Lixing Yang

M.A., Graduate Institute of Linguistics, National Taiwan University 2018 - 2022

linkedin.com/in/li-xing-yang-1a2037230
github.com/Retr0327
lixingyang.dev@gmail.com
retr0327.github.io

Experiences

Fullstack Engineer, TRAPA Security, Taipei/Taiwan Apr 2023 - Ongoing

Django, TypeScript, VueJS, Celery, Redis, RabbitMQ, MySQL, Docker, Nginx, gRPC

- Backend:

1. Designed and built a **cybersecurity training platform** allowing **30000** users online with **2.17** response time assessed by JMeter.
2. Implement **DDD** (Domain-Driven Design) and **various design patterns** (e.g., factory, repository, IOC, etc.) for architecture design, ensuring system scalability and maintainability.
3. Integrated **dependency injection** into Django, and developed NestJS-inspired IOC containers, enhancing module/controller providers communication through interfaces.
4. Developed a fully **type-safe CQRS module** for synchronous and asynchronous message exchanging interface at the application level.
5. Optimized the background cron job to enhance the health check mechanism for **virtual machines** and **docker containers.**
6. Integrated a dockerized validation system through **gRPC** and **RabbitMQ**, allowing asynchronous batch communications across VMs.

- Frontend:

1. Guided a frontend developer to improve efficiency and code quality by introducing best practices in component design, state management, and code reusability.
2. Enhanced the platform's UI/UX, conducting user research and feedback analysis to identify usability issues.
3. Identified and resolved props drilling by using provide and inject.
4. Addressed security anti-patterns and introduced a cookie-based token granting and token refreshing mechanism.
5. Streamlined data fetching process in local development through NodeJS **proxy server** and resolved **CORS** issues.

- DevOps

1. Architected **CI/CD** using **GitHub Actions** and implemented custom Bash scripts for **database migrations** and **MySQL replica synchronization**.
2. Utilized **Docker Compose** and **Nginx** for effective routing, streamlining container management, enhancing development process and boosting web application availability.
3. Implemented **Redis Sentinel** for automatic failover and **MySQL replication** for **CQRS** and data durability, ensuring high system resilience and scalability.
4. Optimized frontend performance using **PM2 clusters** in the production environment.

Side Projects

Eyrie

NestJS, Python (FastAPI, Pydantic, Dependency Injector)

- Developed a **NestJS**-like backend framework built on top of **FastAPI,** including features such as **decorators**, **dependency injection**, **provider registration** and **module configuration.**
- Designed an **IOC container** and a **graph inspector** for managing application providers, boosting module reusability.

- Implemented a **dynamic routing system** incorporating middleware handling to ensure efficient request processing.
- Utilized Pydantic Models and TypeAdapter to enhance code maintainability, readability, and runtime type safety across the entire application.

Taiwan Social Media Corpus

NextJS (ReactJS), NestJS, Typescript, Redis, RabbitMQ, PostgreSQL, Docker, Nginx

- Backend:

1. Launched and designed web application for corpus management and **API token** generation.
2. Streamlined process for collecting and storing social media data using docker-compose and a bash script.

- Frontend:

1. Adopted **Mantine** component library and PostCSS to enhance UI development, and leveraged SWR for optimized data fetching.
2. Migrated the application from NextJS version 11 to 14 incorporating the latest framework improvements.

Py-knot

Python (Dependency Injector, Celery)

- Developed a **message bus system** utilizing Python's generic type annotations to enforce **type correctness** across command and query handlers
- Introduced a distributed task execution model using **Celery** to process asynchronous commands and queries.

Async Ptt Crawler

Python, Scrapy, CKIP, Docker

- Refactored legacy Python Scrapy project "**scraptt**" to support async callables
- Implemented asynchronous programming techniques to increase **crawling** and **tokenizing** speed by **800% - 1000%**