Kobe Shi

New York, NY | (917) 288-8228 | kobeshi@umich.edu | LinkedIn | Portfolio | Github

Education

University of Michigan Ann Arbor, MI
Bachelor of Science in Computer Science *Aug 2020 – Apr 2023*

Clubs: WolvSec - Cybersecurity, WolvSoft - Game Development, MHackers - Professional Development

Relevant Coursework: Data Structures and Algorithms, Computer Architecture, Theory of Computation, Game Design and Development, Software Engineering, Operating Systems, Web Systems, Introduction to Computer Security

Technical Skills

Languages & Frameworks: C/C++/C#, Python, R/R-Studio, SQL, HTML/CSS/JavaScript, Xml, Xaml, WPF, Unity, ReactJS, Flask, Jinja2, MapReduce - Hadoop, Assembly, bash, JSON, Markdown, LaTeX, Cypress, SAFe

Skills: OOP, Distributed Systems, Multi-Threading, TCP/UDP, Pipelining, REST API, Data Structures & Algorithms

Tools: VS/VSCode, Git, Docker, Ghirda, Wireshark, Linux, Jira, Amazon Web Services, Azure Devops/TFS

Work Experience

Hexagon - Asset Lifecycle Intelligence Huntsville, AL - Remote
Software Developer - Full Time *July 2023 – Present*

- Contribute towards the development of Smart3D (S3D), a large-scale 3D modeling software, while acquiring experience in 3D modeling, through rigorous training, testing, and development.
- Dedication to the development, testing, and debugging of S3D's Rules Configuration Manager in order to deliver tailored solutions to customers, improve the end-user experience, and align with overall industry standards.
- Utilized many tools in the development of S3D, such as C++/C#, WPF, Xml, Xaml, TFS, Visual Studios, and concepts such as OOP, while incorporating Scaled Agile Framework (SAFe) in the development process.

Costco, Pittsfield Township Ann Arbor, MI
Front End Assistant - Part Time *Sept 2021 – Apr 2023*

- Accrued invaluable experience in leadership, teamwork, and effective communication through routine work, and fostered relationships with not only my colleagues but members of Costco as well.

Projects

Search Engine - (HTML5, Python3, REST API, MapReduce, Pipelining, Multi-Threading) Apr. 2023

- Built a scalable search engine while incorporating information retrieval concepts such as text and link analysis.
- Created segmented inverted index of web pages using MapReduce and Pipelining, while utilizing threads to scavenge through the populated segmented inverted indexes.
- Built an index server and a REST API application that returns search results in JSON in order to create our search server that returns our desired search results.

MapReduce - (Python3, Multi-Threading/Thread Pools, Distributed System) Mar. 2023

- Implemented a MapReduce framework that executes MapReduce programs with distributed processing on a cluster of computers to learn distributed systems, fault tolerance, OS-provided concurrency facilities, and networking.
- To minimize latency, we implemented a manager thread responsible for overseeing a pool of worker threads. This manager thread assigns mapping and reducing tasks to the workers, enabling them to finish tasks concurrently.
- Manager and workers communicate via TCP and UDP messages such that crucial data is handled by TCP messages, while UDP messages are dispatched from the worker to the manager, signaling the worker's activity.

Network File Server (C++, Multi-Threading, Distributed System) Nov. 2022

- To understand file-systems, socket programming, client-server systems, and security protocols, we implemented a multi-threaded, secure network file server, that allows clients to interact with it via network messages.
- Incorporates threads for concurrency between clients. A thread is created per client that connects to the server.
- Utilizes hand-over-hand locking to prevent data hazards between files.

Thread Library (C++, Multi-Threading, Kernel, RAII) Oct. 2022

- Re-implemented a Thread library, in C++, with a 4 class interface, cpu, mutex, cv, and thread, to understand how threads and monitors are implemented on uni-processor and multi-processor systems.
- To ensure atomicity, we manipulated interrupts and implemented a timer-based interrupt system, that generates interrupts periodically and is managed by an RAII abstraction.