Jen-Ting Chang (Tim Chang)

213-207-0194 | jentingc@gmail.com | linkedin.com/in/jenting-chang/ | github.com/timjtchang | United States

Work Experience

Software Backend Engineer at Crypto Arsenal - Taipei City, Taiwan(Remote) June 2024 – Present

- **Led system design** and agile development for a crypto payment gateway, collaborating **cross-functional** teams to deliver **scalable REST APIs**, object models, and **core functionalities** within **3 months**
- **Developed backend logic in Go** for HD wallet address derivation, payment creation, on-chain balance retrieval via **RPC**, payment record updates, and **MySQL** database interaction
- **Built** and **managed** payment gateway **infrastructure** on **AWS**, using **Secrets Manager** for seed security, **API Gateway and Lambda** for serverless computing, and **RDS** for payment data storage
- **Engineered** a **CI/CD pipeline** with **GitHub Actions**, using **OIDC** and **JWT** for AWS role assumption, automated **unit tests** and **Docker containerization**, and deployed to **ECR**, reducing deployment time by **50%**
- **Developed 20+ unit tests** to validate backend logic for wallet address derivation, payment creation, and on-chain balance retrieval by applying production-tested cloud-native development fundamentals
- **Reduced CloudWatch** log data volume by **64**% across **50**+ repos by implementing environment-sensitive logging frameworks and parsers to consolidate **10+ Kubernetes** log entries into **single JSON** objects

Software Engineer at Realplus Tech - Taoyuan City, Taiwan(Remote) September 2019 – August 2023

- **Developed** a Blockly-based coding **framework** in **JavaScript**, integrating Google Blockly to enable citizen developers to create custom quadcopter flight programs, driving a **500+** user adoption rate within **2** months
- **Engineered** a server-side **runtime pipeline** in **Node.js** for Blockly code, dynamically rewriting code to integrate quadcopter libraries and executing it in isolated **child processes**
- **Led** the design of 4WS Vehicle **backend system** including hardware interface via **ROS** and UI via **REST API**
- **Developed** API and service development in **Python** on **Linux** for real-time hardware status updates, UI and controller command processing, and 4WS vehicle control

Education

University of Southern California January 2023 – December 2024

Master of Science in **Computer Engineering** Los Angeles, CA, USA

- Coursework: Computing Principle (**DS&A**), **Parallel & Distributed Computation**, **Internet Cloud Computing**, **Database Systems**, **Analysis of Algorithms**, **Computer Networks**

Tamkang University September 2015 – July 2019

Bachelor of Engineering in **Electrical and Computer Engineering** New Taipei City, Taiwan

- Coursework: **Data Structures**, **Algorithms**, **Operating System**, **Web-Based Programming**

Projects

Trojan Map - Algorithm Implementation March 2023 - May 2023

- **Developed** a delivery **map** system using **C++** with **20K**+ data entries of places
- **Implemented** location **search** using **Trie** for auto-completion and **Edit Distance** for misspelling correction
- **Built** delivery **routes** under gas **budget** using **topological sort** for visit order and **Dijkstra** for shortest paths
- **Improved** shortest path calculation **speed** by **94**% by replacing the **Bellman-Ford** with **Dijkstra**

Game Management System - Fullstack Project January 2023 - March 2023

- **Built Node.js** backend, **React.js** frontend, and **MongoDB** database, deployed on **AWS EC2**
- **Developed 20+ REST APIs** to manage game matches and player data, including create, read, update, delete
- **Designed 100**+ Mocha **unit tests** for game matches and player information across diverse operational scenarios
- **Engineered** a **GraphQL data pipeline** from backend to **MongoDB** to optimize data integrity and retrieval

Blake2b Parallel Encryption Accelerator - Multi-threaded Implementation November 2023 - January 2024

- **Accelerated** Blake2b **hashing** by reordering independent calculations for parallel processing across **4 threads**, and synchronizing shared-memory access with **mutex locks**
- **Reduced** processing time by **20**% for **32KB** files and **30**% for **65MB** and **500MB** files through parallelization

Core Tech Stacks

Programming Languages: **Python**, **Java**, **JavaScript**, **C++**, **TypeScript**, **Go**
Web and Database: **SQL**, **NoSQL**, **React**, **InfluxDB**, **PostgreSQL**
DevOps and ML: **Docker**, **Kubernetes**, **AWS**, **Git**, **CI/CD pipelines**, **TensorFlow**, **Spark**