Edgar Rafael Jr. – TN42

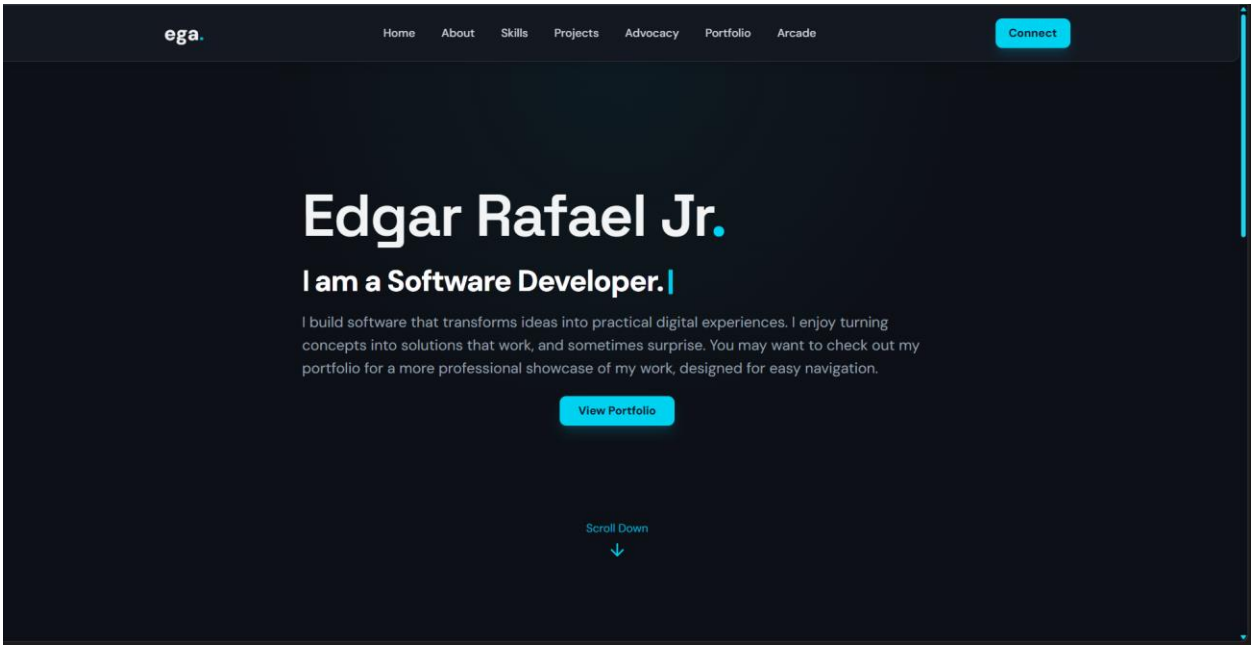
Live now: https://edgarrafaeljr.vercel.app/

Github: https://github.com/usergaia/usergaia-website/

Table of Contents

1. Introduction
2. WebApp Navigation Pages
3. Functionalities / Features
4. Development Overview
    - 4.1 Tech Stack
    - 4.2 Key Components & Architecture
5. Code Navigation
    - 5.1 Folder Structure
    - 5.2 Folder Description

Portfolio Website Documentation



1. Introduction

My website showcases my personal background, skills, and projects. I built it using React for structure instead of traditional HTML, TypeScript for functionality instead of plain JavaScript, and TailwindCSS for styling instead of standard CSS. This combination builds on the basics of **HTML, CSS, and JS,** while offering several advantages.

React's component-based approach allows me to create reusable UI elements easier, making the code more manageable and scalable. TypeScript adds static typing to JavaScript, which not only catches errors early but also improves developer productivity and overall coding quality by acting as a linter for mismatched values and types. TailwindCSS, a utility-first CSS framework, lets me style elements directly in the markup with utility classes without having a need to create an external CSS file, simplifying styling and reducing repetitive code, which enhances coding efficiency and productivity.

Together, these tools reflect modern web development practices and give me experience and opportunity to improve with widely used, in-demand technologies and stack.

I'm also planning a Single Page Application (SPA) version of the portfolio, available on the Portfolio page. It will contain the same content but in a cleaner, more professional, and streamlined layout, making it easier for visitors to navigate and explore my work.

2. WebApp Navigation Pages

- **Home**
  - **Description:** Features a hero section with a CLI-inspired introduction to me and the website.
  - **Navigation:**
    - "Home" button on the navigation bar
    - Direct URL: `/`
- **About**
  - **Description:** Includes information about my background, education, career milestones, and awards.
  - **Navigation:**
    - "About" button on the navigation bar
    - Direct URL: `/about`
- **Skills**
  - **Description:** Lists certifications, technical stacks, and key proficiencies.
  - **Navigation:**
    - "Skills" button on the navigation bar
    - Direct URL: `/skills`
- **Projects**
  - **Description:** Features personal, collaborative, and contributor projects with detailed highlights and information.
  - **Navigation:**
    - "Projects" button on the navigation bar
    - "View Projects" button in the Home page's CLI section
    - Direct URL: `/projects`
- **Advocacy**
  - **Description:** Contains my guiding principles in coding and software development.
  - **Navigation:**
    - "Advocacy" button on the navigation bar
    - Direct URL: `/advocacy`
- **Portfolio**
  - **Description:** Provides an SPA (Single Page Application) version of the website for a streamlined, professional presentation. **[WIP]**
  - **Navigation:**
    - "View Portfolio" button in the hero section
    - "Portfolio" button on the navigation bar
    - Direct URL: `/portfolio`

- **Arcade**
    - **Description:** Showcases some games I play to add more personality in the webapp.
    - **Navigation:**
        - "Arcade" button on the navigation bar
        - "View Arcade" link in the About section
        - Direct URL: `/arcade`

3. Functionalities / Features

- **Mobile Responsiveness:**
    - Navigation includes a hamburger dropdown overlay for mobile devices for easier access.
- **CLI-Terminal Feature:**
    - Terminal-like section showing personal trivia.
    - Includes a "fortune" feature that fetches quotes using the API from `https://dummyjson.com/docs/quotes`.
    - Optional and not required for hirers.
- **Animations:**
    - Page transitions
    - On-view component animations
    - Hover effects
    - Implemented using `Framer Motion`, `TailwindCSS`, and `ShadCN` components from `Reactbits`.
- **Interactive Components:**
    - Some components redirect to related links or showcase additional content.
- **Paginations:**
    - Used for Certifications and Tech Stacks sections to organize content clearly.

4. Development Overview

4.1 Tech Stack

Frontend:

- `Next.js`
- `React`

- TypeScript
- TailwindCSS

Libraries / DevTools:

- Framer Motion
- ShadCN components (Reactbits)
- Lucide React
- React Icons
- ESLint
- Prettier
- clsx

4.2 Key Components & Architecture

- **Page Components:**
  - Located under: /app, /about, /skills, /projects, /advocacy, /portfolio, /arcade
  - Each page has its own page.tsx file, which handles page routing and allows Next.js to manage implicit (file-based) routing automatically.
- **Data Handling:**
  - All data is stored in the data directory.
  - Each category has its own TypeScript file for organized and scalable data fetching.

5. Code Navigation

5.1 Folder Structure

```
/src
├── app
│   ├── about/
│   ├── projects/
│   ├── skills/
│   ├── advocacy/
│   ├── portfolio/
│   └── arcade/
├── components/
```

```
├── data/
└── types/
```

5.2 Folder Descriptions

- **`/app`**
  - Contains all main page directories. Each subfolder corresponds to a page and includes its `page.tsx` file for routing and page content.
  - Example:
    - `about/` – Contains the About page and related components.
    - `projects/` – Contains the Projects page.
    - `skills/` – Contains the Skills page.
    - `advocacy/` – Contains the Advocacy page.
    - `portfolio/` – Contains the SPA (Portfolio) page for professional viewing.
    - `arcade/` – Contains the Arcade page with games and interactive elements.
- **`/components`**
  - Contains reusable React components used across multiple pages and section components for the `page.tsx` in `/app.`
- **`/data`**
  - Contains structured data used by the site, organized by category.
- **`/types`**
  - Contains TypeScript type definitions used across the project to enforce type safety and structure:
    - `ProjectType.ts` – Defines the structure of project objects.
    - `SkillType.ts` – Defines the structure of skill objects.