

Beginner's Guide to Building a Portfolio Website for Developers

Subtitle: Create a personal portfolio to showcase your projects, skills, and attract opportunities.

Website Name: haas.dev

Website Link: <https://dev-roast-app.vercel.app>

Introduction

A portfolio website is the **best way to showcase your skills and projects** as a developer. This guide helps beginners design, structure, and build a professional-looking portfolio that impresses potential clients, employers, or collaborators.

Step 1: Plan Your Portfolio

- Decide on sections:
 - About Me / Introduction
 - Skills & Tools
 - Projects / Case Studies
 - Contact / Social Links
 - Sketch layout and navigation flow
-

Step 2: Set Up Your Project

- Use **HTML, CSS, and JavaScript** for a basic site
- Optional: Use frameworks like **React** or **Next.js** for modern dynamic sites
- Folder structure example:

```
portfolio/  
├── index.html  
├── styles/  
│   └── style.css  
├── scripts/  
│   └── main.js  
└──
```

Step 3: Build Core Sections

1. **Header / Navigation:** Clear links to each section
2. **About Me:** Short intro, photo/avatar, and tagline
3. **Skills:** Use icons or progress bars for tech stack
4. **Projects:** Cards with project screenshots, description, and links
5. **Contact:** Form with validation (name, email, message)

Step 4: Make It Interactive

- Add smooth scrolling for navigation
- Highlight active section on scroll
- Add hover effects on project cards
- Optional: Animate skill progress bars using JS

Step 5: Add Projects Dynamically (Optional)

- Store projects as an array of objects:

```
const projects = [  
  { title: 'To-Do App', description: 'Manage tasks', link: '#' },  
  { title: 'Weather App', description: 'Fetch live weather data', link: '#' }  
];  
  
const projectContainer = document.querySelector('#projects');  
projects.forEach(p => {  
  const card = document.createElement('div');  
  card.classList.add('project-card');  
  card.innerHTML = `

### ${p.title}</h3><p>${p.description}</p><a href="${p.link}</a></p>`; projectContainer.appendChild(card); });


```

Step 6: Make It Responsive

- Use **media queries** for mobile and tablet layouts

- Test across devices to ensure usability
 - Optional: Use frameworks like **Bootstrap** or **Tailwind CSS**
-

Step 7: Deploy Your Portfolio

- Free hosting options: GitHub Pages, Netlify, Vercel
 - Connect a **custom domain** if desired
 - Ensure projects are live and links are working
-

Step 8: Mini Project Ideas

- Developer Portfolio with 3–5 projects
 - Interactive Resume with skills and achievements
 - Blog section to share tutorials or notes
-

Step 9: Best Practices

- Keep design **clean and minimal**
 - Ensure **fast load times** (optimize images, minimize scripts)
 - Include **call-to-action buttons** (contact, hire me)
 - Update portfolio regularly with new projects
-

Step 10: Key Takeaways

- Portfolio websites demonstrate your **skills and credibility**
 - Use HTML, CSS, and JS to create a functional and interactive site
 - Hosting and deployment make your portfolio accessible globally
 - Regular updates reflect your growth as a developer
-

Visit **haas.dev** for portfolio templates, step-by-step guides, and beginner-friendly developer projects.

Website Name: haas.dev

Website Link: <https://dev.react.app> vercel.app

website LINK. <https://dev-toast-app.vercel.app>