SWE 363

Personal Portfolio Website

A responsive, interactive single-page application built with modern web technologies.

Ali Almatrook

s202267500

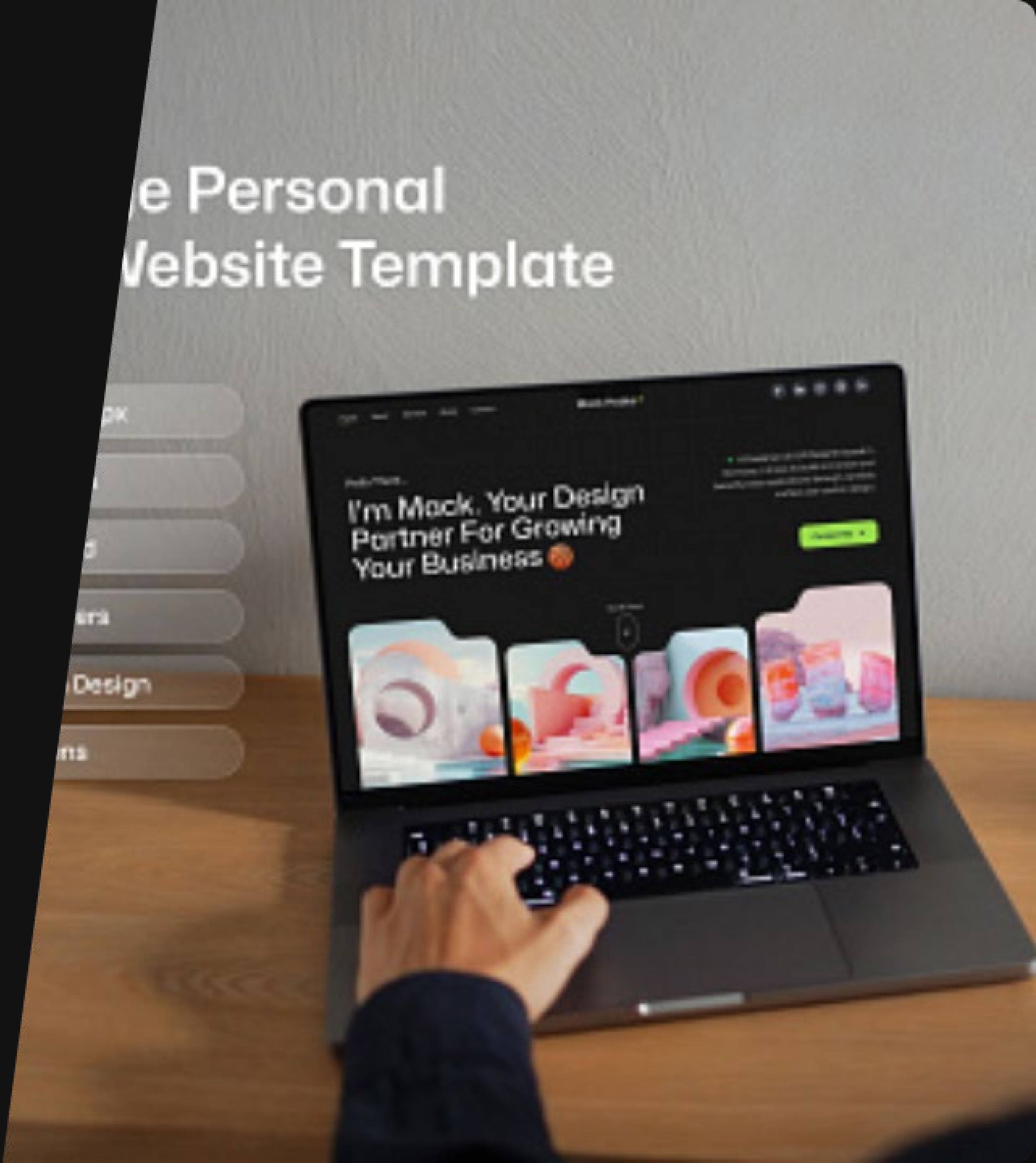
Project Overview

The Digital Hub

This project is a dynamic, single-page portfolio application designed to serve as a professional hub for my academic and personal projects.

Core Philosophy

Built on the principles of **semantic HTML**, **modular CSS**, and **Vanilla JavaScript**, the site demonstrates full-stack potential without relying on heavy frameworks.



Objectives & Motivation

Showcase Mastery

Demonstrate deep understanding of fundamental web technologies including DOM manipulation, CSS Grid/Flexbox, and ES 6+ JavaScript.

Living Resume

Create a centralized, interactive platform to display my projects (Python Banking, Java Game) to potential employers for internship opportunities.

State Management

Challenge myself to build a robust state management system (theming, user preferences) using only client - side storage mechanisms.

Technical Architecture

Frontend Stack

Built with **Semantic HTML 5** for accessibility and SEO. Styling utilizes **CSS3 Variables** for efficient theming and complex Flexbox/Grid layouts for responsiveness. No external CSS frameworks were used.

Logic & Data

Powered by **Vanilla JavaScript**. Implements an appState object for centralized state management. Uses async/await for the GitHub REST API and localStorage for data persistence.

Key Features & Live Demo

Toggles between Dark/Light modes and persists preference via Local Storage.

"Set Name" feature updates the greeting dynamically and remembers the visitor.

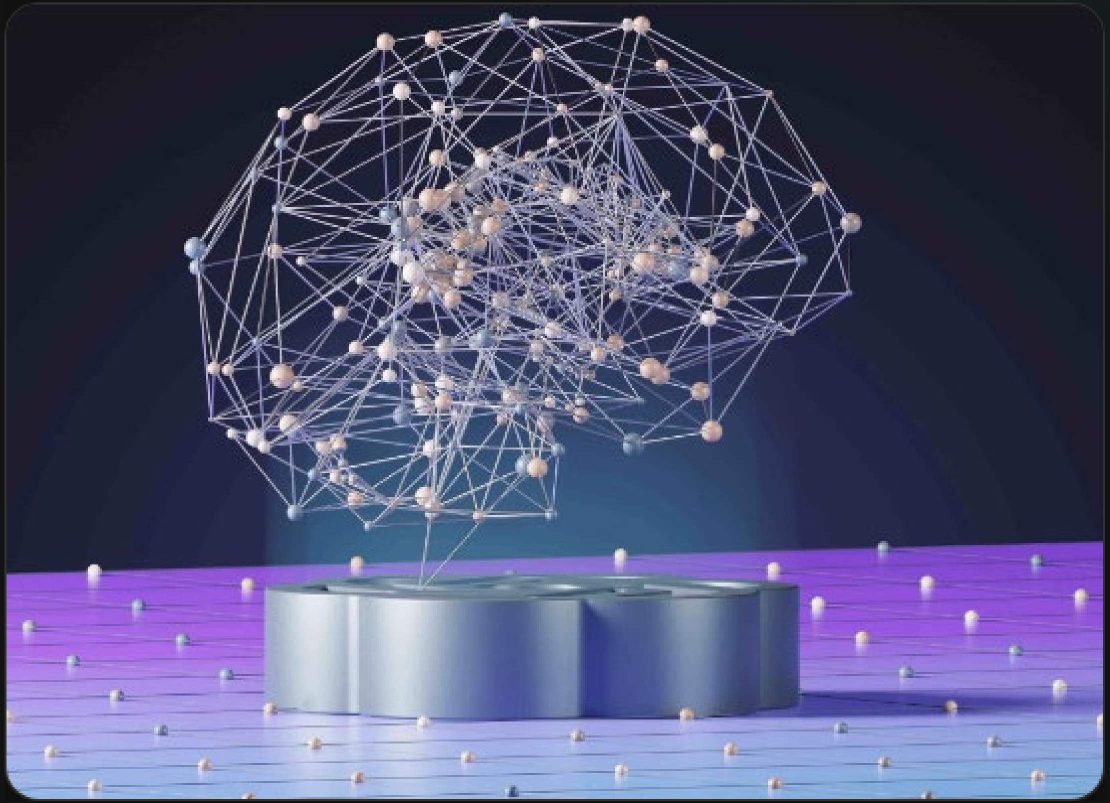
Projects can be filtered by category (Web, Game, Desktop) and sorted by date.

Fetches and displays real-time repository data from GitHub using the Fetch API.

AI Integration

Accelerated development by generating boilerplate code for event listeners and modal logic, reducing repetitive typing.

Assisted in refactoring the appState logic for cleaner state management and generated comprehensive documentation.



Deep Dive: State Management

The Challenge

Creating a personalized experience (remembering if a user hid projects or set a name) without a backend database is difficult. A simple refresh usually wipes all application state, leading to a poor user experience.

The Solution

I implemented a getStoredBool helper function and a unified appState object. This object acts as a "single source of truth," instantly syncing any change (like toggling a button) to the browser's localStorage.

Efficient Data Handling

Modular Logic

The project uses a data -driven approach for the UI. Instead of hardcoding HTML for every project, a JavaScript array of objects stores the data.

This allows for O(n) filtering and sorting operations. The renderProjects() function dynamically rebuilds the DOM based on the current filter state, ensuring high performance.

Conclusion & Future Roadmap

●

Current Status

Deployed single -page app with 95%+ Lighthouse performance and accessibility scores.

○

Phase 2

Connect the contact form to a NodeJS/Express backend for real email handling.

○

Phase 3

Integrate a Headless CMS (like Strapi) to add a dynamic blog section.

Thank you

Thank you for your attention.

✉ s202267500 @kfupm.edu.sa