Completing my Portfolio

The tech stack being used is as follows:

**🌟 Varun's Portfolio Tech Stack (2025)**

**🖼️ Frontend**

| **Tech** | **Purpose** |
| --- | --- |
| **React** | Core UI framework (component-based, powerful, flexible) |
| **Vite** | Super-fast build tool and dev server (replaces Create React App) |
| **Tailwind CSS** | Utility-first CSS framework (no need to write manual CSS) |
| **Framer Motion** | Animation library for smooth transitions, scroll/hover effects |
| **Spline** | Embeddable 3D objects and interactive visuals |
| **Heroicons / Lucide** | Modern, open-source icon sets for buttons and UI elements |
| **JSX + ES6** | The language you’ll write everything in (JavaScript + XML style markup) |

**🧠 Content Management**

| **Option** | **Purpose** |
| --- | --- |
| **Static JS/JSON files** | Project list, skills, experience (easy, local, fast) |

**🪄 Interactivity / Animations**

| **Tool** | **Purpose** |
| --- | --- |
| **Framer Motion** | Animate page transitions, cards, hover effects, reveals |
| **Spline** | Drag-n-drop 3D editor; export as React components or <iframe> |
| **React State / Props** | Control interactivity, animations, and UI states |

**🌐 Backend (Optional)**

If you later want form submissions, authentication, or dynamic content:

| **Tool** | **Purpose** |
| --- | --- |
| **Node.js** | Backend server with API endpoints |
| **EmailJS** | Serverless contact form handling |

**🚀 Deployment**

| **Platform** | **Features** |
| --- | --- |
| **Vercel** | Seamless deployment from GitHub, optimized for React + Vite |
| **Custom Domain** | Buy from Namecheap/Google Domains and link to Vercel in 2 clicks |

**🔧 Tooling & Dev Setup**

| **Tool** | **Role** |
| --- | --- |
| **Node.js** | Runtime that powers npm, dev server, and build tools |
| **npm** | Package manager to install Tailwind, React, Framer Motion, etc. |
| **Git + GitHub Desktop** | Version control and online repo hosting |
| **.gitignore** | Keeps build files and secrets out of GitHub |
| **VS Code** | Recommended editor (with Tailwind and React plugins) |

**📝 License**

| **Choice** | **Why** |
| --- | --- |
| **MIT License** | Allows others to view/learn from your code, with credit |
| *(Optional)* No License | If you want to keep it all-rights-reserved (not recommended) |

Sure! Here's a **GitHub Desktop Cheat Sheet** for your local-to-online workflow:

**🖥️ Local Workflow (Using GitHub Desktop)**

**✅ 1. Make changes locally**

* Edit files in your project folder.

**✅ 2. Stage & Commit**

* Open **GitHub Desktop**
* See changed files in the left panel.
* Add a **summary** (commit message)
* Click **"Commit to main"** (or whatever your branch is called)

**✅ 3. Push to GitHub**

* After committing, click **"Push origin"** (top bar)
* Your code is now live on GitHub.com

**🌐 Syncing with Remote Repository**

**🔄 4. Pull updates from GitHub**

* If changes were made online or by teammates:
  + Click **"Fetch origin"**
  + Then **"Pull origin"** to sync your local copy

**🧪 Extra Tips**

* **New files not showing?** → Click **Repository > Show in Explorer** to verify file location.
* **New branch?** → Click **Current Branch > New Branch**.
* **Ignore files (e.g., .env)** → Add to .gitignore file.

Resources:

* <https://web.dev/learn/css/animations>
* [React Demo 1](https://www.youtube.com/watch?v=ifOJ0R5UQOc&t=3390s&ab_channel=PedroTech)
* <https://cssreference.io/animations/>
* [React Demo 2](https://www.youtube.com/watch?v=LGdPMf-SgBA)
* [React Demo 3](https://www.youtube.com/watch?v=_63mEm3AMSY)

**Steps:**

1. Create a local Git repository using GitHub Desktop, and add Node to the *.gitignore* file. In addition to this, I selected the MIT License to protect my work, while also allowing others to learn from it.
2. Initialized Vite and React with the following statement:
   1. npm create vite@latest . -- --template react
   2. npm install
   3. npm run dev (to run code locally)
3. Installing Tailwind CSS and additional packages:
   1. <https://tailwindcss.com/docs/installation/using-vite>
   2. Have your plugins like this in the vite.config.js:
   3. A screen shot of a computer code

      AI-generated content may be incorrect.
   4. Make sure it runs npm run dev with no issues
   5. Install framer motion with npm install framer-motion
   6. lucide-react for certain icons
   7. react-router-dom for routes to various pages, if making a multipaged site
   8. tailwind-merge to create a util function to combine class names
   9. @radix-ui/react-toast class-variance-authority clsx are packages used to allow for toaster messages (pop-up prompts)
   10. emailjs-com for email management (create free account)
4. Adding a not found page, and home page, import BrowserRouter, Route, and Routes, this will allow you to jump from page to page: [React Demo 1](https://www.youtube.com/watch?v=ifOJ0R5UQOc&t=3390s&ab_channel=PedroTech)
   1. To add the spline model, simply copy the embed public link from spline and add the iframe to the div, but add the class name className="w-screen h-screen overflow-hidden" to the div and className="w-full h-full" allowFullScreen settings to the iframe to ensure the full page is taken up.
5. Create a Resume page similar way of NotFound, and add the iframe container to handle the resume, with the resume stored in public, as that is where all static files should be
6. Delete references to the default App.css and delete all template code given by vite, from there begin to configure your styling and animations needed for the website in index.css, this is dependent on personal preferences and needs of the project.
7. In components create a .jsx file “Navbar” which will feature the navigation or jump bar for the portfolio website.
8. Create the contact form, use demo 1 to build the full layout and toaster, then use demo 2 to integrate EmailJS
   1. When using an api key, since we will be uploading this to a public github, we need to create a .env file, and add them as environment variables, make sure to have the variables prefixed with VITE\_ .
   2. Then add .env to the .gitignore file. TO do these additions use echo and > or >>.
   3. Call the env variables using import.meta.env.\_\_name\_\_
   4. Finally, later when the website is uploaded, add the env variables into Vercel.
9. For the skills section, use the stylings of React tutorial 3, and then for the container, content, and text, use tutorial 1, and then style the boxes to you own needs.
10. For projects, combined it with with the research, and used inspiration from another video for the navigation, however mostly gpt-ed as to accommodated the current Tech stack.