Project Development Journal — Portfolio Website

Summary

This journal documents the development of the portfolio website located at `project root`. It provides a chronological log of work completed during each session, including implementation details, issues encountered, solutions, reproduction steps, and references. The goal is to enable any reader to reproduce the website and understand the non-class techniques applied.

Project Structure

index.html – Main landing page
styles.css – Global site styling
/assets/js/ – Script files (`header-injector.js`, `navbar.js`, `main.js`) for component injection, navigation behavior, and small UI interactions
/components/ – Reusable HTML fragments (`header.html`, `footer.html`)
/assets/img/ – Images for cards, profile, and page visuals
/projects, **/about**, **/contact** – Additional pages linked from the main site

Session Log

Session 1 — Repository Setup and Initial Scaffold (2025-09-10)

Goal: Establish the foundational structure of the website.
Work Completed:

- Created `index.html` with hero section, featured projects, skills grid, and contact call-to-action.
- Added Bootstrap CDN and custom `styles.css`.
- Added component placeholders using `data-inject` attributes.
  Issues: None.
  Outcome: Basic static site successfully opened with Live Server.

Session 2 — Component Injection System (2025-09-12)

Goal: Implement reusable header/footer components.
Work Completed:

- Built `header-injector.js` to fetch and inject HTML fragments.

- Added fetch error-handling and module syntax support.
  Issues: `file://` protocol prevented fetch requests.
  Resolution: Operated site via a local development server (Live Server or `python -m http.server`).
  Reproduction: Start server, then navigate to `http://localhost:8000/index.html`.

Session 3 — Responsive Layout and Accessibility (2025-09-18)

Goal: Optimize responsiveness and accessibility.
Work Completed:

- Applied Bootstrap responsive utilities and refined layout using custom CSS.

- Added ARIA attributes and alt text for screen readers.
  Issues: Images overflowed on narrow screens.
  Resolution: Added `max-width: 100%` and enabled column layout for smaller viewports.

Session 4 — Project Card Refinements (2025-09-22)

Goal: Improve card consistency and layout.
Work Completed:

- Standardized layout by moving metadata to `.card-footer`.

- Added secure external linking: `target="_blank"` with `rel="noopener"`.
  Issues: Uneven card heights.
  Resolution: Converted cards to flex containers and pushed footers to the bottom using `margin-top: auto`.

Session 5 — Accessibility and SEO Updates (2025-10-03)

Goal: Strengthen SEO and meet accessibility standards.
Work Completed:

- Added meta descriptions and improved heading hierarchy.

- Ensured proper semantic structure.
  Issues: Lighthouse flagged insufficient contrast.
  Resolution: Updated color palette and re-tested until scores improved.

Session 6 — Java Project Integration (2025-10-18)

Goal: Integrate Java-based projects and include advanced documentation.
Work Completed:

- Added GitHub links for SoccerGame, 15SquaresGame, and Scrabble.

- Wrote project summaries and created detail pages.

- Documented advanced concepts such as 15-puzzle solvability (inversion counting and parity) and simple physics/networking used in SoccerGame.
  Issues: Inconsistent READMEs across repositories.
  Resolution: Consolidated and standardized README instructions.

Session 7 — Final QA and Deployment Preparation (2025-11-28)

Goal: Prepare for production deployment and hosting.
Work Completed:

- Verified all internal routes and standardized paths for hosting.

- Confirmed that header/footer injection works across all pages.
  Issues: GitHub Pages uses different asset path rules when no custom domain is used.
  Resolution: Chose to retain absolute-root paths for production hosting and rely on CI adjustments when using GitHub Pages.

Implementation Details

Component Injection (`header-injector.js`)

Purpose: Dynamically inserts shared components across pages using `fetch`.
Key Notes:

- Requires an HTTP server; cannot run via `file://`.

- Should include graceful fallback for failed fetch requests.

Navigation Behavior (`navbar.js`)

Purpose: Handles link highlighting, scroll behavior, and mobile nav interactions.
Best Practice:

- Use `IntersectionObserver` instead of scroll events for better performance.

Small Page Interactions (`main.js`)

Purpose: Provides enhancements such as smooth scrolling and keyboard accessibility.

Reproducible Setup (Windows)

- `git clone https://github.com/abhiakkal/project.git`

    - `c:\Users\abhia\webdev\project`
    - `cd c:\Users\abhia\webdev\project`
    - `python -m http.server 8000`

Open:
`http://localhost:8000/index.html`
Verify successful component injection in the browser console.

Deployment Notes

- GitHub Pages may require relative paths if not using a custom domain.
- Test deployments using a dedicated `gh-pages` branch or GitHub Actions.
- Subpath deployments require removing leading slashes from asset URLs.

Common Issues and Solutions

- **Fetch errors:** Always use a local server.
- **Broken images:** Confirm paths and remove unintended leading slashes.
- **Uneven card layouts:** Use flexbox with auto expansion.
- **Contrast issues:** Validate using Lighthouse or WebAIM contrast checker.
- **New tab links:** Include `rel="noopener"` for security.

External References

MDN Web Docs
Bootstrap Documentation
Lighthouse Tools
WebAIM Resources
A11Y Project
Wikipedia (15-puzzle solvability)
GitHub Pages Documentation
Stack Overflow (CORS and path issues)
Java/Gradle Tutorials
Godot/Unity Documentation (related project research)

Additional Concepts Learned

- Dynamic component loading using fetch-based injection
- IntersectionObserver for performant scroll-driven UI changes
- Image optimization strategies (WebP and `srcset`)
- 15-puzzle inversion-count algorithm
- Path management across multiple hosting environments
    -