The Docs 2.0 Process

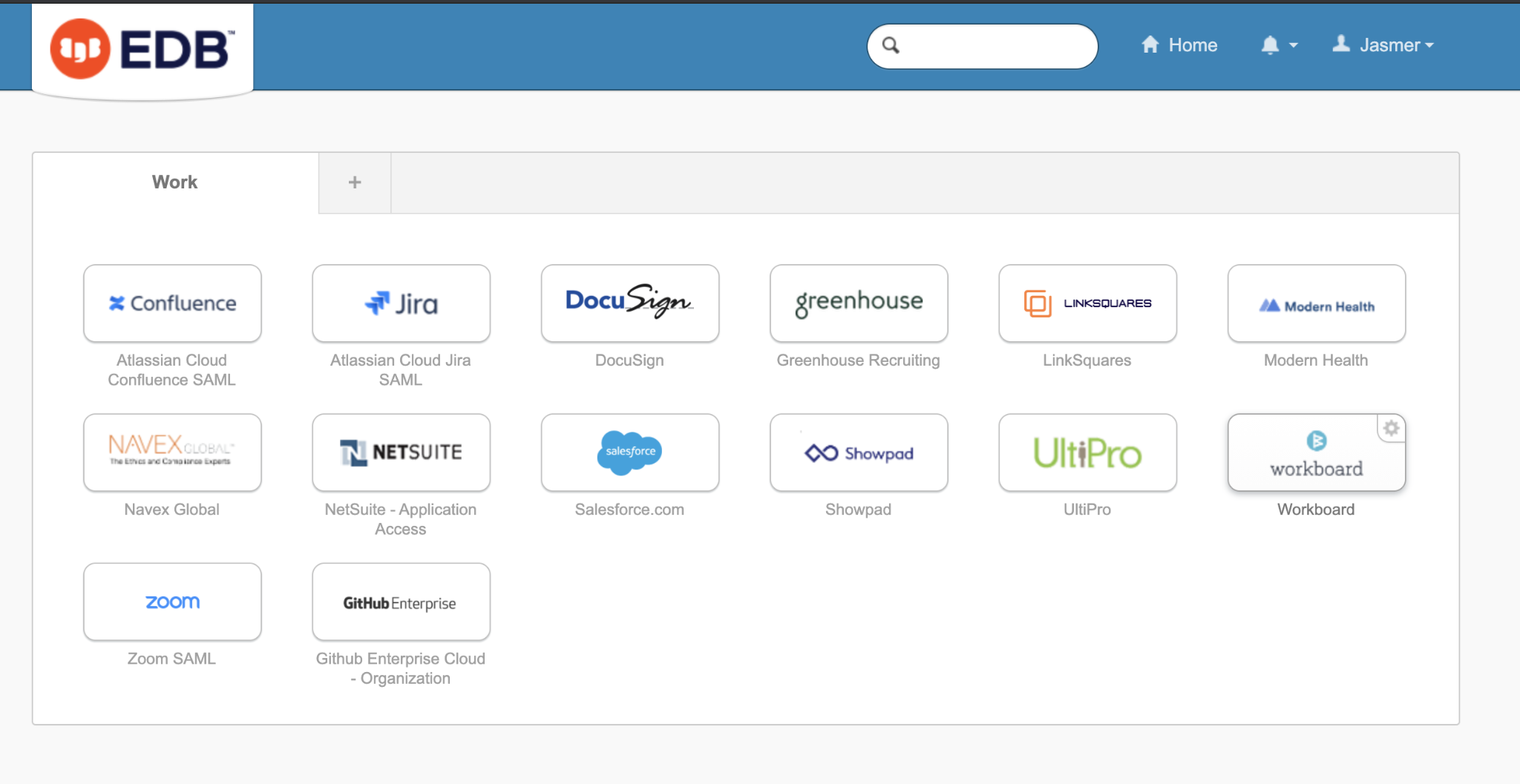
By Jasmer Sethi

## The Docs 2.0 process involves taking PDFs of implementation guides put together by partners and edited by the Tech Partner team and converting them to Markdown files in order to take advantage of the EDB Docs formatting and having it published on the Docs homepage. This document explains the process in stages that take the user from installing Markdown on his or her OS all the way through to creating a Pull Request that sends out the implementation guide for publishing.

## Installing Markdown on MacOS:

The directions on how to install and run Markdown on MacOS can be viewed in greater detail and found at <https://github.com/EnterpriseDB/docs/blob/develop/README.md> but here is a brief overview:

1. Ensure that you have access to GitHub on your Okta account. If you don’t have it, contact the IT Helpdesk ([helpdesk@enterprisedb.com](mailto:helpdesk@enterprisedb.com)) to request access to GitHub.



*Look for the GitHub enterprise icon on your Okta page.*

1. Install Homebrew on your computer, if you already don’t have it.
2. Get permission from the IT Helpdesk to access the Docs repository on GitHub.
3. Install git and git-lfs using Homebrew with command brew install git git-lfs.
4. Set up an SSH key on GitHub (https://github.com/settings/keys) , follow the instructions on the following page to set it up

<https://docs.github.com/en/github-ae@latest/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent>

1. Clone the Docs repository to your computer through GitHub desktop.
2. Navigate to the cloned repo directory in your Terminal and create a .env.development file: cp env.development.example .env.development
3. Install Docker on your computer using the command brew install --cask docker

If you get a message saying that you already have Docker installed, check which version is installed using these commands

brew ls --formula docker and brew ls --cask docker

If the first command yields results, enter the following command to uninstall the formula version and to install the cask version

brew uninstall -f docker && brew install --cask docker

1. If you're a member of the EnterpriseDB Github Org, follow the instructions below to enable icons in the docs application.

<https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/creating-a-personal-access-token>

Create a Github token which can pull private packages.

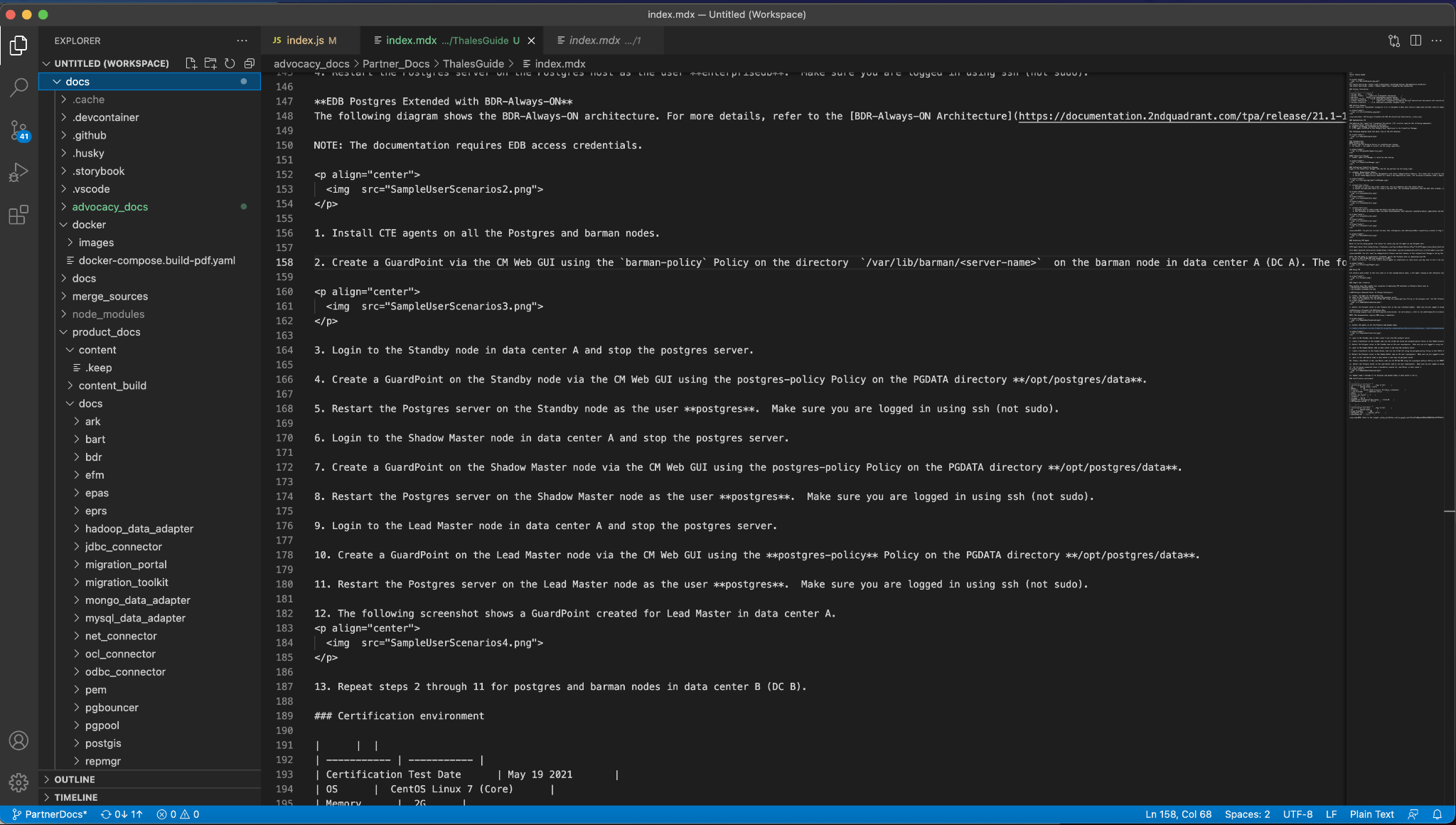
* 1. The minimum scope the token requires is read:packages
  2. Make sure to enable SSO for your token or it may not work correctly

1. Once you have that token, update your .env.development file with this line: NPM\_TOKEN=your-token-here. That will be utilized in the NPM\_TOKEN variable in the .npmrc file.
2. Go to Applications Folder on your computer and open the Docker Application.
3. Execute command docker --version and if the following error is displayed then restart the Docker application.

zsh: command not found

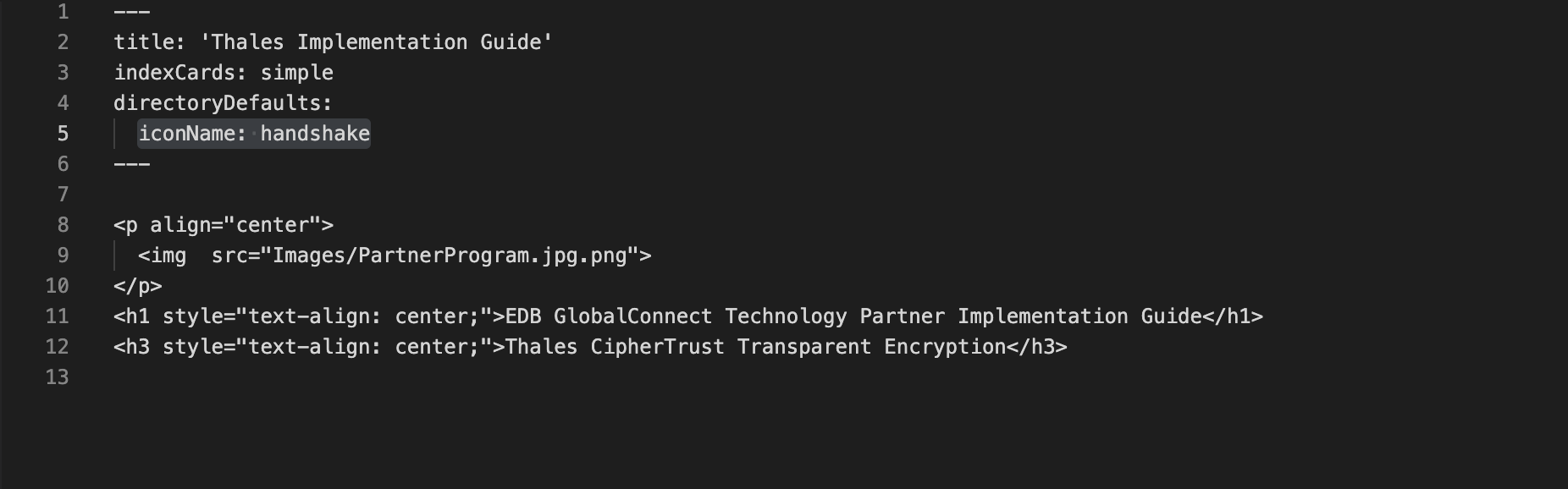
1. Create your own branch in GitHub desktop in order to ensure that you have your own local copy of the docs repo that you can use in order to create your documents in the Docs 2.0 standard format while making sure that it doesn’t interfere with what gets published online.

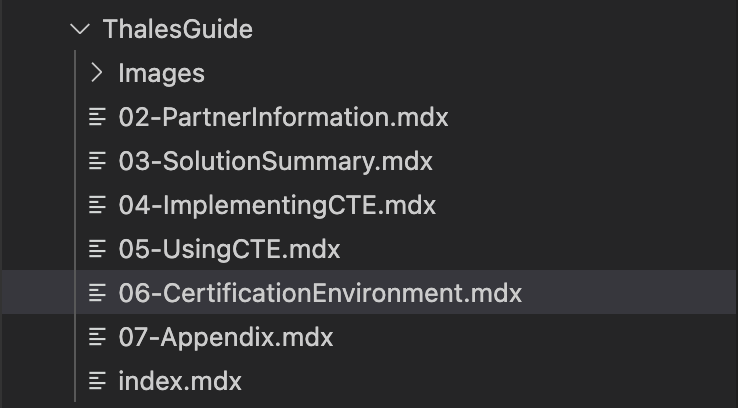
The end result should be that you have a local copy of the docs repo in a local folder that you created, as shown on the left side of the screenshot below:



## Creating Implementation Guides in Markdown on MacOS:

More information on how to use Markdown syntax to write documents can be found at <https://www.markdownguide.org/cheat-sheet/> but here is a brief overview:

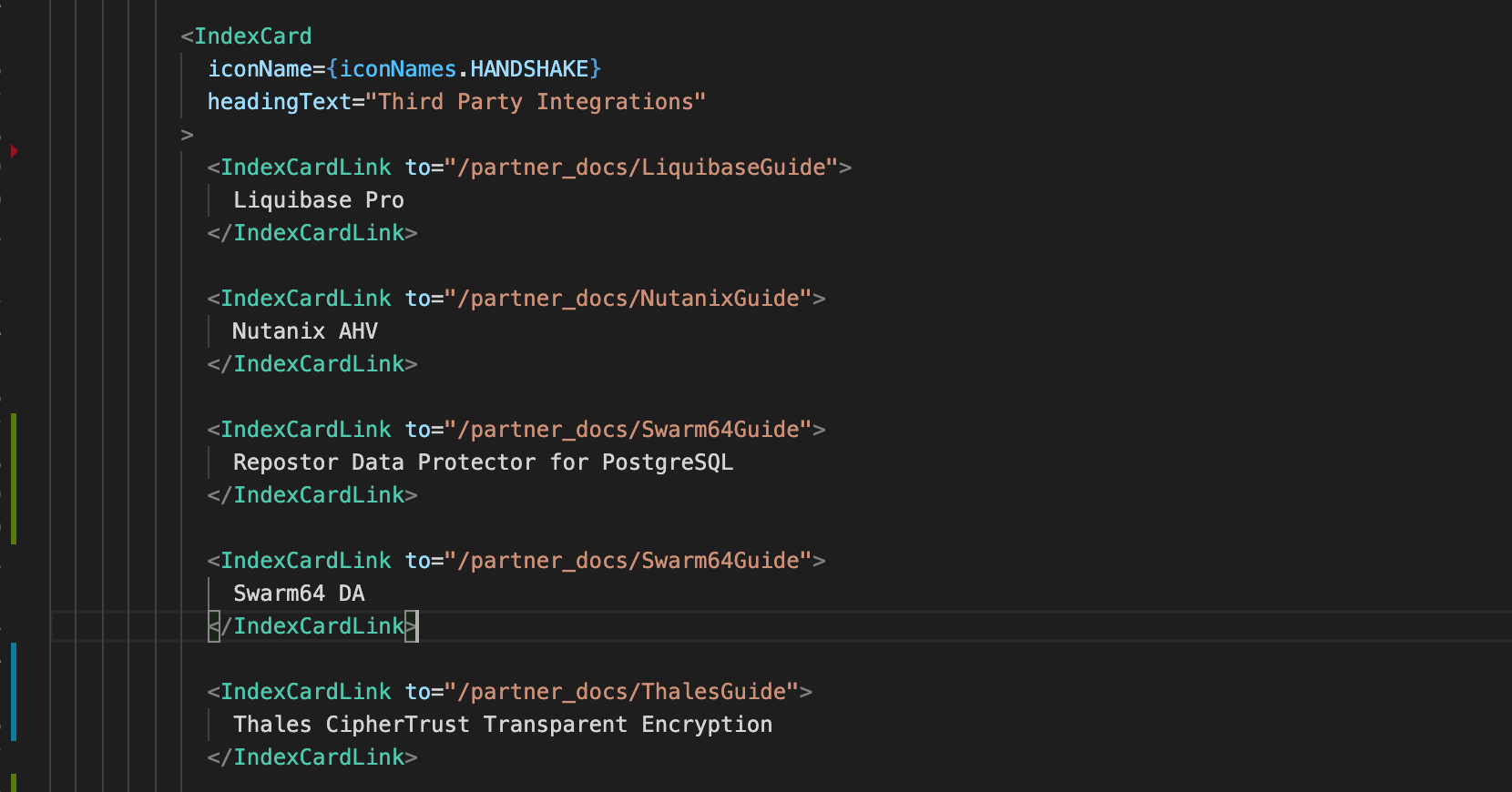
1. Once you have cloned the Docs repository and have set up the Markdown environment in Visual Studio Code, create a folder in which you’ll create the implementation and another folder within that for the images you’ll include in the guide.
2. Create a file titled index.mdx. This file will serve as the homepage for your implementation guide in Markdown. You must include a header that gives a title for the document (ex: “title: 'Thales Implementation Guide' “), indicates that several files will be brought together (“indexCards: simple”), and has an icon that will be used on the left side of the guide (“iconName: handshake”). Look at the image below for example of an index.mdx file: 
3. In that same index.mdx file, insert the Partner Program image using the html syntax displayed in the image above. Then, utilize the code below the image syntax to indicate that this is an EDB GlobalConnect Technology Partner implementation guide and mention the name of the company and the product underneath that.
4. Each section of the implementation guide should be written up in their own .mdx file and should be marked with a number at the beginning in order to organize them as they were in the pdf version.
   1. For example, the Thales Implementation Guide has the following sections:
      1. Partner Information
      2. Solution Summary
      3. Implementing CipherTrust Transparent Encryption (CTE)
      4. Using CipherTrust Transparent Encryption (CTE)
      5. Certification Environment
      6. Appendix
   2. These sections should be recreated as their own .mdx files within the guide folder as saved as “02-PartnerInformation.mdx”, “03-SolutionSummary.mdx”, and so forth so that they get ordered.
   3. This is how a folder for an implementation guide recreated in Markdown should look:



* 1. Set the title as “(company) Implementation Guide” in the index.mdx file of the folder (ex: “title: Thales Implementation Guide”)
  2. This is how the implementation guide should look on your local host:

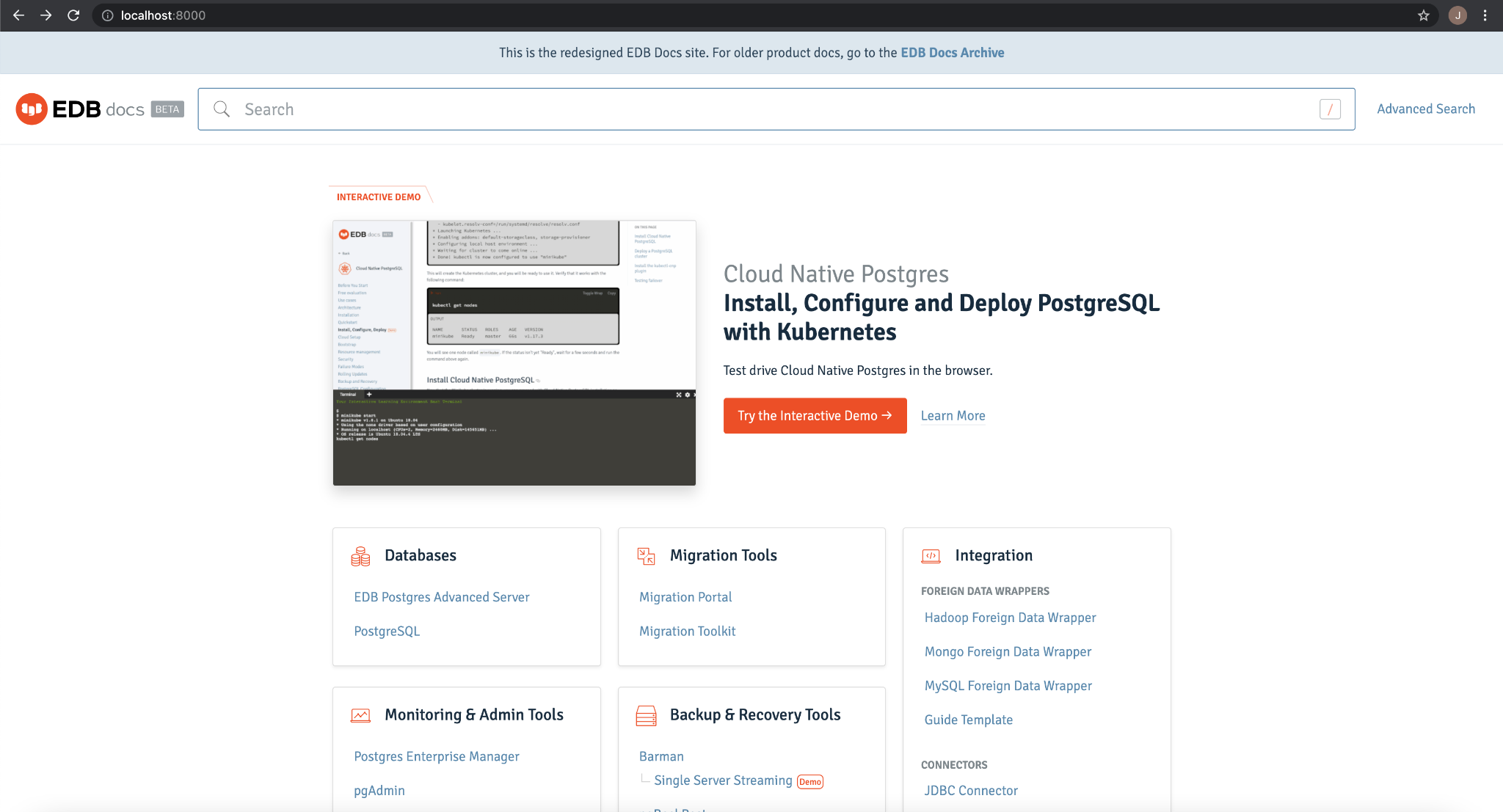


*Notice that the sections of the Implementation Guide are ordered as they are in the PDF after having the numbering in the title of the .mdx file.*

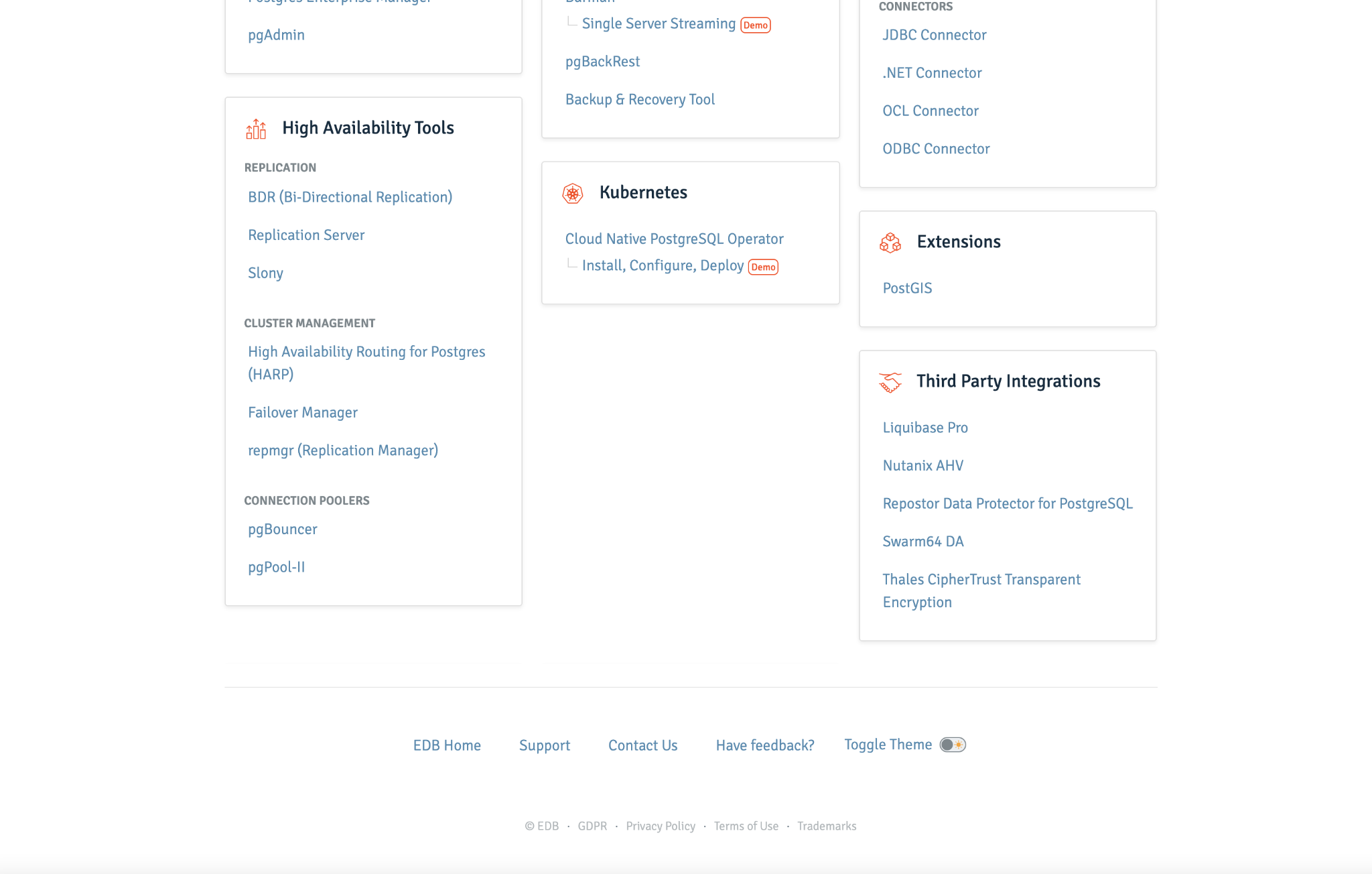
* 1. Create a box for the Implementation Guides on your local host of the Docs homepage and title it **Third Party Integrations**
     1. Go to the **index.js** file under the **pages** subfolder within the **src** folder
     2. Implement the following syntax in order to create the box:
     3. List the guides in alphabetical order.
     4. Make sure that the **Third Party Integrations** box is the last box, it should be the last one coded after all the other ones in the index file.

## Viewing the Implementation Guides on Your Local Host:

1. Open up Terminal on your computer.
2. Change the directory location to where you’ve stored GitHub.
   1. Example: If I stored GitHub in my Documents folder, I would type “cd Documents”.
3. Change the directory location to the docs repo.
   1. Example: Type “cd docs”.
4. Type “yarn develop” to generate the local host.
5. Once Terminal is done running the “yarn develop” command, look for the URL of the local host.
   1. It should look like this: [http://localhost:**8000**/](http://localhost:8000/)
6. Copy and paste the local host URL in your web browser.
7. You should be taken to the local version of the Docs website:



1. Scroll down to the bottom to find the **Third Party Integrations box**:

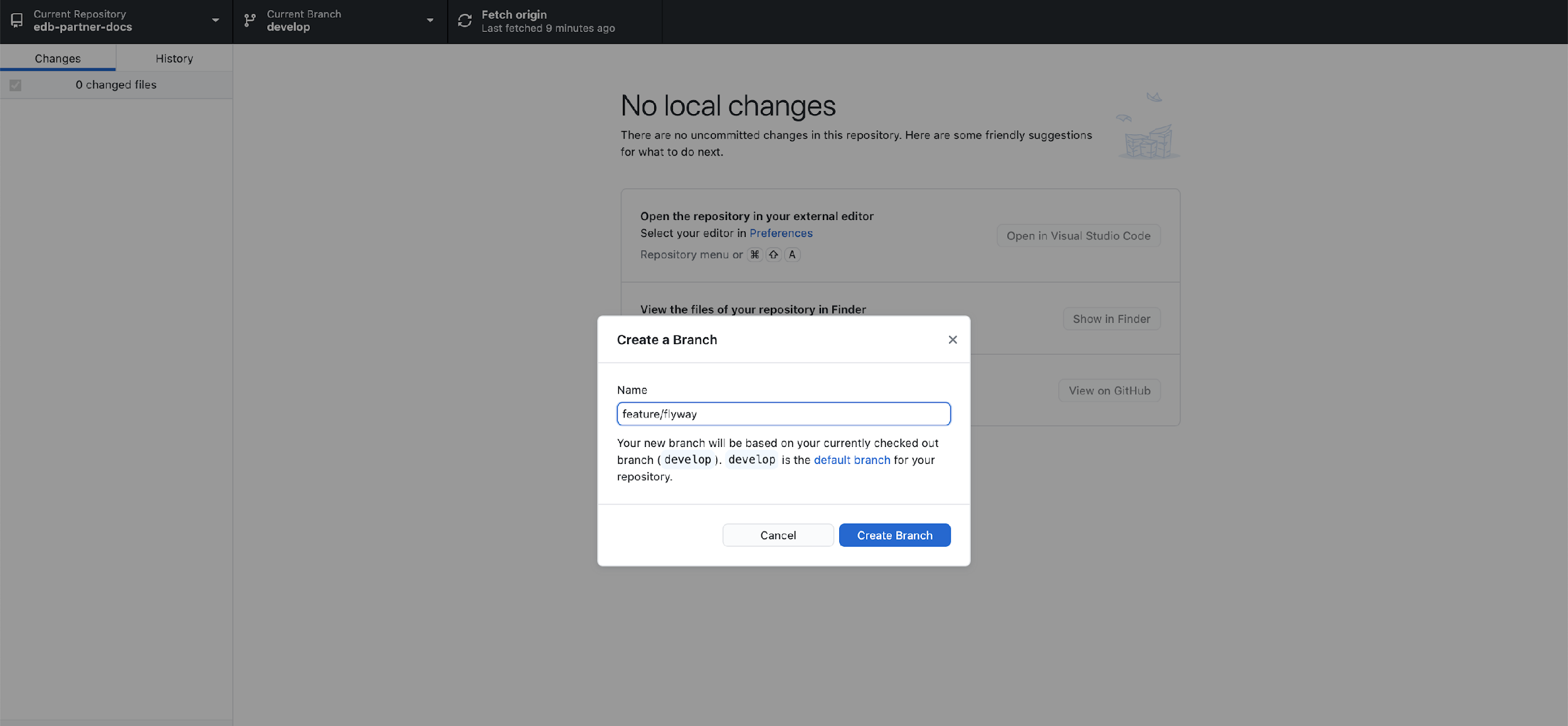


1. Click on any implementation guide that you wish to access from there.

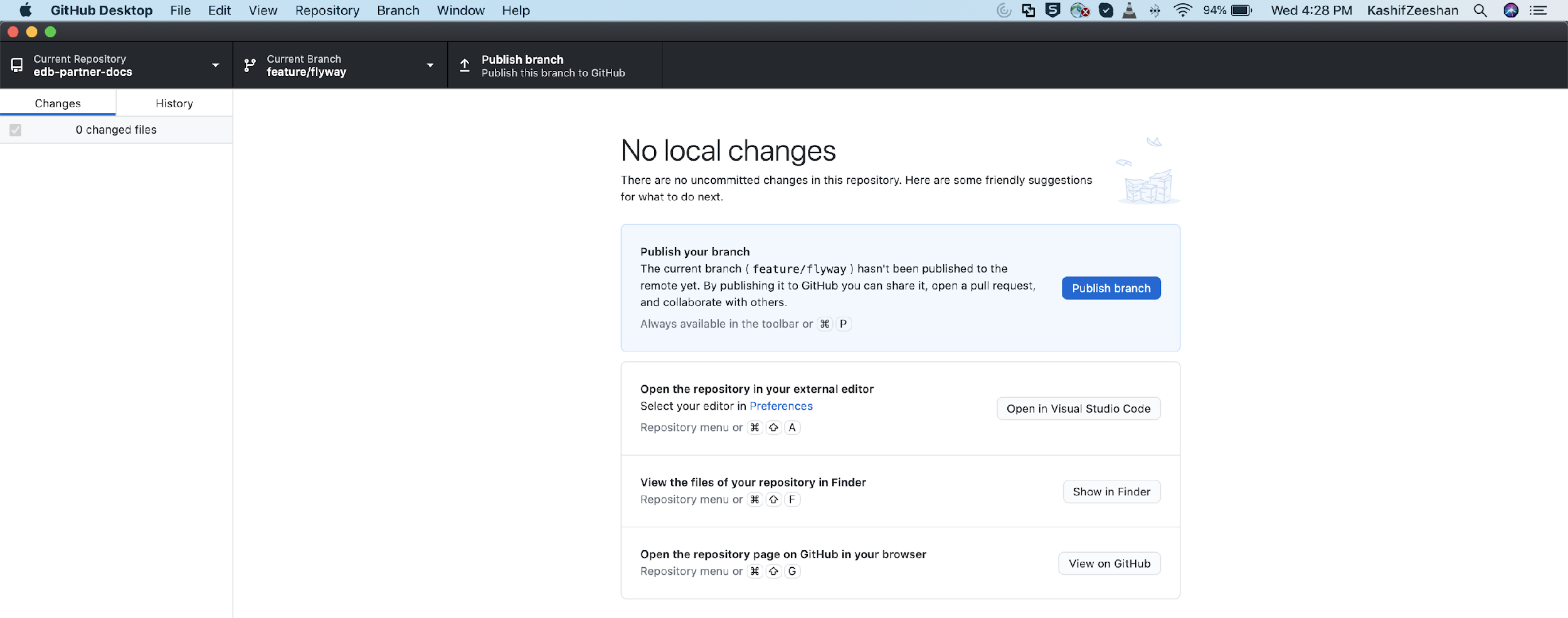
## Publishing and Making Changes to the Implementation Guides on the Docs Homepage:

The tutorial on how to push changes in GitHub goes into greater detail on publishing and making changes to the guides and can be found [here](https://docs.google.com/document/d/1JwCuDdK1txupwg9cCLsDi_uXv8fj8ANYw2L3oeuI-Z8/edit?usp=sharing).

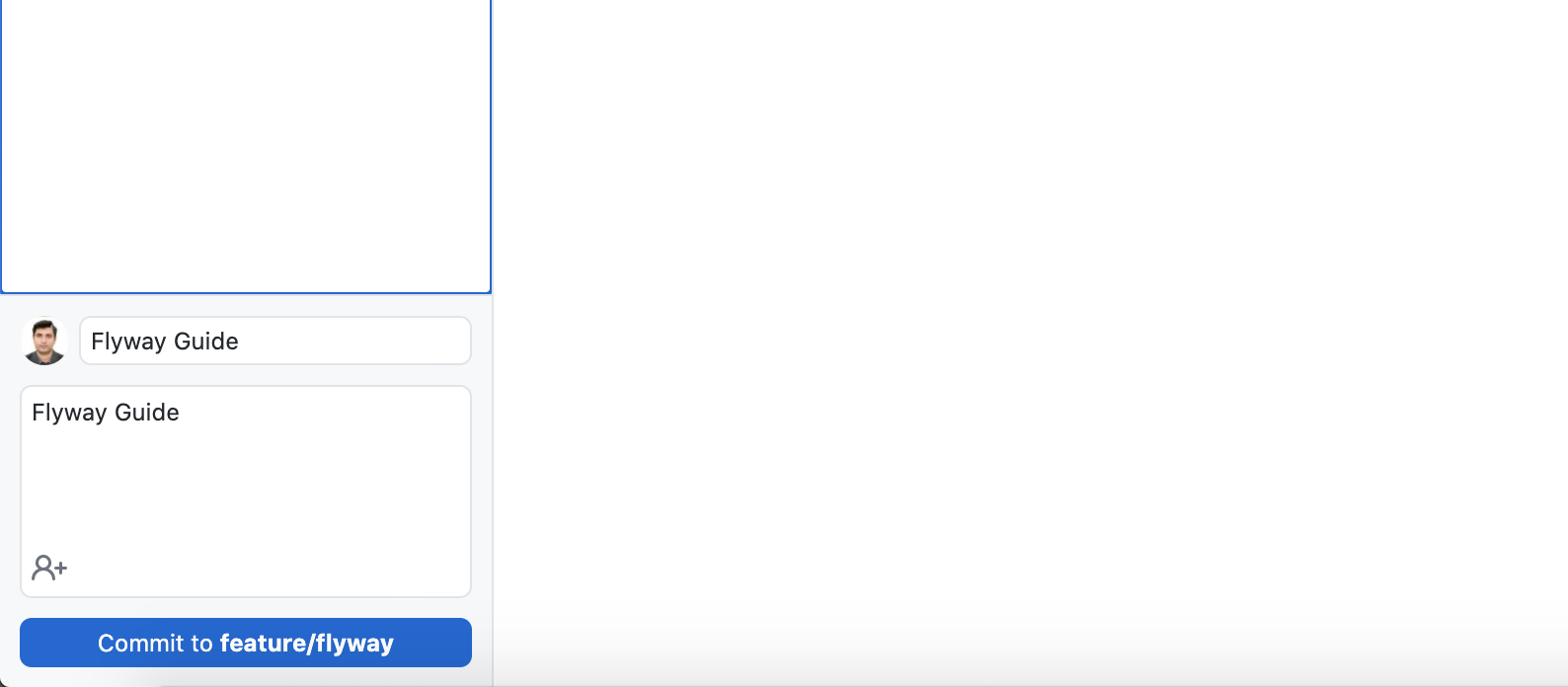
1. Create a new branch in order to store any changes that you need to make to the implementation guide by clicking on the Branch menu, “New Branch”, and then “Create Branch”.



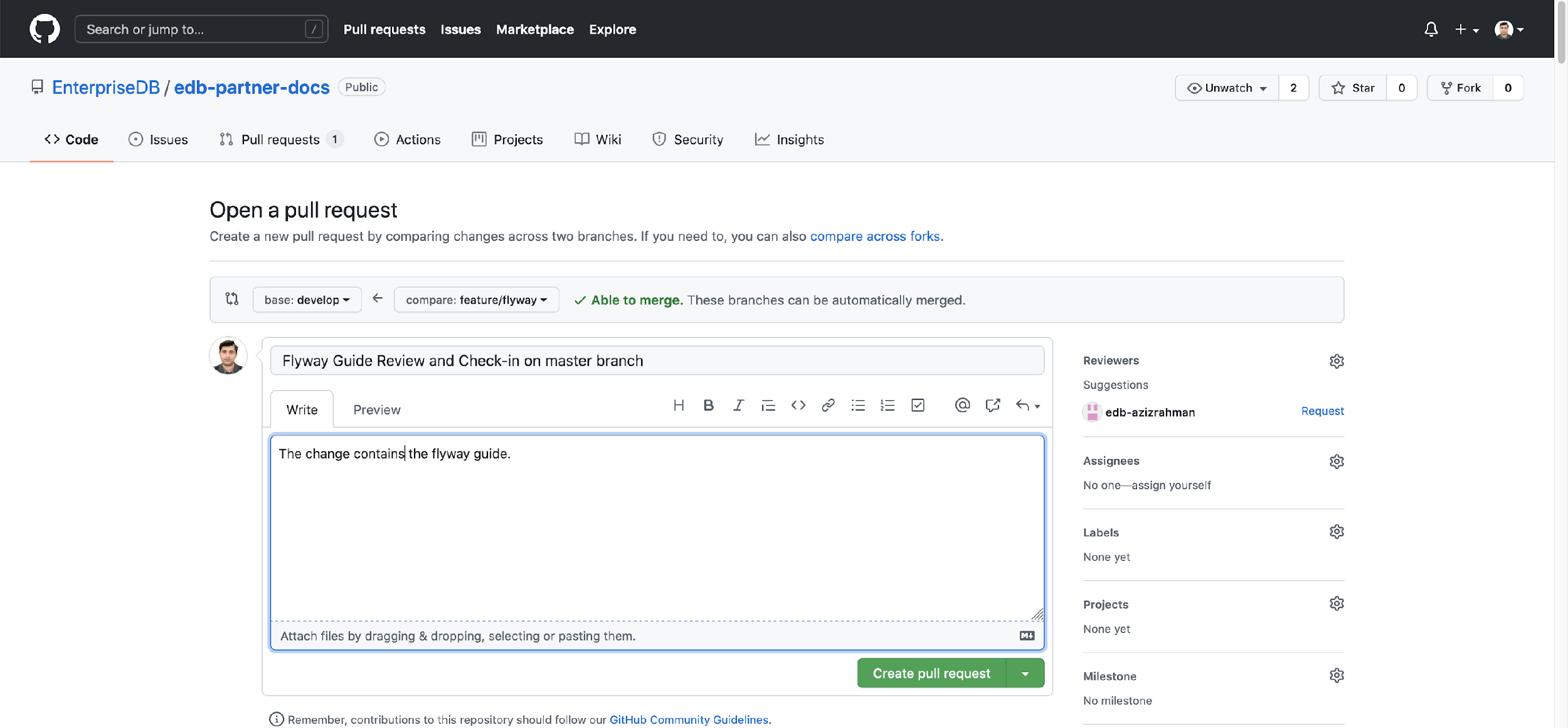
1. Publish the branch so that you can eventually create a pull request, have it reviewed by your team, and get it approved for publishing.



1. Save all the changes you made to the implementation guide by saving the folder in your branch.
2. Make sure your branch is refreshed. In your terminal, run the following commands in order to do so:
   1. **If you've no local changes or commits:** git reset --hard (branch name)
   2. **If you do:** git pull --rebase
3. Once any changes have been made, click on the blue “Commit to ‘...’ “ in order to commit them to the branch that you have created.



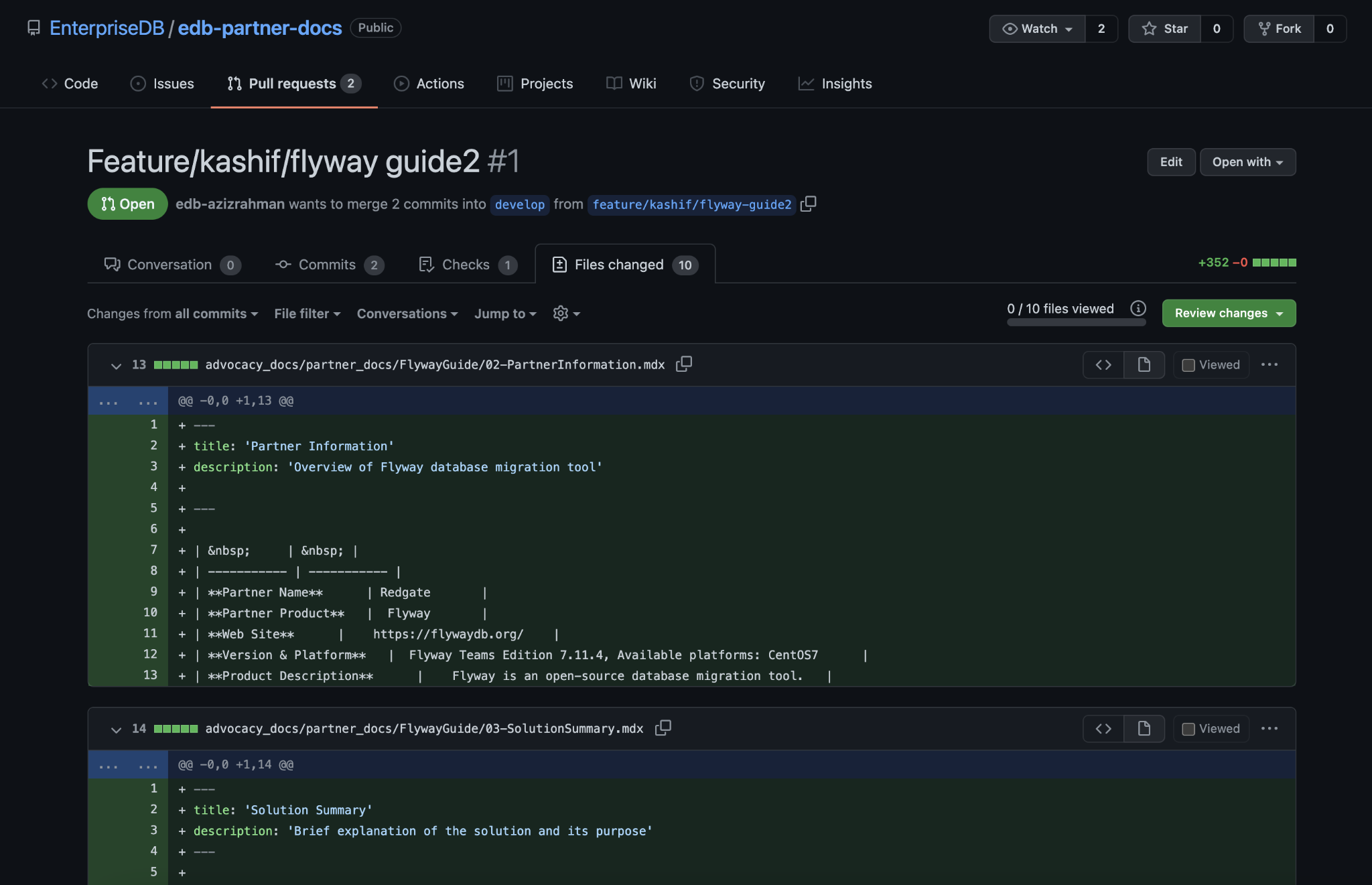
1. Once the changes have been committed, press the “Push Origin” button in order to push the changes to the repository.
2. Go to the GitHub enterprise website and find the copy of the docs repository that you had created.
3. Click on the drop-down list of branches to find the one to which you had committed changes.
4. Find the one that has the implementation guide(s) that you’re trying to publish.
5. Once you have found the correct branch, click on the “Create Pull Request” button in order to have your changes reviewed and approved to be merged with the main develop branch.
6. Reach out to a member from the Docs team and have him/her added as a reviewer of the Pull Request under the Reviewers section on the right side of the page.
7. Add your teammates (if any) so that they can be able to review your changes as well.



1. At the bottom of the Pull Request page, you should be given an option to make the Pull Request ready for review.
2. Your reviewer will review the Pull Request, make suggested changes if there are any, and will be able to merge it for publication.
3. If you want to see how the guide looks in the developer environment, put a “deploy” label on the pull request and a website link to the guide in the developer environment will be generated within 30-45 minutes.

## Reviewing and Commenting on the Implementation Guides:

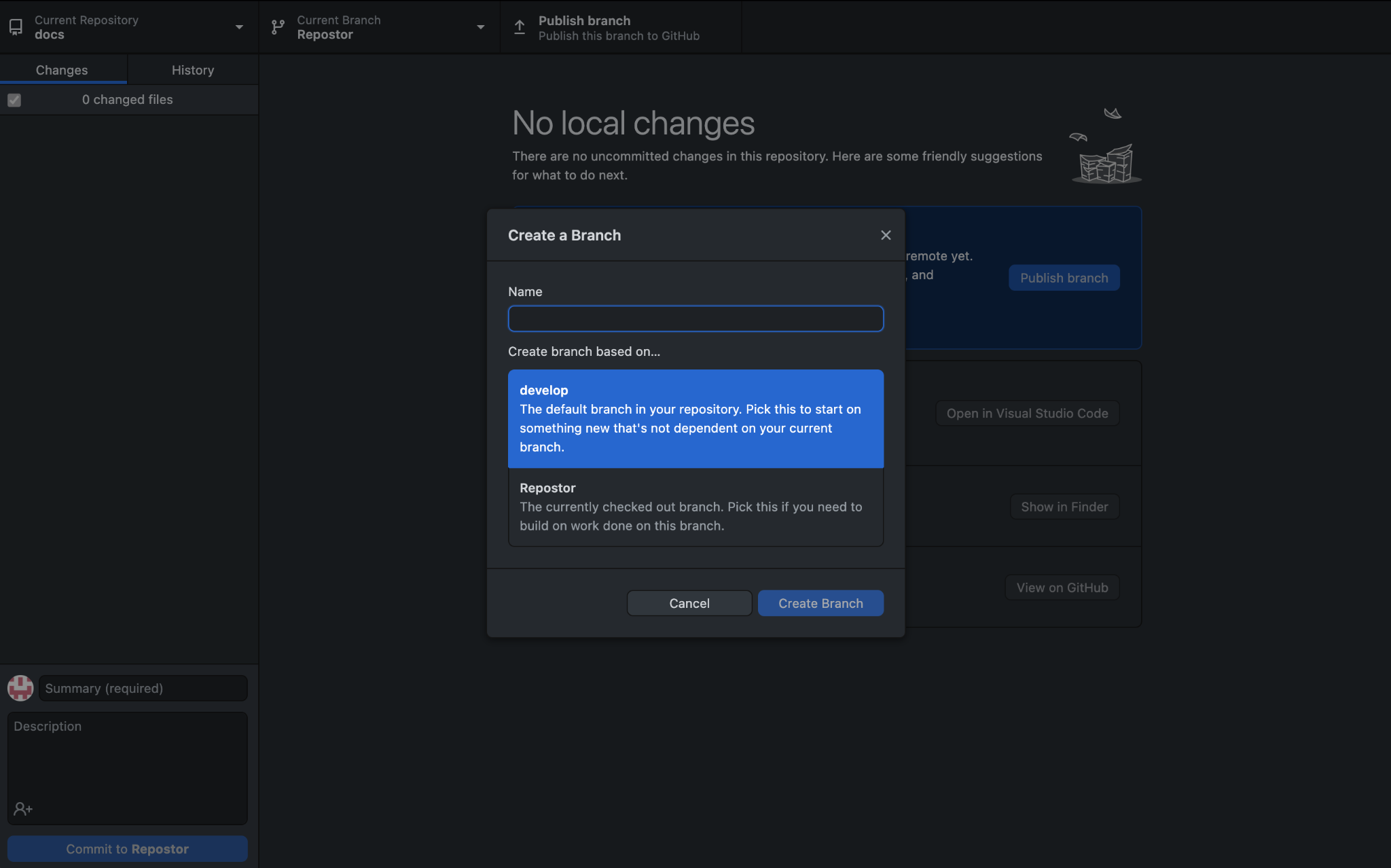
1. Go to the page of the pull request you created on GitHub Enterprise, click on the “Files Changed” tab, and you’ll see a green button on the right titled “Review Changes”.



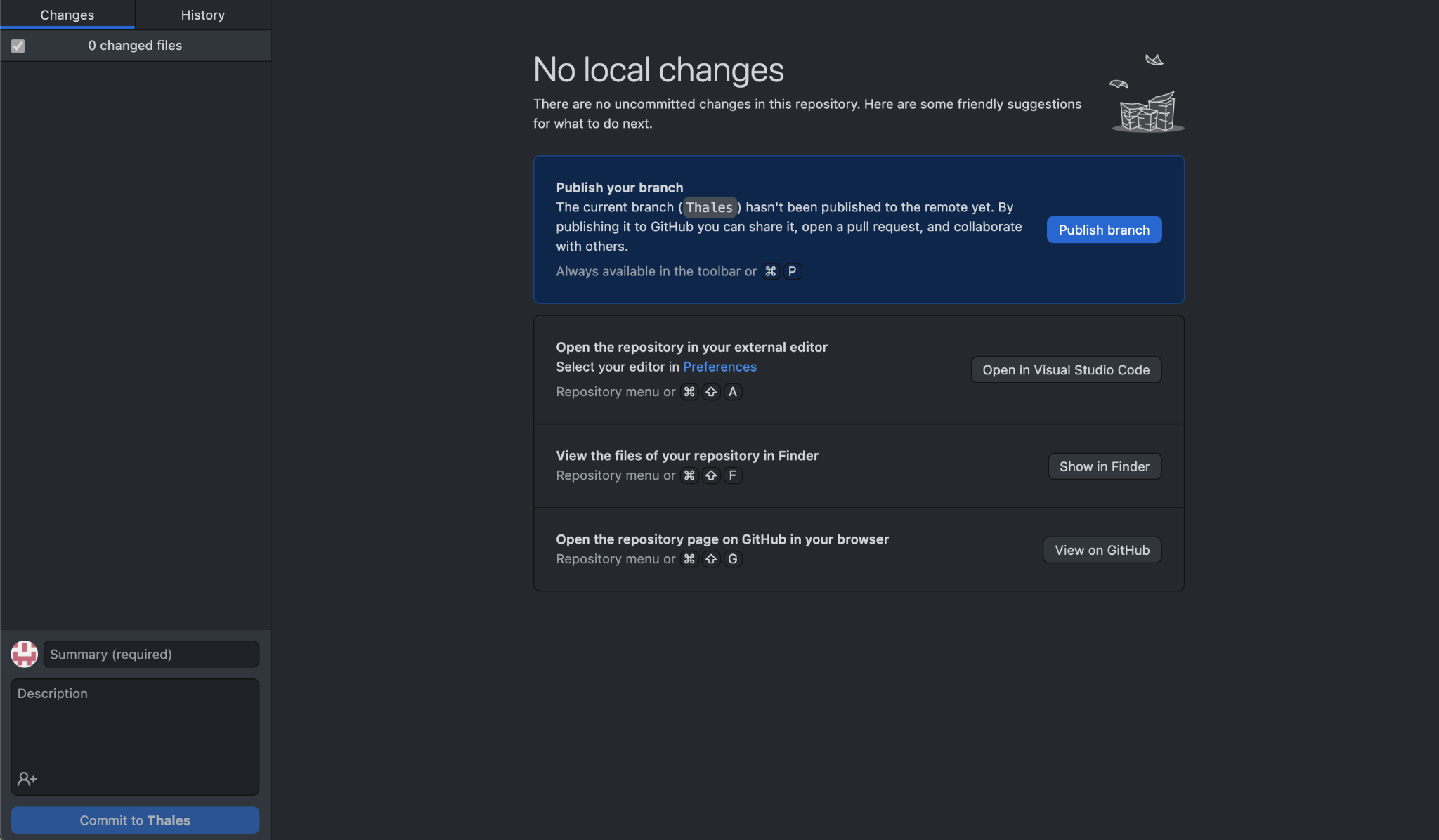
1. Once you click that button, you’ll be given 3 options that will allow you to review and comment on the guides themselves:
   1. **Commenting:** Submitting general feedback without explicit approval
   2. **Approving:** Submitting feedback and approving the merging of these changes.
   3. **Requesting changes:** Submitting feedback that must be addressed before merging
2. Once these changes have been approved by those who have been asked to review the document and there are no conflicts within the branch, then the branch is ready to be merged in with the develop branch.

## Updating Existing Implementation Guides:

1. Create a copy of the develop branch while keeping the docs repository as your current repository in GitHub desktop.



1. Name the branch however you would like, although it would be good practice to incorporate the name of the guide that you’re making changes to.
2. Go back into Visual Studio Code, go into the partner\_docs folder within the advocacy\_docs folder and find the file of the implementation guide you’re making changes to.
3. Make your needed changes to the file, save it, and then go back to GitHub Desktop.
4. Whatever changes you have made to the file will show up in the changes section on the left side, click on the blue “Commit to” button at the bottom to commit these changes to your branch.



1. Once you’ve committed your changes to the branch, click on the blue “Publish branch” button to create a pull request that will be reviewed and merged, pending approval.

## Docs 2.0 Process Diagram:

