

Vibecoding Deployment Guide: AI-Powered Web Development to AWS

Complete End-to-End Documentation

Table of Contents

1. [Introduction to Vibecoding](#)
 2. [Phase 1: Setting Up Replit](#)
 3. [Phase 2: Prompt Validation](#)
 4. [Phase 3: Local Development with Cursor AI](#)
 5. [Phase 4: AWS Account Setup](#)
 6. [Phase 5: S3 Bucket Configuration](#)
 7. [Phase 6: Domain Registration](#)
 8. [Phase 7: SSL Certificate Setup](#)
 9. [Phase 8: CloudFront Distribution](#)
 10. [Phase 9: Route 53 DNS Configuration](#)
 11. [Troubleshooting Guide](#)
 12. [Best Practices & Security](#)
-

1. Introduction to Vibecoding {#introduction}

Vibecoding is a modern development workflow that leverages AI-powered code generation tools to rapidly prototype and deploy web applications. This methodology combines the power of AI assistants like Replit AI, ChatGPT, and Cursor AI with enterprise-grade cloud infrastructure on AWS.

What You'll Accomplish

By the end of this guide, you will have:

- Created a fully functional website using AI-generated code
- Set up a professional development environment with Cursor AI
- Deployed your website to AWS S3 with global CDN distribution
- Configured a custom domain with SSL/TLS encryption
- Implemented industry-standard security and performance optimizations

Prerequisites

- A computer with internet connection (Windows, Mac, or Linux)
- Basic understanding of websites and web browsers
- Email address for account registrations
- Credit/debit card for AWS and domain registration (minimal costs involved)

Estimated Time

- Initial setup: 2-3 hours
 - Learning curve: 1-2 days for complete familiarity
 - Deployment time: 30-45 minutes once familiar with the process
-

2. Phase 1: Setting Up Replit {#phase-1-replit}

Replit is an online integrated development environment (IDE) that allows you to code directly in your browser with powerful AI assistance.

Step 2.1: Create Your Replit Account

1. Navigate to Replit

- Open your web browser (Chrome, Firefox, Safari, or Edge)
- Go to <https://replit.com>
- You'll see the Replit homepage with a "Sign Up" button

2. Sign Up Process

- Click the "Sign Up" button in the top-right corner
- Choose your preferred sign-up method:
 - **Google Account:** Fastest option, uses your existing Google credentials
 - **GitHub Account:** Ideal if you're familiar with version control
 - **Email Address:** Traditional sign-up with email and password

3. Complete Your Profile

- Enter your display name (this will be visible to others)
- Choose a unique username (this becomes part of your Replit URL)
- Verify your email address by clicking the link sent to your inbox
- Complete any onboarding questions about your experience level

4. Choose Your Plan

- **Free Plan:** Sufficient for learning and small projects
- **Hacker Plan:** \$7/month, includes more resources and private repls
- **Pro Plan:** \$20/month, adds AI features and collaboration tools
- For this guide, the **Free Plan** works perfectly fine

Step 2.2: Access Replit AI

1. Understanding Replit AI

- Replit AI is an intelligent coding assistant built into the platform
- It can generate complete projects from natural language descriptions
- Available in the chat interface on the left sidebar

2. Opening Replit AI

- Click the "Create Repl" button on your dashboard
- Select "Generate with AI" option
- Alternatively, open any existing Repl and click the AI chat icon (sparkle/star icon)

3. Familiarize Yourself with the Interface

- **Left Panel:** File explorer and AI chat
- **Center Panel:** Code editor
- **Right Panel:** Browser preview and console
- **Top Bar:** Run button, settings, and sharing options

Step 2.3: Craft Your Initial Prompt

The quality of your AI-generated code depends heavily on your prompt. Here's how to write effective prompts:

1. Prompt Structure Template

Create a [TYPE OF WEBSITE] that includes:

CORE FEATURES:

- [Feature 1 with specific details]
- [Feature 2 with specific details]
- [Feature 3 with specific details]

DESIGN REQUIREMENTS:

- [Color scheme/theme]
- [Layout preferences]
- [Responsive design needs]

TECHNICAL SPECIFICATIONS:

- [Technologies to use: HTML, CSS, JavaScript, React, etc.]
- [Any specific libraries or frameworks]
- [Browser compatibility requirements]

FUNCTIONALITY:

- [Interactive elements]
- [Forms or data collection]
- [Any animations or special effects]

2. Example Prompt

Create a modern portfolio website for a web developer that includes:

CORE FEATURES:

- Hero section with animated gradient background
- About me section with profile image and bio
- Projects showcase with filterable categories
- Skills section with progress bars
- Contact form with email validation
- Smooth scroll navigation

DESIGN REQUIREMENTS:

- Dark mode theme with accent color #00D9FF
- Minimalist, clean design with plenty of white space
- Mobile-first responsive design
- Modern sans-serif font (Inter or similar)

TECHNICAL SPECIFICATIONS:

- Pure HTML, CSS, and vanilla JavaScript
- No frameworks required
- CSS animations for smooth transitions
- Optimized images with lazy loading

FUNCTIONALITY:

- Hamburger menu for mobile navigation
- Project cards that expand on click
- Form submission with client-side validation
- Smooth scrolling between sections
- Social media links in footer

3. Submit Your Prompt

- Paste your prompt into the Replit AI chat
- Press Enter or click "Send"
- Wait for Replit AI to generate the code (usually 30-60 seconds)

4. Review Generated Code

- Replit AI will create necessary files (index.html, style.css, script.js, etc.)
- Click the "Run" button to preview your website
- The preview appears in the right panel

Step 2.4: Iterate and Refine

1. Test Your Generated Code

- Click through all navigation links

- Test forms and interactive elements
- Resize the browser window to check responsiveness
- Check for any console errors (press F12 in the preview)

2. Request Modifications

- If something isn't working, describe the issue to Replit AI
- Example: "The contact form isn't validating email addresses properly"
- Replit AI will update the code accordingly

3. Add More Features

- You can continue the conversation with Replit AI
 - Example: "Add a testimonials section with a carousel"
 - The AI will integrate new features into existing code
-

3. Phase 2: Prompt Validation {#phase-2-validation}

Before downloading your code, validate your prompt with additional AI tools to ensure it's comprehensive and follows best practices.

Step 3.1: Validate with ChatGPT

1. Access ChatGPT

- Go to <https://chat.openai.com>
- Sign in with your OpenAI account (or create one)
- Free tier is sufficient for prompt validation

2. Validation Prompt Template

I'm creating a website using AI-generated code. Please review my prompt and suggest improvements for:

- Missing features that would enhance user experience
- Best practices for web development
- Accessibility considerations
- Performance optimizations
- Security considerations

Here's my prompt:

[PASTE YOUR ORIGINAL PROMPT HERE]

Please provide:

1. A rating of the prompt quality (1-10)
2. Specific improvements to add
3. Potential issues to avoid
4. An enhanced version of the prompt

3. Analyze ChatGPT's Feedback

- Read through all suggestions carefully
- Note any accessibility features you missed (alt text, ARIA labels, keyboard navigation)
- Consider performance recommendations (image optimization, lazy loading)
- Review security suggestions (input sanitization, HTTPS requirements)

4. Refine Your Prompt

- Incorporate valuable suggestions from ChatGPT
- Create an enhanced version of your original prompt
- Keep a document with both versions for comparison

Step 3.2: Cross-Reference with Perplexity AI

1. Access Perplexity AI

- Navigate to <https://www.perplexity.ai>
- No account required for basic searches
- Perplexity provides real-time web research capabilities

2. Research Best Practices Query

What are the current best practices for [YOUR WEBSITE TYPE] in 2024-2025?

Include:

- Modern design trends
- Essential features
- Performance benchmarks
- SEO requirements
- Accessibility standards (WCAG 2.1)

3. Gather Industry Standards

- Perplexity will provide sourced information from recent articles
- Review cited sources for credibility
- Note any trending technologies or approaches
- Compare against your prompt requirements

4. Technology Stack Validation

Compare the following technologies for building [YOUR PROJECT TYPE]:

- HTML/CSS/JavaScript
- React
- Vue.js
- Next.js

Consider: ease of deployment, performance, SEO, learning curve, and AWS S3 hosting compatibility

5. Create Final Enhanced Prompt

- Combine insights from both ChatGPT and Perplexity
- Ensure your prompt includes:
 - Core functionality
 - Responsive design
 - Accessibility features
 - Performance optimizations
 - SEO best practices
 - Browser compatibility
 - Security considerations

Step 3.3: Document Your Validation Process

Create a validation checklist document:

markdown

Prompt Validation Checklist

Original Prompt

[Your initial prompt]

ChatGPT Feedback Summary

- Rating: X/10
- Key improvements: [bullet points]
- Security considerations: [bullet points]
- Accessibility features added: [bullet points]

Perplexity Research Findings

- Current trends: [bullet points]
- Industry standards: [bullet points]
- Technology recommendations: [bullet points]

Final Enhanced Prompt

[Your improved prompt incorporating all feedback]

Implementation Notes

[Any specific considerations for AWS deployment]

4. Phase 3: Local Development with Cursor AI {#phase-3-cursor}

Cursor AI is an AI-powered code editor built on VS Code that provides intelligent code completion and modification capabilities.

Step 4.1: Download Generated Code from Replit

1. Prepare for Download

- Ensure your Replit project is complete and tested
- Click the three-dot menu (⋮) in the file explorer
- Select "Download as ZIP"

2. Download Process

- Your browser will download a file named `repl-name.zip`
- The ZIP contains all your project files and folders
- Default download location: usually your `Downloads` folder

3. Extract Files

- **Windows:** Right-click the ZIP → "Extract All" → Choose destination
- **Mac:** Double-click the ZIP file (auto-extracts)
- **Linux:** Right-click → "Extract Here" or use `unzip repl-name.zip`

4. Organize Your Files

- Create a dedicated projects folder: `Documents/WebProjects/MyWebsite`
- Move extracted files to this folder
- Verify all files are present (HTML, CSS, JS, images, etc.)

Step 4.2: Install Cursor AI

1. Download Cursor AI

- Visit <https://cursor.sh>
- Click the "Download" button
- Select your operating system:
 - **Windows:** Downloads `.exe` installer
 - **Mac:** Downloads `.dmg` file (Apple Silicon or Intel)
 - **Linux:** Downloads `.AppImage` or `.deb` package

2. Installation Process For Windows:

- Double-click the downloaded `.exe` file
- Click "Yes" when prompted by User Account Control
- Follow the installation wizard
- Check "Add to PATH" option (important)
- Select "Create desktop shortcut" if desired
- Click "Install" and wait for completion
- Launch Cursor from desktop or Start menu

For Mac:

- Open the downloaded `.dmg` file
- Drag Cursor icon to Applications folder
- Go to Applications folder
- Right-click Cursor → "Open"
- Click "Open" in security dialog (first time only)
- Allow any system permissions requested

For Linux:

```
bash
```

```
# For .deb package (Ubuntu/Debian)
```

```
sudo dpkg -i cursor_*.deb
```

```
sudo apt-get install -f
```

```
# For AppImage
```

```
chmod +x Cursor-*.AppImage
```

```
./Cursor-*.AppImage
```

3. First Launch Configuration

- Cursor will ask to import VS Code settings (optional)
- Sign in with GitHub account (recommended) or email
- Choose your theme (dark/light)
- Enable Cursor AI features when prompted

Step 4.3: Set Up Your Project in Cursor

1. Open Project Folder

- Launch Cursor AI
- Click "File" → "Open Folder"
- Navigate to your extracted project folder
- Click "Select Folder" (Windows) or "Open" (Mac)

2. Explore the Interface

- **Left Sidebar:** File explorer, search, source control, extensions
- **Center:** Code editor with tabs
- **Right Sidebar:** AI chat panel (Ctrl+L or Cmd+L)
- **Bottom Panel:** Terminal, problems, output, debug console

3. Install Recommended Extensions

- Click Extensions icon (four squares) in left sidebar
- Install these essential extensions:
 - **Live Server:** Local development server with live reload
 - **Prettier:** Code formatter
 - **ESLint:** JavaScript linter
 - **HTML CSS Support:** Enhanced HTML/CSS IntelliSense
 - **Auto Rename Tag:** Automatically renames paired HTML tags

4. Configure Live Server

- Right-click on `index.html` in file explorer
- Select "Open with Live Server"
- Your default browser opens with your website
- Any changes you make auto-reload the page

Step 4.4: Use Cursor AI for Code Modifications

1. Understanding Cursor AI Features

- **Cmd+K** (Mac) or **Ctrl+K** (Windows): Inline editing
- **Cmd+L** (Mac) or **Ctrl+L** (Windows): Chat with AI
- **Tab**: Accept AI suggestions
- **Esc**: Reject AI suggestions

2. Inline Editing with Cmd+K / Ctrl+K

- Select the code section you want to modify
- Press Cmd+K (Mac) or Ctrl+K (Windows)
- Type your instruction in plain English
- Example: "Add smooth scrolling to all anchor links"
- Press Enter, review the changes, and accept or reject

3. Chat-Based Editing with Cmd+L / Ctrl+L

- Press Cmd+L (Mac) or Ctrl+L (Windows) to open AI chat
- Ask questions about your code
- Example queries:
 - "How can I improve the performance of this JavaScript?"
 - "Add responsive navigation menu for mobile devices"
 - "Implement form validation for the contact form"
 - "Optimize images for web performance"

4. Common Modification Patterns Add New Features:

Select relevant file → Cmd+K → Type:

"Add a dark mode toggle button in the header that persists using localStorage"

Fix Bugs:

Highlight problematic code → Cmd+K → Type:

"This function isn't working on mobile. Fix cross-browser compatibility"

Improve Code Quality:

Select function → Cmd+K → Type:

"Refactor this code for better readability and add comments"

Optimize Performance:

Select code → Cmd+L → Ask:

"How can I optimize this code to reduce page load time?"

5. Testing Your Modifications

- Save files (Ctrl+S or Cmd+S)
- Check Live Server preview (auto-reloads)
- Test on different screen sizes (browser DevTools - F12)
- Check browser console for errors
- Validate HTML at validator.w3.org

Step 4.5: Prepare Code for Deployment

1. Final Code Review Checklist

- ☐ All images optimized (compressed, proper formats)
- ☐ Remove console.log() statements
- ☐ Minify CSS and JavaScript (for production)
- ☐ Check all links work correctly
- ☐ Test forms thoroughly
- ☐ Verify responsive design on mobile, tablet, desktop
- ☐ Run accessibility checker
- ☐ Check browser compatibility

2. Build Production Files

- Create a new folder called `dist` or `build`
- Copy all necessary files to this folder:
 - HTML files
 - CSS files
 - JavaScript files

- Images and assets
 - Favicon
 - Remove any development-only files
 - Ensure file paths are correct and relative
-

5. Phase 4: AWS Account Setup {#phase-4-aws}

Amazon Web Services (AWS) is a comprehensive cloud platform that will host your website with enterprise-grade reliability and performance.

Step 5.1: Create Your AWS Account

1. Navigate to AWS

- Go to <https://aws.amazon.com>
- Click "Create an AWS Account" in the top-right corner

2. Account Information

- **Email Address:** Use a professional email you check regularly
- **AWS Account Name:** Choose a descriptive name (e.g., "MyCompany Production")
- Click "Continue"

3. Contact Information

- Select account type:
 - **Personal:** For individual projects and learning
 - **Professional:** For business use (requires company details)
- Fill in your complete contact information:
 - Full legal name
 - Phone number (with country code)
 - Complete address (must match credit card billing address)
- Read and accept AWS Customer Agreement
- Click "Create Account and Continue"

4. Payment Information

- Enter credit/debit card details
- AWS requires a valid payment method for verification
- **Important:** You won't be charged during the Free Tier period
- AWS will make a temporary \$1 authorization (refunded immediately)

- Enter billing address if different from contact address
- Click "Verify and Continue"

5. Identity Verification

- Choose verification method:
 - **Text message (SMS):** Fastest option
 - **Voice call:** Alternative if SMS unavailable
- Enter your phone number
- Enter the security code displayed on screen
- Wait for verification code (arrives within 1-2 minutes)
- Enter 4-digit code received
- Click "Verify Code"

6. Select Support Plan

- **Basic Support (Free):** Sufficient for this guide
 - 24/7 access to customer service
 - Documentation and forums
 - AWS Trusted Advisor (limited)
- **Developer Support (\$29/month):** For active development
- **Business Support (\$100/month):** For production workloads
- Select "Basic Support - Free"
- Click "Complete Sign Up"

7. Account Confirmation

- You'll receive a welcome email within minutes
- Account activation may take up to 24 hours (usually instant)
- Once activated, you can sign in to AWS Management Console

Step 5.2: Secure Your AWS Account

1. Sign In to AWS Console

- Go to <https://console.aws.amazon.com>
- Enter your root account email
- Click "Next"
- Enter your password
- Click "Sign In"

2. Enable Multi-Factor Authentication (MFA)

- Click your account name (top-right) → "Security Credentials"
- Scroll to "Multi-factor authentication (MFA)" section
- Click "Assign MFA device"
- Choose MFA device type:
 - **Virtual MFA device:** Use authenticator app (recommended)
 - **Hardware TOTP token:** Physical device
 - **Hardware MFA device:** U2F security key

3. Set Up Virtual MFA (Most Common)

- Select "Virtual MFA device" → Click "Continue"
- Install an authenticator app on your phone:
 - **Google Authenticator** (iOS/Android)
 - **Microsoft Authenticator** (iOS/Android)
 - **Authy** (iOS/Android/Desktop)
- Open the authenticator app
- Scan the QR code displayed in AWS console
- Enter two consecutive MFA codes from your app
- Click "Add MFA"
- **Important:** Store backup codes in a secure location

4. Create IAM User (Best Practice) Why IAM Users Matter:

- Root account has unlimited access (dangerous for daily use)
- IAM users have limited, specific permissions
- Better audit trails and security

Creating Your IAM User:

- In AWS Console, search for "IAM" in the top search bar
- Click "IAM" service
- Click "Users" in left sidebar → "Add users"
- Enter username (e.g., "admin-user")
- Select access type:
 - ☒ AWS Management Console access
 - ☒ Programmatic access (for CLI/API)
- Console password:

- Choose "Custom password" and create a strong password
- ☐ Uncheck "Require password reset" (optional)
- Click "Next: Permissions"

5. Set IAM Permissions

- Select "Attach existing policies directly"
- Search and select these policies:
 - **AdministratorAccess**: Full access (for your admin user)
 - **AmazonS3FullAccess**: For S3 operations
 - **CloudFrontFullAccess**: For CDN management
 - **AWSCertificateManagerFullAccess**: For SSL certificates
 - **Route53FullAccess**: For DNS management
- Click "Next: Tags" → "Next: Review"
- Review settings → Click "Create user"
- **Critical**: Download CSV file with access keys
- Store this file securely (you can't retrieve secret key later)

6. Set Up Billing Alerts

- Click account name → "Billing Dashboard"
- Click "Billing preferences" in left sidebar
- Enable these options:
 - ☒ Receive PDF Invoice By Email
 - ☒ Receive Free Tier Usage Alerts
 - ☒ Receive Billing Alerts
- Enter alert email address
- Click "Save preferences"

7. Create CloudWatch Billing Alarm

- Go to CloudWatch service
- Click "Alarms" → "Billing" → "Create alarm"
- Click "Select metric"
- Choose "Billing" → "Total Estimated Charge"
- Select USD → "Select metric"
- Set threshold: \$10 (or your comfort level)
- Click "Next"

- Configure email notification
- Create alarm

Step 5.3: Understand AWS Free Tier

What's Included (12 Months Free):

- **S3 Storage:** 5 GB storage, 20,000 GET requests, 2,000 PUT requests
- **CloudFront:** 50 GB data transfer out, 2,000,000 HTTP/HTTPS requests
- **Route 53:** Hosted zone (\$0.50/month - not free, but minimal)
- **Certificate Manager:** SSL certificates (completely free, always)

Monitoring Your Usage:

- Go to Billing Dashboard
- Click "Free Tier" in left sidebar
- View current usage vs. limits
- Set up alerts before exceeding limits

After Free Tier:

- S3: ~\$0.023 per GB (first 50 TB)
 - CloudFront: ~\$0.085 per GB (first 10 TB)
 - Route 53: \$0.50 per hosted zone per month
 - For a small website: typically \$1-5/month
-

6. Phase 5: S3 Bucket Configuration {#phase-5-s3}

Amazon S3 (Simple Storage Service) will host your static website files with high availability and durability.

Step 6.1: Create S3 Bucket

1. Access S3 Service

- Sign in to AWS Console
- Search for "S3" in top search bar
- Click "S3" to open the S3 dashboard

2. Create New Bucket

- Click orange "Create bucket" button

- **Bucket Name Guidelines:**

- Must be globally unique across all AWS accounts
- 3-63 characters, lowercase only
- No spaces, use hyphens for readability
- Should match your domain (recommended)
- Example: `my-awesome-website-2024` or `www.yourdomain.com`

3. **Configure Bucket Settings AWS Region:**

- Choose a region close to your target audience
- **US East (N. Virginia):** us-east-1 - Good for US/global
- **EU (Frankfurt):** eu-central-1 - Good for Europe
- **Asia Pacific (Mumbai):** ap-south-1 - Good for India
- **Note:** Region affects latency and cost (minimal difference)

Object Ownership:

- Select "ACLs disabled (recommended)"
- This simplifies permissions management

Block Public Access Settings:

- **UNCHECK** "Block all public access" ☐
- This allows your website to be publicly accessible
- Check the acknowledgment box that appears:
 - ☒ "I acknowledge that the current settings might result in this bucket and the objects within becoming public"
- **Why:** Your website needs to be publicly accessible

Bucket Versioning:

- Select "Disable" (sufficient for most websites)
- Enable if you need version history (costs extra)

Default Encryption:

- Select "Server-side encryption with Amazon S3 managed keys (SSE-S3)"
- This is free and encrypts data at rest

Advanced Settings:

- Object Lock: Leave disabled
- Tags: Optional, add `Environment: Production` if desired

4. **Create Bucket**

- Click orange "Create bucket" button at bottom

- Wait for confirmation message
- You'll see your new bucket in the list

Step 6.2: Configure Static Website Hosting

1. Open Bucket Settings

- Click on your bucket name in the list
- Navigate to the "Properties" tab

2. Enable Static Website Hosting

- Scroll down to "Static website hosting" section
- Click "Edit"
- Select "Enable" 

3. Configure Hosting Settings

- **Hosting type:** Static website hosting
- **Index document:** `index.html`
 - This is your homepage file
- **Error document:** `error.html` (optional)
 - Create a custom 404 page for better UX
 - If not specified, S3 returns generic XML error
- **Redirection rules:** Leave empty for now
- Click "Save changes"

4. Note Your Website Endpoint

- After saving, you'll see a "Bucket website endpoint"
- Format: `http://bucket-name.s3-website-region.amazonaws.com`
- Example: `http://my-website.s3-website-us-east-1.amazonaws.com`
- **Save this URL** - you'll need it for testing
- **Note:** It's HTTP (not HTTPS) - we'll add SSL via CloudFront later

Step 6.3: Set Bucket Permissions

1. Configure Bucket Policy

- Go to "Permissions" tab in your bucket
- Scroll to "Bucket policy" section
- Click "Edit"

2. Add Public Read Policy

- Paste this policy (replace `YOUR-BUCKET-NAME` with your actual bucket name):

```
json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PublicReadGetObject",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::YOUR-BUCKET-NAME/*"
    }
  ]
}
```

- **Explanation:**
 - `"Principal": "*" - Allows anyone to access`
 - `"Action": "s3:GetObject" - Allows reading objects only`
 - `"Resource": ".../*" - Applies to all files in bucket`
- Click "Save changes"
- You may see a warning about public access - this is expected

Step 6.4: Upload Your Website Files

1. Prepare Files for Upload

- Open your project folder (the production/dist folder from Cursor)
- Verify all files are present:
 - index.html (required)
 - CSS files
 - JavaScript files
 - Images and assets
 - error.html (optional but recommended)
 - favicon.ico (recommended)

2. Upload via AWS Console

- Go to "Objects" tab in your S3 bucket

- Click orange "Upload" button
- **Option 1:** Drag and drop files/folders
- **Option 2:** Click "Add files" and "Add folder" buttons

3. Configure Upload Settings

- **Permissions:** Leave default (inherits bucket permissions)
- **Properties:**
 - Storage class: "Standard" (default)
 - Encryption: Use bucket settings
- **Optional:** Add metadata (cache-control headers)
- Click orange "Upload" button at bottom
- Wait for "Upload succeeded" message

4. Verify File Structure

- Ensure index.html is at the root level (not in a subfolder)
- Check that file paths in your HTML match uploaded structure
- Example structure:

```
my-bucket/  
├── index.html  
├── error.html  
├── css/  
│   └── style.css  
├── js/  
│   └── script.js  
├── images/  
│   ├── logo.png  
│   └── background.jpg  
└── favicon.ico
```

Step 6.5: Test Your S3 Website

1. Access Website Endpoint

- Copy your S3 website endpoint from Properties tab
- Paste into browser address bar
- Press Enter

2. Verify Functionality

- ✓ Homepage loads correctly

- ✓ All images display
- ✓ CSS styling applies
- ✓ JavaScript functions work
- ✓ Navigation links work
- ✓ Forms submit (if applicable)

3. Check Different Pages

- Test all internal links
- Try accessing non-existent page to verify error.html
- Test on mobile device or browser DevTools mobile view

4. Troubleshooting Common Issues **Issue:** "403 Forbidden" error

- **Solution:** Check bucket policy allows public read access
- Verify bucket's "Block Public Access" settings are correct

Issue: "404 Not Found" for index.html

- **Solution:** Ensure index.html is at root level, not in subfolder
- Check file name is exactly "index.html" (case-sensitive)

Issue: Images don't load

- **Solution:** Check file paths in HTML are correct
- Ensure images uploaded to correct folder structure
- Verify image file names match exactly (case-sensitive)

Issue: CSS/JS not working

- **Solution:** Check file paths in HTML
- Verify files uploaded successfully
- Check browser console (F12) for 404 errors

7. Phase 6: Domain Registration {#phase-6-domain}

A custom domain gives your website a professional appearance and makes it easier to remember.

Step 7.1: Choose Your Domain Name

1. Domain Name Best Practices

- **Short and memorable:** 15 characters or less ideal
- **Easy to spell:** Avoid complex words or unusual spellings
- **Brandable:** Unique and professional

- **Extension choice:** .com (most recognized), .io (tech startups), .dev (developers)
- **Avoid:** Numbers, hyphens, trademarked terms

2. Check Availability

- Use domain checker tools:
 - [Namecheap.com](https://namecheap.com)
 - [GoDaddy.com](https://godaddy.com)
 - AWS Route 53 domain registration
 - [Google Domains](https://domains.google)

Step 7.2: Register Domain (Using Namecheap Example)

1. Search for Domain

- Go to registrar website (e.g., Namecheap)
- Enter desired domain name
- Review available options and pricing
- Typical cost: \$8-15/year for .com

2. Purchase Domain

- Add domain to cart
- Choose registration period (1-10 years)
- **Recommended add-ons:**
 - ☒ WhoisGuard (free privacy protection on Namecheap)
 - ☐ Skip web hosting (you're using AWS)
 - ☐ Skip email (use Gmail or AWS WorkMail separately)

3. Complete Registration

- Create account or sign in
- Enter payment information
- Complete purchase
- Check email for confirmation

7. Phase 7: SSL Certificate Setup {#phase-7-ssl}

AWS Certificate Manager (ACM) provides free SSL/TLS certificates to encrypt traffic to your website.

Step 7.1: Request Certificate in ACM

1. Access Certificate Manager

- Open AWS Console
- **Important:** Switch region to **US East (N. Virginia) us-east-1**
- This is required for CloudFront certificates
- Search for "Certificate Manager" or "ACM"

2. Request Certificate

- Click "Request a certificate"
- Select "Request a public certificate"
- Click "Next"

3. Add Domain Names

- Enter your domain:
- Click "Add another name to this certificate"
- Add wildcard:
- This covers www and all subdomains

4. Select Validation Method

- Choose **DNS validation** (recommended)
- Click "Request"

5. Complete DNS Validation

- View certificate details
- Expand domain names
- Click "Create records in Route 53" (if domain in Route 53)
- Or manually add CNAME records to your DNS provider

8. Phase 8: CloudFront Distribution {#phase-8-cloudfront}

CloudFront is AWS's CDN that delivers your content globally with low latency and HTTPS support.

Step 8.1: Create Distribution

1. Open CloudFront

- Search for "CloudFront"
- Click "Create distribution"

2. Origin Settings

- **Origin domain:** Select your S3 bucket endpoint
- **Origin path:** Leave blank
- **Origin access:** "Origin access control settings"
- Create new OAC (recommended for security)

3. Default Cache Behavior

- **Viewer protocol policy:** "Redirect HTTP to HTTPS"
- **Allowed HTTP methods:** GET, HEAD
- **Cache policy:** "CachingOptimized"

4. Distribution Settings

- **Alternate domain names (CNAME):** Add `yourdomain.com` and `www.yourdomain.com`
- **Custom SSL certificate:** Select your ACM certificate
- **Default root object:** `index.html`

5. Create Distribution

- Click "Create distribution"
- Status changes from "Deploying" to "Enabled" (10-15 minutes)
- Note your CloudFront domain: `d1234abcd.cloudfront.net`

9. Phase 9: Route 53 DNS Configuration {#phase-9-route53}

Route 53 connects your domain name to your CloudFront distribution.

Step 9.1: Create Hosted Zone

1. Open Route 53

- Search for "Route 53"
- Click "Hosted zones"
- Click "Create hosted zone"

2. Configure Hosted Zone

- **Domain name:** `yourdomain.com`
- **Type:** Public hosted zone
- Click "Create hosted zone"

3. Update Nameservers

- Copy the 4 NS (nameserver) records from Route 53

- Go to your domain registrar (Namecheap, etc.)
- Update nameservers to AWS nameservers
- Wait 24-48 hours for propagation (usually faster)

Step 9.2: Create DNS Records

1. Create A Record for Root Domain

- Click "Create record"
- **Record name:** Leave blank (root)
- **Record type:** A
- **Alias:** Yes
- **Route traffic to:** CloudFront distribution
- Select your distribution
- Click "Create records"

2. Create A Record for WWW

- Click "Create record"
- **Record name:**
- **Record type:** A
- **Alias:** Yes
- **Route traffic to:** CloudFront distribution
- Select same distribution
- Click "Create records"

3. Verify DNS Propagation

- Use dnschecker.org
- Enter your domain
- Check that it resolves to CloudFront

10. Troubleshooting Guide {#troubleshooting}

Common Issues and Solutions

Certificate Pending Validation

- Ensure CNAME records added to DNS
- Wait up to 30 minutes for validation

- Check DNS propagation with online tools

CloudFront 403 Error

- Update S3 bucket policy to allow CloudFront OAC
- Verify default root object is set to `index.html`

Domain Not Resolving

- Check nameservers updated at registrar
- Verify Route 53 A records point to CloudFront
- Clear DNS cache: `ipconfig /flushdns` (Windows) or `sudo dscacheutil -flushcache` (Mac)

Website Loads Without HTTPS

- Ensure CloudFront viewer protocol is "Redirect HTTP to HTTPS"
 - Verify SSL certificate is attached to distribution
-

11. Best Practices & Security {#best-practices}

Performance Optimization

- Compress images before upload
- Minify CSS/JS files
- Enable CloudFront compression
- Set appropriate cache headers
- Use WebP image format

Security Measures

- Enable S3 bucket encryption
- Use CloudFront with OAC (not OAI)
- Implement Content Security Policy headers
- Enable AWS CloudTrail for audit logs
- Regular security reviews

Cost Management

- Monitor AWS billing dashboard weekly
- Set up billing alerts at \$5, \$10, \$20
- Delete unused S3 versions
- Use S3 Lifecycle policies for old content
- Monitor CloudFront data transfer

Maintenance Tasks

- Update website content via S3 upload
 - Invalidate CloudFront cache after updates: `/*`
 - Review CloudWatch logs monthly
 - Update SSL certificates before expiry (auto-renews)
 - Test website across browsers quarterly
-

12. Quick Reference Commands

Invalidate CloudFront Cache

```
bash

aws cloudfront create-invalidation \
  --distribution-id YOUR_DISTRIBUTION_ID \
  --paths "/*"
```

Sync Local Files to S3

```
bash

aws s3 sync ./dist s3://your-bucket-name \
  --delete --cache-control max-age=31536000
```

Check DNS Records

```
bash

nslookup yourdomain.com
dig yourdomain.com
```

Conclusion

You've successfully deployed a production-ready website using AI-assisted development and AWS infrastructure! Your site now features:

- ✓ AI-generated, custom-coded website
- ✓ Global CDN delivery with CloudFront
- ✓ HTTPS encryption with free SSL certificate
- ✓ Custom domain with professional DNS
- ✓ Scalable, enterprise-grade hosting
- ✓ Cost-effective solution under AWS Free Tier

Next Steps

- Add Google Analytics for traffic monitoring
- Implement SEO optimization
- Set up automated backups
- Create staging environment
- Integrate CI/CD pipeline with GitHub Actions

Additional Resources

- [AWS Documentation](#)
- [Cursor AI Documentation](#)
- [Web.dev Best Practices](#)
- [MDN Web Docs](#)

Document Version: 1.0

Last Updated: December 2025

Author: Vibecoding Deployment Guide

This guide is maintained and updated regularly. For the latest version, visit the documentation repository.