Esther Xiao

424-428-4523 | yiquxiao@gmail.com | github.com/felicixawe felicixawe.github.io

Education

University of California, Los Angeles — Los Angeles, CA
Bachelor of Science in Mathematics of Computation — *Sept. 2021 – Dec. 2023*

Arizona State University — Lake Havasu, AZ
Master of Science in Computer Science — *Mar. 2024 – Aug. 2025*

Related Coursework: Operating System, Computer Networking, Algorithms, Object Oriented Design, Software Construction

Technical Skills

Languages: JavaScript, Python, Java, Yaml, Json, SQL, HTML, CSS
Tools and Frameworks: MERN(MongoDB, Express, React, Node), Django, SpringBoot, MySQL, DynamoDB, Maven, Npm, Docker, AWS, Postman, JUnit, JEST, PyTest, Jenkins, Github Actions

Internship Experience

Software Engineer Intern — Jun. 2023 – Aug. 2023
Qianzhiyi Tech Inc. — *Guizhou, China*

- Devised frontend advanced websites(layouts, navigation, animation button and icons) with **Bootstrap**, font-awesome using **HTML5**, **CSS3** and **JavaScript**.
- Connected the frontend to **RESTful APIs** using **ReactHooks** and validated behaviors with **Jasmine** framework tests.
- Contributed to backend development utilizing **Node.js**, and **MongoDB**, implemented RESTful and GraphQL endpoints, achieving over 200qps during peak times.
- Conducted manual API testing through **Postman** and automated basic integration tests using **JEST** testing framework achieving unit test coverage of 99%.
- Built the application using **Yarn** and created **CI/CD** testing and deployment workflows in **Jenkins** to improve efficiency greatly for internal developers.

Projects

MERN Based Shopping Platform — May 2023 – Jun. 2023

- Devised a shopping platform based on **MERN**(MongoDB, Express, React, Node.js) and **AWS**.
- Implemented **MongoDB** based CRUD operation using **Node.js** and connected the frontend with backend APIs through **React** (UseEffect and UseEvent).
- Enabled session based user authentication and authorization using **JWT** and **Cookie**.
- Created docker-compose files to build the **micro-services** and deployed the services through **Docker**.
- Conducted manual API testing using **Postman** and wrote unit tests for isolated services and integration and end-to-end tests based on **JEST** testing framework.
- Deployed the service on **AWS** using **Docker** and automated the build and deployment pipeline through **Github Actions**.
- Setup monitoring system for the application based on Google open source **cloud-prober** and created a dashboard to visualize the status in **Grafana**.

Recipe Management System — Jun. 2023 – July 2023

- Designed and developed a recipe management system harnessing the power of **Python Django**, **PostgreSQL**, and **AWS**.
- Crafted docker-compose configurations to scaffold a local development server, deploying all services using **Docker**.
- Utilized **django-ORM** to persist data in **PostgreSQL**, establishing foundational CRUD operations.
- Orchestrated **RESTful** APIs to underpin recipe management functionalities, leveraging **django-rest**.
- Incorporated **Django Token** for user authentication and authorization, ensuring role-based access and permissions.
- Introduced a caching mechanism, optimizing system performance via Django Cache **Middleware**.
- Deployed the application to an **AWS EC2** instance utilizing AWS CLI, ensured API integrity with tests in **Postman**, and streamlined build and deployment workflows using **Github Actions**.
- Integrated automatic API documentation with **Swagger**, enhancing the developer experience and API clarity.

Student Registration Application — Jan. 2023 - Mar. 2023

- Designed and developed a Student Registration application using Java **SpringBoot**, **React** and **DynamoDB**.
- Used SpringBoot **H2** in-memory database to cache the frequently used student info based on **LRU**.
- Provided both session-based authentication and authorization via **JWT** and **Spring Security**.
- Built the frontend based on Ant design library and **JSX** and connected the frontend with APIs through **React Hooks**.
- Built the application using **Maven** and deployed on an **AWS EC2** instance, created a **DynamoDB** table through AWS console and wired up using an **IAM** account.