Jay Mistry

jay@jellyware.co.uk | www.jellyware.co.uk | www.github.com/jaym-01

EDUCATION

Imperial College London - Electronic and Information Engineering:

- Obtained the **Dean's List** in my 1st and 2nd year for academic performance (top 10% of the year).
- Built a pipelined RISC-V 32-I (CPU) using System Verilog & C++.
- Built a C90 to RISC-V compiler with C++, Flex and Bison.
- Created a marble game controlled by an FPGA. Built the Python FastAPI backend, with an SQLite database, deployed it to an AWS EC2 instance and contributed to the FPGA codebase written in C.

A-Levels, 2020-2022: Maths **(A*) |** Physics **(A*) |** Computer Science **(A*)** (Coursework: designed and built a normalised MySQL database, and a desktop app to perform CRUD operations (C# and WPF)) **|** EPQ **(A)**
AS-Level, 2021-2022: Further Maths (self-taught) **(A)**

PROFESSIONAL EXPERIENCE

Full stack engineer at OncoFlow (Health tech AI startup) (June 2024 – September 2024):

- Developed the frontend and backend (implementing REST APIs) from scratch for the main app and company site independently, using Next.js (with **Typescript**), Tailwind CSS, **Python** FastAPI and Next.js server actions.
- Developed a demo environment to simulate the data environment used in the NHS, to test the app before integration.
- Added optimisations such as caching to reduce the use of AI models, reducing cost and latency.
- Implemented unit, integration and end-to-end tests using pytest, Jest and Playwright.
- Created local testing environments to simulate production, using Docker and Bash scripts.
- Worked in a fast-paced environment, adding features and fixing bugs within a few hours or days.
- Collaborated with GitHub: pushed code to production with pull requests and participated in code reviews.
- Maintained high code quality and consistency with ESLint, Prettier, flake8 and mypy.

Mentor at Zero Gravity: Helped a student from a disadvantaged background get into a top University, by informing and aiding decision-making on the application process, and providing feedback and support on their application. Both listening to them and coherently presenting ideas improved my communication skills.

SKILLS

Programming Languages/Frameworks: (Proficient): TypeScript, Tailwind CSS, Python; (Intermediate): React.js, Next.js, Jest, Playwright, C++, SQL.
Familiarity with **Unix development**, **Bash Scripting, Git**, **GitHub**, and designing UIs with **Figma**.

PROJECTS

CTO of Elise – an app that containerises and deploys backends to AWS (Typescript and Python):

- Built the frontend for the app using Next.js, Tailwind CSS and shadcn/ui.
- Fixed bugs concerning pulling GitHub repositories from the Python FastAPI backend and enabled custom run commands for the docker file.
- Learnt about creating docker files, storing container images in a container registry and deploying them to AWS.
- Connected with founders and engineers by sending LinkedIn messages and emails to receive feedback and sell.
- Developed a Chrome extension to track leads with my co-founder, using JavaScript and Supabase.

AI mobile app automating processes in a pharmacy (Typescript and Python) (https://github.com/jaym-01/Rx2Label):

- Built the mobile app with React Native and Expo, and the backend with Python FastAPI.
- Used the Mistral API to connect to Pixtral 12B (LLM) and built a data pipeline, to perform OCR of a prescription, perform checks against medical literature and generate data required for the prescription label, if valid.
- Used Supabase storage buckets to store the prescription image temporarily for use with the Mistral API.

Implemented a Redis Server in C++ (https://github.com/jaym-01/jellis):

- Used boost ASIO library to create TCP sockets that handle multiple clients using an event loop.
- Implemented a parser and encoder for the Redis serialisation protocol (RESP).

Dashboard and embedded programming for Mini Power Grid (Typescript & Python):

- Built a full-stack app with Next.js and Tailwind CSS that displays real-time data using MQTT.
- Deployed an MQTT broker to an AWS EC2 instance and shared the dashboard using ngrok.
- Programmed Raspberry Pi Picos to send and receive data from the dashboard using MQTT, with micro python.
- Designed and implemented an algorithm to synchronise with an HTTP server, with data changing periodically, that minimised latency and resource consumption.