Kaiyuan Li

984-259-8342 | kl406@duke.edu | linkedin.com/in/kaiyuan-li

Education

Duke University Durham, NC
Master of Engineering(M.Eng.), Electrical and Computer Engineering; GPA:3.8/4.0 Aug. 2021 – May. 2023(Expected)

ShanghaiTech University Shanghai, China
Bachelor of Engineer(B.E.), Electronic Information Engineering; GPA:3.45/4.0 Sep. 2017 – Jun. 2021

Skills

Languages: C++, C, Python, Matlab, Java, JavaScript, HTML, CSS
Frameworks: Django, Spring, BootStrap
Database & Tools: PostgreSQL, Docker, AWS, React, Postman, Git, Github, Linux

Work Experience

Software Engineer Intern @ StarInvest | Python, React, AWS amplify, Flask

- Designed and developed a full-stack web application with the features of search, create, delete, like and unlike tweets with the integration of Twitter APIs in **Flask**
- Backend: Applied **AWS** Amplify/Lambda function to handle input from frontend and calling Twitter APIs in the backend service
- Frontend: Constructed UI components in the frontend using **React** to display tweets
- Performed data scraping on YouTube to collect more than 600 channel data points including the number of followers, views, social web links etc using Python Selenium and BeautifulSoup
- Applied Python Selenium and smart proxy to scrape **1.6 million** user reviews and social websites from kickstarter

Projects

Ride Sharing Service Web-App | Python, Django, PostgreSQL, Docker

- Implemented a full-stack web application modeling ride sharing service using **Django** and **PostgreSQL**
- Supported functions of user login/logout/registration, request/select/drive for/join/view riders, check driver status as well as personal/vehicle info
- Implemented with **html**, **CSS** and **bootstrap** library, used **Docker** for deployment

Thread-safe Malloc Library | C, multi-threading

- Implemented malloc library in C based on **First Fit** and **Best Fit** strategies, which features merging/splitting adjacent free regions
- Enabled thread-safe with the implementation of lock and lock-free versions using mutex synch strategy and thread-local storage to prevent race conditions
- Handle **1 billion** memory allocation and deallocation within **60 seconds** in both First Fit and Best Fit strategies

Mini Amazon/UPS System | Python, Java, Django, Gradle, PostgreSQL, Protocol Buffer

- Developed a full-stack web application modeling UPS system paired with amazon ordering and delivery system in Django. Simulated the whole process from buying products to getting the package delivered.
- Stored data in **PostgreSQL** and used **Django ORM** to interact with it.
- Used **Google Protocol Buffer** to communicate with world-simulator and mini amazon system partner.
- Designed UI with **CSS**, **HTML** and **Bootstrap**, realized featured web pages with search bar and user login/logout.

HTTP Caching Proxy Server | C++, HTTP, TCP Socket, LRU Cache, OOD

- Built a **HTTP proxy** server in **C++** capable of handling GET, POST and CONNECT API requests.
- Implemented a **LRU cache** to cache server requests, a logging system to record request events and utilized multi-threads to handle concurrent requests
- Implemented server memory management with **RAII**; handled malformed requests and external failures with exceptions to guarantee robustness