Professional Portfolio

Assignment 4 - Personal Website Project

Showcasing Advanced Web Development Skills

Yousef Alhadlaq
KFUPM SWE363

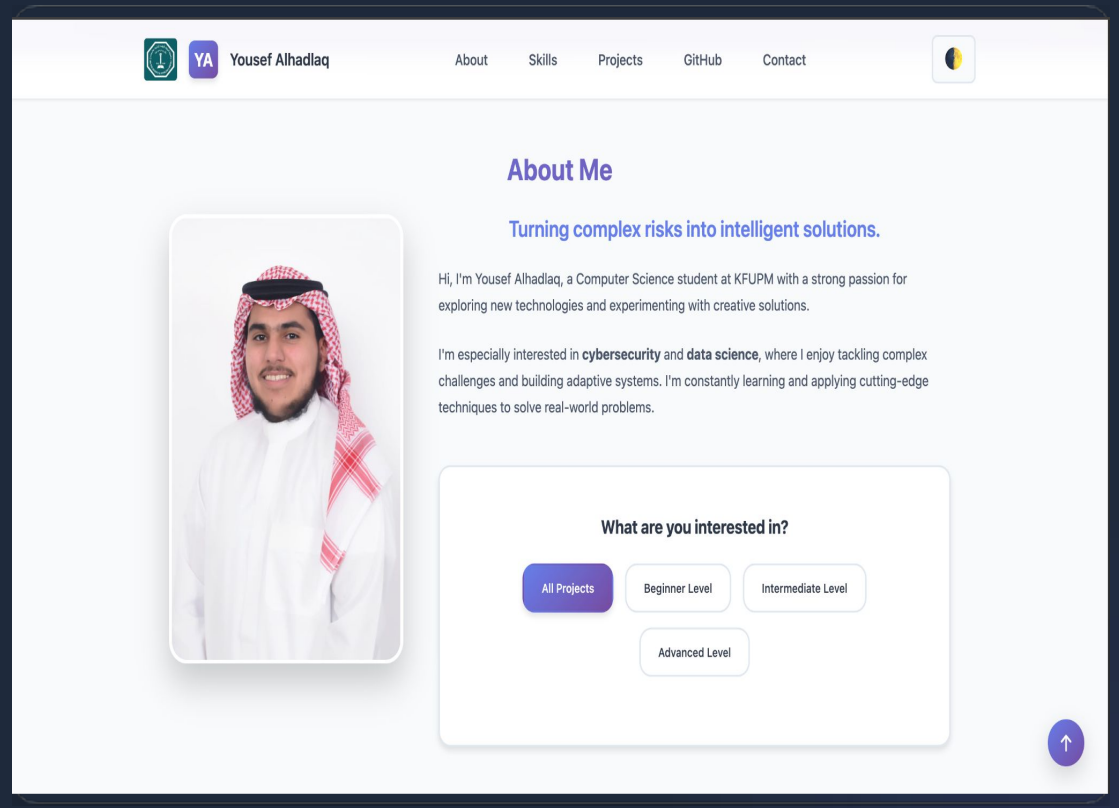
Project Overview

Objective

Develop a professional, responsive personal portfolio website to showcase academic projects and technical skills.

Core Philosophy

- **Mobile-First Design:** Prioritizing experience on smaller screens.
- **Performance:** Optimized asset loading and clean code.
- **Interactivity:** Engaging animations and real-time feedback.



Technology Stack

HTML5

Semantic Structure
~600 Lines

CSS3

Grid & Flexbox
CSS Variables
~1000 Lines

JavaScript

ES6+ Features
Async/Await
~700 Lines

GitHub

REST API Integration
Pages Deployment

Key Features: User Experience

Download CV

A prominently placed, professionally styled button allowing recruiters to instantly access my resume. Implements a direct download attribute for seamless access.

Animated Skills

Interactive visualization of 12 key technical skills categorized into Frontend, Backend, and Tools. Progress bars animate upon scrolling into view using Intersection Observer.

Key Features: Functionality

Live GitHub Projects

Real-time integration with the GitHub API. The portfolio automatically fetches and displays my latest public repositories, ensuring content is always up-to-date without manual edits.

Theme Toggle

A persistent Dark/Light mode toggle that saves user preference to localStorage. Uses CSS variables for instant global color updates without page reloads.

UI/UX Improvements

Visual Polish

Moved away from generic Bootstrap styles to a custom design language focusing on readability and modern aesthetics.

- **Typography:** Implemented 'Outfit' for headings and 'Inter' for body text for better hierarchy.
- **Color Scheme:** Adopted a cohesive purple/blue gradient palette to convey creativity and tech competence.
- **Feedback:** Added loading spinners and error states for all asynchronous operations.

Code Highlights: JavaScript

API Data Fetching

Demonstrating clean asynchronous code using async/await. This snippet handles the GitHub API request, checks for response validity, and parses the JSON data.

Key Concepts:

- Error Handling (try/catch)
- DOM Manipulation
- Loading State Management

```js
async function fetchProjects() {
  try {
    const response = await fetch(API_URL);
    if (!response.ok) throw new Error('Failed');
    const data = await response.json();
    renderProjects(data);
  } catch (error) {
    showError('Could not load projects.');
  } finally {
    hideLoader();
  }
}
```

Code Highlights: CSS Animations

```css
.skill-bar-fill {
  width: 0;
  transition: width 1.5s cubic-bezier(0.22, 1, 0.36, 1);
}

.slide-in {
  animation: slideUp 0.8s forwards;
  opacity: 0;
}

@keyframes slideUp {
  from { transform: translateY(30px); opacity: 0; }
  to { transform: translateY(0); opacity: 1; }
}
```

Smooth Interactions

CSS transitions and keyframe animations bring the static interface to life without relying on heavy JavaScript libraries.

Techniques:

- **Cubic-Bezier:** Custom timing functions for natural-feeling movement.
- **Hardware Acceleration:** animating transform and opacity for 60fps performance.
- **Observer Pattern:** Classes like .slide-in are triggered via JS IntersectionObserver.

Responsive Design

Mobile-First Approach

The site was built starting from mobile viewports, ensuring core content is accessible on small screens before scaling up.

- **Hamburger Menu:** A custom-built collapsible navigation menu for screens under 768px.
- **Flexible Grids:** Project cards use grid-template-columns: repeat(auto-fit, minmax(300px, 1fr)) to automatically adjust to
- available width.

Touch Targets: Buttons and links sized appropriately for finger tapping.

API Integration & Data Flow

1. User Visit

Page Load triggers the init() function in JavaScript.

2. Request

fetch() sends a GET request to api.github.com/users/yousef/rep

3. Process

JSON response is filtered for specific projects and sorted by date.

4. Render

HTML cards are dynamically generated and injected into the DOM.

Challenges & Solutions

Theme Persistence

Challenge: Theme reverting to light mode on page refresh.

Solution: Used localStorage to store the user's preference string and check it immediately upon DOM load.

Async Data Delays

Challenge: Blank space appearing while GitHub data fetched.

Solution: Implemented a "Skeleton Loading" state to provide visual feedback before content arrives.

Mobile Menu

Challenge: Menu state management on resize.

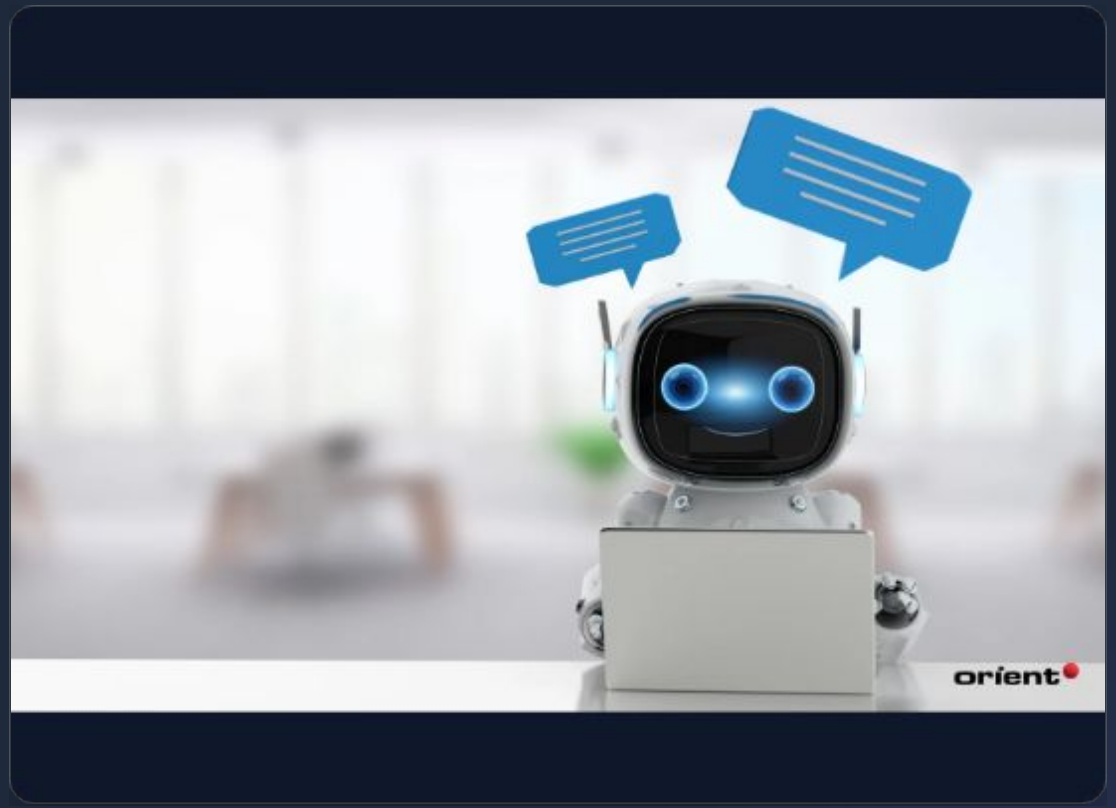
Solution: Added event listeners to close the mobile menu automatically if the window is resized to desktop width.

AI Usage & Learning Outcomes

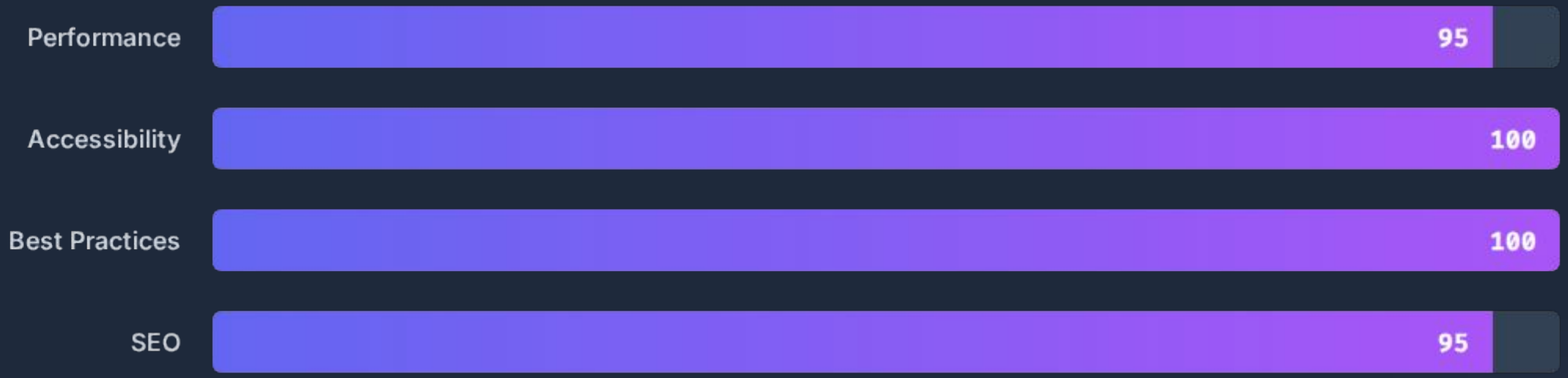
Leveraging AI Tools

Used tools like GitHub Copilot and ChatGPT for:

- **Debugging:** Quickly identifying syntax errors in complex logic.
- **Refactoring:** optimizing CSS selectors for better maintainability.
- **Regex Generation:** Creating validation patterns for email forms.



Testing & Performance



Lighthouse Audit Scores

Live Demo

View the project live on GitHub Pages:

https://yousefalhadlaq.github.io/assignment-4/

Questions?

Thank you for your time.

youssefalhadlaq@gmail.com