Songhao Li

Portfolio: https://songhao-li.com/
Github: https://github.com/Songhao-LI
Email: sl10500@nyu.edu
Mobile: +1-201-726-0401
LinkedIn: https://www.linkedin.com/in/songhao-li-nyu-mscs/

Education

- **New York University** — New York City, United States
  Master of Science - Computer Science — *Sep. 2023 - May 2025*

Skills Summary

- **Languages**: Python, TypeScript, Java, C++, JavaScript, HTML, CSS, C#, SQL, Bash
- **Frameworks**: Electron, Django, Spring Boot, React, Vue, Django, Flask, NodeJS, WebSocket, RPC, IPC, Scikit, JUnit, Selenium
- **Tools**: Git, Swagger, Tailwind CSS, Three.js, Vite.js, Unity3D, Postman, CMake, JNI, JDBC, Kafka, RabbitMQ
- **Database and Deployment**: Redis, PostgreSQL, MongoDB, MySQL, SQLite, Docker, AWS, GCP, GKE, Apache Bench

Experience

- **University of Pennsylvania - Huang Group** — Philadelphia, United States
  Software Engineer Intern / Research Assistant — *Sep. 2024 - present*
  - Conducted toolkit research. Developed an **open-source** library for image segmentation.
  - Utilized **Electron** to develop TissueLab, a cross-platform application focused on pathologist-AI collaboration, enabling low-code access to Vision-Language Models for physicians.
  - Utilized **IPC** to enhance collaboration between processes, optimize resource utilization, and support complex multitasking operations in a system, particularly leveraging the notification mechanism for third-party authentication.
  - Streamlined maintenance operations by designing and implementing a JavaScript linear plotting library, resulting in a substantial **reduction in costs** (from a team of 3 to a single maintenance personnel).
- **DreamFlow** — San Francisco, United States
  Full-stack Software Engineer Intern — *June 2024 - present*
  - Created real-time AI-generated video live streaming platform. Replaced the HTTP polling and WebSocket-based video streaming solutions with **WebRTC**, significantly improving video live streaming performance and reducing latency.
  - Integrated an **AI model** into Django microservices and utilized **gRPC** for **high-QPS**, low-latency inter-service communication. Implemented optimizations such as load balancing, caching, and asynchronous handling to ensure the system could scale to handle high concurrency. Collaborated with the author of the LCM and LCM-LoRA models, tuning the inference framework to achieve a 10x speed increase compared to mainstream models.
  - Utilized **TypeScript** and **Next.js** to develop the 3D landing page. Through UI optimization by three.js and **framer-motion**, and performance improvements, successfully elevated the Product Hunt ranking from over 2700 to 78.
  - Utilized Key-Value **distributed database** based on Raft algorithm to manage the meta, status, configure of Kubernetes.
  - Implemented Continuous Integration and Continuous Delivery (**CI/CD**) pipelines by GitHub Actions and Nginx on AWS for back-end, Cloudflare for front-end, ensuring seamless updates and maintenance of the application with each new commit and resulting in a 95% reduction in deployment costs.
- **New York University - Career Center** — New York, United States
  Full-stack software engineer — *March 2024 - May 2024*
  - Developed a web application to help users to compare tech products. Utilized **Tailwind CSS** for responsive layouts and dark mode. Embedded **Stripe** form within app for checkout.
  - Implemented java micro-services architecture using **Kafka** for asynchronous messaging, and utilized **Kubernetes** for scalable and resilient containerized deployment. Integrated **Spring Cloud Gateway** to manage API access.
  - Utilized PostgreSQL to keep data and implement transaction management using JDBC. Supporting Multi-Version Concurrency Control (**MVCC**), improving performance and consistency up to 80% in high-concurrency environments.
  - Improving loading speed over 60% by using **Redis** to cache frequently used high-frequency data. Used Bloom filters to avoid cache penetration and employed **Spring Task** scheduled tasks to preheat caches to avoid cache avalanche by 100%.
  - Utilized Redis and Lua to build **distributed locks**, alleviating the issue of coupon overselling in high concurrency environments by 100%.
  - Utilized **GCP OAuth 2.0** for User Authentication and Access Management. Configure **Spring Cloud Gateway** to use the JSON Web Key Set(JWKS) from to verify the **JWT** signature.
  - Generated API documentation based on **Swagger** and enhanced Swagger documentation by integrating Knife4j, reducing time cost in writing API documents by 99%.
- **Chinese Academy of Sciences** — Beijing, China
  Full-Stack software engineer (Full-time) — *July 2021 - April 2023*
  - Developed cross-platform mobile apps using **React Native** as a Full-Stack developer, integrated with Cesium and **Three.js** libraries to leverage **WebGL** technology for delivering a Web 3.0 experience with immersive 3D graphics.
  - Implemented critical performance optimizations by integrating C++ code as a shared library into Java code running in a JVM via the **CMake** and **Java Native Interface** (JNI) to, achieving significant reductions in processing times.
  - Employed **Test-Driven Development** (TDD) methodologies, using Swagger, Jest, Postman to ensure reliable and maintainable code, which reduced the **bug rate** by up to 75% and a streamlined feature release cycle.
  - Contributed to the development of a **fuzzy search** engine. Built a front-end framework that reduced the cost of modifying frontend UI code by over 80%.
- **Chinese Academy of Sciences** — Beijing, China
  Full-Stack software engineer Intern — *Sep 2020 - June 2021*
  - Developed the mobile app using the **Vue** framework. Engaged in UI optimization, employing **regular expressions** for automatic result highlighting and graphics plotting. Improved smoothness by over 50% with CSS animations.
  - Implemented JWT token technology for login and identity authentication, **custom interceptors** for user authentication, and **ThreadLocal** for token validation, determining user login status and resolving HTTP request statelessness.
  - Solved data consistency issues with active updates and delayed deletion strategies to meet data consistency requirements.
  - Enabled real-time updates for client order status and notifications through **WebSocket**. Implemented state transitions of orders based on the **Spring state machine**. And Utilized Spring Task and **Delay Queue** for scheduled order status processing, automatic order cancellation, and other timing functions.