Cambridge MA 02139

Sheng Huang

(315) 704-9115
shengh@mit.edu

Experience

AI Tutor Software Developer **MIT Media Lab** **Current 2024**

- Utilized **SQL** and **localstorage** for database, resulting in more efficient data storage.
- Designed overall **API architecture** which led to better results in performance and security of software.
- Implemented parser for RAG context, resulting in an **automated process** of context retrieval for model.

Software Engineer, Intern **Capital One** **Summer 2023**

- Developed **Spring Boot API** that generates custom file config which resulted in a dynamic web page.
- Implemented **additional security layer** of input validation, leading to enhanced back-end security.
- Created and **deployed failed back-end request logging**, leading to a 30% decrease in resolution time.
- Established foundational class templates for back-end, decreasing code development for the team by 70%.

Software Developer **MIT EECS** **Summer 2022**

- Engineered algorithms to filter n-grams from data set, creating 1GB of organized and comprehensible data.
- Developed a **robust data filter system** using Google API, decreasing irrelevant data appearance by 65%.
- Optimized software performance by **implementing caching**, resulting in a 50% reduction in execution time.
- Reduced file storage by 75% through a unique file format and ASCII characters, leading to 20% faster loading.

Education

Cambridge, MA **Massachusetts Institute of Tech.** **May 2025**

- **M.Eng. in Computer Science, May 2025** GPA: **4.7** / 5.0
  - **Graduate Courses:** Computer Networks; Systems Security; Algorithm Engineering; Distributed Systems;
- **B.S.E. in Computer Science, May 2024** GPA: **4.7** / 5.0
  - **Undergraduate Courses:** Computational Architecture; Design and Analysis of Algorithms; Software Engineering; Computer Systems Engineering; Computer and Network Security; *Performance Engineering of Software Systems*;

Projects

Raft by Ongaro and Ousterhout **Go / RPC / Parallelization**

- Implemented **entire Raft Distributed System**, using **RPC** and **Go**.
- Optimized worker and leader algorithms through parallelized RPC calls.
- Thoroughly tested the implementation using Go's robust testing framework.

Map Reduce by Google **Go / RPC / Parallelization**

- Implemented the distributed processing system **MapReduce**, using **RPC**.
- Improved efficiency by fine-tuning worker and leader with parallelized RPC calls.
- Ensured reliability through rigorous testing using Go's comprehensive testing utilities.

Ray tracer Multi-Body Simulator **C / C++ / AWS / OpenCilk**

- Utilized **OpenCilk** for **multi-core processing**.
- Optimized algorithms based on **Span** and **Work** of **paralleled code**.
- Utilized **AWS** for better performance testing.

StarBattle Video Game **Typescript / NPM**

- Implemented feature where the application handles **concurrent** inputs from users.
- Created **asynchronous back-end server communication** to retrieve and store games.

Tools

- **Languages**: Go; C; C++; Java; Python; Typescript;
- **Frameworks and Libraries**: OpenCilk; AWS; Springboot; Fast API; NPM; Git; Linux; Maven; Gradle;