QINZHOU SONG

qinzhounick@wustl.edu | LinkedIn | (267) 595-7202

EDUCATION

Washington University in St. Louis — **St. Louis, MO**

Master of Science in Computer Science, GPA: 3.8/4.0 — 08/2022 – 05/2025

Bachelor of Science in Computer Engineering, GPA: 3.8/4.0

Coursework: OOP, Systems Software, Multi-Paradigm in C++, Operating Systems, Computer Architecture, Computer Systems Design, HPC Systems, Digital IC Design and Architecture

Sewanee: The University of the South — **Sewanee, TN**

Bachelor of Science in Computer Science, GPA: 3.5/4.0 — 08/2019 – 05/2022

WORK EXPERIENCES

WashU Mckelvey School of Engineering — **St. Louis, MO**

Firmware Engineer — 05/2023 – Present

- Developed and simulated a noise cancellation algorithm in **C++** using single and double threshold methods, improving localization accuracy by 20%
- Analyzed energy levels of Gamma-ray events using **Vitis HLS** implementing integration and prefixed sum methods that reduced latency below 300 cycles
- Simulated data flow from front end ASICs to FPGA in **Vivado** using FIFOs, PynqMicroblaze, PL control
- Contributed to the development and testing of a cutting-edge Gamma-ray telescope demonstrator

CW Software (A startup providing local news & alerts) — **Shanghai, China**

Software Engineer Intern — 05/2024 – 08/2024

- Built in-memory caching layer using **Redis** for the *Like* feature, to reduce direct communications to persistence layer. Helped to design the Redis data structures to store the *Like* mappings between posts and users. Wrote server-side functions to read/write data via Redis. Implemented primary and replica for availability and failover. Database overhead is reduced by 2% and page load time is increased by 20%
- Wrote stored procedures to feed data into a user preference prediction model based on user *Like* data, to make personalized content recommendations and accurate target advertising
- Analyzed and visualized user growth and usage data for KPI and quarter reports using **Tableau**

PROJECTS

SocketRelay

- Developed a robust client-server application in **C** utilizing TCP/IP sockets and epoll, demonstrating strong operating systems fundamentals, inter-process communication (IPC), and **UNIX** system programming skills.
- Implemented comprehensive error handling and short-read/write recovery mechanisms through low-level system calls, reinforcing practical experience with OS-level networking and resource management.
- Automated multi-client testing using **UNIX shell scripting**, showcasing adeptness in UNIX environments and scripting—with a foundation that easily translates to macOS/iOS development and debugging using tools like lldb and DTrace.

AI Class Copilot App

- Created an iOS app that listens to lectures, extracts insights, and answers questions using **Swift** and **XCode**
- Integrated with **Deepgram API** for transcribing audio into text transcripts and **OpenAI API** to process transcripts for insight extractions and Q&A
- Utilized asynchronous **FastAPI** calls to improve server concurrency and handle multiple I/O-bound requests efficiently, improving the throughput and performance of the iOS app

TECHNICAL SKILLS

Programing Language: Python**,** C++**,** Java, Rust, C#, JavaScript, HTML5, CSS, SQL, Shell Scripting, Node.js, CUDA

Framework: React, NumPy, Pandas, FastAPI, Matplotlib, Unity, Unreal Engine

Machine Learning: TensorFlow, PyTorch, Scikit-Learn

Data Store & Streaming: MySQL, PowerBI, DAX, Tableau

Architecture & Methodology: Microservices, FPGA, Object-Oriented Programming, Caching, Design Patterns, Distributed Systems, Relational Database, Cloud Computing, A/B Testing, EDA

CI/CD & Development Tools: Git, Docker, Visual Studio Code, Eclipse, Anaconda