

task_one

Public

generated from [education/codespaces-project-template-js](#)

🔑 sara/update-home-image

had recent pushes 26 seconds ago

Compare & pull request

🔑 sara/update-packages

had recent pushes 25 seconds ago

Compare & pull request

🔑 sara/update-site-gif

had recent pushes 25 seconds ago

Compare & pull request

🔑 main ▾

🔑 8 Branches

🔑 0 Tags

🔑 🔍 Go to file t

Go to file

+

Add file ▾

<div>🧩 Shahzaib-Gandapoor</div>	Initial commit	f2cccf5 · 1 minute ago	🕒 1 Commits	⋮
📁 .devcontainer	Initial commit	1 minute ago		
📁 .github/workflows	Initial commit	1 minute ago		
📁 __images__	Initial commit	1 minute ago		
📁 src	Initial commit	1 minute ago		
📁 translations	Initial commit	1 minute ago		
📄 .eslintrc	Initial commit	1 minute ago		
📄 .gitignore	Initial commit	1 minute ago		
📄 .prettierrc	Initial commit	1 minute ago		
📄 CODE_OF_CONDUCT.md	Initial commit	1 minute ago		
📄 LICENSE	Initial commit	1 minute ago		
📄 README.md	Initial commit	1 minute ago		
📄 SECURITY.md	Initial commit	1 minute ago		
📄 SUPPORT.md	Initial commit	1 minute ago		
📄 package-lock.json	Initial commit	1 minute ago		
📄 package.json	Initial commit	1 minute ago		
📄 slides.pptx	Initial commit	1 minute ago		

About

No description, website, or topics provided.

📖 Readme

📄 MIT license

📄 Code of conduct

📄 Security policy

📈 Activity

🌟 Code

0 stars

👁 1 watching

🔗 0 forks

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

Languages

🟡 JavaScript 85.2%

🟣 CSS 11.1%

🔴 HTML 3.7%

Suggested workflows

Based on your tech stack

🌀 Deno

Configure

Test your Deno project

✉ SLSA Generic generator

Configure

Generate SLSA3 provenance for your existing release workflows

📦 Publish Node.js Package to GitHub Packages

Configure

Publishes a Node.js package to GitHub Packages.

JavaScript Portfolio Site with GitHub Codespaces and Copilot

[More workflows](#)[Dismiss suggestions](#)

View these instructions in [Spanish](#) or [Portuguese](#)

Create, customize and deploy your own portfolio website in minutes. ✨

In this template repository we have the development environment and base set and ready to go. So that you can immediately launch your [Codespace](#) environment and start customizing your site using [Copilot](#) to help you write code faster.

- **Who is this for?** Anyone looking to create a portfolio site, learn web development, or test out Codespaces.
- **How much experience do you need?** Zero. You decide how much you want to customize based on your experience, and time available.
- **Tools needed:** None. No need to install anything! All you need is a web browser.
- **Prerequisites:** None. This template includes your development environment and deployable web app for you to create your own site.

About this portfolio template

In this "choose your own adventure" template portfolio, we have a [React](#) based web application ready for you to easily customize and deploy using only your web browser.



Quick Start

1. Click the **Use this Template** button and then **Create a new repository**
2. Select the repository owner (e.g. your GitHub account)
3. Enter a unique name for your new repository
4. Click the **Code** button
5. Click **Create Codespace on main** button
6. [Customize your portfolio site](#) with Copilot
7. [Deploy your site](#)

► 📺 To learn more about Codespaces, watch our video tutorial series

JavaScript Portfolio template

This repo is a GitHub template to build a JavaScript personal portfolio frontend web application using the React framework. The goal is to give you a template that you can immediately utilize to create your own website through Codespaces.

The repo contains the following:

- `/.devcontainer`
 - `.devcontainer/Dockerfile` : Configuration file used by Codespaces to determine operating system and other details.
 - `.devcontainer/devcontainer.json` : Configuration file used by Codespaces to configure Visual Studio Code settings, such as the enabling of additional extensions.
- `/src` : HTML, JS and CSS files used to build your portfolio site.
- `.eslintrc` : Settings for [ESLint](#) that is included for code consistency and quality.
- `.prettierrc` : Settings for [Prettier](#) that is used to format code.
- `package.json` and `package-lock.json` : Defines the project information for [Node.js](#), dependent packages and the versions needed of each.



Getting started

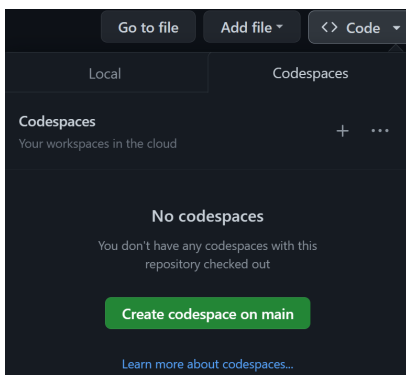
This portfolio site project is filled with sample data so that you can immediately open Codespaces, see it running, and deploy at any point.

Your development environment is all set for you to start. Based on our [JavaScript Codespace Template \(React\)](#), here is what is already setup and ready for you to use:

- Simple [React](#) application with components for each section of the portfolio site
- [Parcel](#) in place to build your site when deploying
- Code linting and formatting using [ESLint](#) and [Prettier](#) for code consistency.

Create your portfolio


1. Create a repository from this template. Use this [create repo link](#). Select the repository owner, provide a name, a description if you'd like and if you'd like the new repository to be public or private.
2. Before creating the Codespace, enable GitHub Copilot for your account. If it is not enabled, take a look at [Customize your portfolio site using Copilot](#).
3. Navigate to the main page of the newly created repository.
4. Under the repository name, use the Code drop-down menu, and in the Codespaces tab, select "Create codespace on main".



5. Wait as GitHub initializes the Codespace.

Setting up your codespace

Connecting...

 **Tip** See your application running with port forwarding. [Learn more](#)

6. When complete you will see your Codespace load with a terminal section at the bottom. Codespaces will install all the required extensions in your container, followed by executing `npm install`. Once the package installs are completed, Codespaces will execute `npm start` to start your web application running within your Codespace.

When the web application has successfully started you will see a message in the terminal that the server is running on port 1234 within your Codespace:

```
Server running at http://localhost:1234
🌟 Built in 1.40s
```

Customize your site in 3 steps

This project is built to be easily customizable. Each section of the site is a separate component, and your information needs to be set in only one spot. This is not only for ease of updating, but so you can see how prop values are passed to React components.

For each step, open the project in Codespaces, then you can make and commit your changes while within your Codespace.

See [Using source control in your codespace](#) for more Codespaces source control how-tos

1 Add your "About me" and social accounts

Within `/src/App.jsx` you will see a variable called `siteProps`. This is a JavaScript object that hold the key value pairs needed to customize your name, title, email, and social accounts.

```
const siteProps = {
  name: "Alexandrie Grenier",
  title: "Web Designer & Content Creator",
  email: "alex@example.com",
  gitHub: "microsoft",
  instagram: "microsoft",
  linkedIn: "satyanadella",
  medium: "",
  twitter: "microsoft",
  youTube: "microsoft",
};
```



Update to the name and title you'd like displayed at the top of your site.

Optional values are email address and social accounts. These are used in the `Footer` component. If any item is not included in the list or set to an empty string ("") it will not display the icon and link.

2 Update images

This portfolio site includes 3 images: top section background, "About me" background and portfolio section (desk). These can be any **landscape** sized images of your choosing from your own collection, or found that have a license allowing you to use without attribution.

A couple possible sites to find photos are [Pixabay](#) and [Unsplash](#). Photos, illustrations, vectors, your choice! When you find your images, save each one to `/src/images` with a short, meaningful file name.

Go to the following below components to update the `import image...` line to reference the new image you downloaded for that section, as well as the `imageAltText` for the image:

- `/src/Components/Home.jsx` - section at top of the page, main image you will see when site loads (woman standing by server wall in sample)

```
import image from "../images/server-wall.jpg";  
const imageAltText = "woman holding laptop standing by server room with glass wall";
```



- `/src/Components/About.jsx` - background behind the detailed "About me" section (abstract mosaic in sample)

```
import image from "../images/mosaic.svg";  
const imageAltText = "purple and blue abstract background";
```



- `/src/Components/Portfolio.jsx` - image highlighted in left hand side of section (design desk photo in sample)

```
import image from "../images/design-desk.jpeg";  
const imageAltText = "desktop with books and laptop";
```



3 Add items you've worked on and detail text

The About section helps to give people a bit more information about your skills and passions. Within `/src/Components/About.jsx` you will find 2 values to update:

- `description` : short sentence or two describing yourself, career goals, and/or passions
- `detailOrQuote` : longer block for you to add more detail about yourself, or even a quote you like

The second section to update is the Portfolio section, where you highlight items you've worked on. These would be articles, videos, logo designs, GitHub projects, anything that highlights you!

Go to `/src/Components/Portfolio.jsx` to the `projectList` variable. This is a JavaScript array of objects. Each item you want to highlight needs: title, description, and URL.

The sample design has 4, but the number you include is up to you.

```
const projectList = [  
  {  
    title: "10 Things to know about Azure Static Web Apps 🍌",  
    description: "Collaboration to create a beginner friendly...",  
    url: "https://dev.to/azure/10-things-to-know-about-azure-static-web-apps-3n4i",  
  },  
  {  
    title: "Web Development for Beginners",  
    description: "Contributed sketch note imagery to accompany...",  
    url: "https://github.com/microsoft/web-dev-for-beginners",  
  },  
  {  
    title: "My Resume Site",  
    description: "Created from Microsoft's resume workshop...",  
    url: "https://github.com/microsoft/workshop-library/tree/main/full/build-resume-website",  
  },  
  {  
    title: "GitHub Codespaces and GitHub.dev",  
    description: "Video interview to explain when to use GitHub.dev...",  
  },  
]
```



```
url: "https://www.youtube.com/watch?v=c3hHhRME_XI",
},
];
```

Deploy your web application

Project includes the setup needed for you to deploy **FREE** to either [Azure Static Web Apps](#) or [GitHub Pages](#). Instructions are included below for both:

Azure Static Web Apps

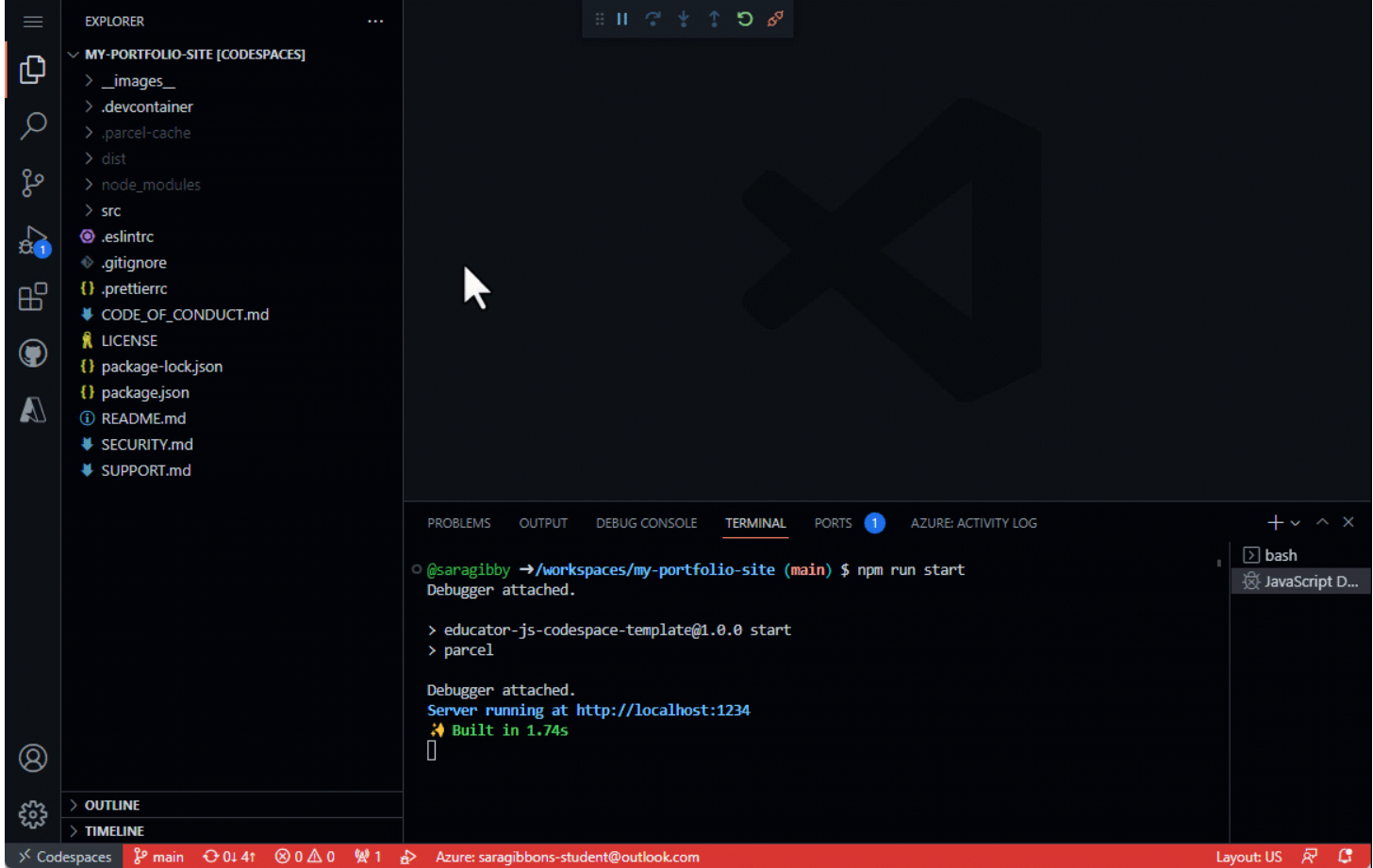
[Azure Static Web Apps](#) is Microsoft's hosting solution for static sites (or sites that are rendered in the user's browser, not on a server) through Azure. This service provides additional opportunities to expand your site through Azure Functions, authentication, staging versions and more.

You'll need both Azure and GitHub accounts to deploy your web application. If you don't yet have an Azure account you can create it from within during the deploy process, or from below links:

- [Create a \(no Credit Card required\) Azure For Students account](#)
- [Create a new Azure account](#)

With your project open in Codespaces:

1. Click Azure icon in the left sidebar. Log in if you are not already, and if new to Azure, follow the prompts to create your account.
2. From Azure menu click "+" sign and then "Create Static Web App".
3. If you are not logged into GitHub you will be prompted to log in. If you have any pending file changes you will then be prompted to commit those changes.
4. Set your application information when prompted:
 - i. **Region:** pick the one closest to you
 - ii. **Project structure:** select "React"
 - iii. **Location of application code:** /
 - iv. **Build location:** dist
5. When complete you will see a notification at the bottom of your screen, and a new GitHub Action workflow will be added to your project. If you click "Open Action in GitHub" you will see the action that was created for you, and it is currently running.



6. To view the status of your deployment, find your Static Web App resource in the Azure tab in the VS Code left side bar.
7. Once deployment is complete, you can view your brand new new publicly accessible application by right clicking on your Static Web App resource and selecting "Browse Site".

Issues? When creating your Static Web app, if you are prompted to select an Azure subscription and are not able to select a subscription, check the "Accounts" tab in VS Code. Make sure to choose the "Grant access to ..." options if those options appear. Should you receive the error-message "RepositoryToken is invalid. ..." switch to Visual Studio Code for the Web (vscode.dev) and repeat the steps there.

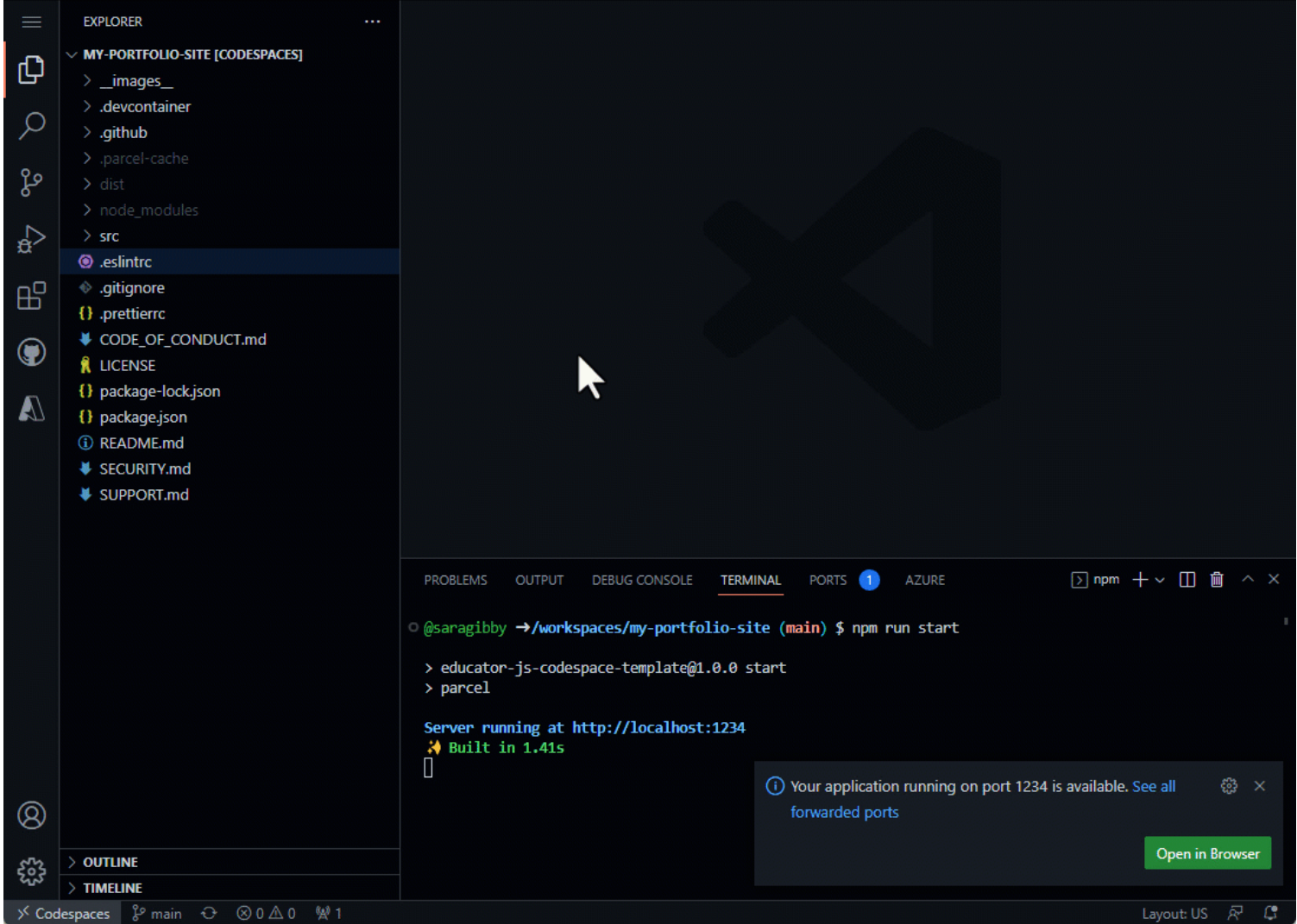
🤪 **Bonus:** [Setup a custom domain for your Azure Static Web App](#)

GitHub Pages

[GitHub Pages](#) allows you to host websites directly from your GitHub repository. This project is already set up for you to get your portfolio deployed to GitHub pages with minimal steps.

With your project open in Codespaces:

1. Open `package.json` and update the following values:
 - i. **homepage:** set to `http://{github-username}.github.io/{repo-name}`, where `github-username` is your GitHub username and `repo-name` is the what you named this portfolio repo when you created it
 - ii. **build-gh:** replace `github-username` with your GitHub username and `repo-name` with the repository name
2. Commit and push those changes to `package.json` to your GitHub remote repo.
3. Open a new terminal from the menu or by pressing `ctrl + shift + `` (or open top left menu, select "Terminal" and "New Terminal")
4. Within the terminal window run `npm run deploy`. This will first run the pre-deploy script to build the project, followed by the deploy script that will push those bundled files to a new branch on your repo (gh-pages) that will be used for you GitHub Pages site.
5. When completed, within your repo, go to Settings and Pages. There you will see that a page has been set up to for you against the gh-pages branch, and you should see the URL (that should match the "homepage" value you set in `package.json`)



🧐 Bonus: [Setup a custom domain for your GitHub pages site](#)

🏆 Customizing with Copilot

Below are 4 additional ways you can continue to customize your Codespace and portfolio site. We'll show you how to use Copilot to make suggestions for faster development, and help you learn more HTML, CSS and JavaScript along the way.

1. [Customize your Codespace](#)
2. [Update to smooth scroll to a section](#)
3. [Animate the desk photo](#)
4. [Add a new section](#)

🧐 Getting Copilot access

If you don't yet have Copilot access, you can [request it here](#). If you are a student, you can get Copilot for **FREE** [following these instructions](#).

To ensure that Copilot is running correctly, navigate to the extension tab in your Codespace and check the status of the Copilot extension. If the status is inactive, recreate the Codespace, and enable the extension to ensure that it is activated.

1. Customize your Codespace

Your environment comes with preinstalled extensions. You can change which extensions your Codespaces environment starts with, here's how:

1. Open file `.devcontainer/devcontainer.json` and locate the following JSON element **extensions**



```
"extensions": [
  "dbaeumer.vscode-eslint",
  "esbenp.prettier-vscode",
  "gitHub.copilot",
  "ms-vscode.azure-account",
  "ms-azuretools.vscode-azurestaticwebapps"
]
```

2. Let's add the `indent-rainbow` extension. To do this, go to the **extensions** list and add:



```
"oderwat.indent-rainbow"
```

What you did above was to add the unique identifier of an extension of the [indent-rainbow](#). This will let Codespaces know that this extension should be pre-installed upon startup.

To find the unique identifier of an extension:

- Navigate to the extension's web page, like so marketplace.visualstudio.com/items?itemName=oderwat.indent-rainbow
- Locate the *Unique Identifier* field under **More info** section on your right side.

★ COPILOT BONUS ★

In `devcontainer.json`, go to the following line in the `settings` values: `"emmet.triggerExpansionOnTab": true`. Add a comma at the end of the line and press enter. See what other settings Copilot recommends and if they'd help you in your Codespace.

💡 Learn more on how to [Personalize Your GitHub Codespace](#)

2. Update to smooth scroll to a section

In your site header you have links to each section below. Click one of these links and watch it scroll the page to that section. Not really a scroll, right?

Let's make this a better user experience by slowing that down so the user has a sense of what is happening, and where they are navigating to on the page.

1. Open `styles.css`, which is the stylesheet for your portfolio application. We need to add a style for `html`. If you look, you'll see right now `html` and `body` styles are being set together, with no style set for `scroll-behavior`. Let's give Copilot a prompt to add this for us.
2. Below the `html` and `body` styling prompt Copilot with a comment of: `/* add a smooth scroll */`
3. Copilot will then suggest the CSS below:



```
html {
  scroll-behavior: smooth;
}
```

4. Press the tab key to accept the suggestion. (Note: If you don't see this exact suggestion from Copilot, continue typing it to get the suggestion to appear.)

Your site should already be running in your Codespace, and the change will reload onto the page automatically. Click a link in the top header to see the smooth scroll in action.

3. Animate desk photo

Animations are a way you can easily add some motion to elements on your page to increase user interactivity and highlight items you want to make sure they notice. Let's animate the desk photo in the portfolio section.

1. Open your site's stylesheet, `styles.css` within your Codespace. Using Copilot, at the bottom of your `styles.css` prompt Copilot with the following comment:

```
/* add a slide in animation */
```



It should then suggest the following animation sequence for you. Press the `tab` key to accept, or continue to type until Copilot completes suggestions, and you have your animation ready to use.

```
@keyframes slideInLeft {  
  0% {
```



[README](#) [Code of conduct](#) [MIT license](#) [Security](#)



```
    100% {  
      transform: translateX(0);  
    }  
  }  
}
```

2. With the animation sequence defined, we can now tell our desk photo to animate itself with our new `slideIn` animation sequence. Open `Portfolio.jsx` and locate the `img` tag. You will see it utilizes inline CSS to set it's styling. Within it's style definition add the following:

```
animation: "1s ease-out 0s 1 slideIn";
```



Your image tag should look something like:

```
<img src={image} style={{ height: "90%", width: "100%", objectFit: "cover", animation: "1s ease-out 0s 1 slideIn" }} />
```



Your site should already be running in your Codespace, and the change will reload onto the page automatically. Scroll up and down the page and watch your desk photo slide onto the page.

★ COPILOT BONUS ★

Use Copilot to help you animate the scroll down arrow in your `Home` component to bounce up and down on your page.

Hint: In your `styles.css` file, use comments to start to tell Copilot what you want to do. See what is suggests, adjust your prompts, and see how it guides you to get your arrow to bounce.

4. Add a new section

We started you off with a few basic sections for your portfolio site, but you have creative freedom to make it your own and add new sections relevant to what you want on your site.

For an example, let's add an education section to your portfolio site.

1. Create a new component for the section within the `Components` folder. Add a new file called `Education.jsx`.
2. Let's have Copilot help get us started. Instead of prompting with a comment, start your `Education.jsx` file with:

```
import React from "react";
```



As you start typing it will pick up what you are doing and may offer an autocomplete of that line.

3. Press `enter` after the import line to have Copilot suggest code to create your `Education` component. Press `tab` to accept the solution or `ctrl + enter` to view all Copilot suggestions and select the one that works for you.

At minimum, you need an `const` defined and the `Education` component exported (example below). The rest is up to you. Experiment with Copilot, nudging it along with what you'd like it to build out.

```
import React from "react";

const Education = () => {
  return(
    <section className="light" id="education">
      <h2>Education</h2>
    </section>
  )
}

export default Education;
```



4. In `App.jsx` import your new `Education` component at the top by adding the following, and watch as Copilot starts to see what you are doing and give you auto-complete suggestions:

```
import Education from "../Components/Education";
```



5. Now add the `Education` component by going to a new line where you want it rendered. Watch Copilot already know you want to render the `Education` component. It should suggest the following that you can accept, and will render your new component:

```
<Education />
```



In your Codespace, your portfolio application should be running and will reload your site with the changes.

★ COPILOT BONUS ★

Now that you are familiar how Copilot can not only help you code faster, but give you suggestions to save you time looking up solutions.

Explore how you can use Copilot to help you:

- add Education to your top navigation

