Meihong Ge

CONTACT

182-2501-2120

mg476@duke.edu

Gemeihong2016 (wechat)

SEEKING

Software Dev

Backend Dev

C++ Dev

ACADEMICS

GPA:

3.58(CQU)/3.8(UC)

4.0(Duke, On-going)

GRE: 161+166

TOFEL: 112

LANGUAGES

Proficient: C/C++

Competent: Python, MATLAB, Mysql

Advanced Beginner: Java, Machine Learning, Deep Learning

Novice: Spark, Docker, Django

AWARDS

University Scholarship

Best Student Leader

Research Excellent Award

Merited Intern

TRAITS

Innovative

Self-learner

Problem oriented

Cooperative

High proficiency in English

EDUCATION

- **Master of Science,** 2020-2022 (Expected)

——Electrical & Computer Engineering Duke University, Durham, NC, U.S.

- **Bachelor of Engineering (Dual bachelor's degree program),** 2015-2020

——Electrical Engineering

Chongqing University, Chongqing, China, 2015-2019

University of Cincinnati, Ohio, U.S, 2019-2020

Core Curriculums:

Data Structure, Algorithms, Networking, Robust Server, Software Engineering, System Programming, Deep Learning, Machine Learning, Embedded Systems, Signals and System, Linear Algebra

PROJECTS

- **Thread Safe Memory Allocation in C**

1. Implemented **memory management** functions **malloc** and **free**, done by **sbrk**() **system call**.
2. Adopted **Doubly Linked List** as **Free List**. Implemented a **merge()** function to reduce **Memory Fragmentation**.
3. Explored two **thread-safe implementations**, more specifically, **mutex lock** and **thread local storage(lts)**.
4. Further explored **glibc's** version of Memory Management which utilize **Arena**, **Bin**, and **Chunk** data structures.

- **HTTP Caching Proxy**

1. Written in **C++** under **Linux** environment, connection done with **TCP socket**.
2. Support **GET**, **POST**, **CONNECT** HTTP method, with a **LRU Cache** to cache response from server with **GET** method. Adhere to **RFC** document for parsing of HTTP request and response.
3. Implemented **Reactor** model, with **Epoll** for **IO-multiplexing** and **thread-pool** to handle tasks.

- **Mini UPS/Amazon**

1. **Simulate UPS/Amazon** website way of interacting.
2. **Frontend** built with **Django; Backend** of websites connected through **TCP socket connection**.
3. **Google Protobuf** is used as **API** for data transmission, facilitating **parsing**.
4. **Frontend** & **backend database scheme** implemented with **Django model API**.
5. Backend data is stored in a **postgresql** database and the whole program is containerized in a **docker container** for easy deployment.

- **Multi-player RISK game written in Java**

1. Done **extensive system design** with **UML diagram** under guidance of **SOLID** design principle before rushing into development; Applied **abstract factory**, **chain of responsibility** and other **design pattern**.
2. Separated **game interface** (running on client server) from **game logic** (running on server).
3. Game has 3-iteration, used **Git features** like **issue**, **branch**, **pull request**, and **CI/CD** for better **team corporation** and **version control**.

WORK EXPERIENCE

- **Test & Development Intern，Two-wheeler Production and Research Department**

——Didi, Beijing, 2021/5-2021/7

1. Participate in the development of the internal remote test platform.
2. Redesigned website's front-end logic from having user actively input test configuration to automatically fetching available configuration from back-end server, smoothed out the test creation process; refactored code on back-end side to exploit modularize, reducing front/back interaction by doing active fetching of data.
3. Tried out internal deployment environment and process. Explored ways of deploying which include steps like compiling/packaging, containerization, and internal supervision system.