0M
CONTENTFUL_ENVIRONMENT_ID	master
CONTENTFUL_MANAGEMENT_TOKEN	CFPAT-EydOCEkDpeMLrYGTfVc8Lf_n99ELruF8thR16PF1SrI
CONTENTFUL_REVALIDATE_SECRET	50e6bcb152860fe84a90ebab4482e8169523418f9268112333c8b602dbd978b0

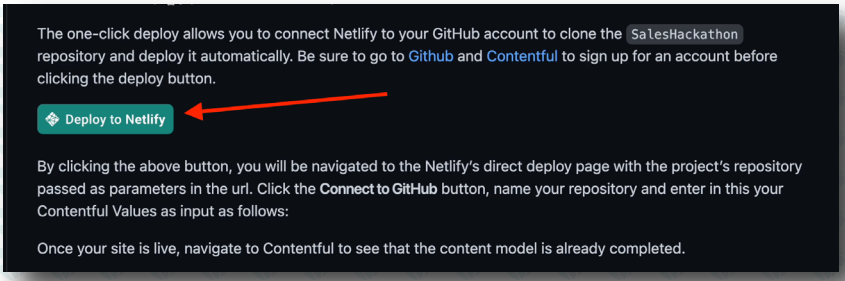
CHAPTER 2

DEPLOYMENT

Congratulations! You've done all of the work needed to get right into the meat of this project. In the following chapter, we're going to show you just how easy it is to launch a new website to Netlify. You may have noticed that by the end of Chapter 1 there was no actual content in your Contentful instance, that is true but luckily we've created a template that will not only connect your Netlify instance to your Contentful account, but it will also populate your Contentful account with dummy data so you can get started quickly!

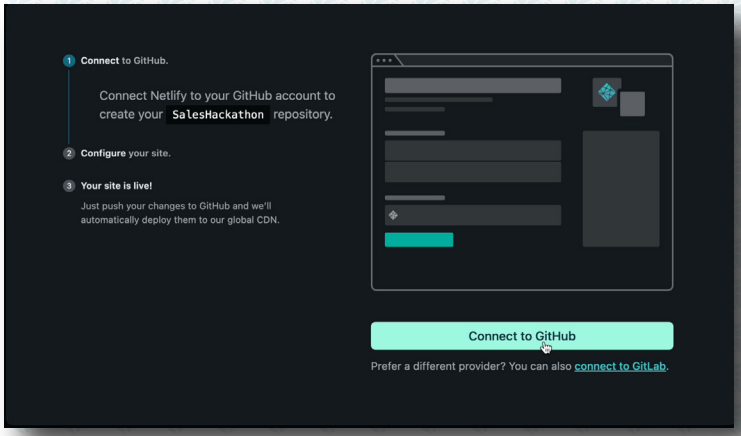
Let's get started!

First, make sure that you are logged into Netlify and GitHub. Next, have the **VALUES** that we copied in the spreadsheet in Chapter 1 ready, as we will be pasting them into our site as an Environment Variable*.

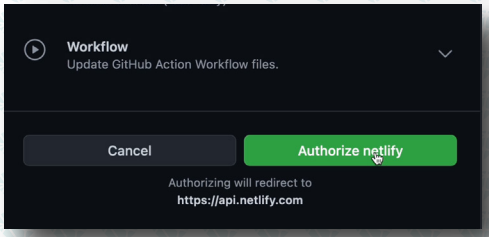


Next, visit https://github.com/Netlify-Moneytronic/SalesHackathon and scroll down until you see the Deploy to Netlify button in the ReadMe. Go ahead and click the button.

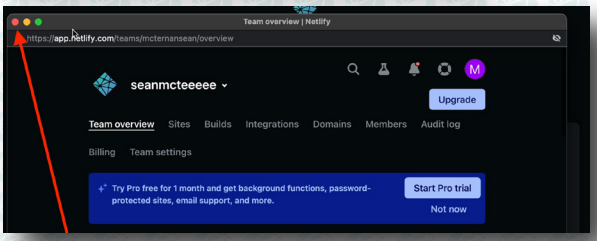
If this is your first time using Netlify and GitHub together, it's going to ask you to connect your GitHub Account, click the button to connect. If you are not seeing this, skip* this section and jump to the steps below where we paste our environment variables.



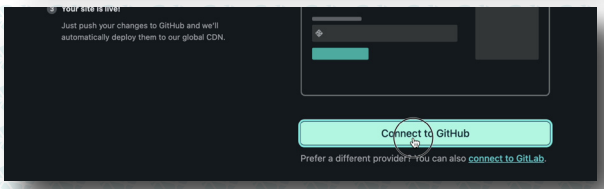
Scroll down until you see the 'Authorize Netlify" button and click on it.



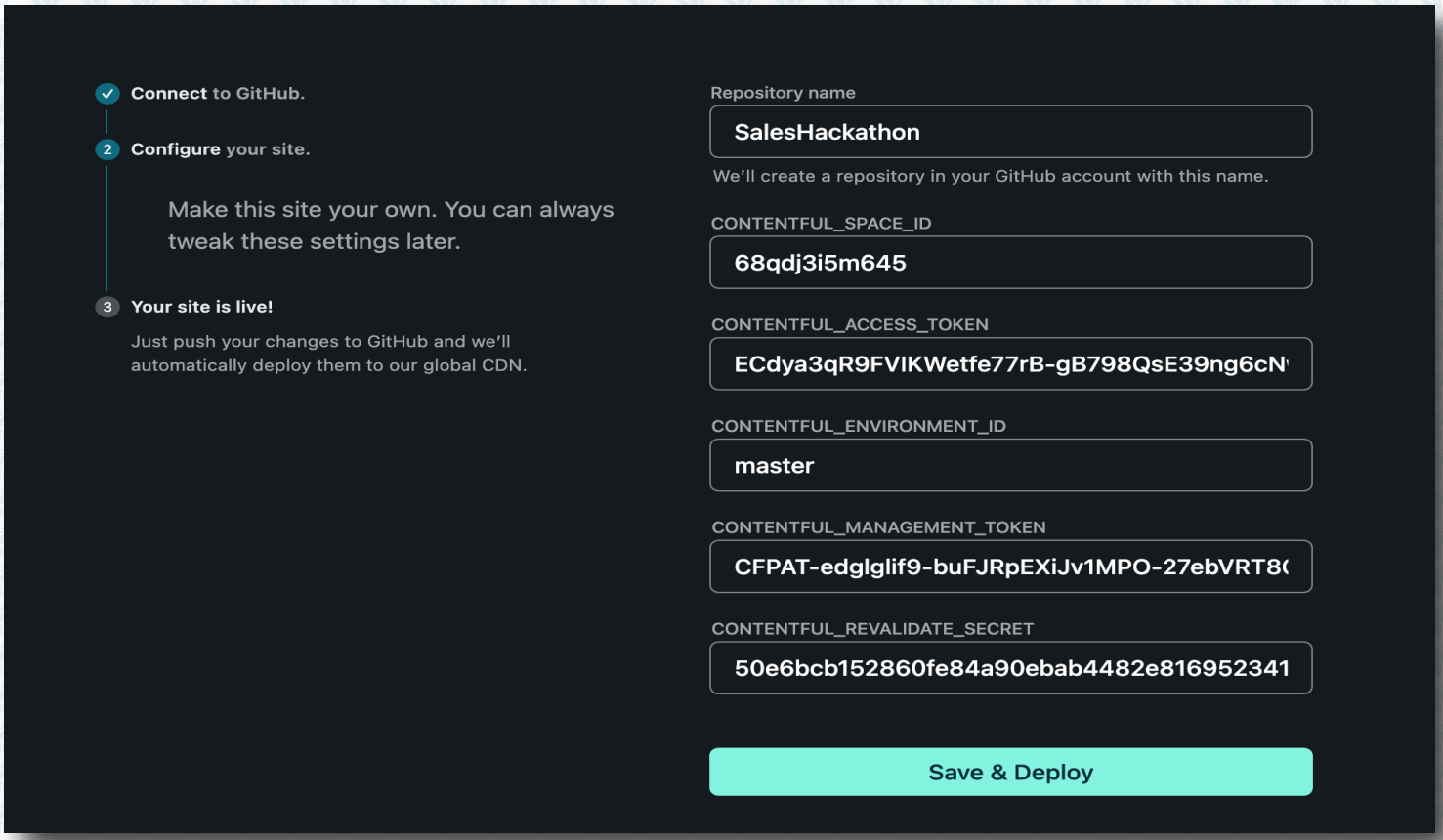
Once authorized, go ahead and close the small popup window.



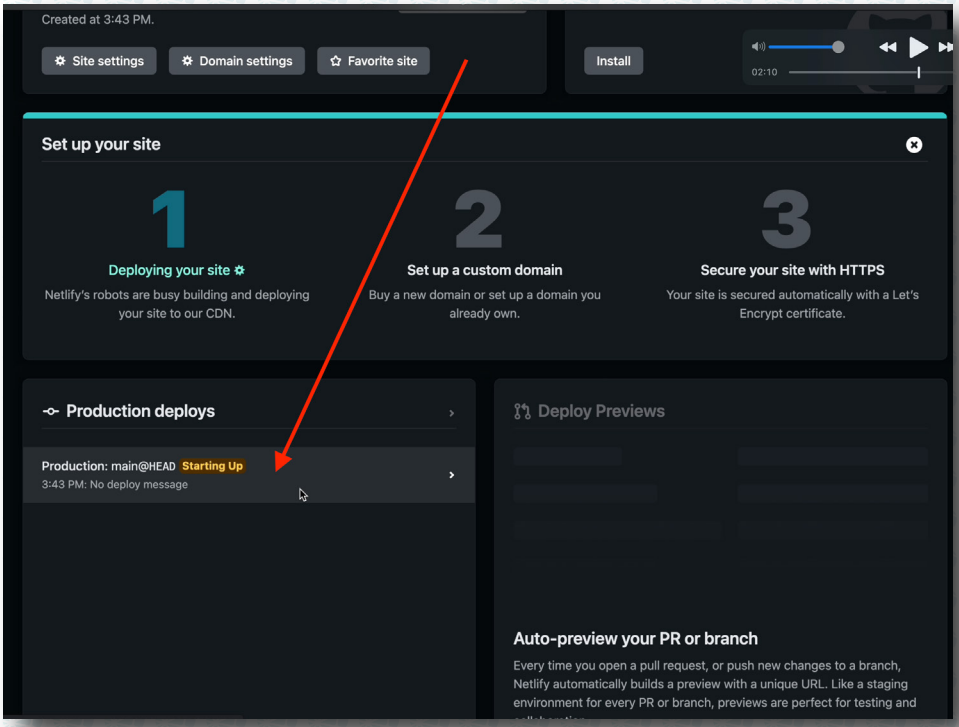
And once again, click on Connect to GitHub again.



Next, paste in the values you collected in your spreadsheet from Chapter 1, into their respective fields. Then click Save & Deploy.



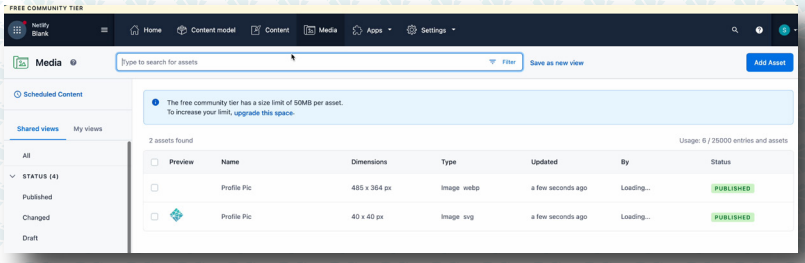
Let's click in, and watch the build happen in real time! If you were a developer, this is where you would read the logs and try to find certain events that help you debug your template - Luckily, we've removed all the bugs for you, but it's nice to take a peek in there and see it working.

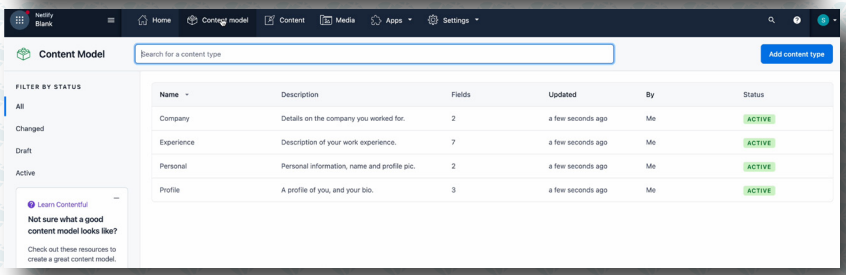


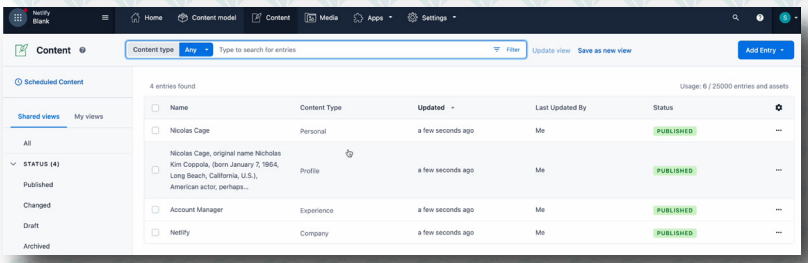
When we've reached line 91 on the build, you can see that it's detected that you don't have the content needed for this template, so we import it for you using our fancy Build Plugin, that we built just for this project - It's a little bit too advanced to go into this, but it's worth understanding the power of this feature when speaking to clients. You can solve almost any problem on Netlify!

```
3:44:29 PM: ────────────────────────────────────────────────────────────────
3:44:29 PM:   2. /plugins/contentful-setup (onPreBuild event)
3:44:29 PM: ────────────────────────────────────────────────────────────────
3:44:29 PM:
3:44:30 PM: We've could not detect the content models needed for this template - We're loading in the dummy
content
3:44:30 PM: ┌──────────────────────────────────────────────────────────────┐
3:44:30 PM: │ The following entities are going to be imported:             │
3:44:30 PM: ├──────────────────────────────────────────────────────────────┤
3:44:30 PM: │ Content Types                                          │ 4   │
3:44:30 PM: ├────────────────────────────────────────────────────────┼─────┤
3:44:30 PM: │ Editor Interfaces                                      │ 4   │
3:44:30 PM: ├────────────────────────────────────────────────────────┼─────┤
3:44:30 PM: │ Entries                                                │ 4   │
3:44:30 PM: ├────────────────────────────────────────────────────────┼─────┤
3:44:30 PM: │ Assets                                                 │ 2   │
3:44:30 PM: ├────────────────────────────────────────────────────────┼─────┤
3:44:30 PM: │ Locales                                                │ 1   │
3:44:30 PM: ├────────────────────────────────────────────────────────┼─────┤
3:44:30 PM: │ Tags                                                   │ 0   │
```

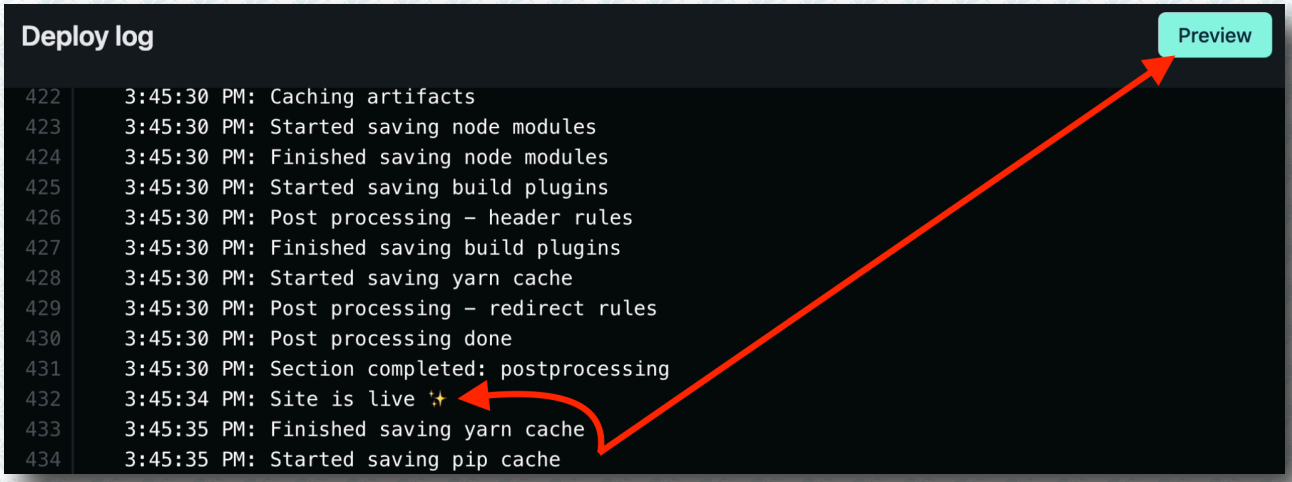
Before it's done building, you can go over to Contentful and see the content we've imported under the Media, Content, and Content Model Tabs:

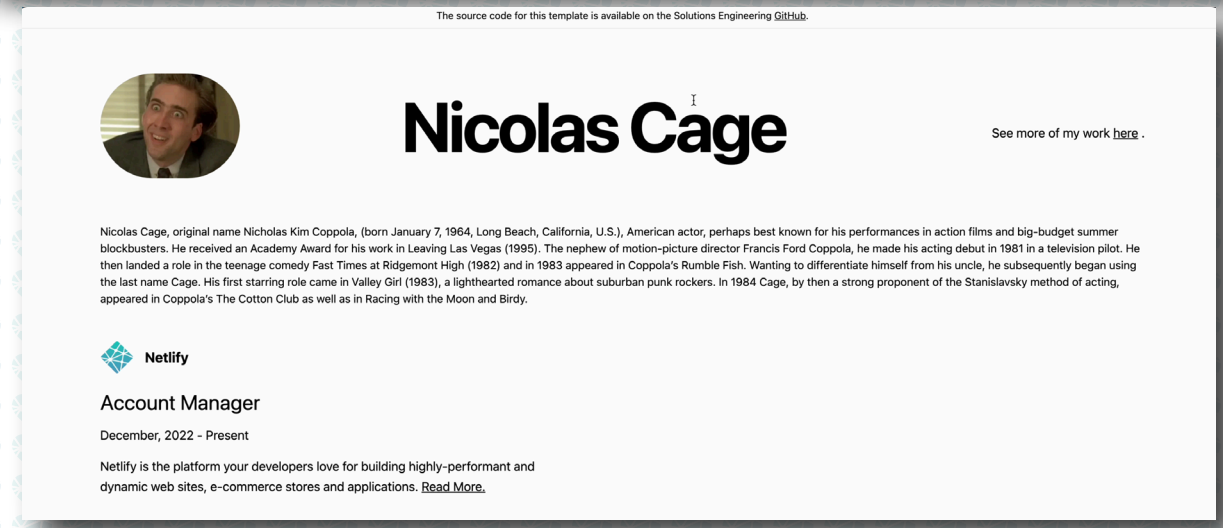






Finally, back over on Netlify when you see *Site is live* and the *Preview* button, you can click on the preview to reveal your shiny new site!





Great! You've successfully deployed a template to Netlify and taken your first glimpse into the world of composable architecture, you should be very proud! You took some code from GitHub, you connected your Contentful account using API keys and you mixed it all together using Netlify and deployed it for the world to see - Well done!

But there's a problem. Unless you're Nicolas Cage, the content does not fit with you or your experience. We'll cover how to update that in Chapter 3.

If by chance you are Nicolas Cage, then hello Mr Cage. Welcome to Netlify and thanks for taking part in the Hackathon, I loved you in Con Air.

CHAPTER 3

CONTENTFUL FEATURES

We're now at the stage in a developer's journey where they need to start thinking about working with other stakeholders outside of the development team. We will of course be playing both the developer and content creator role in this Hackathon, but to fully encompass the benefits of decoupled architecture/the Jamstack we need a way to notify Netlify when new content has been published.

Why is it so important?

In the days before decoupled architecture, like the one we are using, many teams had to coordinate to release content to the world. If I was a content creator, I would likely be restricted to a release schedule that is controlled tightly by the site reliability engineer, or dev ops teams. This is because they would need to rebuild the entire site themselves, each time I wanted to release new content, and they would have to clear the cache* on the CDN* to ensure that our users are getting the latest content. Using the decoupled approach, users of a modern CMS like contentful can simply hit publish via the UI and behind the scenes, Netlify will rebuild your site with the latest content, clear the cache, and push out to the CDN. When we tell our customers we save them a day a week in development time, it's because of features like this.

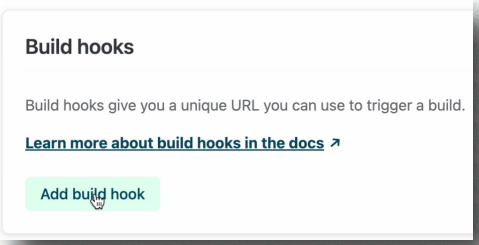
Let's get that going!

SETTING UP YOUR BUILD HOOKS

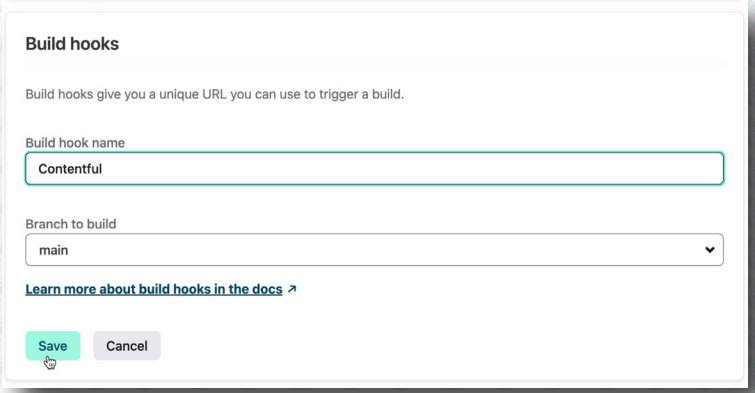
Head over to the deploy section setting section of your site:

https://app.netlify.com/sites/**YOUR-SITE-NAME-HERE**/settings/deploys

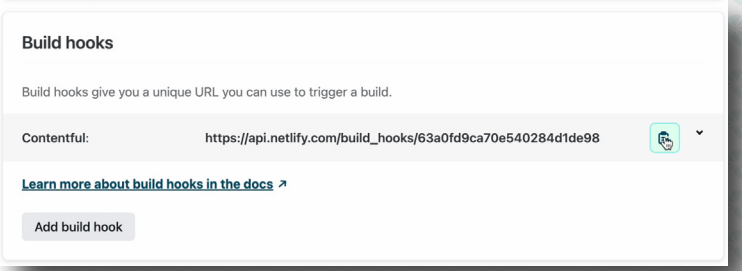
Scroll down until you see Build Hooks*, then select *Add build hook*.



Give your build hook a name, and click save.

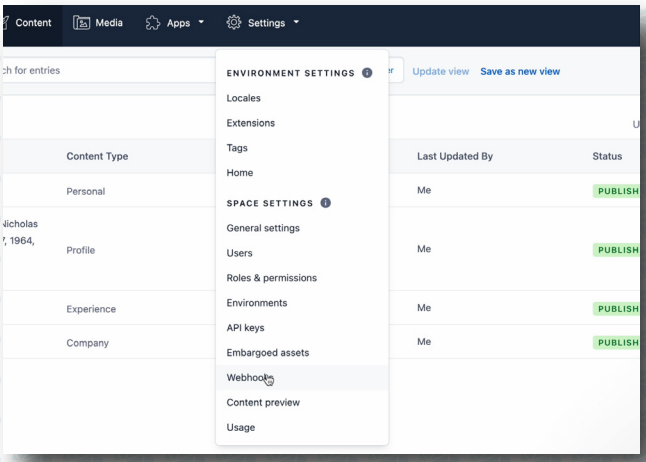


Copy the URL that is generated using the icon.

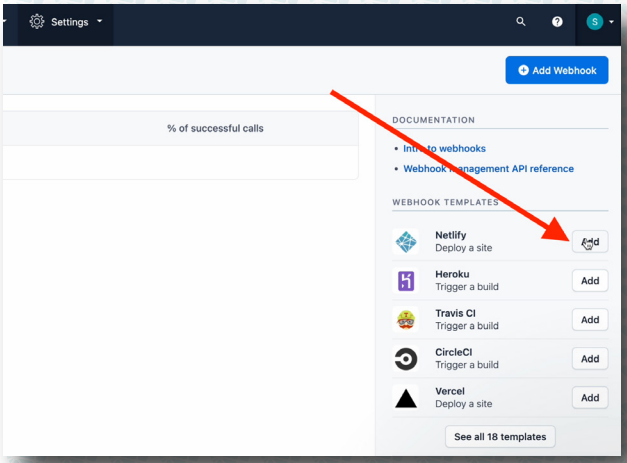


Great! You've just generated your first build hook! Put very simply, these are little subscription services. You paste them into the Webhooks* sections of other services, like Contentful so you can subscribe to a particular event - And when that event happens, your site starts to build again! Make sure to check out the glossary section where we cover Webhooks in more detail - they are a vital piece in composable architecture/the Jamstack!

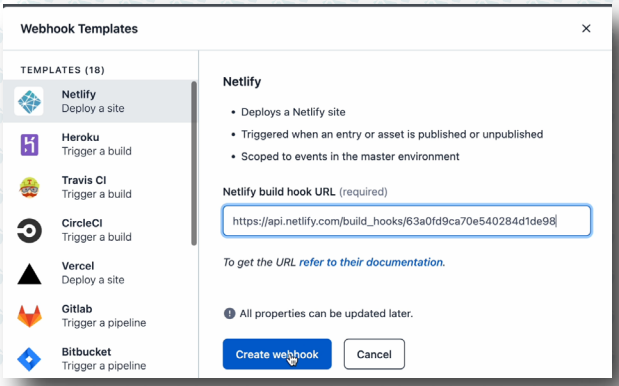
Next, head over to your Contentful page and go to Settings -> Webhooks.



Thanks to our wonderful partnerships team, Contentful has a Webhook integration specifically for us! Click on the add button beside Netlify on the right-hand side of the page.



Paste the URL you copied from the previous step into the *Netlify build hook URL* and click *Create webhook*

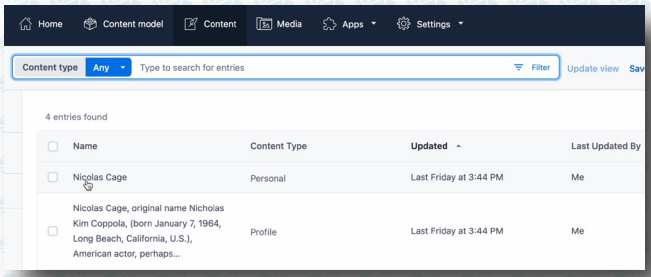


Let's take this build hook for a test drive, shall we!?

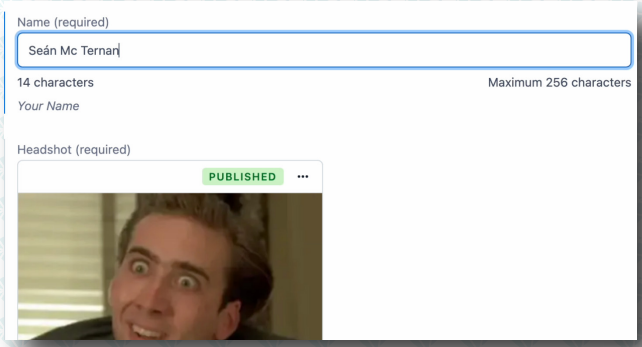
UPDATING YOUR CONTENT

EDITING EXISTING CONTENT

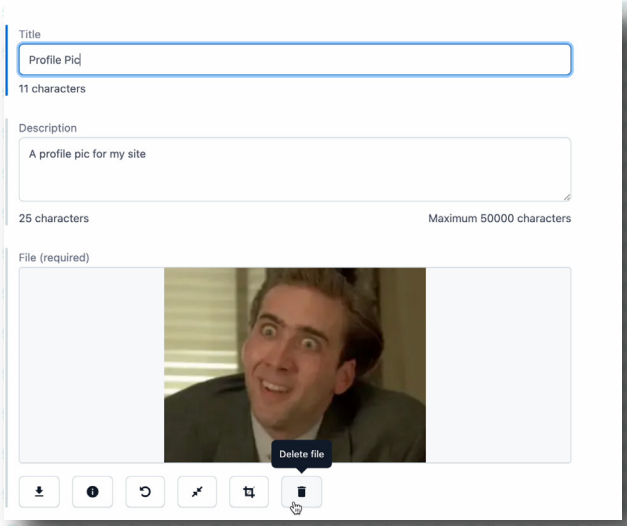
First, we're going to update the dummy content we added in when we deployed the template to Netlify. Click back into your Content tab, then click on the first item, it should be Nicolas Cage with the Content Type Personal.



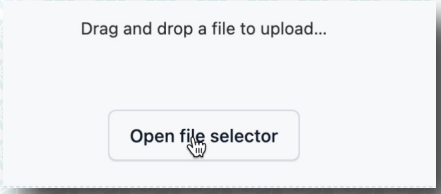
Enter your name, then click on the *Headshot* image



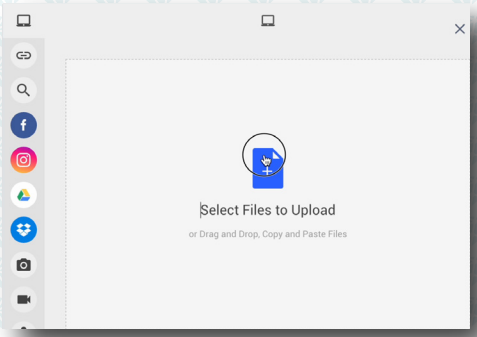
In the new window, select the delete button



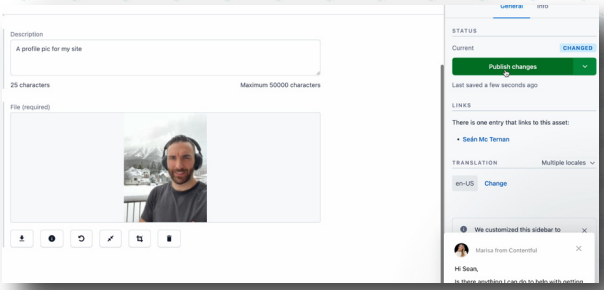
Then choose, the file selector steps



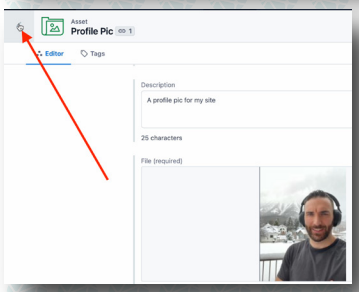
Select a file headshot from your computer to upload the new image. Any image will do, but if you have an image available from your Slack/LinkedIn you can save that image to your desktop and upload it here.



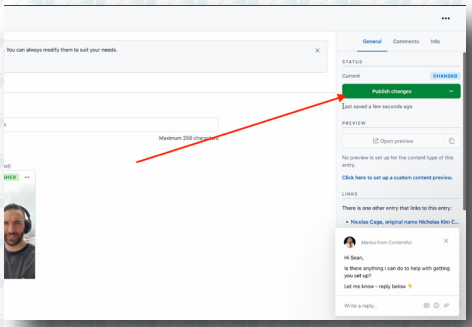
Once that is updated, click Publish changes in the top right corner. **Feel free to ignore any pop up asking if you want to explore the API!**



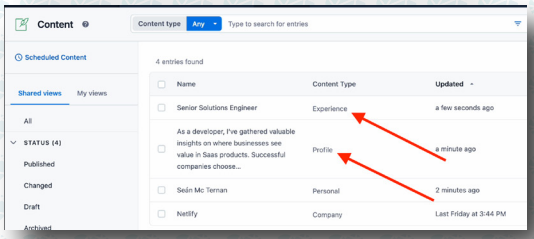
Click on the back arrow in the top left corner to *Personal* section



Then click *Publish changes*

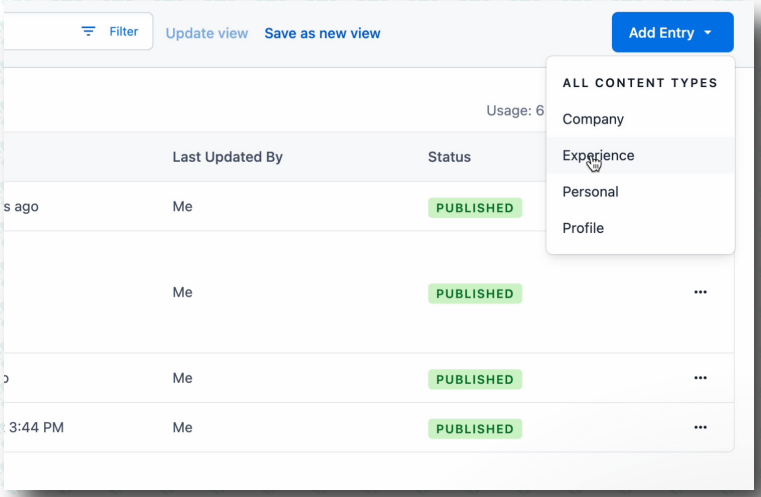


Next, click into the Content tabs, repeat the same steps you had taken previously to update your experience and profile information



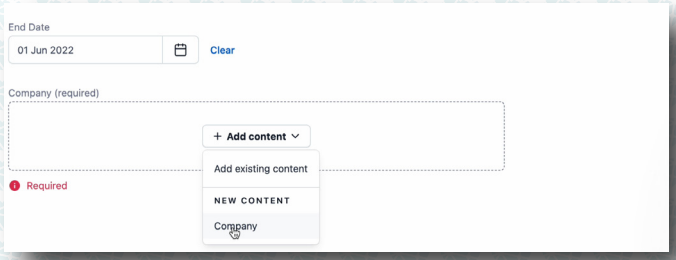
ADDING NEW CONTENT

Next, we're going to start adding new experiences and company information to your site. Click on Add Entry in the top right side of the page, then click on Experience. Use this to add entries relating to your experience before your current role at Netlify. Add the details of your experience



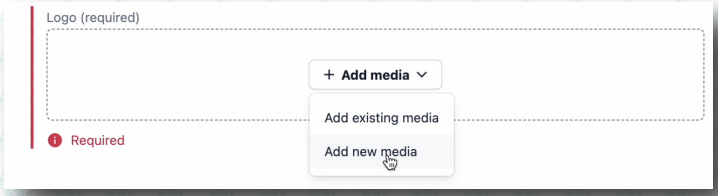
Note that the slug will be the corresponding address for each webpage that is generated on your site, it must be unique for each of your experiences. It will automatically generate based on the Job Title so if it matched a previous entry just add some letters or numbers at the end to make it unique.

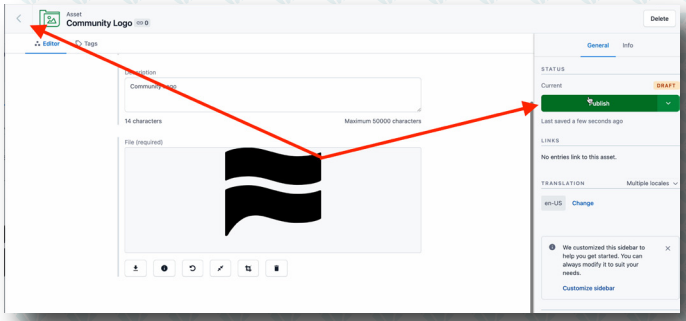
When you reach the company section, you can add an existing company (for example, Netlify is already in your account) or select a new company to add. Below we added a new company

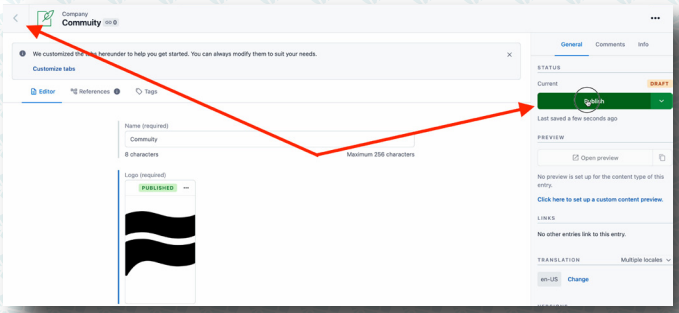


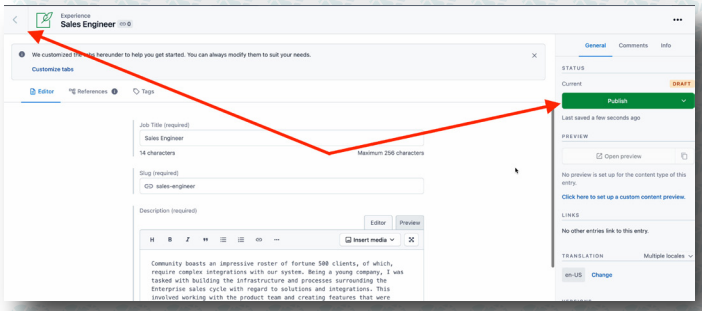
Tip:** If you need to grab a company logo from a previous company, you can do so by finding the image online and doing a screen capture of it on your mac by selecting **Cmd + Shift + 5.

Follow the steps to add new media by giving it the details needed and clicking publish then back once completed.

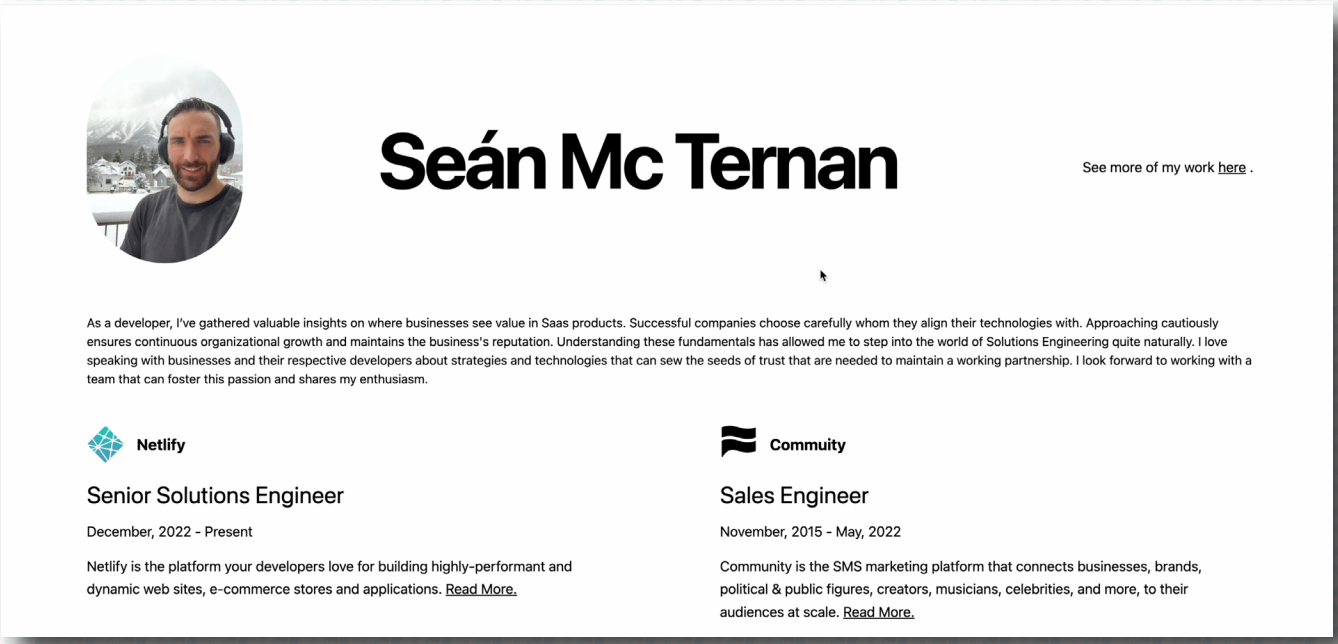








Phew! That was a lot of steps, we know! You just got a glimpse into the life of a content creator, however, because we set up the build hooks and we are using a modern Jamstack approach, your changes have been picked up by Netlify via the build hook! Head on over to your Netlify account and open up your site, you should see the changes that we created, deployed and are viewable to the world via your site!



If you click on the company logos or the jobs titles, you can see it will open an entirely new page that was generated with the content you created - Pretty neat!

Well, you set up a real-life webhook, lived the life of a Contentful user, and saw how Netlify was able to automatically re-build your website without any additional effort after you made changes in Contentful. **Pretty cool**! Head over to your assignment workbook and reflect on your work using Contentful and how it fits into the general workflow for building and maintaining sites on Netlify.

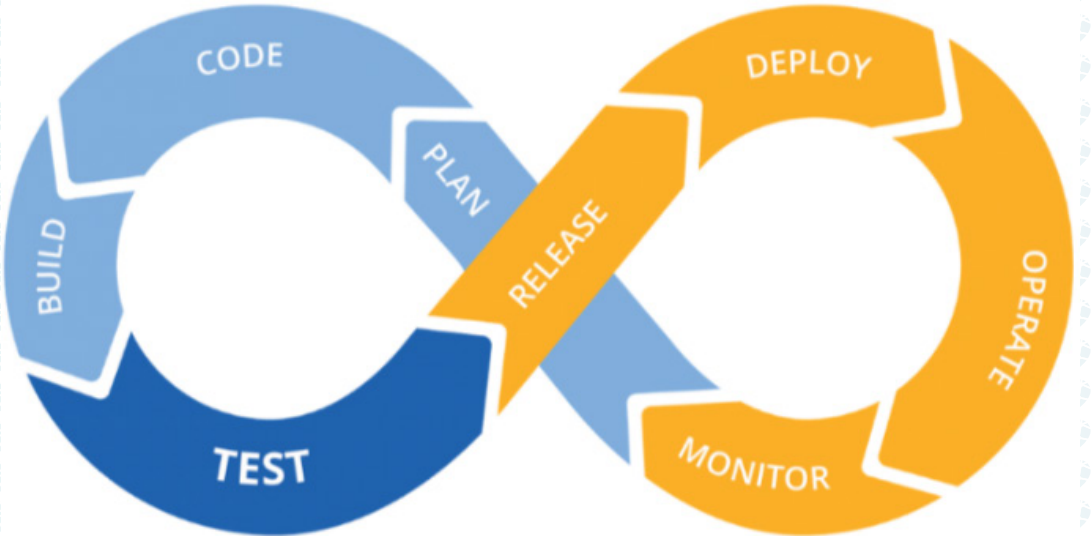
CHAPTER 4

NETLIFY FEATURES

Why the Netlify platform is so valuable to developers is the concept of Continuous Integration/ Continuous Development, or more commonly known as CI/CD. CI/CD is a software engineering concept that aims to shorten the software process from code to pushing it out for all the world to see.

It can be thought of as ongoing actions to write code >> go to staging >> go to QA >> have the senior devs rip into your code >> go home, cry and reevaluate your life >> comeback the next day >> make your changes >> push code to staging >> push to QA >> Senior checks >> Go to production. Start all over again!

As you all know, we SE's of Netlify love to show visuals, so here is the visual of the CI/CD process:



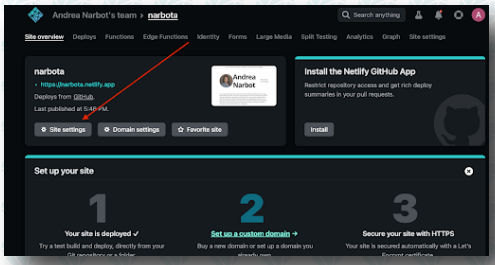
Now with all of this said, you can imagine that configuring this on your own or building different tools to accommodate this can get complicated!

Netlify is here to take the complexity out of this process by automating it and putting it on one platform and incorporating tools to make life easier for developers and DevOps!

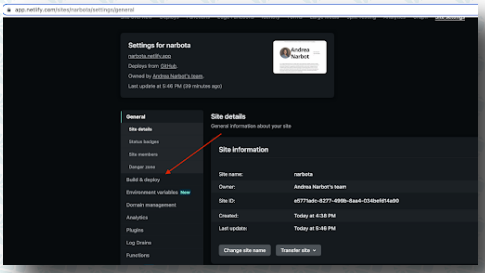
BRANCH DEPLOYS

The Branch Deploys feature gives you and your development team so much flexibility. You no longer have to worry about setting up additional hosting environments* for staging, testing, QA, production, etc. You can spin up unlimited environments within Netlify's build and hosting infrastructure just by creating a branch in your code repository and configuring it in Netlify!

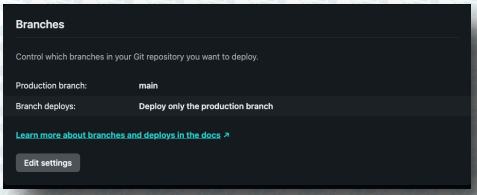
To do this, go to your sites in the list of sites available (maybe only one site - the one you just built at this point). Go to the 'Site Settings' button and click it.



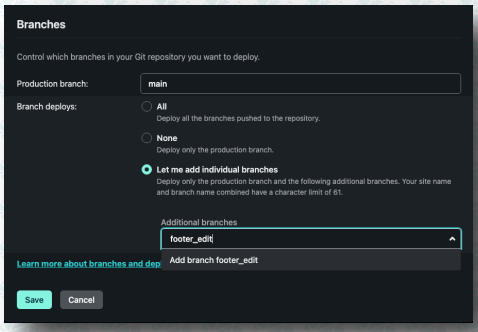
You will see the general settings page for your site. Click on '*Build and Deploy*'.



Once you have clicked '*Build & Deploy*', scroll down on the right hand side until you see 'Branches'. This is where you will enable Branch deploys. Click the "edit settings" button to edit your branch settings.



- Select 'Let me add individual branches'
- type in **footer_edit** then
- click *Add branch footer_edit* followed by *Save*

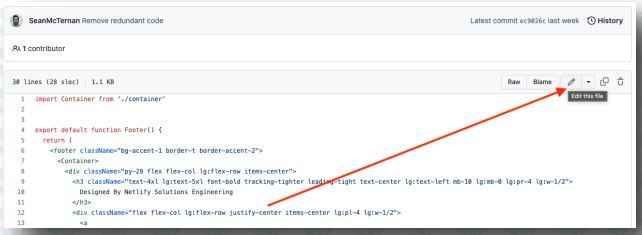


Next, head on over to GitHub to edit the footer of your site. You can get there by changing the URL below to include the name of your own GitHub Account.

https://github.com/**YOUR-GITHUB-ACCOUNT**/SalesHackathon/blob/main/components/footer.js

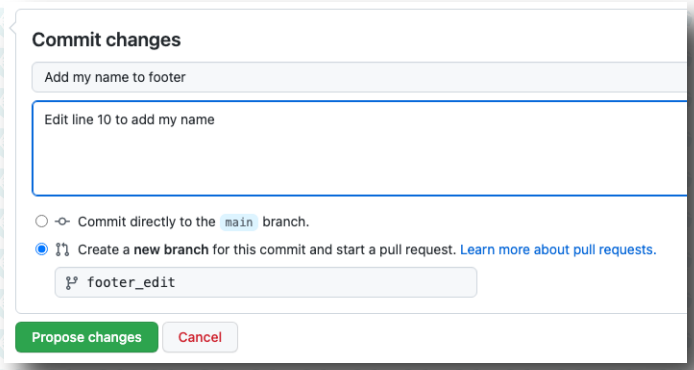
If you have difficulty finding that page, you can simply open your repo in GitHub, click on the *components* folder, then click on *footer.js*

Once you've arrived at the right place, click on the *edit this file* button:

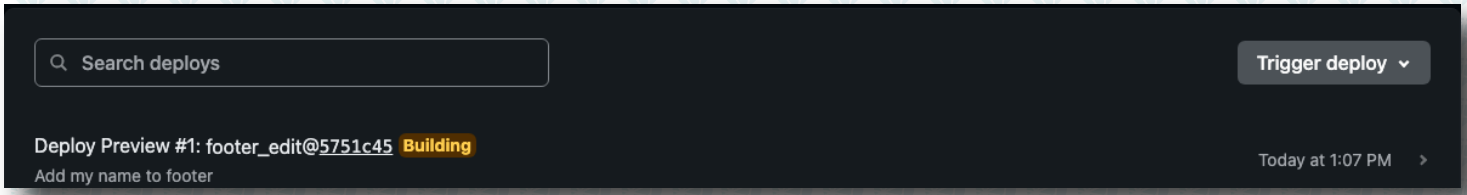


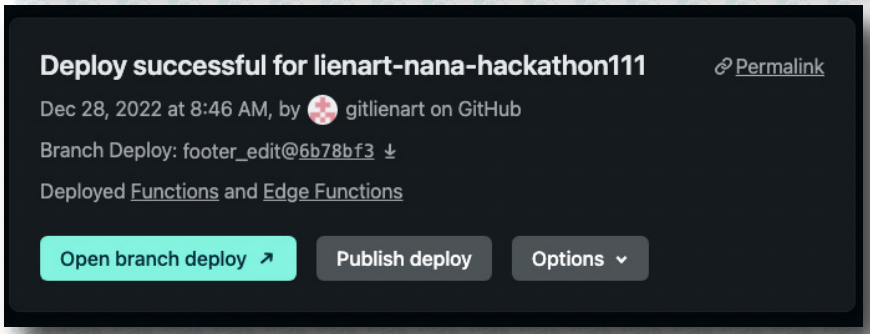
On line 10, change the name after *Designed By* to your own name

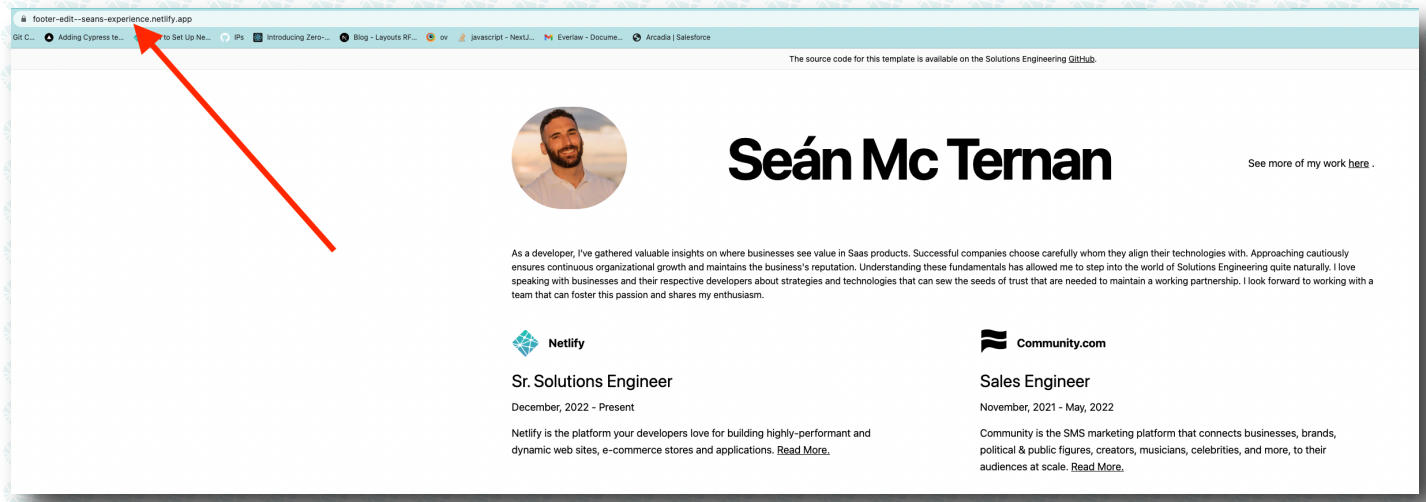
Scroll down, then select Create a new branch for this commit and start a pull request. Make sure you name the branch the same name you entered over in Netlify i.e. **footer_edit**. Click *propose changes*.



Head over to Netlify **https://app.netlify.com/teams/YOUR_TEAM_NAME/sites** and you will see your Branch Deploy being build! Click into the build and select *Open branch deploy* when the build is complete.







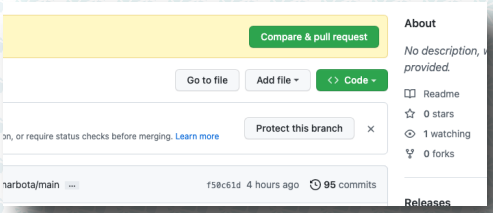
Once you hit the branch deploy button, You can see the unique deploy url that is generated when you create a branch deploy. It's all a part of the continuous deployment feature that Netlify offers!

Head over to your assignment workbook and reflect on what you think this feature means to our web developer customers and their workflow.

COLLABORATIVE DEPLOY PREVIEWS

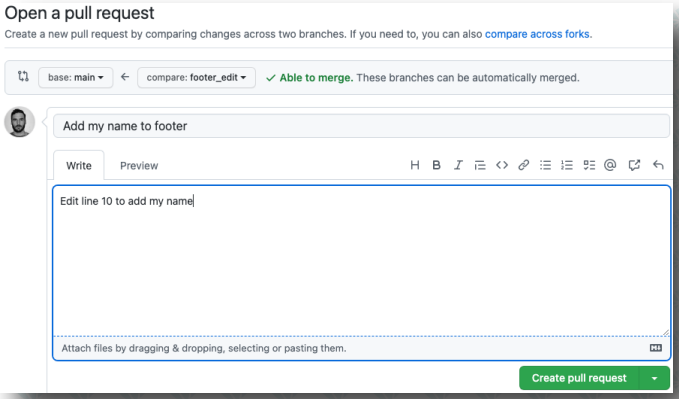
Get instant feedback with Collaborative Deploy Previews (CDP)! Developer teams can give feedback, review projects using most of the popular productivity tools. With this feature you no longer have to wait until a developer makes the change in the project, but all collaborators and stakeholders can make changes with deploy previews from pull or merge requests.

To see this in action, go back to GitHub, you should see a pull request on your main page https://github.com/**YOUR ACCOOUNT**/SalesHackathon - *Click on Compare & pull request*.

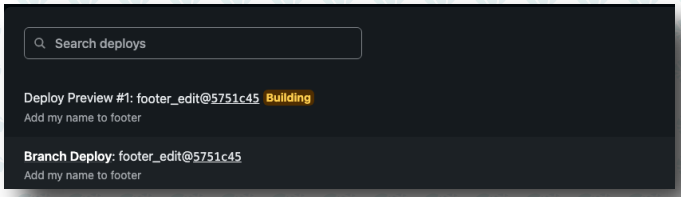


In the comment box, enter a comment and select '*Create Pull Request*'

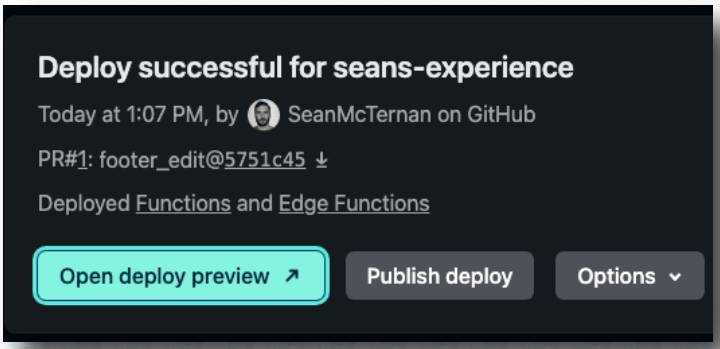
Note: *These comments would be an area where non-technical and technical people alike can leave feedback on proposed changes to the site. In our case, we just made a small change to the footer and you were your own boss, but in larger teams, these comments are a vital piece within the collaboration it takes to make a site come together.*



Go over to Netlify under your site https://app.netlify.com/sites/**YOUR-SITE**/deploys You should see a deploy preview being built!



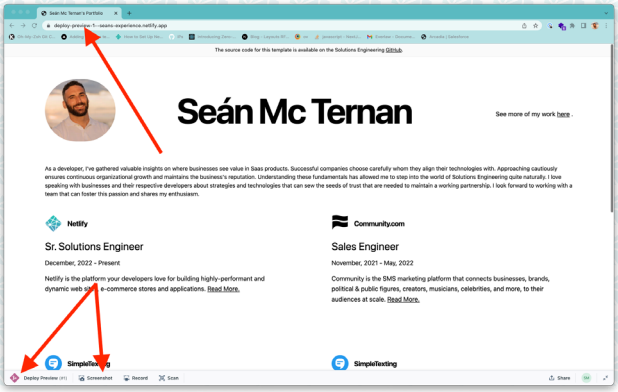
When it's done building, you can click on the build itself to expand the window. Then click on the Go To Deploy Details button. From here, click the Open deploy preview button and your site will open in Deploy Preview mode.



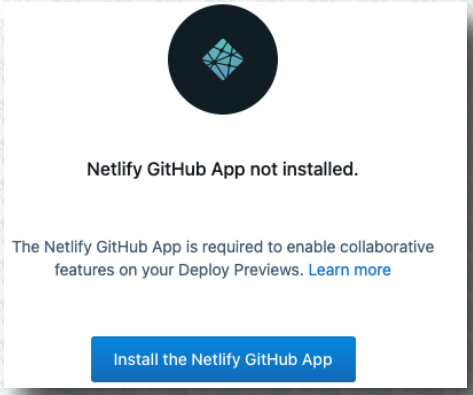
Notice the unique URL and the menu at the bottom of the page. These are unique features to Netlify!

You will see your page and the deploy preview URL and in the lower left-hand corner, you will see the link to begin collaborating.

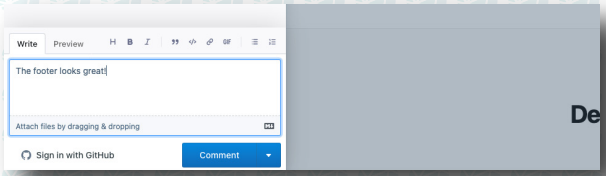
To Collaborate, click on the ‘deploy preview’ link in the bottom left corner.



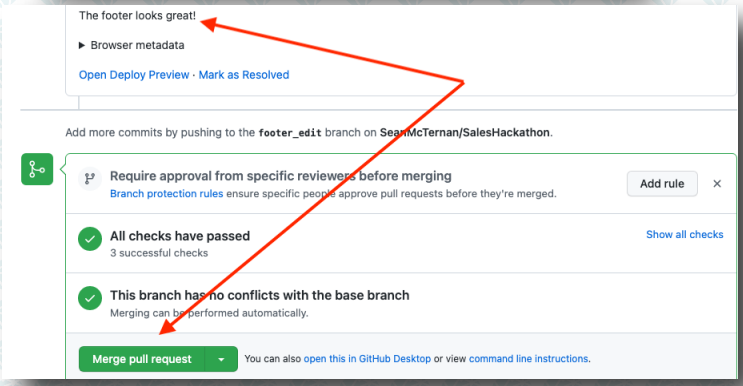
Note: If you see ‘Log in to Netlify’ instead of ‘deploy preview’ you may need to login/install the Netlify GitHub app. Follow instructions to do so by following the prompts.



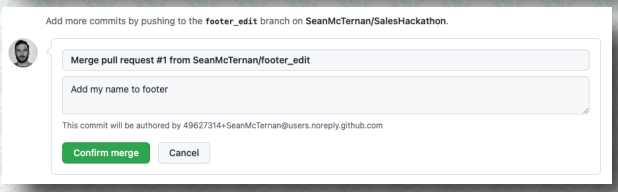
Once you’re logged in on the deploy preview window - Write a comment - Click *Comment*



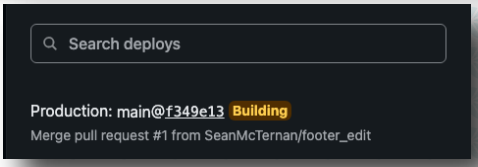
Jump back over to GitHub - The url should be https://github.com/**YOUR-ACCOUNT**/SalesHackathon/pull/1 and see your comment. You can merge the request by hitting Merge pull request.



Confirm the merge by clicking *Confirm merge*



Your main site (not your branch deploy) should now start building again over on Netlify! CI/CD at work once again!



NICE WORK! By making a new branch, opening up a pull request, and merging a change, you've just stepped into the shoes of a developer working in a CI/CD workflow! Most importantly, your site was live the entire time, while you made code base changes behind the scenes - This is why Netlify is so powerful for teams, with very little effort users never lose access to your site while you continue to make improvements; fast, often and with ease.

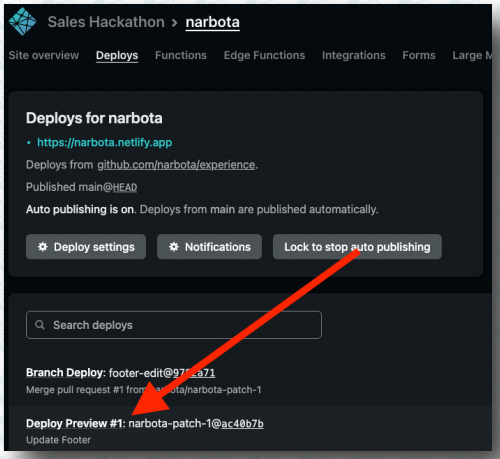
Head over to your assignment workbook and reflect on what you think this feature means to our web developer customers and their workflow.

ROLLBACKS

Rollbacks, also known as the *Save my A*** feature (Ok, it's not, but that is what I call it) are a very powerful tool in the tool belt of a developer. Say, you shipped a change to the footer in the steps above that broke it entirely, and it was a worst case scenario - Meaning the customer can see it. As a developer you're getting heat from all sides, your boss, your boss's boss, the customer - what do you do!? Go back in time to when it wasn't broken of course! Because we store every version of your site on Netlify, we can simply publish a version that was working in just a few clicks. That means you can fix whatever you broke behind the scenes without leaving the bad version in front of the customer - Let's get your Delorean up to 88 MPH and go back in time to one of your previous deploys, shall we?

Go to your list of deploys as shown below:
https://app.netlify.com/sites/YOUR-SITE/deploys

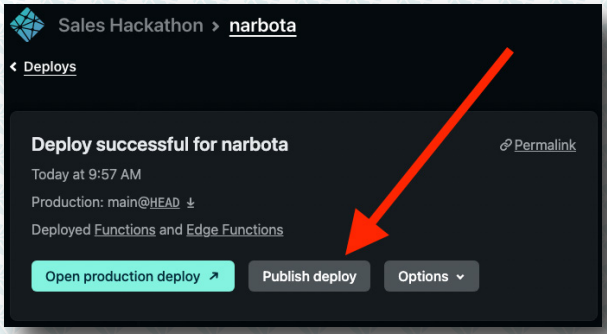
Select one of your Deploy Previews. You may need to scroll down until you see one whose title starts with *Deploy Preview #1* Click it.



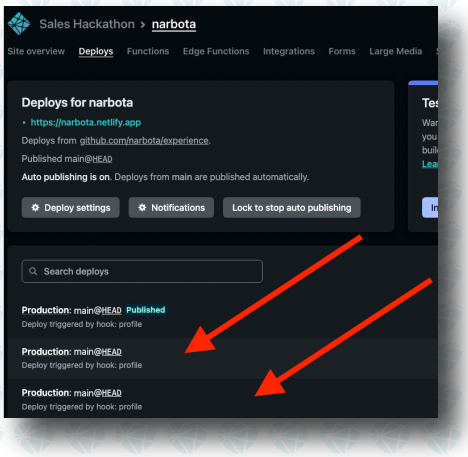
Once you are on that page, you will see a button that says **Publish Deploy**. Click on that button to publish this change.

Oh No! We just published we just sent something to prouction that does not belong there, customers are going to see, we have to unpublish it!

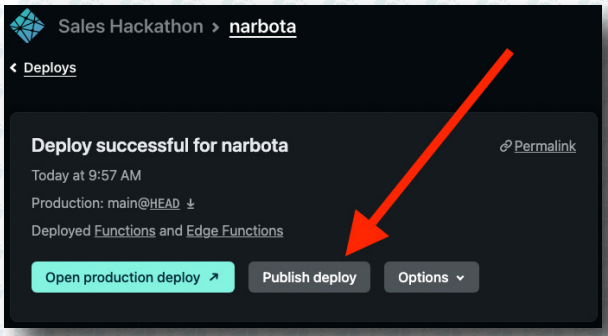
Note: *To keep things simple we published bad changes on purpose via the user interface. In reality, these mistakes could be pushed via code updates from GitHub by developers making changes to the code*



Let’s see how easy it is to go back to a previous deploy and publish this instead! You see how the ‘deploy preview’ says ‘published’. Well, let’s go ahead and publish the production version instead. Click on a previous deploy, one that says ‘**Production**’, preferably.



Click on the ‘*Publish Deploy*’ button; confirm it and voila! No more silly messages!

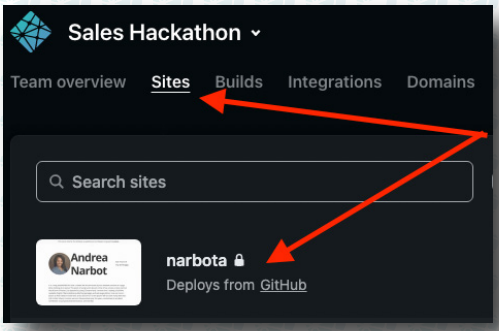


It’s easy peasy lemon squeezy to rollback any published changes!

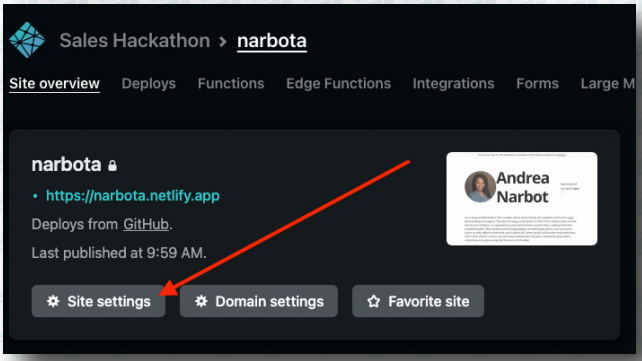
SITE TRANSFER

Now it's time to show your awesome work to the rest of the team and transfer your **SHINY** new site

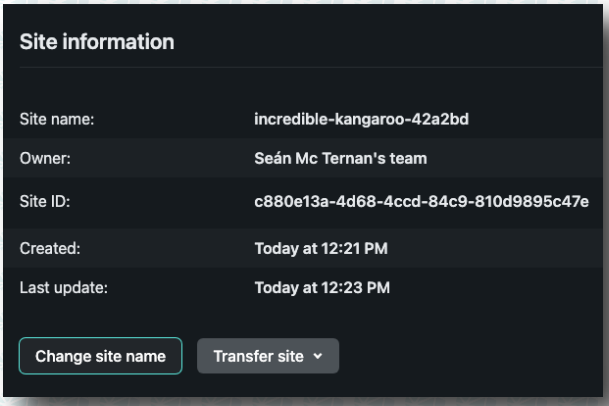
Click on *Sites* then click on your site

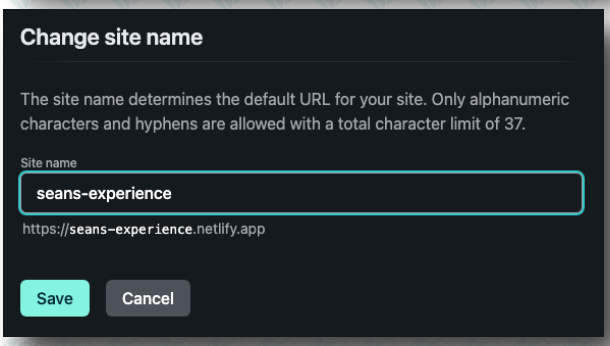


Then select '*Site settings*'



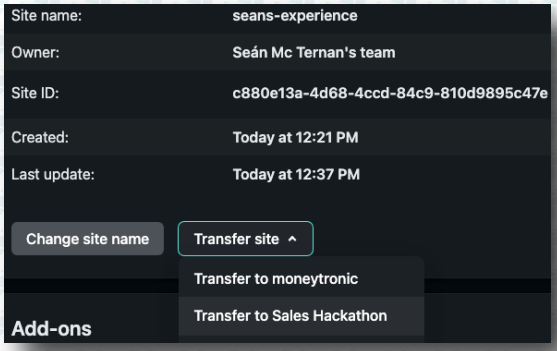
Select *Site information* then Select - *Change Site Name.* Enter your name, or a site name related to you, then choose *Save*

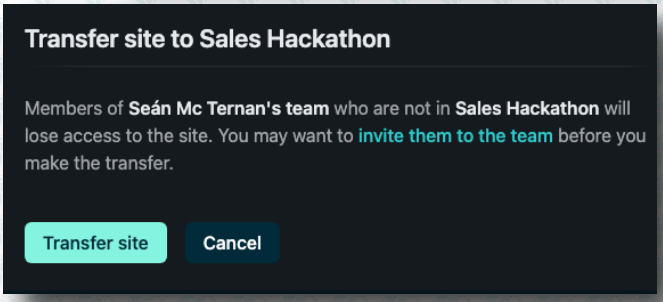




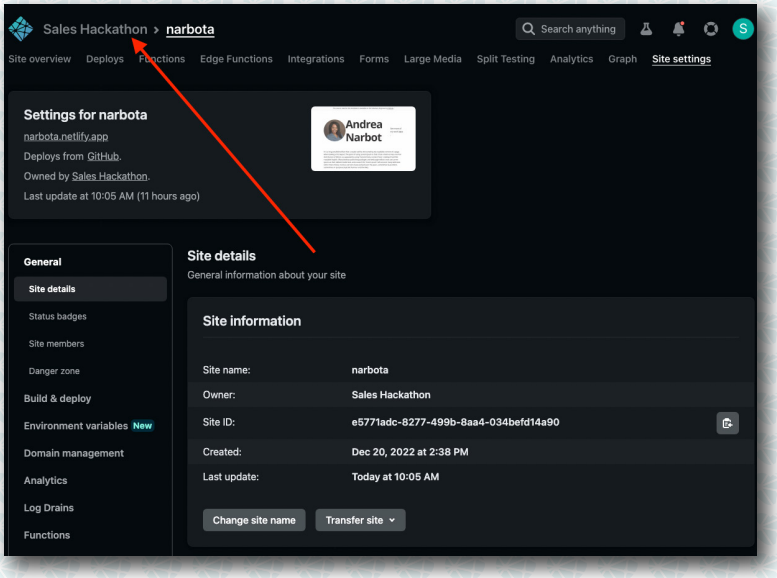
Next click on the drop down *Transfer Site* and select *Transfer to Sales Hackathon* then click *Transfer Site*

Note: *If you do not see the this option, you may not have accepted the invitation to join the team. Check your email and be sure to accept the invitation to the team. If you did not see an invitation, please reach out to Andrea Narbot*





You should then see that you are now in the Sales Hackathon team!



Head over to your assignment workbook and reflect on what you learned and what you took away by going through this chapter.

CHAPTER 5

BONUS CHALLENGES

For all you overachievers who completed this challenge with ease and would like extra brownie points. We have put these bonus challenges to help you dig more into the Netify platform and the features that aid in developer productivity. Up to this point we've been guiding you through the steps, however, for the bonus section we're going to put you in the shoes of a real developer and utilize the documentation on Netlify!

INSTALL LIGHTHOUSE PLUGIN

Lighthouse is a tool that analyzes web pages and generates a report on the performance of the page. The report includes a score for various performance-related metrics, such as the time it takes for the page to load, the size of the page, and the use of best practices. These scores can be useful for anyone who is interested in understanding how well a web page is performing and identifying potential areas for improvement.

One reason lighthouse scores are useful is that they provide a clear, objective measure of a web page's performance. Rather than relying on subjective assessments or personal experience, lighthouse scores provide a quantitative way to compare the performance of different pages.

Another reason lighthouse scores are useful is that they can help identify specific areas where a web page is performing poorly. The lighthouse report provides detailed information about the various performance metrics that are being measured, as well as specific recommendations for how to improve those scores. This can be valuable for non-technical users who want to understand what is causing a web page to perform poorly and how they can fix it.

Using the plugin on Netlify, we can provide instant feedback to developers regarding the changes they've made to the site. We can even fail builds on purpose if their scores are too bad, so the public won't see any sites with bad scores. Go ahead and install this plugin and trigger a build via Contentful to see it in action.

https://www.netlify.com/blog/2021/03/26/netlify-build-plugin-of-the-week-lighthouse/

Why stop at Lighthouse? Using Netlify's vast ecosystem of Integrations, why not add Algolia Search, or Sign Up for New Relic and get detailed analytics about your site?

Turn on Analytics

Web analytics are important because they provide insights into how users are interacting with a website. By collecting data on things like page views, traffic sources and unique visitors. Web analytics tools can help website owners understand how people are using their site and identify areas for improvement.

One of the main benefits of web analytics is that they allow website owners to make informed decisions about their website based on data, rather than assumptions. The website owner might conclude that the page is not meeting the needs of its users and take steps to improve it.

Web analytics can also help website owners understand the effectiveness of their marketing efforts. By tracking the sources of traffic to a website, web analytics can help website owners understand which marketing channels are driving the most traffic and conversions. This information can be used to optimize marketing efforts and allocate resources more effectively.

Go ahead and turn on Analytics for your site:

https://www.netlify.com/products/analytics/

Password Protection

Sometimes you want to keep things on the hush-hush before showing them to the rest of the team, or your customers! Go ahead and add some password protection to your deploy previews, or branches:

https://docs.netlify.com/visitor-access/password-protection/

Subdomains / HP Edge:

We've gone ahead and enabled HP Edge on the teams account, so there is no need to configure that on your end. Getting added on to the HP Edge on an account that is enabled with it, is as simple as creating a Subdomain from the main domain when it's registered with Netlify DNS! Subdomains are used in web development to create separate areas of a website with their own unique content and functionality. They allow for better organization and can also be used to target specific groups of users or to test new features. For our purposes, we're going to use it to identify all the hardwork you've put in in this Hackathon, so make sure you to name your subdomain after yourself. For example https://sean-mct.sales-hackathon.com. You can enable a subdomain on your main site, using the docs below.

https://docs.netlify.com/domains-https/netlify-dns/delegate-a-subdomain-to-netlify-dns/#delegate-a-subdomain-to-netlify-dns-through-site-settings

Head over to your assignment workbook and reflect on what you learned and what you took away by going through this chapter.

CONCLUSIONS

Congratulations! You've completed the technical portions of this hackathon, and that is no small feat. Learning new things is always challenging, however, when this is coupled with a highly technical industry it complicates things, to say the least. Therefore you should be very proud of what you've achieved so far. A quick recap:

- You signed up for new developer productivity tools
- Deployed a site to the world
- Connected that site using webhooks to Netlify
- Created different staging environments in the form of branch deploys
- Edited code through GitHub
- Collaborated with team members via Collaborative Deploy Previews
- Last but not least, you utilized one of the most prized features on Netlify - Atomic Deployment - In the form of Rollbacks!

That's a heck of a lot of progress to make in a single Hackathon.

But let's add a little context here. Let's imagine you had to do all of this without Netlify. In order to do the same on AWS for example, you would need two different pipelines to push code and content changes to the world. The reason being, each pipeline has its own checks and balances that need to be met before they finally merge when the site is built. This would involve many team members who have to build and maintain those pipelines so that the changes you make do not break other parts of the site. It doesn't stop there. Another team would take over at the end of those pipelines to distribute that content to the CDN so that the world can see it. They would have to scale their servers up and down in different parts of the world depending on how much traffic their site gets, balancing the demand on their site against a budget they've been allocated. They also would need to watch out for bad actors trying to take the site down and fight against them using another suite of tools. If this sounds like a juggling act, well - It is! All of these parts need to move together so as to not break the machine. And when that unfortunate time comes, you need to get all hands on deck to figure out what went wrong and where! The result for many teams is that they have to schedule their releases of content and code on a weekly/monthly cycle slowing down their productivity considerably. By comparison, through this project you (with no offense intended) a non-technical person could achieve everything we outline above in our AWS example by yourself in just a few hours. That is the power of Netlify. We take all of that headache away from the customer so that their engineers can push code, and the content writers can get their content out immediately without the need to wait for one another. Most importantly they can do this multiple times per day, without having to wait for a release schedule. That is **HUGE**!

Finally, thank you so much for taking part in this Hackathon, we hope you had as much fun participating as we did building this for you!

Until next time, hackers - Peace ✌️

Solutions Engineering Team

GLOSSARY

API Keys
An API key is a unique identifier that is used to authenticate requests to an API (Application Programming Interface). It is a way for the API to identify the client that is making the request and to determine if the client has permission to access the data or functionality being requested. An API key is typically a long, randomly generated string of characters that is provided by the API provider. When a client makes a request to the API, the client includes the API key in the request header. The API then checks the key to determine if the client is authorized to make the request.

Build Hooks
Build hooks are URLs you can use to trigger new builds and deploys. You can read more about them here.

Cache
Caching is the process of storing data in a temporary location so that it can be accessed more quickly and efficiently in the future. One of the primary functions of a CDN is to cache content in order to improve the performance and speed of content delivery. When a user requests a piece of content from a website, the CDN will check to see if it has a copy of the content stored in one of its cache servers. If it does, it will serve the cached copy to the user, which is faster than retrieving the content from the origin server. If the CDN does not have a copy of the content in its cache, it will retrieve it from the origin server and serve it to the user.

CDN
A CDN (Content Delivery Network) is a distributed network of servers that are used to deliver content, such as web pages, images, and videos, to users based on their geographic location.

CI/CD
CI/CD stands for Continuous Integration and Continuous Delivery. It's a way to improve efficiency in software development by integrating code changes frequently, automatically building and testing them, and deploying them quickly without manual intervention. This helps reduce errors, improve quality, and speed up delivery of new features and updates.

CMS
A CMS (Content Management System) is a software platform used to manage and publish digital content, such as text, images, and videos. It allows users to create, edit, and publish content without the need for technical expertise.

DevOps

DevOps is a set of practices and tools that helps organizations improve the collaboration and communication between their development and operations teams. The goal of DevOps is to enable organizations to deliver software and other products more quickly and reliably, by automating and streamlining the build, test, and deployment processes.

Environments
In web development, an environment refers to the specific context in which the website or application is being used or tested. This can include the production environment, which is the live version of the site that is publicly available to users, or the staging environment, which is a testing environment used to preview and test changes before they are made live. There can also be other environ-ments such as a QA (quality assurance) environment, which is used for more thorough testing before the changes are pushed to the production environment. These different environments help developers to ensure that the site is functioning correctly and is ready for public use before making any changes live.

Environment Variable
In web development, environment variables are often used to store sensitive information, such as database credentials or API keys, that should not be stored directly in the codebase. By using environment variables, developers can keep sensitive information separate from the code and avoid committing it to version control (GitHub). This helps to improve the security of the application and prevent sensitive information from being accidentally exposed.

Frameworks
A frontend framework is a collection of libraries and tools that are used to build the user interface (UI) of a web application. It provides a set of pre-designed and pre-built UI components, such as buttons, forms, and layout containers, which can be easily customized and assembled to create a functional and visually appealing UI.

Headless
In web development, "headless" refers to the separation of the frontend and backend of a web application. A headless web application is one that has no user interface (UI) and is accessed only through an API (Application Programming Interface).

Webhooks
Webhooks are a way for one application to send a message or trigger an event to another application over the internet. They are often used to enable real-time communication and integration between different applications and services.

If you would like to see how they work visit this website: https://webhook.site and copy the generated URL into the webhooks section of your Contentful account as outlined in Chapter 3, then update some content. You can see the **Webhook.Site** update with the Webhook Sent from Contentful.