Vercel Deploy

A comprehensive guide for beginners

October 6th, 2024

Created by Hrishikesha Kyathsandra

Table of Contents

1. Introduction

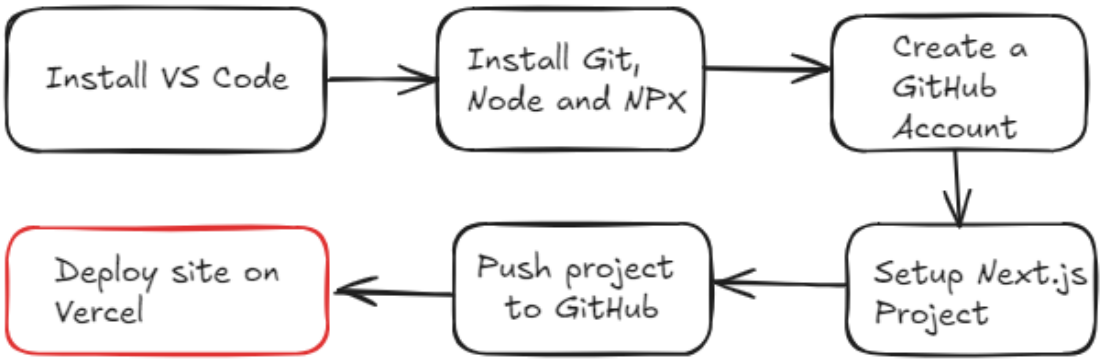
Welcome to your first guide on deploying a web project! In this manual, we'll walk you through creating and deploying a simple Next.js web app using Vercel, a popular platform for hosting modern web applications.

What You'll Learn:

- How to install **Node.js** (a key tool for building web applications) and **NPX** (a tool to manage Node.js projects).
- How to set up a simple project using **VS Code** (a code editor) and **Next.js**, a framework for building fast, dynamic web pages.
- How to use **GitHub** to store your code online.
- How to deploy your web project live using **Vercel**, so anyone on the internet can see it!

Who is This Guide For?

This guide is for absolute beginners. If you've never deployed a web project or don't even know what Node.js is, don't worry. We'll break everything down into easy steps. By the end of this guide, you'll have a live web page you can share with others.



2. Installing VS Code

Visual Studio Code (VS Code) is a free, lightweight code editor that's perfect for beginners. It makes it easy to write and manage code, and it works well with the tools we'll be using in this guide.

2.1 Why VS Code?

VS Code is popular among developers because it's simple to use, and it has built-in features to help you write organized code. Plus, it's free and works on all major operating systems.

Downloading VS Code

Follow these steps to download and install VS Code on your computer:

1. Go to the VS Code Website:

   You can begin the installation process by going **here**

2. Download the Installer:
   - For **Windows** users: Click the "Windows" download button.
   - For **macOS** users: Click the "macOS" download button.
   - For **Linux** users: Choose your linux Distribution
   - It should like something as seen in Figure 2.1

VS Code is like your coding studio—it gives you all the tools and space you need to create, debug, and organize your projects

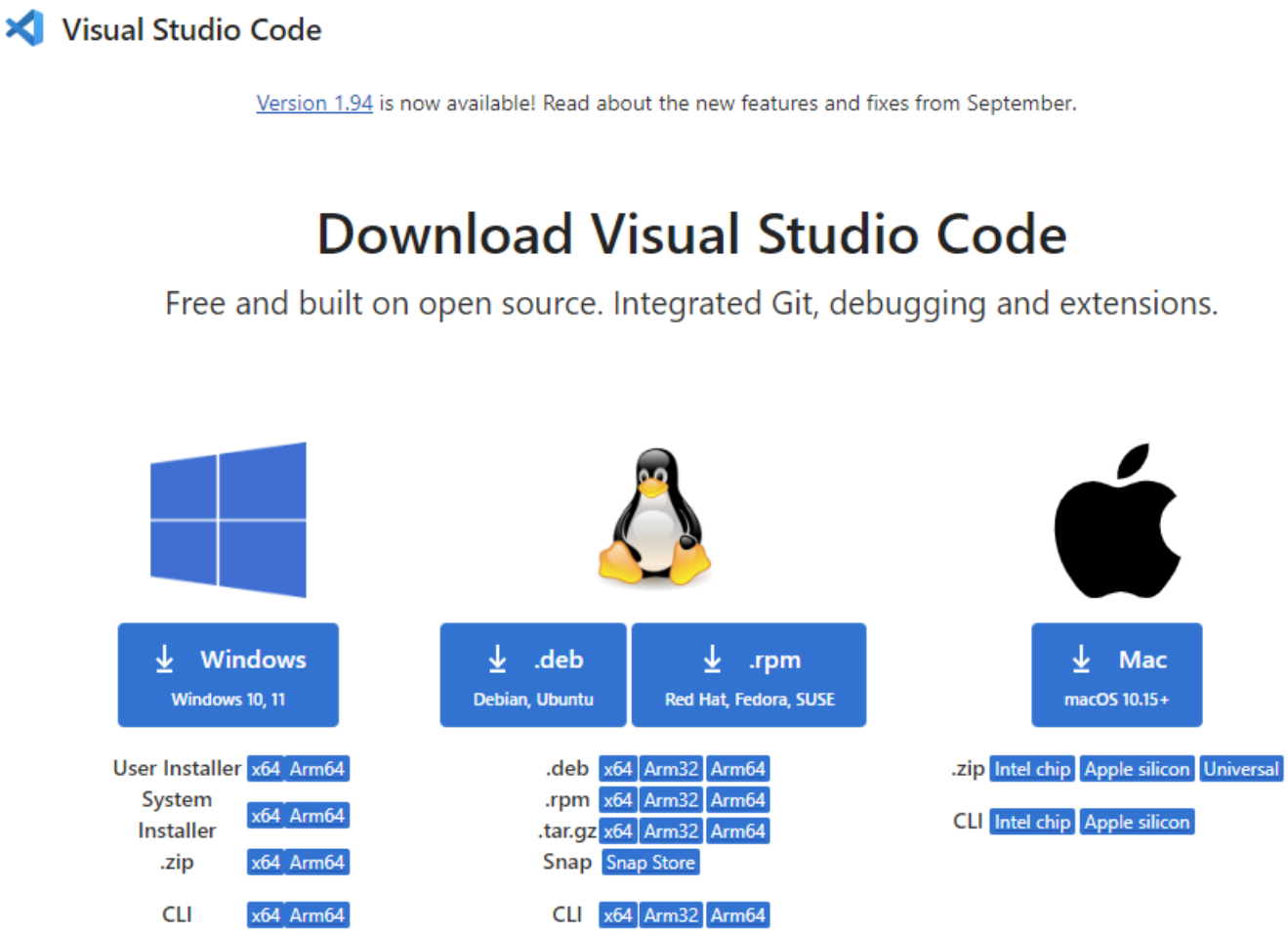


Figure 2.1: VS Code download page

3. Run the Installer:

   Once the download finishes, find the file in your downloads folder and double-click it to open the installer.

   Click **Next** for the prompts and agree with the default values.

4. Finish the Installation:

   Once the installation is complete, click **Finish.**

   You can now open VS Code on your system.

3. Installing Node.js and NPX

Before you can create a Next.js app, you need to install **Node.js**. Think of Node.js as the foundation for building web applications. Without Node.js, we can't run the code needed to make our website work.

What is Node.js?

Node.js is a software that allows you to run JavaScript outside a browser. It's widely used by developers to build web applications, and it includes a tool called **npm** (Node Package Manager), which lets you easily install tools and libraries like Next.js.

What is NPX?

NPX is a tool that comes with Node.js, and it allows you to quickly run commands and scripts from npm packages without installing them globally. We'll use NPX to quickly set up our Next.js project.

3.1 Downloading Node.js

Now that you know what Node.js is, let's install it.

Step-by-Step Instructions:

1. **Go to the Node.js website**:
   You can begin Node.js installation **here**. It should look something like **Figure 3.1.1.**

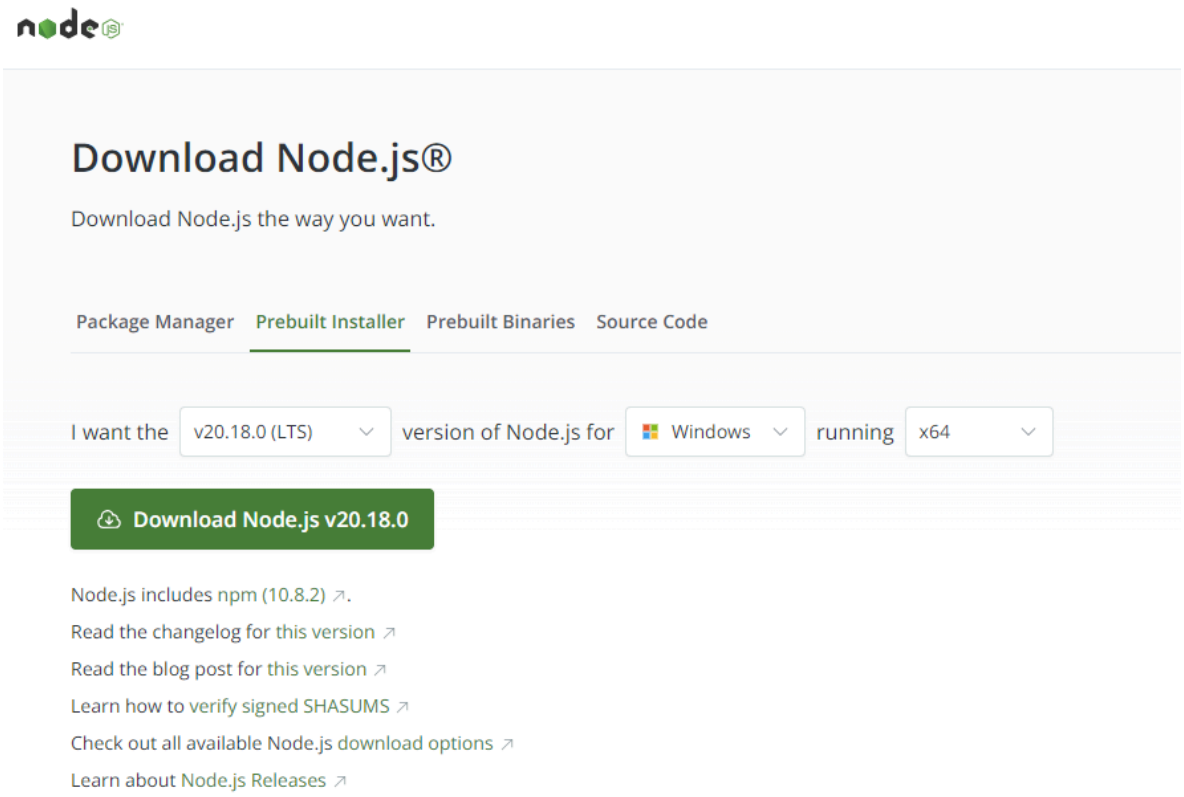


Figure 3.1.1 Node js Download Page

2. **Choose your Operating System and click Download**:
   On this page, you will see a list of different versions, operating systems and architecture. You can leave the version and architecture to its default, but make sure to choose your operating system. Look at **Figure 3.1.2** for more details.

NOTE:
Make sure to choose your PC's operating system in the drop down menu.

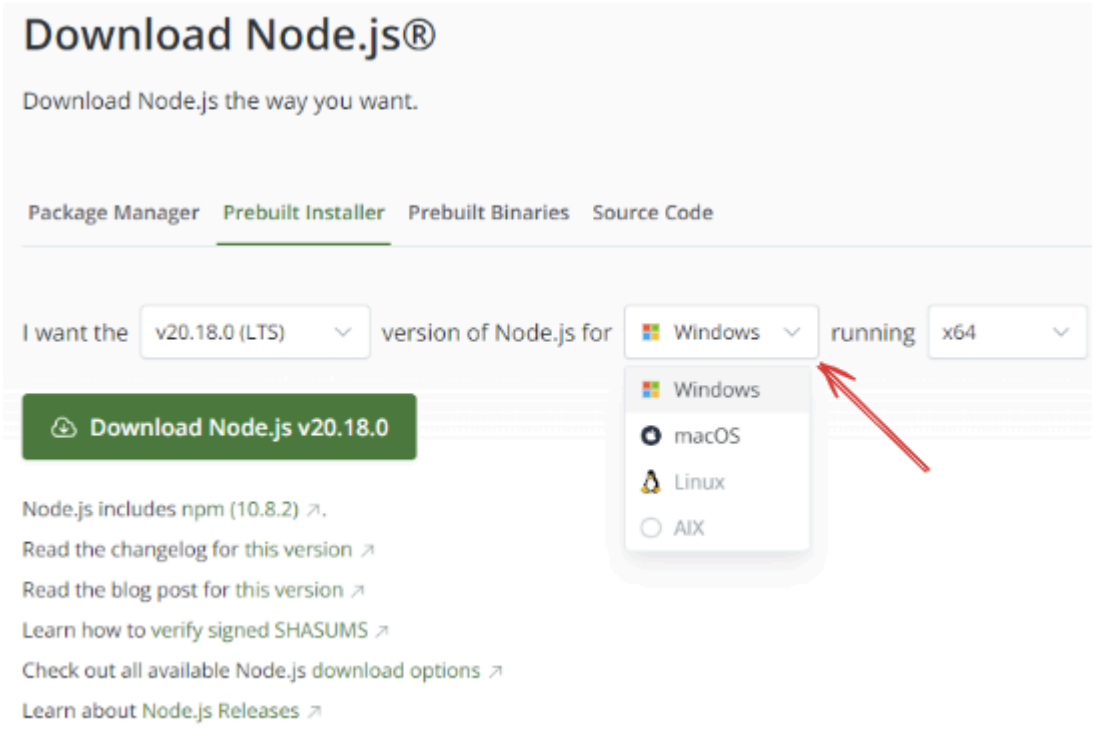


Figure 3.1.2: Dropdown to choose OS

3. **Run the installer**:
   After downloading the file, find it in your downloads folder and double-click it to run the installer.
4. **Follow the installation prompts**:
   The installer will guide you through the process. Make sure to accept the default settings, and click **Next** at each step.
5. **Finish the installation**:
   Once the installation is complete, click **Finish**.

Think of Node.js as the engine of your web app—it makes everything run behind the scenes

3.2 Verifying Installation

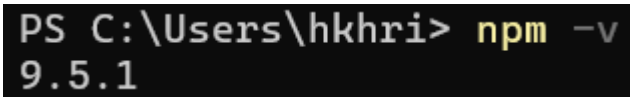
Now that Node.js is installed, let's verify it to make sure everything is working properly.

Step-by-Step Instructions:

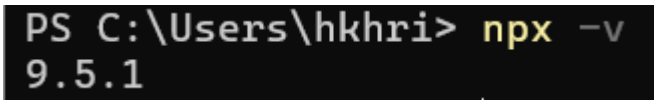
1. **Open your terminal/command prompt:**
   If you're using:
   - **Windows**: Press the Windows Key, type "Command Prompt", and press Enter.
   - **macOS**: Press Command + Space, type "Terminal", and press Enter.
   - **Linux**: Open your terminal application.

2. **Check Node.js version:**
   In the terminal, type the following command and press Enter:
   `node -v`
   This should show something like

3. **Check npm version:**
   Now, type this command and press Enter:
   `npm -v`



4. **Install NPX (if not installed automatically):**
   NPX typically comes with npm, so it should already be installed.
   To verify, type:
   `npx -v`



   If it's not installed, you can install NPX with the following command:
   `npm install -g npx` and then verify as seen above.

4. Installing Git

Git is an essential tool for managing the different versions of your code and collaborating with others. We use Git to keep track of changes in our project and to store our code on platforms like **GitHub**.

In this section, we'll explain why Git is important and show you how to install it on your computer.

4.1 Why Git?

Git is a **version control system** that helps you manage changes to your code. It's like a time machine for your project—you can save your work at different points and go back to any previous version whenever you want.

Why do we use Git?

1. **Backup your code**: Git keeps a history of all your changes, so you won't lose any progress.
2. **Easily collaborate**: If you're working with others on a project, Git helps you combine everyone's work without overwriting anyone else's code.
3. **Track your progress**: Git shows you what changes were made and when, making it easy to identify problems and revert back if needed.

Think of Git as a journal for your project—it keeps a detailed history of every change you make, so you can look back anytime.

4.2 Installing Git

Let's now install Git on your computer. The installation process is simple, and we'll guide you step-by-step for **Windows**, **macOS**, and **Linux**.

For Windows Users

1. **Go to Git for Windows Website**:
    - You can begin download by going **here**
    - Click the **Download** button to get the installer.
2. **Run the Installer**:
    - After downloading, open the file and run the installer.
    - During installation, accept the default settings. If you're unsure, just

keep clicking **Next**.

3. **Complete the Installation**:
    - When the installer is finished, click **Finish**.
4. **Verify the Installation**:
    - Open your **Command Prompt** (Windows Key + type "Command Prompt" and press Enter).

      Type the following command and press Enter:
      git --version

      You should expect to see something like this

```
PS C:\Users\hkhri> git --version
git version 2.43.0.windows.1
```

For macOS Users

1. **Open Terminal**:
    - Press Cmd + Space, type **Terminal**, and press Enter to open the terminal.
2. **Install Git Using Homebrew**:

   If you have Homebrew installed, run the following command
   brew install git

   If you don't have Homebrew installed, first install it by following the instructions from **here**, and then run the command above to install Git.

For Linux Users

The installation process depends on your Linux distribution:

1. **For Ubuntu/Debian-based distributions**:

   Open your terminal and run the following command:
   sudo apt install git

2. **For Fedora-based distributions**:

   Open your terminal and run the following command:
   sudo dnf install git

NOTE:
You can verify the installation on macOS and Linux the same way we did for windows.

5. Creating a GitHub Account

GitHub is a platform where you can store your code online, collaborate with others, and manage different versions of your projects. It works with Git to make sharing and managing your code easier. In this section, we'll guide you through signing up for GitHub and creating your first repository (a place to store your project).

5.1 Signing Up for GitHub

To start using GitHub, you need to create a free account.

Step-by-Step Instructions:

1. **Go to GitHub's Website**:
   - Visit **the GitHub Website**
2. **Sign Up for a New Account**:
   - On the GitHub homepage, click the **"Sign up"** button in the top-right corner.

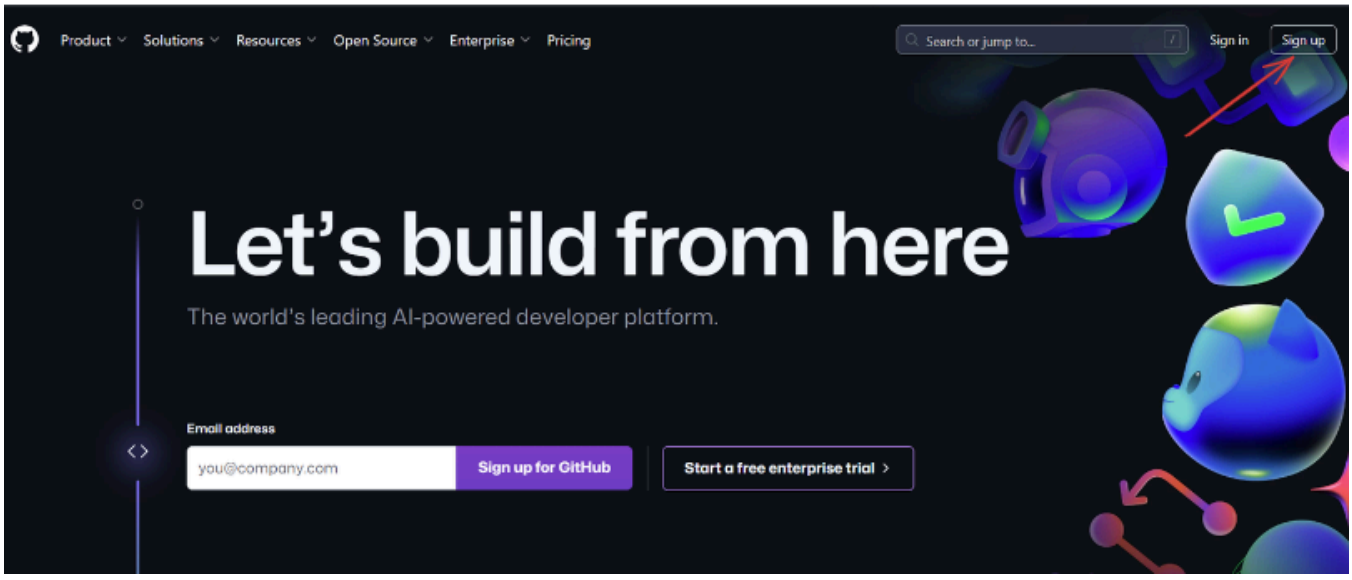


Figure 5.1.1: GitHub Homepage

3. **Fill Out Your Details**:
   - Enter your **email address**, **password**, and a **username**.
   - GitHub will ask if you want to receive updates—choose your

preference.
- Click **“Continue”** and then follow the rest of the verification steps and reCAPTCHA

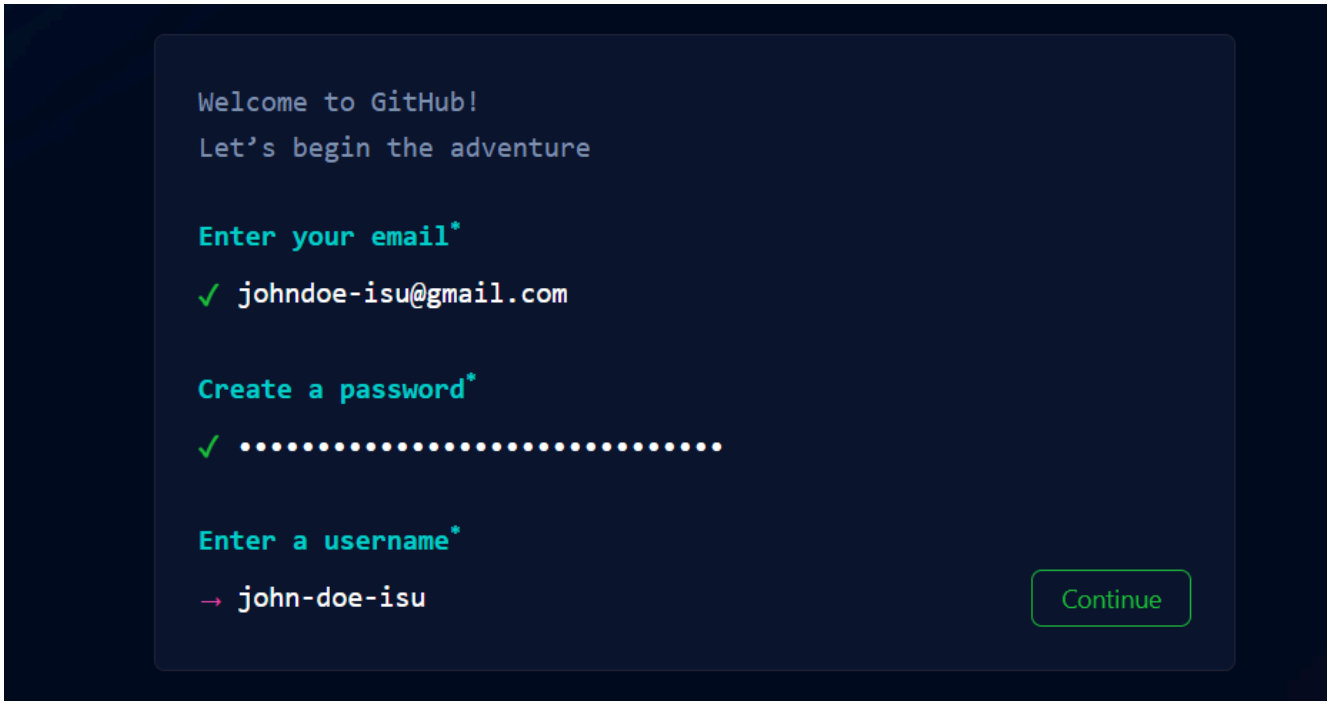


Figure 5.1.2: GitHub Sign Up Page

4. **You're In!**:
   - Once your account is set up, you’ll be taken to your **GitHub dashboard**.

Think of GitHub as an online library for your code—it stores your projects safely in one place, so you can access and share them from anywhere.

5.2 Creating a Repository

A **repository** is where your project lives in GitHub. It’s like a folder that contains all the code, files, and history for your project.

Step-by-Step Instructions:

1. **Go to Your Dashboard**:
    - Once you're logged into GitHub, click on your **profile icon** (top-right corner), then click **"Your repositories"** from the dropdown menu.

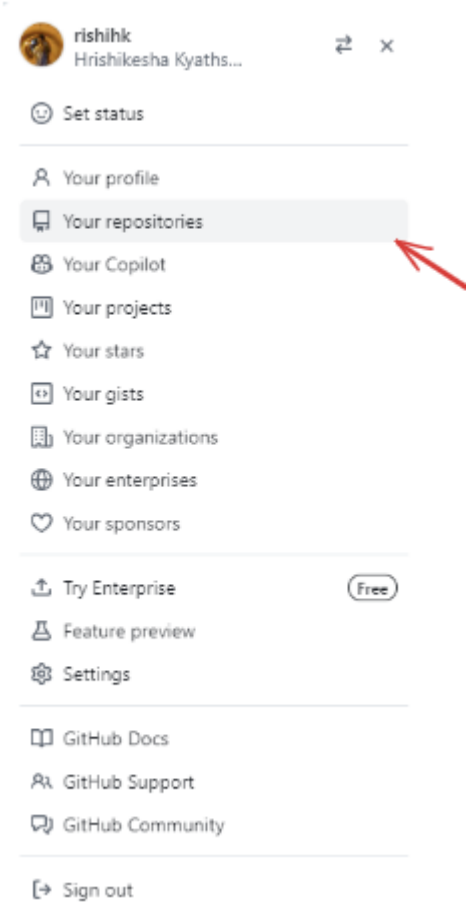


Figure 5.2.1: Repositories

2. **Create a New Repository**:
    - In the **Repositories** section, click the green **"New"** button on the right-hand side.

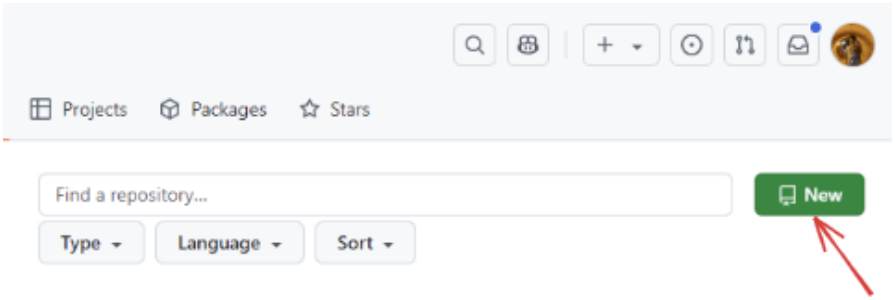


Figure 5.2.2: New Repository

3. **Name Your Repository**:
    - In the **Repository name** field, type a name for your project (e.g., my-portfolio).
    - I will be naming mine vercel-deploy-user-manual
    - You can add a description (optional).
    - Choose between making the repository **public** or **private**
4. **Add a README (Optional but Recommended)**:
    - Check the box that says **"Initialize this repository with a README"**.
    - The **README** file will give people information about your project when they visit your repository.
5. **Create the Repository**:
    - Once you've filled in all the fields, click the green **"Create repository"** button at the bottom. Refer to **Figure 5.2.1** for reference.

NOTE:
A public repository can be viewed by anyone whereas a private repository can only be viewed by you and people you invite.

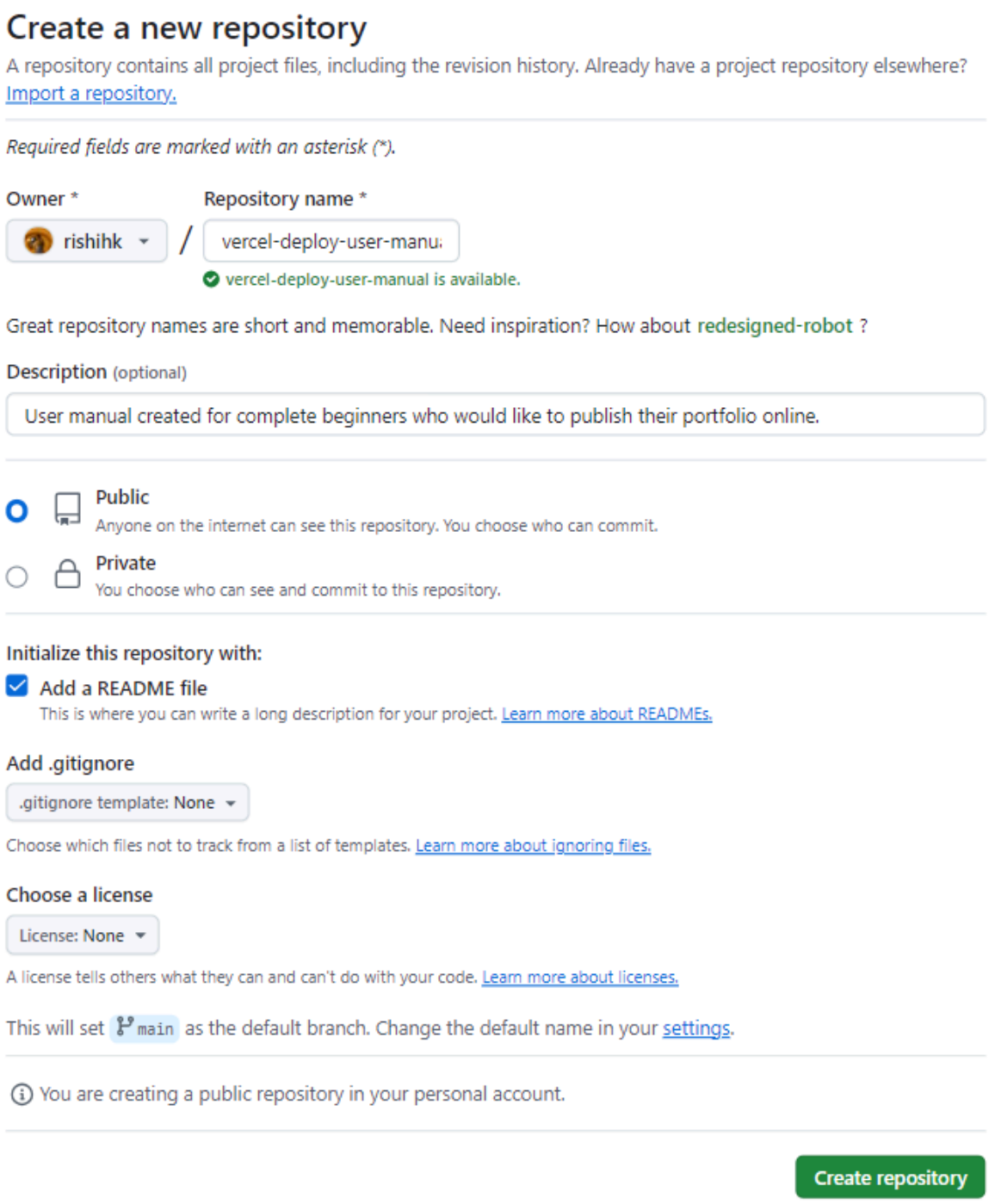


Figure 5.2.3: Repo Config

6. **Done!**:
    - You've just created your first GitHub repository! This is where you'll store and manage your project's code.

5.3 Cloning the Repository to Your Local Machine

Once you've created a repository on GitHub, the next step is to **clone** (copy) the repository to your computer so you can work on it locally. We'll guide you through the steps of cloning your repository to your **Desktop**.

Step-by-Step Instructions:

1. **Go to Your GitHub Repository**:
   - In your browser, go to **GitHub** and navigate to the repository you just created.
2. **Find the Clone Button**:
   - On the repository page, you'll see a green **Code** button near the top right. Click on it to reveal a dropdown with the URL for your repository.
3. **Copy the Repository URL**:
   - In the dropdown, under **Clone with HTTPS**, click the **copy icon** next to the URL. This will copy the repository's web address to your clipboard.

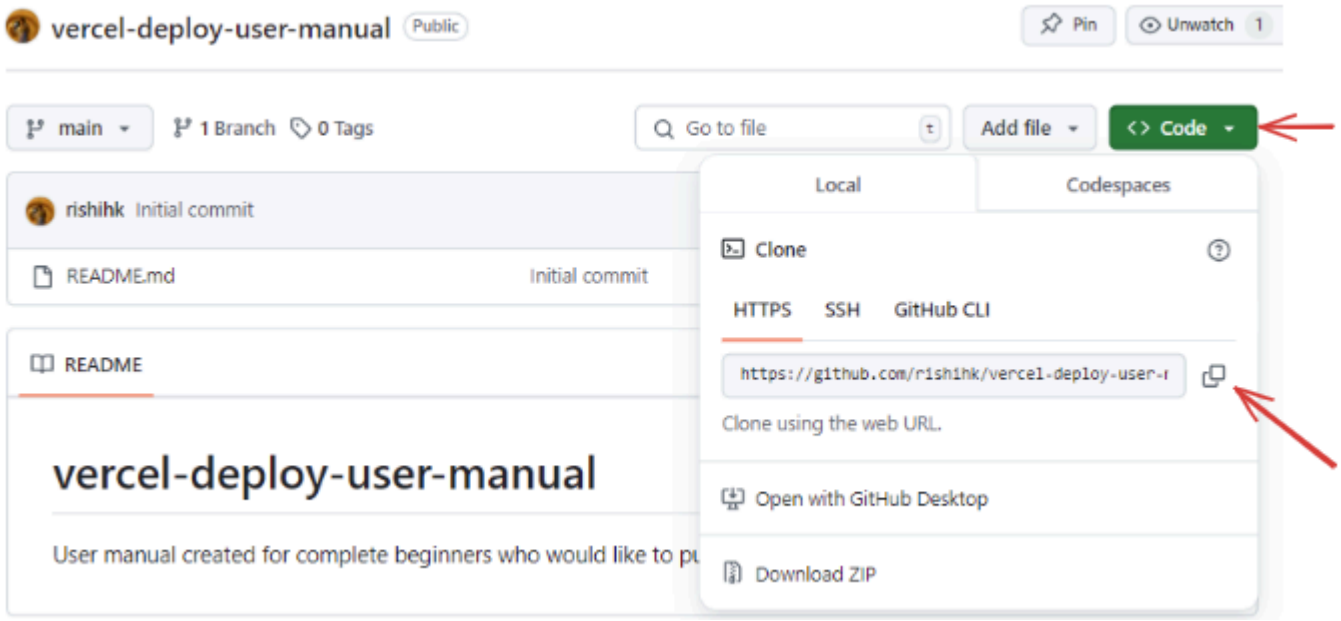


Figure 5.3.1: Clone Repo

4. **Open Terminal**:

   Windows: Press the Windows Key, type **Command Prompt**, and hit Enter.

macOS: Press Cmd + Space, type **Terminal**, and hit Enter.

Linux: Open your terminal application.

5. **Navigate to Your Desktop**:

   In your terminal, type the following command to move to your desktop folder:

   cd Desktop

   This will ensure your repository is cloned to your desktop.

6. **Clone the Repository**:

   Now, use the git clone command to clone the repository. Type the following:

   git clone https://github.com/yourusername/your-repo-name.git

   This command will create a folder on your desktop with all the files from your GitHub repository. The output should look like something as seen in **Figure 5.3.1**

NOTE:
Replace `yourusername` and `your-repo-name` in the link above with your actual GitHub username and the repository name you created.

Figure 5.3.2: CLI Clone

7. **Check Your Desktop**:
   - Go to your desktop, and you'll see a new folder with the same name

as your GitHub repository.
- You can now open this folder in **VS Code** and start working on your project!

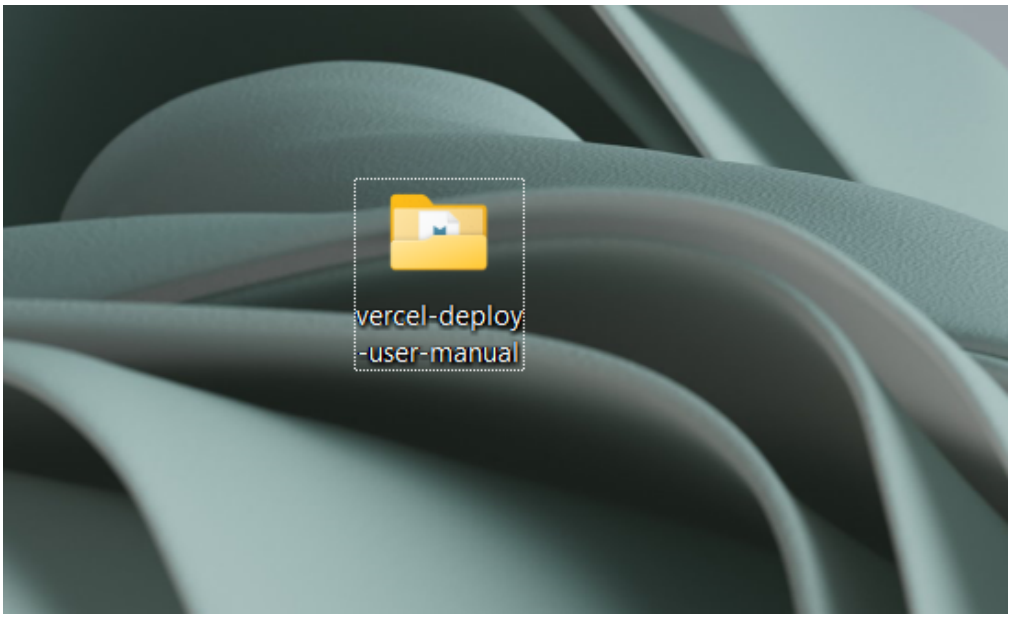


Figure 5.3.3: Repo on Desktop

The cd command stands for change directory. If doing cd Desktop shows an error saying cannot find path, try doing cd OneDrive/Desktop.

5.4 Opening the Folder in VS Code

Now that you've cloned your repository to your desktop, let's open it in **VS Code** so you can start working on your project.

Step-by-Step Instructions:

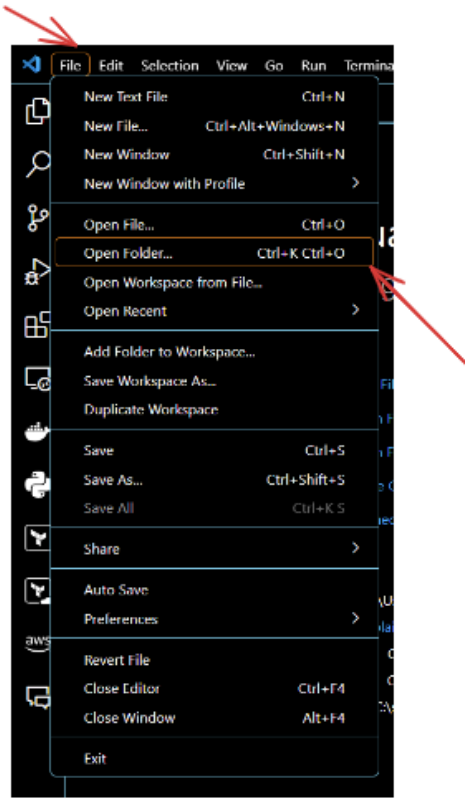
1. **Open VS Code**:
   - Launch **VS Code** by either:

     Searching for **"Visual Studio Code"** in your start menu (Windows), launchpad (macOS), or application menu (Linux).

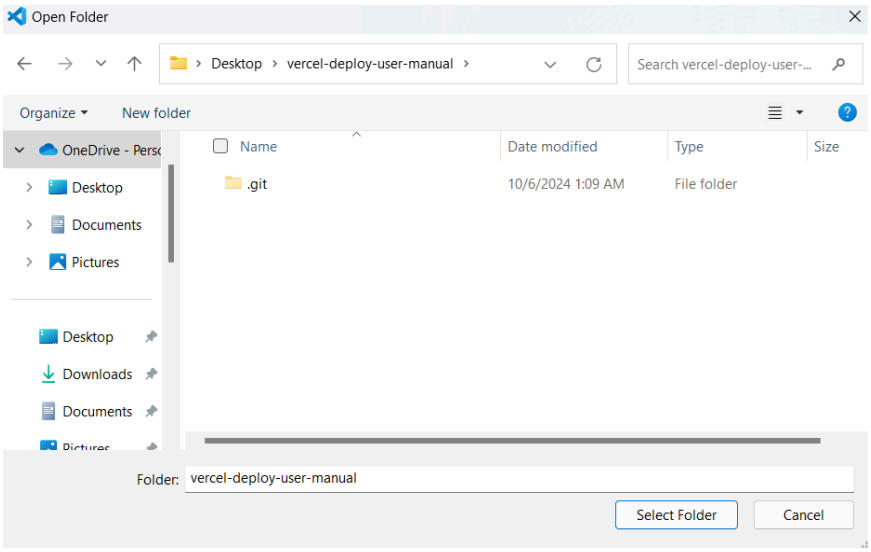
     Double-clicking the **VS Code** icon on your desktop.

2. **Open the Repository Folder**:

- In VS Code, click on **File** in the top-left corner, then select **Open Folder...**



3. **Select the Repository Folder**:
   - A file explorer window will pop up. Navigate to your **Desktop** and find the folder with the name of your cloned repository (the one you just cloned in the previous step).
   - Select the folder and click Select folder or Open



4. **You're All Set!**:
    - Your repository is now open in VS Code. You should see the files listed on the left-hand side (in the Explorer panel). You can now start editing your code or adding new files to your project.

6. Setting Up a Simple Next.js Project

In this section, we'll guide you through setting up a simple **Next.js** project. Next.js is a popular framework that makes building web applications faster and easier. We'll explain what it is, how to set up your project, and how to run your development server.

6.1 What is Next.js?

Next.js is a tool that helps you build websites quickly and easily. It's based on **React**, a popular way to create websites using JavaScript. Next.js makes your website faster and more efficient by handling some tasks automatically, like loading pages quicker and improving performance.

Why Use Next.js?

- **Automatic pages**: Every time you create a new file in the project, it automatically becomes a new page on your website.
- **Faster websites**: It helps your website load faster by preparing parts of the site before they're sent to visitors.
- **Easy setup**: You can get a project running with just a few simple commands.

Think of Next.js as a tool that helps you build modern websites quickly and efficiently!

6.2 Creating a Next.js Project with NPX

We'll now create your first **Next.js** project using **NPX**. NPX allows us to quickly run commands without installing them globally, making setup fast and easy.

Step-by-Step Instructions:

1. **Open VS Code and Terminal**:
    - Open **VS Code**, and if it's not already open, open the terminal by:

      Windows/Linux: Press `Ctrl + `.

      macOS: Press `Cmd + `.

2. **Navigate to Your Cloned Repository**:
    - If you cloned the repository onto your Desktop as we had done earlier, navigate to it by typing the command below and hitting Enter
      cd Desktop/your-repo-name

NOTE:
Replace `your-repo-name` with your actually repository name

3. **Run the Create Next.js App Command**:
    - In the terminal, type this command and hit enter to set up a new Next.js project.
      npx create-next-app@latest
    - You'll be prompted to answer several configuration questions.
4. **Answer the Setup Questions**:
    - Type y and hit Enter when prompted for installation permission.
    - **Project Name:** Type in a name for your project and hit Enter. (Maybe something like my-website) I will be naming mine tutorial-website.
    - **TypeScript**: Choose **No** (For simplicity)
    - **ESLint**: Choose **No** (For simplicity)
    - **Tailwind CSS**: Choose **No** (For simplicity)
    - **src/ directory**: Choose **Yes**
    - **App Router:** Choose **Yes**
    - **import alias**: Choose **No** (For simplicity)
    - You can refer to **Figure 6.2.1** for more reference.

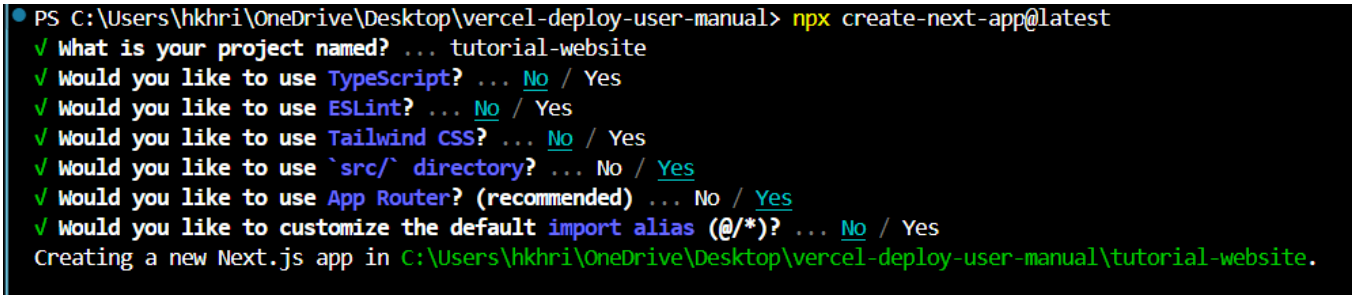


Figure 6.2.1

NOTE:
You can switch between the Yes or No options using the left and right arrow keys on your keyboard.

5. **Wait for the Setup to Complete**:
    - NPX will now install all the necessary dependencies and set up your project folder. This may take a minute or two.
6. **Move into Your Project Folder**:
    - Once the setup is complete, navigate into the new project folder with: `cd your-app-name`

NOTE:
Replace `your-repo-name` with the name you chose for your application in the project configuration step.

7. Now, you're ready to start building your Next.js website!

6.3 Running the Development Server

After creating your Next.js project, you'll want to see it in action! Let's start the **development server** so you can view your website in your browser.

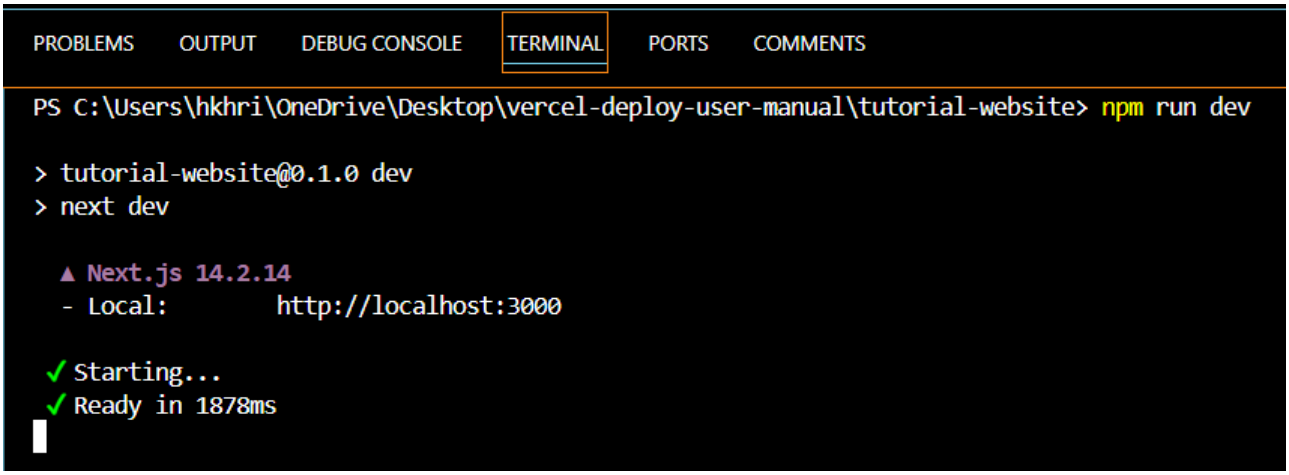
Step-by-Step Instructions:

1. **Start the Development Server**:

    In the terminal (while inside your project directory), type the following command:

npm run dev

This starts the development server, and your project will be accessible locally.



2. **Open Your Browser**:

   Open your web browser and go to the following URL: http://localhost:3000

   - You should see the default **Next.js welcome page**. **(Figure 6.3.1)**

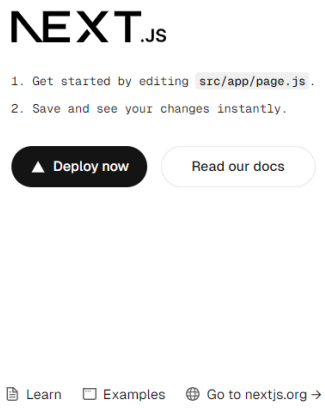


Figure 6.3.1

7. Modifying the Project

In this section, we'll guide you through modifying your **Next.js** project. You'll learn how to navigate the project structure, edit the homepage, use **MUI components**, view your changes locally, and push your code to **GitHub**.

7.1 Navigating the Project Structure

When working with Next.js using the **App Router** and **src** folder structure, here is the key folder and file.

1. src/app/ folder:
    - This is where your page files live. The main file you'll be editing for your homepage is located at src/app/page.js.
    - Each file in the app folder represents a different route (page) on your website.

7.2 Editing src/app/page.js

To modify the homepage, follow these steps to update the content.

Step-by-Step Instructions:

1. **Open src/app/page.js:**
    - In **VS Code**, navigate to src/app/ and open **page.js.** This is the file that controls your homepage.
2. **Access the Template Code**:
    - We've provided the code you'll need to paste into this file. You can find it **here.**
3. **Replace the Default Code**:
    - Once there, copy the entire code provided, and paste it into **page.js**. The code includes everything needed for the base page.
4. **Install MUI**:

    Before running your project, you'll need to install **MUI** and its icons. Run the following command in your terminal and hit Enter.

    npm install @mui/material @mui/icons-material @emotion/react @emotion/styled

Think of MUI as a toolkit that helps you easily build stylish websites with ready-made components like buttons and cards, saving you time and effort!

5. **Save the File**:
    - Once you've pasted the code and installed **MUI**, save the file by pressing **Ctrl + S** (Windows/Linux) or **Cmd + S** (Mac).

7.3 Viewing the Changes Locally

Now that you've edited your homepage, let's view the changes locally.

Step-by-Step Instructions:

1. **Start the Development Server**:

    In the terminal, run the following command to start your development server:

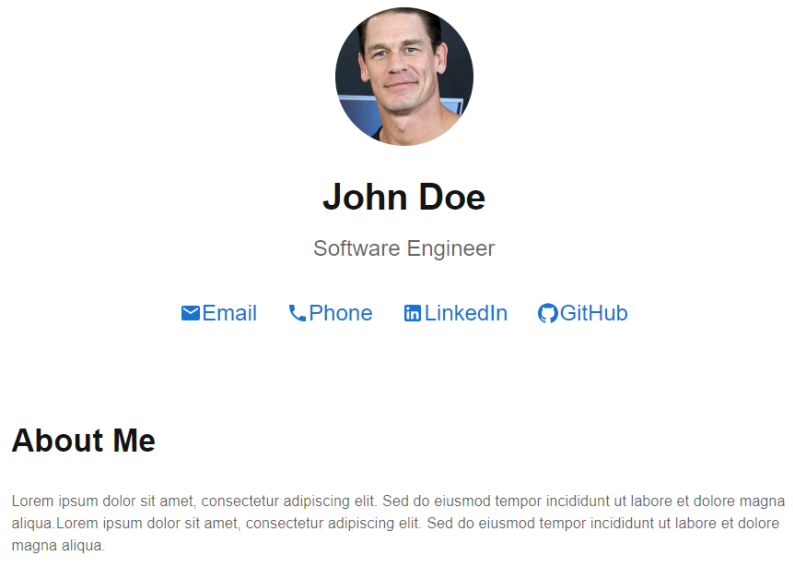
    ```
    npm run dev
    ```

    This command will start the Next.js development server, and your project will be available for viewing.

2. **Open the Project in Your Browser**:

    Open your web browser and go to:
    http://localhost:3000

    - You should now see the updated homepage.



3. **Real-Time Updates**:
    - Any changes you make to **page.js** will automatically refresh in the browser, so you can see the updates instantly.

NOTE:
Though you can see your website on localhost:3000, sharing this link with others does not mean they can view it too. localhost is **local** to your machine. This is why we want to deploy it so that everyone can access it.

7.4 Pushing the Code to GitHub

Once you're happy with your changes, it's time to push the updated code to **GitHub** so that it's saved and accessible online.

Step-by-Step Instructions:

1. **Check the Status of Your Changes**:

   In the terminal, type the following command to see the changes you've made:
   git status

2. **Add the Changes**:

   Stage the modified files by running:
   cd ../ (To get into the root)
   git add .

3. **Commit the Changes**:

   After staging the changes, commit them with a message:

   git commit -m "add base portfolio code"

4. **Push the Changes to GitHub**:

   Finally, push the committed changes to your GitHub repository:

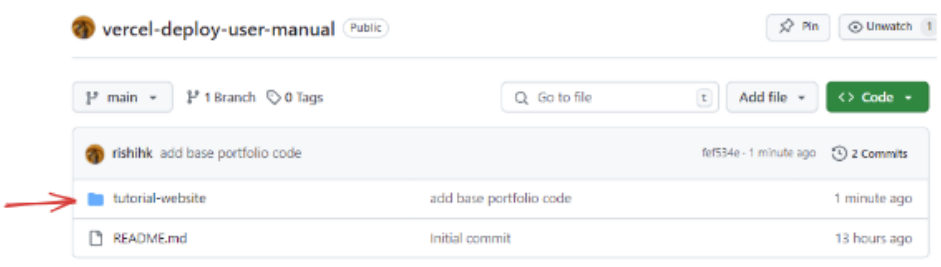
   git push origin main or just git push

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  COMMENTS

PS C:\Users\hkhri\OneDrive\Desktop\vercel-deploy-user-manual> git add .
warning: in the working copy of 'tutorial-website/.gitignore', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'tutorial-website/README.md', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'tutorial-website/jsconfig.json', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'tutorial-website/next.config.mjs', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'tutorial-website/package-lock.json', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'tutorial-website/package.json', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'tutorial-website/src/app/globals.css', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'tutorial-website/src/app/layout.js', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'tutorial-website/src/app/page.js', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'tutorial-website/src/app/page.module.css', LF will be replaced by CRLF the next time Git touches it
PS C:\Users\hkhri\OneDrive\Desktop\vercel-deploy-user-manual> git commit -m"add base portfolio code"
[main fef534e] add base portfolio code
 14 files changed, 2141 insertions(+)
 create mode 100644 tutorial-website/.gitignore
 create mode 100644 tutorial-website/README.md
 create mode 100644 tutorial-website/jsconfig.json
 create mode 100644 tutorial-website/next.config.mjs
 create mode 100644 tutorial-website/package-lock.json
 create mode 100644 tutorial-website/package.json
 create mode 100644 tutorial-website/public/images/cena.jpg
 create mode 100644 tutorial-website/src/app/favicon.ico
 create mode 100644 tutorial-website/src/app/fonts/GeistMonoVF.woff
 create mode 100644 tutorial-website/src/app/fonts/GeistVF.woff
 create mode 100644 tutorial-website/src/app/globals.css
 create mode 100644 tutorial-website/src/app/layout.js
 create mode 100644 tutorial-website/src/app/page.js
 create mode 100644 tutorial-website/src/app/page.module.css
PS C:\Users\hkhri\OneDrive\Desktop\vercel-deploy-user-manual> git push
Enumerating objects: 23, done.
Counting objects: 100% (23/23), done.
Delta compression using up to 8 threads
Compressing objects: 100% (19/19), done.
Writing objects: 100% (22/22), 192.33 KiB | 5.20 MiB/s, done.
Total 22 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/rishihk/vercel-deploy-user-manual.git
   40d3dd1..fef534e  main -> main
PS C:\Users\hkhri\OneDrive\Desktop\vercel-deploy-user-manual>
```

NOTE:
You can ignore the warnings. It is just a message by GitHub explaining how it manages new lines.

5. **Check Your GitHub Repository**:
    - Once the push is complete, you can go to your GitHub repository to see your updated code.



8. Deploying on Vercel

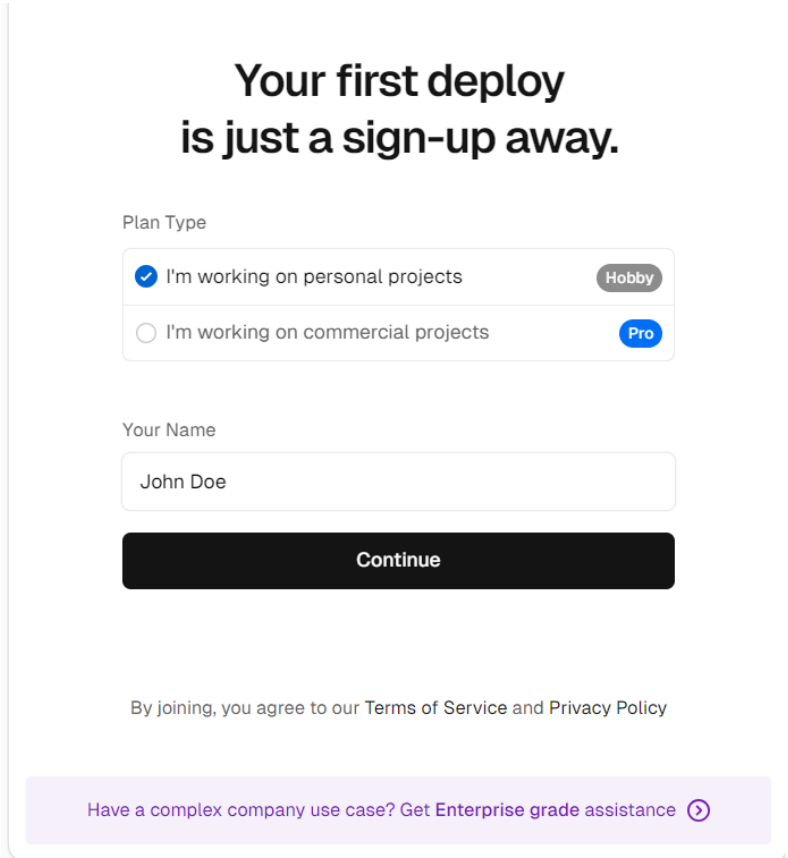
Once your project is ready, it's time to deploy it live so others can see it on the internet! Vercel makes it easy to deploy **Next.js** projects, and you can also customize your domain to something like **yourname.vercel.app**, as long as it's available.

8.1 Signing Up for Vercel

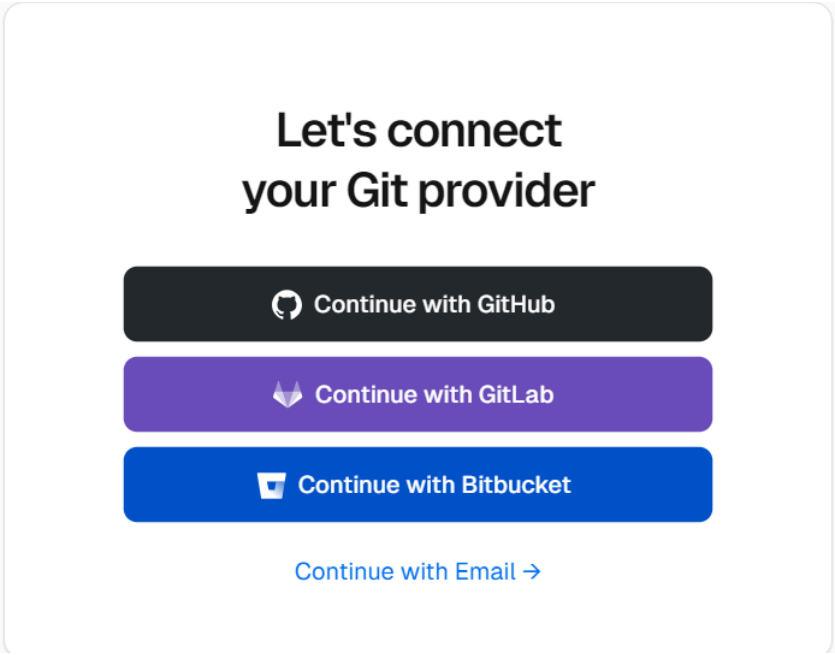
To deploy your project, you'll first need to sign up for an account on **Vercel**.

Step-by-Step Instructions:

1. **Go to Vercel**:
    - Go **here** to begin signing up.
2. **Create an Account**:
    - Choose the free Tier and enter your name



Sign up through your GitHub account (**recommended)**



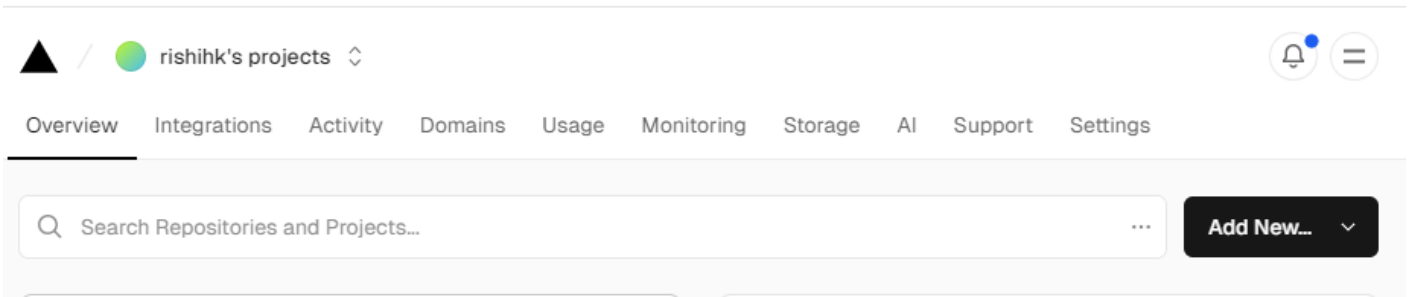
3. **Authorize GitHub (if using GitHub)**:
    - If you choose to sign up with **GitHub**, Vercel will ask you to authorize access to your GitHub repositories. Allow the authorization so Vercel can deploy projects directly from your GitHub account.

8.2 Connecting Vercel to GitHub

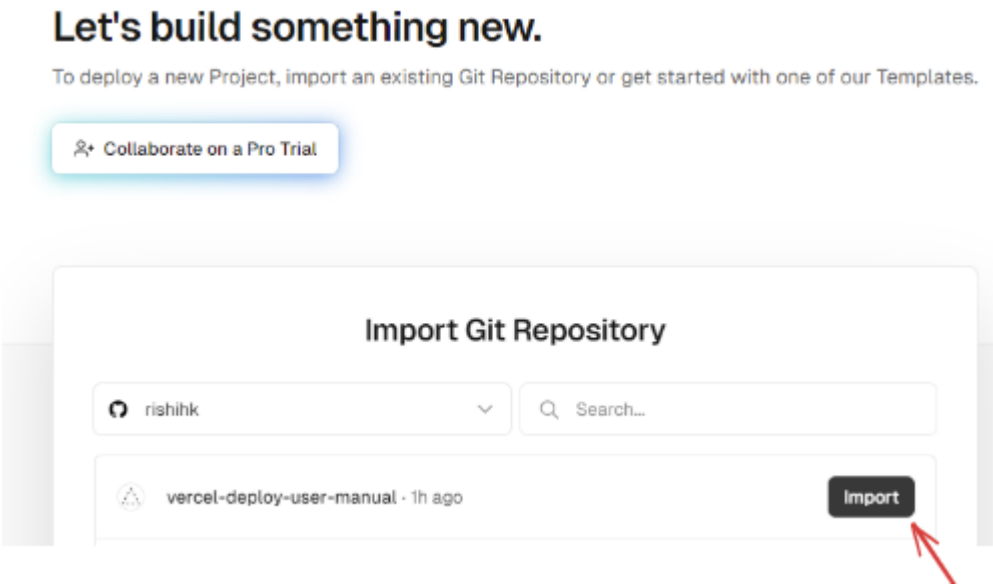
Next, we'll connect your **GitHub repository** to **Vercel** so that your project is automatically deployed whenever you push updates.

Step-by-Step Instructions:

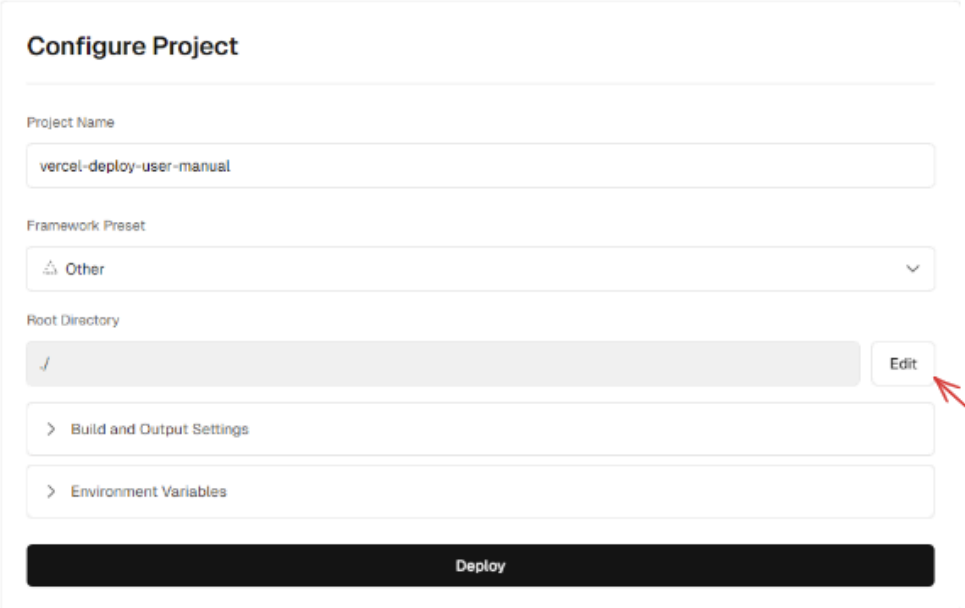
1. **Go to the Vercel Dashboard**:
    - After signing in, you'll be taken to the Vercel dashboard. Here, you can manage all your deployed projects.
2. **Click on "New Project"**:
    - On the dashboard, click **Add New** and then **Project**

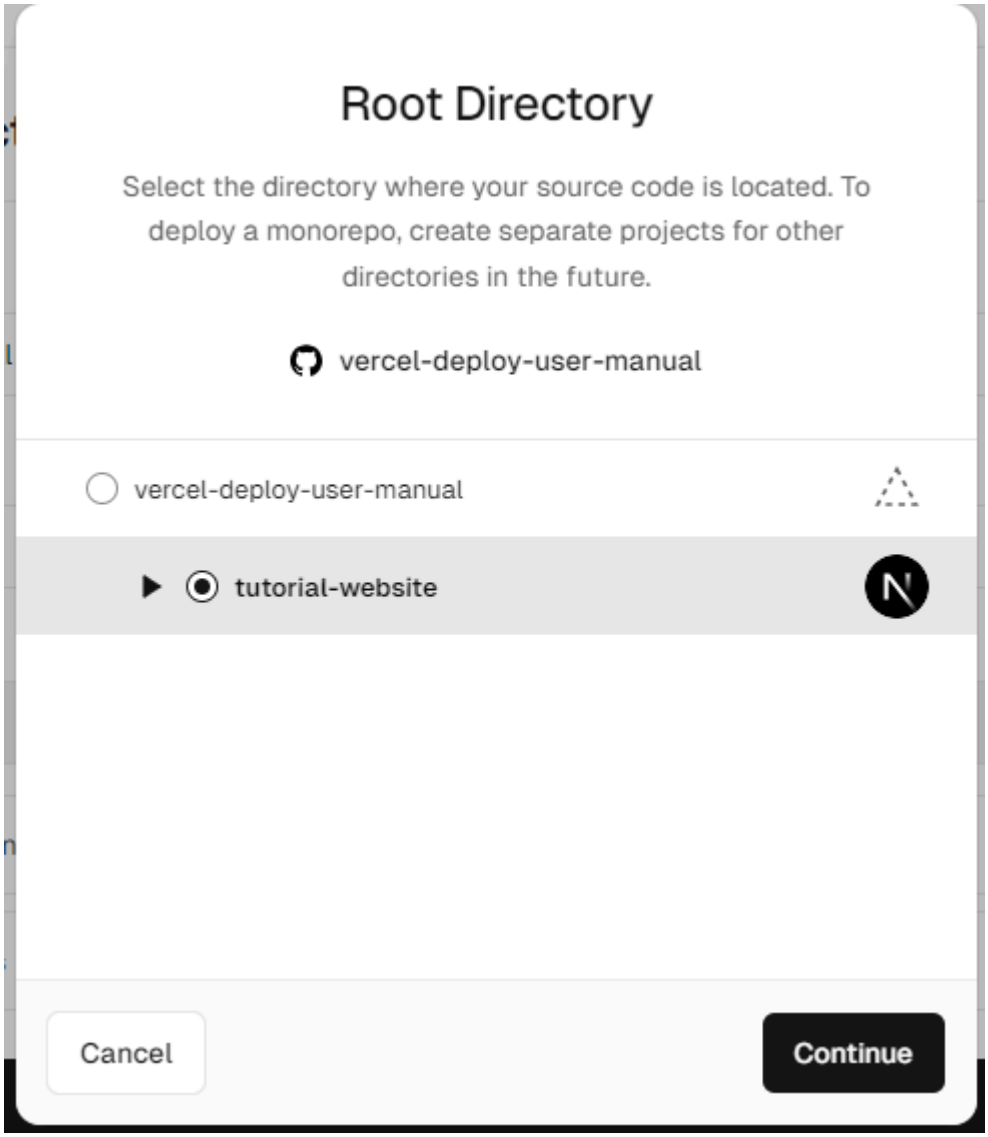


3. **Import Your GitHub Repository**:
    - Vercel will show a list of your GitHub repositories. Find the repository for your **Next.js** project and click **Import**.

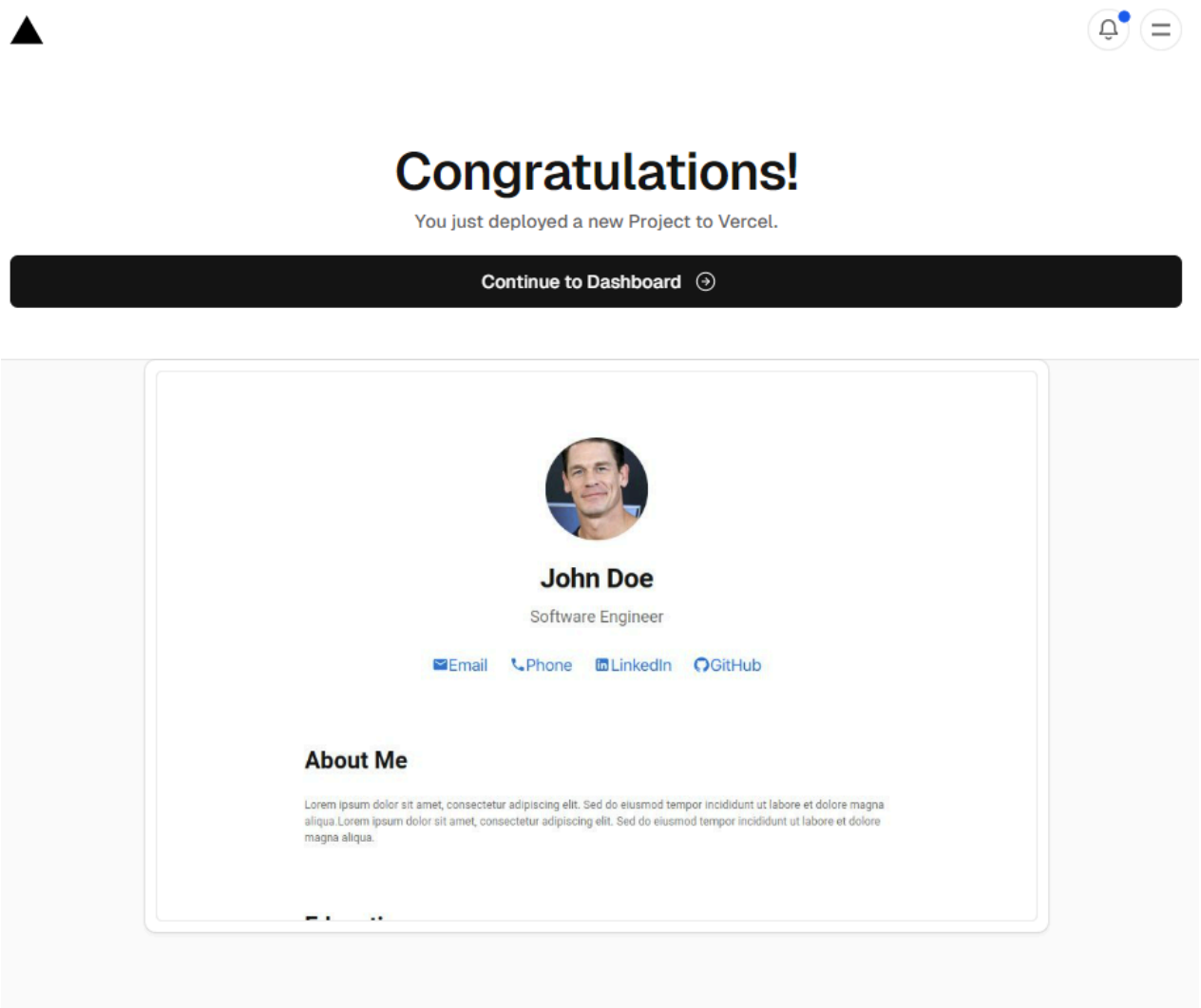


4. **Configure the Project**:
    - Edit the root directory to point to the next-js app.
    - Click Continue.





5. **Click "Deploy"**:
    - Once the configuration looks good, click **Deploy**. Vercel will start building and deploying your project.



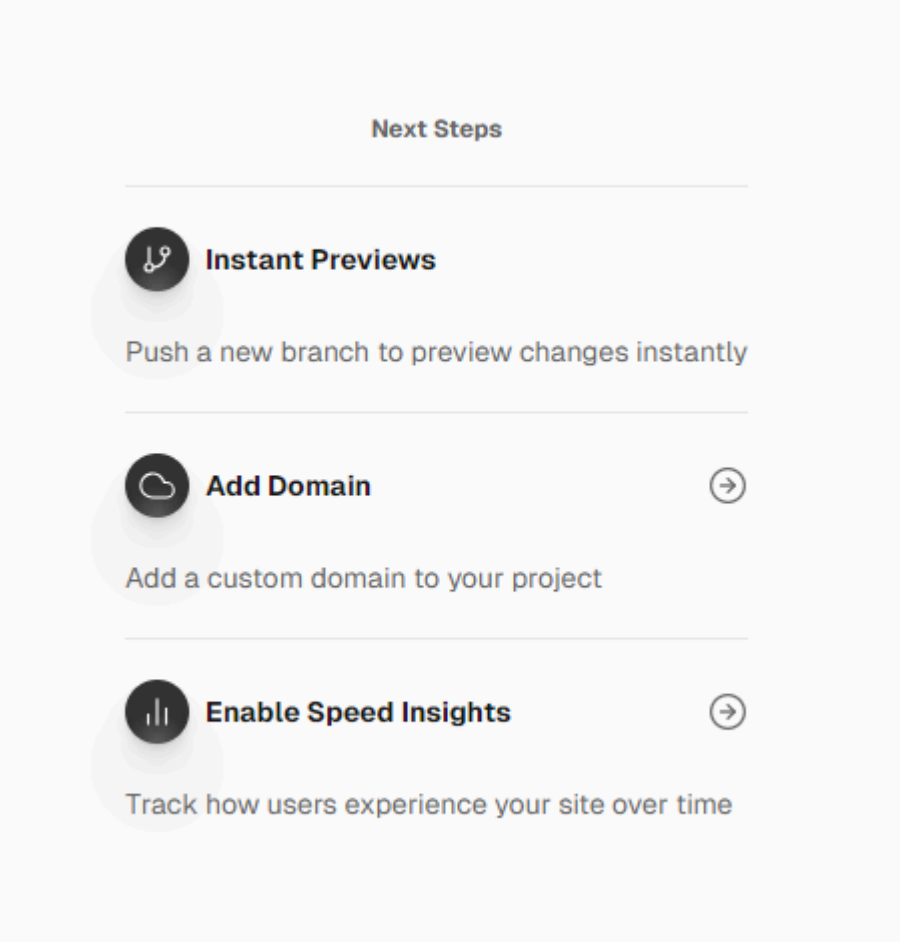
8.3 Changing the domain name

Now that the deployment process has completed, let's look at the next steps of the process and working on changing the domain name to whatever you like

Step-by-Step Instructions:

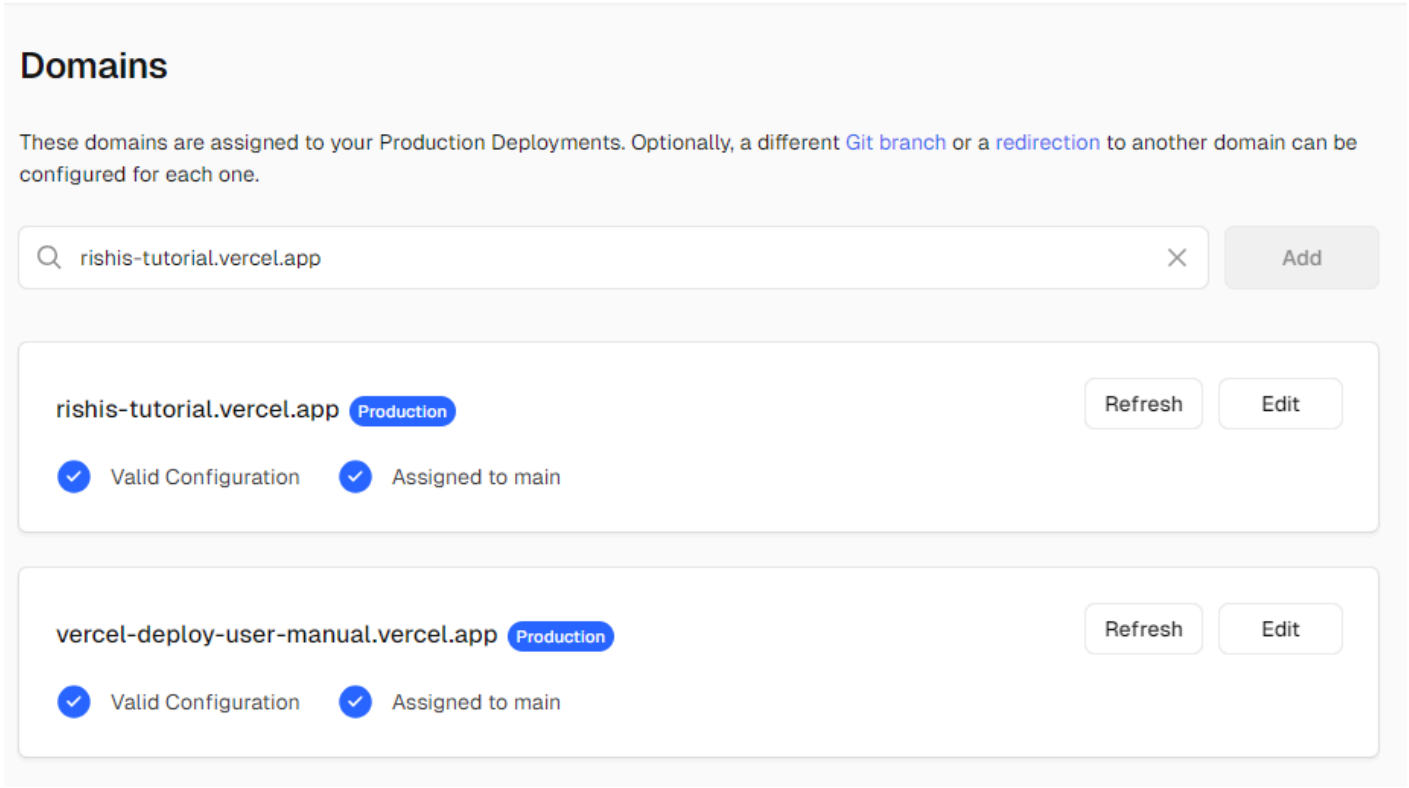
1. **Customizing Your Domain Name**:
    - You can customize your live URL to something more personal (e.g., **yourname.vercel.app**) if the name is available. To do this:
        1. Scroll down on the same congratulations page

2. Click on the section that says add domain.



3. Type in a valid domain name (unused) and click **Add**

NOTE:
If you would like a domain like yourname**.com**, instead of yourname.**vercel.app** you will have to buy the domain on google and then configure it on vercel to use it.



8.4 Viewing the Live Project

Once the deployment is complete, you can visit your live website!

Step-by-Step Instructions:

1. **Go to the Live URL**:
    - Click on the new domain you just added.
    - You should see your fully functioning website, including the homepage you've customized!
    - Here is **mine**
2. **Test Your Website**:
    - Navigate through your site, click on different links, and make sure everything works as expected.

9. Conclusion

Congratulations! 🎉 You've successfully created a **Next.js** project, customized it, and deployed it live using **Vercel**, with your own personalized domain name!

Here's a quick recap of what you've accomplished:

- Installed tools like VS Code, Git, and NPM to begin your web developing journey.
- Created a GitHub account and made a repository for your website.
- Set up a **Next.js** project from scratch.
- Modified a Next.js project
- Deployed your project live with **Vercel**, making it accessible to the world.
- Customized your live URL to make it more personal and professional (e.g., **yourname.vercel.app**).

10. Next Steps:

- You can continue improving your website by learning more about **Next.js, React, JavaScript** and code more features, modify the design, or create additional pages.
- Whenever you push changes to **GitHub**, Vercel will automatically rebuild and redeploy your site. You **never** have to worry about the deployment part now. Just your site!
- You can check out my official Portfolio site **here** for inspiration.

Your website is now live and ready to be shared with anyone. Keep experimenting and building!

Works and References Cited

1. “Develop. Preview. Ship. For the Best Frontend Teams – Vercel.” *Vercel.com*, vercel.com/.
2. Git. “Git - Documentation.” *Git-Scm.com*, 2019, git-scm.com/doc.
3. “Git - Downloading Package.” *Git-Scm.com*, 2024, git-scm.com/downloads/win.
4. GitHub. “GitHub.” *GitHub*, github.com/.
5. “GitHub.com Help Documentation.” *Docs.github.com*, docs.github.com/en.
6. “Homebrew.” *Homebrew*, brew.sh/.
7. “Node.js — Download Node.js®.” *Nodejs.org*, 2024, nodejs.org/en/download/prebuilt-installer. Accessed 6 Oct. 2024.
8. Vercel. “Getting Started | Next.js.” *Nextjs.org*, 2024, nextjs.org/docs.
9. Visual Studio Code. “Visual Studio Code.” *Visualstudio.com*, 14 Apr. 2016, code.visualstudio.com/Download.