Ziqiao Xi

717-875-7551 zixi@ucsd.edu LinkedIn Github

Education

University of California, San Diego **2024 - 2026**
M.S. in Computer Science (C.S.75)

University of Michigan Ann Arbor **2021 - 2024**
B.S in Computer Science & Mathematics *3.8/4 GPA*

Work Experience

Quantum Compliance **May 2024 – August 2024**
Full Stack Developer Intern *Ann Arbor, MI*

- Developed an end-to-end solution for parsing and extracting data from chemical safety data sheets using **GPU**-accelerated **OCR** (Tesseract) and clustering techniques, reducing processing time from 15 to 3 minutes per PDF.
- Deployed locally hosted LLMs (**Mistral, Llama3** via Ollama) to ensure data privacy, orchestrated queries using LangChain, and integrated selective **OpenAI API** usage for enhanced accuracy.
- Implemented and optimized **RabbitMQ** message queues to distribute PDF parsing tasks across GPU-backed servers, incorporating retry logic and **dead letter queues** to handle failures and analyze problematic messages for reliability.
- Implemented an Excel export feature in a **Spring Boot** and **Angular** application, integrating a seamless UI button and leveraging Apache POI to generate well-structured spreadsheets for tracking workers' certificate statuses.
- Optimized legacy **SQL** queries by resolving the N+1 problem, consolidating multiple inefficient loops into a single query strategy, reducing data retrieval and file generation time from 1 minute to 5 seconds
- Deployed the service on GPU-backed servers using **Docker** containers and designed **Nginx load balancers** to ensure efficient distribution and scalability.
- Designed and implemented a **Retrieval-Augmented Generation (RAG)** chatbot using **LangChain**, incorporating a sliding window technique to deliver real-time, context-aware answers to chemical regulation queries.
- Scraped chemical regulation websites using **Python**, preprocessed data into multilingual vector spaces with **Hugging Face**'s GPT-2 tokenizer, and employed **Faiss** for vector storage, similarity search, and automated index updates.

Project

Meditation App **www.ziqiaoxi.com**
Independent Full Stack Developer

- Developed a scalable meditation-sharing platform where users can record, post, and share meditation experiences with text, images, and location data using **Node.js** and **React**.
- Designed an optimized **MySQL** database with ER diagrams for user posts, comments, and location data, implementing **indexing** and query optimization techniques to improve retrieval speed.
- Optimized MySQL query performance by implementing **Redis** as an in-memory cache with LRU eviction policies, fine-tuned key design, and expiration strategies to efficiently cache user data and session tokens for faster access.
- Built scalable **RESTful APIs** using **Node.js** and **Express.js** to handle CRUD operations on MySQL, with **pagination** implemented for efficient retrieval of large datasets, such as user posts and comments.
- Integrated **RabbitMQ** as a message broker for tasks like asynchronous image uploads and processing, enabling efficient handling of large image files while decoupling services for better maintainability and scalability.
- Designed **Elasticsearch** indexes using an inverted index structure for efficient text and location-based search, enabling fast and accurate retrieval of user posts based on keywords or proximity to specific locations
- Configured **Prometheus** for real-time monitoring of system metrics, setting up **Slack-based** anomaly alerting.
- Deployed the app on **Dockerized EC2** instances with **Kubernetes**; implemented **GitHub Actions** and **Argo CD** for CI/CD, automating unit and integration tests with **Jest/Supertest** and multi-architecture Docker build.

Operating Systems **January 2024 - May 2024**
Course Project *Ann Arbor, MI*

- Built a custom user-level thread library using **C++**, supporting context switching, thread creation, and FIFO-based scheduling with safe concurrency through interrupt management and global atomic guards.
- Developed a demand-paged **virtual memory manager** with swap/file-backed pages, clock-based page replacement, copy-on-write optimizations, and a robust state machine for efficient page lifecycle management.
- Designed a secure, **multi-threaded** network file server with password-based encryption, an upgradeable reader-writer lock scheme, and a non-hierarchical file system structure to ensure crash safety and file system consistency.

Disributed System **January 2024 - May 2024**

Course Project *Ann Arbor, MI*

- Designed and implemented a distributed **MapReduce** framework with a Manager and Workers communicating over **TCP/UDP**, featuring job queuing, task scheduling, and dynamic worker registration.
- Implemented a **concurrency** model using upgradeable reader-writer locks to synchronize threads, prevent data races, and manage job queues, worker states, and task assignments.
- Optimized performance with a **streaming** approach for mapper and reducer input/output,,and incorporating fault tolerance via UDP **heartbeat** monitoring to detect Worker failures and reassign tasks seamlessly.

Web Search Engine **January 2024 - May 2024**

Course Project *Ann Arbor, MI*

- Developed a web search engine using **Flask** with a multi-stage **MapReduce** pipeline, Index and Search servers, and a frontend interface for ranking and displaying query results using **React**
- Built an inverted index using a MapReduce pipeline to compute **tf-idf** scores, segmented indices by document ID, and integrated **PageRank** data to enhance document ranking with user-configurable weight adjustments..
- Developed a **RESTful API** in the Index server to retrieve query results as JSON, optimized query processing by leveraging precomputed index segments stored in memory.
- Incorporated **upgradeable reader-writer locks** to ensure thread-safe operations in the Index and Search servers, enabling concurrent read operations and writes for in-memory updates and query result merging.