ABDELHADI AMHAMDI

FULL STACK ENGINEER

Marrakech | aamhamdi943@gmail.com | +212 690 73 14 52 | aamhamdi.me

linkedin.com/in/abdelhadi-amhamdi/ | github.com/Abdelhadi-Amhamdi | leetcode/Abdelhadi-Amhamdi

SUMMARY

I am a passionate computer science student with a deep interest in software development, web technologies, and problem solving. I am always eager to learn new things and apply my knowledge to build efficient and scalable solutions.

EDUCATION

1337 CODING SCHOOL, khouribga

Coursework: c , c++ , linux, unix, system administration, computer graphic, data structures, javascript, typescript , python, reactJs, nextJs, tailwindCss, django, docker, react-native, etc...

PROJECTS

CodeRead Mobile Application

This React Native app consumes the DEV.to public API to display a list of developer articles. Implements dynamic tag filtering, enabling users to tailor their feed based on selected technologies or topics. The app showcases efficient state management, optimized API calls, and user-friendly design principles suitable for content-heavy mobile apps.

Tools Used: **Typescript, React-native**

React and Django-Driven Gaming Hub

This project involves building a full-stack single-page application (SPA) using React and TailwindCSS for the frontend, with Django as the backend. The app offers a variety of interactive features:

1. User Authentication: Log in and manage accounts.
2. Social Features: Send friend requests, chat with friends, and engage in real-time communication.
3. Ping Pong Game: Play ping pong in various modes
4. A user profile page with an elegant and data-rich dashboard.
5. A settings page for personal preferences.

The backend uses PostgreSQL for database management and Redis for real-time functionalities, particularly with Django Channels to handle WebSocket connections. The entire application is containerized using Docker, providing a scalable and efficient environment for development and deployment.

Tools Used: **Html, Tailwindcss, Typescript, ReactJs, Django Rest Framework**

Multi-Container Web Solution

This project focuses on Docker and containerization, aiming to build an infrastructure application using multiple Docker containers. The primary requirements include:

- Running an Nginx container with TLS for secure connections
- Deploying a WordPress container with PHP-FPM for dynamic content
- Setting up a MariaDB container for database management
- Creating two volumes: one for WordPress files and another for the database
- Adminer for database management
- Redis for caching (configured with WordPress)
- FTP for file transfers
- Node.js container for a separate web application

- cAdvisor for monitoring container performance

Beyond practical implementation, this project delves into Docker's architecture and its evolution as the leading container technology. You will explore the container creation process—from the initial Docker client request to the daemon, and how dockerd, runc, and the shim manage container operations.

Tools Used: **docker, docker-compose,bash, nginx , mariadb, wordpress**

C++ChatServer: A Lightweight IRC Solution

This project is a fully functional IRC (Internet Relay Chat) server developed in C++ using socket programming. The server handles multiple simultaneous client connections, enabling real-time text communication in a lightweight and efficient manner. With support for private messages, public chat rooms, and basic IRC commands.

1. Authentication Commands
2. Channel Manipulation Commands
3. Other Commands

The server is built from the ground up using raw socket APIs, offering low-level control over network communications and focusing on high performance and scalability. Ideal for those interested in understanding network protocols, the project showcases the fundamentals of concurrent client handling, message parsing, and protocol adherence in a structured, robust C++ environment.

Tools Used: **C++, C++ standard containers, sockets**

MinShell: A Lightweight Unix Shell

This project focuses on creating a simple shell that supports both built-in commands and external command execution. Key features include: (command execution, redirection, pipes logical operators and wildcards) The project is divided into two main parts:

1. **Parsing:**
   - Display a prompt for the user to input commands.
   - Tokenize and parse the input.
   - Build an Abstract Syntax Tree (AST) to represent the structure of the commands.
2. **Execution:**
   - Receive the AST and iterate over it to execute commands.
   - Handle all mentioned features, including redirection, pipes, logical operators, and wildcards.

This shell provides foundational experience in command parsing, process management, and execution flow control in a Unix-like environment.

Tools Used: **C programing language, ast (abstract syntax tree), linked list**

TECHNOLOGIES

Languages: C++, C, Javascript, Typescript, Python, Bash

Technologies: ReactJs, React-Native, NextJs, Django, Docker, Docker-compose, Tailwindcss, Linux, Unix