Personal Website Architecture Documentation

Overview

This document describes the architecture of the personal portfolio website, including its components, deployment strategy, and interaction flows.

1. Component Architecture

Frontend Components

- **UI Layer**
    - index.html: Main structure and content
    - styles.css: Styling and layout
    - script.js: Client-side interactivity
- **Image Gallery**
    - Filter Bar: A-Z filtering and search functionality
    - Image Grid: Responsive image display
    - Modal View: Full-size image viewing
- **Contact Form**
    - Form Validation: Client-side input validation
    - Form Submission: AJAX submission handling

Backend Components

- **Server Layer**
    - server.js: Express server configuration
    - routes/gallery.js: Image gallery API endpoints
- **Storage**
    - File System: Local image storage

Development Tools

- ESLint: Code quality and style checking
- Mocha Tests: Unit testing framework
- GitHub Actions: CI/CD pipeline

2. Deployment Architecture

Development Environment

- **Local Development Server**
    - Node.js with Express

    - npm for package management
    - Development build configuration
    - Local testing and debugging

Production Environment

- **Production Server**

    - Node.js with Express
    - Production-optimized build
    - Static asset serving
    - Error handling and logging

Deployment Flow

1. Code pushed to GitHub repository
2. GitHub Actions triggers CI/CD pipeline
3. Tests and linting run automatically
4. On success, deployment to production

3. User Interaction Flows

Page Load Sequence

1. User visits website
2. Server sends HTML/CSS/JS
3. JavaScript initializes
4. Image gallery loads

Image Gallery Interactions

1. **Filtering**

    - User clicks filter button
    - JavaScript filters images
    - Display updates instantly

2. **Search**

    - User enters search term
    - JavaScript searches images
    - Results update in real-time

3. **Modal View**

    - User clicks image
    - Modal opens with full-size image
    - Close button dismisses modal

Contact Form Flow

1. User fills out form
2. Client-side validation runs
3. Form submits via AJAX

4. Success/error message displays

4. Technical Stack

Frontend

- HTML5
- CSS3
- Vanilla JavaScript
- Responsive Design

Backend

- Node.js
- Express.js
- File System API

Development

- Git/GitHub
- ESLint
- Mocha
- GitHub Actions

5. Security Considerations

Input Validation

- Client-side form validation
- Server-side sanitization
- File type restrictions

Error Handling

- Graceful error recovery
- User-friendly error messages
- Server error logging

Code Security

- Dependencies kept updated
- Security headers configured
- Safe file handling

6. Performance Optimizations

Frontend

- Minimized CSS/JS
- Optimized images
- Lazy loading

- Responsive design

Backend

- Caching strategies
- Efficient file handling
- Response compression

7. Maintenance and Scaling

Version Control

- Feature branching
- Pull request workflow
- Semantic versioning

Testing Strategy

- Unit tests
- Integration tests
- Automated CI/CD

Documentation

- Code comments
- API documentation
- Setup instructions

8. Future Enhancements

Planned Features

- Backend form processing
- Image optimization service
- Dark/light mode toggle
- Performance monitoring

Technical Debt

- Add TypeScript
- Enhance test coverage
- Implement caching
- Add logging service