Portfolio Website

Project Overview

This project is a **static personal portfolio website** developed using **HTML5 and CSS3**. The primary goal of the project is to present professional information—such as skills, projects, and contact details—in a clean, visually appealing, and structured manner.

Goals and Objectives

- Showcase personal and technical profile in a professional format
- Present skills and projects in a clear, readable layout
- Apply modern CSS styling for improved UI/UX
- Ensure responsiveness and accessibility across devices
- Maintain clean and well-organized code

Setup Instructions

This project does not require any external libraries, frameworks, or build tools.

Installation & Configuration Steps

1. Download or clone the project repository.
2. Ensure the following files are present in the same directory:
    - index.html
    - style.css
3. Open the HTML file in a web browser:
    - Double-click the file
      or
    - Right-click → Open with → Any modern browser (Chrome, Firefox, Edge)

No additional configuration or server setup is required.

Code Structure

The project follows a simple and well-organized file hierarchy:

portfolio-website/
|

```
├── index.html # Main HTML file containing page structure and content
├── style.css # External CSS file for styling and responsiveness
└── README.md # Project documentation
```

File Responsibilities

- **index.html**
    - Defines the structure and content of the website
    - Contains sections such as Header, Introduction, Skills, Projects, and Contact Form
    - Links to the external stylesheet
- **style.css**
    - Handles layout, typography, colors, and spacing
    - Implements responsive design using media queries
    - Applies hover effects and visual enhancements

Technical Details

Architecture

- **Frontend-only static architecture**
- Separation of concerns:
    - HTML for structure
    - CSS for presentation

Algorithms and Data Structures

- No complex algorithms or data structures are used
- The project focuses on **content organization and UI rendering**
- Form validation is handled using native HTML attributes (required, type="email")

Key Technical Concepts Used

- Semantic HTML elements (header, section, form, ul, li)
- CSS Flexbox for layout (skills list)
- CSS transitions and hover effects
- Responsive design using media queries

- External font integration via Google Fonts

Testing Evidence

Testing was performed through **manual validation** across different scenarios.

Sample Test Cases

Test Case	Description	Expected Result
Page Load	Open website in browser	Page loads without errors
Responsive Layout	Resize browser window	Layout adapts correctly
Project Links	Click GitHub links	Opens correct repositories
Form Validation	Submit empty form	Browser blocks submission
Email Validation	Enter invalid email	Browser shows validation error

Validation Results

- No console errors on page load
- HTML and CSS render consistently across modern browsers
- Responsive behavior verified on desktop and mobile viewports