Gisma University of Applied Sciences

Department of Computer and Data Sciences

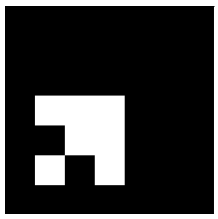
Individual Final Project

Student Portolio Project Report

Ali M Abdou

GH1033452

SS0325



Gisma University of Applied Sciences
Department of Computer and Data Sciences

Paper Title

Individual Final Project: Student Portolio Project Report

GitHub Repository

`https://github.com/quack-b1/quack-b1.github.io`

Portolio Link

`https://aliabdou.de`

Report by:

Ali Mohamed Abdou GH1033452 alimohamed.fathi@gisma-student.com

Submitted in fulfillment of the final assessment for the module

B201 Computer Science Lab

Lecturer

William Baker Morrison

Module Leader

Prof. Dr. Mohammad Mahdavi

Submission Quarter

SS0325

I confirm that this project report is my own work and that I have documented all sources and materials used.

Berlin, 1 July 2025

Word Count: 1,767

Contents

1. Introduction

When applying for a Working Student position in computer science, a well-organized technical portfolio is essential for demonstrating both skills and professionalism. This report outlines the development of my computer science portfolio, accessible at `https://aliabdou.de`, and publicly hosted on GitHub Pages via the repository `https://github.com/quack-b1/quack-b1.github.io`.

The primary objective of the portfolio is to exemplify my academic credentials, technical expertise, and selected projects in a professional and accessible manner. Designed as a single-page, responsive website with multilingual capabilities, it utilizes contemporary web development tools, including Jekyll and Bootstrap CSS, to optimize performance, usability, and aesthetic appeal. The GitHub repository preserves a clean and well-structured codebase, adheres to best practices, incorporates a modular architecture, and contains comprehensive documentation to facilitate straightforward updates.

This report offers a comprehensive analysis of the portfolio's structure, design decisions, and implementation methodology. It additionally evaluates the strengths and limitations inherent in the current iteration and suggests potential improvements for the future. Through this initiative, I demonstrate my ability to translate theoretical knowledge into practical, goal-oriented applications, a vital competency for any aspiring computer scientist.

2. Portfolio Structure

2.1. Website Overview

The portfolio website adopts a one-page layout that emphasizes readability, responsiveness, and intuitive navigation. The design is centered on enhancing user experience and logical progression, thereby functioning as a clutter-free developer portfolio. The website features the following sections:

- **Hero Section**: Features a profile image, name, and contact icons linking to GitHub, LinkedIn, and email.
- **About Me**: A quick and efficient introduction to my background, interests, and career goals.
- **Experience**: Lists relevant internships and roles with brief descriptions, dates, and locations..
- **Education**: Highlights my academic qualifications, including the IB Diploma and current BSc studies.
- **Certifications**: Displays key technical and extracurricular certifications using a clean list layout.
- **Projects**: Showcases selected personal and academic projects with a carousel interface for interactivity.
- **CV Download**: A download button that provides access to my LaTeX-generated curriculum vitae in PDF format.
- **Language Toggle**: Users can seamlessly switch between English and German using a toggle button.

2.2. Repository Overview

The GitHub repository is structured to reflect a clean separation of concerns, improving maintainability and collaboration. The leading directories and files include:

- `/_includes`:
  - `/sections` and `.sectuibs_de`: Markdown content files for each section in English and German, respectively.
  - `project-carousel.html`: The single file including HTML, CSS, and JavaScript that enables projects to appear as tiles in a carousel on the main page.
- `/_layouts`:

  - `default.html`: The base layout template used to render the site.
  - `error.html`: The error layout template used to render the 404 Not Found page.
  - `projects.html`: The layout template for each project's page (project1, project2, etc.).
- `/_projects`: Contains different project markdown files, such as project1.md
- `/_assets`:
  - `/css/styles.css`: The main stylesheet using manual CSS modifications and external formatting from Bootstrap.
  - `/cv`: Contains the LaTeX file (.tex) and PDF output for my downloadable CV.
  - `/images`: All images used in the site, such as my profile picture and project thumbnails.
  - `/flags`: Contains the Deutschland and United Kingdom flags in SVG format for use in the language switcher
- `404.html`
- `_config.yml`: Jekyll configuration file defining site metadata and build behavior.
- `index.md` and `index_en.md`: Entry points for the German and English homepages, written in Markdown.

Other notable files:

- `.gitignore`: Lists files and directories excluded from version control.
- `CNAME`: Used for domain configuration with GitHub Pages.
- `Gemfile` and `Gemfile.lock`: Specify Ruby gems like `jekyll` and `minima` required to build the site.

- `README.md`: A brief description of the repository, currently empty apart from a short title and copyright.

Version control is actively used to track changes, maintain historical versions, and ensure deployment consistency. The repository is configured to deploy automatically via GitHub Pages from the main branch.

3. Design and Implementation

The design and implementation of my portfolio website aim to establish a professional, clean, and responsive platform that accurately reflects both my technical competencies and personal brand. The emphasis was placed on usability, modularity, and efficient deployment.

3.1. Design Justifications

The design approach follows a minimalist aesthetic that prioritizes clarity and accessibility. The layout is a single-page scrollable format, which ensures that the entire content is easily discoverable without requiring navigation between pages. The visual hierarchy is maintained through font scaling, spacing, and component layout, utilizing the Jekyll Minima theme and Bootstrap CSS. Key design choices include:

- **Color Scheme**: A fake-black background with white text was selected to reflect a calm and professional tone while maintaining readability across different devices. The overall dark theme coincides with every programmer's favorite code editor theme (the light attracts bugs!).
- **Typography**: A simple sans-serif font ensures legibility across devices, eliminating any possibility of design flaws due to unavailable fonts.
- **Layout**: The use of a one-pager with all content ensures that the website is confined to precisely what it was designed for: a portfolio. The project carousel ensures that an infinite number of projects do not compromise the simplistic design of the portfolio; instead, they are presented on a single line rather than a long list.

- **Responsiveness**: The use of a Jekyll theme ensures that the design adjusts gracefully for mobile, tablet, and desktop screens.

These design choices were informed by modern UI/UX principles and best practices. I asked myself what I would look for in a candidate's portfolio if I were a recruiter and attempted to create the simplest, most straightforward portfolio I could think of, given the considerable time constraints.

3.2. Tools and Technologies Used

- **Jekyll**: A static site generator that integrates seamlessly with GitHub Pages. It enables modular content through layouts, _includes, and Markdown-based sections. This eliminates the need for tedious CSS configuration and writing HTML from scratch, as the Jekyll package automatically converts Markdown files into HTML files upon rendering in a web environment.

- **GitHub Pages**: Used for free and reliable hosting, with continuous and automatic deployment triggered by any push to the main branch of the repository.

- **LaTeX**: Used to generate a clean, typographically sound CV. The resulting PDF is hyperlinked in the portfolio and available for download. LaTeX was also used to create this very same report.

- **Git**: All developments were version-controlled through Git, with regular commits and rollbacks for experimentation and failed design attempts. I specifically used GitHub Desktop to manage all commits and version control, as it provides a graphical interface to interact with Git, rather than using a command-line interface (CLI).

3.3. Multilingual Support

To make the portfolio accessible to both local (German-speaking) and international audiences, a language switcher is placed at the top of the site. Content is manually translated into different languages on various pages, allowing users to switch between English and German versions with a single page reload. This feature demonstrates both

technical skill and cultural sensitivity. I initially attempted to have the page dynamically translated using JSON files, but this would require abandoning the Jekyll theme and converting all the site's content to pure HTML5. Due to time constraints and other commitments, I chose the easier option of providing English and German versions on separate permalinks (`/` for German, `/en` for English).

4. Reflection and Critical Evaluation

Constructing this portfolio website not only demonstrated my technical competencies but also functioned as a practical exercise in full-stack web development, project planning, and self-assessment. This section assesses the project's strengths, identifies its limitations, and recommends future enhancements.

4.1. Strengths

- **Professional Design and User Experience**

  The use of Bootstrap CSS and a modular design structure allowed for a clean, consistent, and responsive layout. The single-page structure ensures ease of navigation and adds a polished, modern feel without overwhelming the user.

- **Multilingual Accessibility**

  Implementing seamless English-German language toggling demonstrates both technical flexibility and cultural awareness. This feature is particularly valuable for communicating with a broader audience in Germany and beyond.

- **Clean Codebase and Repository Organization**

  The repository is logically structured, with content, layouts, assets, and configuration files separated clearly. This improves readability, maintainability, and scalability for future development or collaboration.

- **Static Deployment Efficiency**

  Hosting via GitHub Pages ensures low-latency access, continuous deployment, and version control, all without additional infrastructure costs. The static nature of the

site also improves security and performance.

- **Professional Integration of Tools**

  By incorporating LaTeX for CV generation, Jekyll for static site generation, Bootstrap for design, and Git for version control, the portfolio demonstrates a mature understanding of multiple developer workflows and toolchains.

4.2. Weaknesses

- **Limited Interactivity and Backend Functionality**

  As a static site, it does not include a contact form, database interaction, or authentication system. While this approach works well for a portfolio, the lack of interactivity may be limited if additional features, such as messaging or analytics, are added.

- **No Real-Time Content Management**

  Adding or editing content requires editing Markdown or HTML files and pushing changes via Git. This makes updating slightly less accessible for non-technical users, preventing real-time content updates.

- **Language Support Scope**

  The multilingual feature currently supports only English and German. Extending this to support additional languages would require significant scaling of the translation infrastructure.

4.3. Future Enhancements

- **Add Contact Form with Backend Support**

  Integrate a form handler, such as Formspree or Netlify Forms, or a simple backend serverless function to allow potential employers to contact me directly rather than have to open their email client and send an email.

- **Incorporate a Lightweight Content Management System (CMS)**

  Tools like Netlify CMS or Forestry.io could be integrated to allow editing content

through a GUI, simplifying future updates.

- **SEO Optimization and Analytics**

  Add metadata for search engine indexing and integrate tools like Google Analytics or Plausible to gain insights into visitor traffic and engagement.

- **Project Filtering and Tags**

  Enable filtering by technology, date, or category in the Projects section to improve the discoverability of relevant work.

- **Site Theme**

  Ultimately, remake the entire portfolio using HTML or a Tailwind CSS theme, eliminating Jekyll. This would allow more customization and better presentation, allowing the incorporation of a more technological theme.

5. Conclusion

My personal computer science portfolio at `https://aliabdou.de` has served as a comprehensive demonstration of my ability to apply theoretical knowledge within a professional context. This project has allowed me to highlight a diverse array of technical competencies, including web development, UI/UX design, content organization, version control, and deployment methodologies. Utilizing Jekyll, Bootstrap CSS, and GitHub Pages, I developed a swift, scalable, and aesthetically consistent portfolio that proficiently presents my academic credentials, certifications, technical projects, and career aspirations. Moreover, the incorporation of multilingual support, interactive functionalities, and a well-structured codebase further enhances the portfolio's professional significance.

Critical reflection on the process has facilitated the identification of areas for future development, notably in backend integration, content management, and user interaction. The project constitutes a robust foundation that I intend to further expand as I gain additional experience and skills during my computer science studies and in future endeavors. Overall, this portfolio represents not only a technical achievement but also

a personal milestone, synthesizing the knowledge acquired thus far with my professional aspirations.