Safet Imamović

Software Engineer

safet.imamovic.22@size.ba

+387 60 34 99 752

Dr. Ćire Truhelke 10 - C

B, B1

safet.dev

linkedin.com/in/safeti

github.com/SafetImamovic

PROFESSIONAL EXPERIENCE

Backend Developer Intern
08/2024 - 09/2024

Symphony SA
Sarajevo (Remote)

- **Tech:** Python(FastAPI), Pydantic, AIemble, Docker, Postman, PyTest.
- Built a RESTful API with **FastAPI**, leveraging **Pydantic** for data validation and **Alembic** for database migrations.
- Implemented comprehensive testing with **Postman** (integration) and **PyTest** (unit/integration tests).
- Fully containerized using **Docker** for seamless deployment.
- **GitHub Repository**

Bank Promoter
03/2022 - 05/2022

UniCredit Bank
Zenica, Bosnia

Sharing promotional material for the banks products.

EDUCATION

Software Engineering
10/2022

University of Zenica, Polytechnic Faculty
Zenica, Bosnia

- Successfully completed all coursework in a 3-year program.
- Bachelor's thesis in **Computer Graphics**: *Web 3D Render Engine using Rust & WebGPU*.
- Planned: **Master's in Software Engineering** (Expected 2027)

CERTIFICATES

freeCodeCamp: Relational Database

freeCodeCamp: JavaScript Algorithms & Data Structures

PROJECTS

ForgeAI - AI-Powered MIDI Generation Tool (VST Prototype)

Tech: Python (Django), JavaScript, Google Gemini API, Stripe, Custom MIDI Algorithms

Developed a **VST plugin prototype** (web-based) that generates MIDI files from text prompts using **Google Gemini AI**.

Core Features:

- Text-to-MIDI conversion via **custom algorithms** (JSON → MIDI).
- Secure auth/subscription system (**Stripe integration**).
- End-to-end pipeline: User input → AI processing → MIDI download.

Architecture:

- **Frontend**: Interactive UI (HTML/CSS/JS).
- **Backend**: Django server handling API calls, data processing, and user management.

Terminal 3D Render Engine (C, CMake)

Tech: C, CMake, ANSI Escape Codes.

Developed a **cross-platform** 3D render engine that outputs directly to terminal.

Pure C implementation using ANSI escape codes for rendering.

Platform-agnostic design:

- Windows/Linux support via **compile-time directives**
- Native console handling for each OS

CMake/Make build system for portable compilation.

Features real-time rendering with optimized display refresh.

Decibel - .NET Music Streaming Service

Designed and deployed a **full-featured music streaming platform** with secure authentication, playlist management, and real-time playback.

Tech: C#, .NET, Entity Framework Core, MS SQL Server, Azure DevOps, Docker.

Key Contributions:

- Built RESTful APIs with **.NET Core** and optimized SQL queries via **Entity Framework**.
- Managed deployments via **SmarterASP** (PaaS) with zero downtime.
- Led project lifecycle using **Azure DevOps**: Agile sprints, version control, and CI/CD pipelines.
- Fully containerized for easy deployment.

Engineered a high-performance backend system featuring:

- **Double Circular Linked List** implementation via stored procedures
- O(1) complexity for critical playlist operations (reordering, track insertion)
- Optimized pagination for large media libraries

Database Innovations:

- Designed SQL Server schema with procedural linked list logic
- Balanced relational integrity with performance needs

System Performance:

- Achieved constant response times for playlist modifications

Note: Proprietary system developed under NDA

Music Streaming Service (Web App) 🔗

Tech: Supabase (PostgreSQL), React/Next.js.

Developed a **Supabase-backed** music streaming platform with user authentication, playlist creation, and artist/album following.

Features:

- Tech: **Supabase (PostgreSQL), React/Next.js**

Audio Player Application (C++ & SFML) 🔗

Tech: C++, SFML, Custom GUI Library.

Designed and implemented an **object-oriented** music player in **C++** using the **SFML** framework for audio handling and GUI.

Key features:

- File system navigation for audio tracks
- Playback controls (play/pause, volume adjustment, track skipping)
- Lightweight, performance-optimized architecture

GRIT (General Rust Interface Tool) 🔗

Tech: Rust.

Designed and built a **modular**, general-purpose command-line toolkit in Rust

Core features:

- Extensible architecture for easy command additions
- Unified interface for diverse developer utilities
- Memory-safe implementation leveraging Rust's ownership model

Focused on:

- Developer ergonomics (intuitive subcommands/flags)
- Performance optimization (zero-cost abstractions)
- Future extensibility (module system)

Embedded Smart Car System 🔗

Tech: Arduino (C++), STM32, HC-SR04, DHT11, I2C OLED, H-bridges, IR (NEC protocol)

- Developed a **dual-MCU embedded system** with Arduino (motor control via PWM) and STM32 (sensor processing).
- Engineered **interrupt-driven communication** between MCUs for seamless mode switching (Drive/Parking).
- Implemented **scalable ultrasonic sensing** (2→N sensors) with haptic feedback (piezo buzzer frequency ∝ obstacle distance).
- Achieved **<5ms response time** for sensor-to-motor actions via optimized ISRs and hardware timers.
- Integrated **environmental monitoring** (DHT11) with live OLED display (SSD1306) using I2C.