SIYANG ZHANG

Vancouver, BC | (236) 989-8162 | siyangzhang1@gmail.com

EDUCATION

Northeastern University — Vancouver, Canada
Master in Computer Science; GPA: 3.9 / 4.0 — *Jan. 2021 - Dec. 2022*

Harbin Institute of Technology — Harbin, China
Bachelor in Financial Management; GPA: 3.7 / 4.0 — *Sept. 2014 - July. 2018*

SKILLS

- **Languages**: Java, Go, Python, TypeScript, SQL, C/C++, HTML, CSS, JavaScript
- **Development**: Cloud & Distributed Systems, Microservices, Database, RESTful Routing, CI-CD Pipeline & Automation, Dev Ops Configuration & Orchestration & Monitor
- **Tools and Technologies**: AWS, Spring, Junit, Mockito, React, Redux, Git, Docker, Kubernetes, Helm, Node, Express, JSON, Kafka, Ansible, Elasticsearch, Logstash, Kibana, Filebeat, Nginx, Redis, Linux, Jenkins, Istio, Prometheus

PROFESSIONAL EXPERIENCE

- **Index Exchange** — *Toronto. Canada*
  Software Engineer Intern, Cloud Platform — *Jan. 2022 - April. 2022*
  - Working at an open source project Vault which is used to store user secrets.
  - Use Ansible playbook to set up Gitlab CI-CD pipeline which enables each individual app's Git request to automatically create its corresponding Vault approle and policy, set up pipeline to achieve automatic secret rotation with ArgoCD.
  - Implement Vault audit monitor pipeline: enable Vault audit log file and server log file to be pass through Kafka into Elasticsearch, Logstash, Kibana(ELK) stack with Filebeat processing.

ACADEMIC EXPERIENCE

- FinanceRisk Analysis: Classify Moody's credit rating of 1700 firms based on 26 financial metrics — *Nov. 2018*
  - Using L1 regularization as feature selection technique to reduce from 33 features to 15 features
  - Conducted majority voting using logistic regression, decision tree and kernel SVM to achieve 80% accuracy with 3%variance, leading to better bias and variance trade off
  - By parameter tuning and ensemble, bagged random forest model managed to achieve 87.1% accuracy classifying(baseline accuracy 70%) Investment Grade (binary classification) and 62.4% accuracy(baseline accuracy 20%-30%) to classify Moody's score (multiclass classification)

PROJECTS

- **Distributed System**: Implement fault-tolerant sharded key/value-service system with log compaction using Go — *Jun. 2022*
  - System shards the keys over replica groups for performance(throughput) and load balance. System provides strong consistency and can handle linearizable concurrent calls and replicas reconfiguration (join and leave)
  - Servers use Raft for replication and can continue to process client requests in spite of failures or network partitions. Implement Raft as a Go object whose interface will support appending index-numbered log entries as replicated logs which will eventually be committed. Enable Raft leader election.
  - Implement log compaction / snapshots to save persistent state (for server reboot & resume): When a server restarts (or catch up), the server will replay log entries based on last snapshot, to save space, server will periodically persistently store a snapshot of its current state, and Raft will discard old log entries that precede the snapshot.
  - Each replica group handles requests for shards subsets in parallel, use Raft to log the sequence of client Puts, Appends, and Gets and log the sequence of reconfigurations. All group members within a replica will agree on when a reconfiguration occurs.
  - A single shard master decides which replica group should serve each shard, clients consult the shard master to find the replica group for a key, and replica groups consult the master to find out which shards to serve.
- Database System: Built a fully functional, optimized database that could perform both simple and nested correlated queries — *May. 2020*
  - Built a parser for SQL statements and wrote an index system with B+ tree
  - Built in logic for joins (partial, inner, outer, equ-joins, etc)
  - Transformed incoming queries into relational algebra and performed Selinger optimization to find the best way to compute the query results given the sizes and clusterings of each dataset involved
- **Ticketing App**: Full stack development of an E-Commerce App using Microservices with Node, React, Docker and Kubernetes — *Sep. 2019*
  - Use React, Hooks and Next JS to provide features include client registry, order booking and payment and JWT-based authorization.
  - Architect a multi-service application, support async, event-based communication between services. Each service is created by Node and Express. Data for each service is held in Redis.
  - App is written with Typescript, deployed and run in Docker containers executed in Kubernetes cluster, enhancing scalability and reusability, deployment by AWS.
- YelpCamp: Full stack of web development (a Yelp.com style website for campgrounds) — *Feb. 2019*
  - Supported full CRUD features such as user log in, posting review and comments, and editing previous submissions
  - Used Bootstrap, the Express framework, NodeJs, and MongoDB database
  - Designed and implemented MongoDB infrastructure to store reviews/comments data and relevant information associated with users and campsites