Jay Mistry

jay@jellyware.co.uk | www.jellyware.co.uk
www.github.com/jaym-01 | www.linkedin.com/in/jay-mistry1

EDUCATION

Imperial College London - Electronic and Information Engineering **Graduated June 2026 with First Class Honours**

- Obtained **Dean's List** in my 1st, 2nd & 3rd year for academic performance (**top 10%** of the year).
- Built a pipelined RISC-V 32-I (CPU): System Verilog & C++.
- Built a C90 to RISC-V compiler with C++, Flex and Bison.
- Built a hardware accelerator for a softcore CPU on an FPGA, providing a 200x reduction in computation time.

Notable Modules

- Advanced Computer Architecture – 89%
- Quantum Computing – 88%
- Machine Learning – 86%
- Digital System Design (FPGA) - 86%
- Compilers – 81%

PROFESSIONAL EXPERIENCE

Meta - Software Engineering Intern (Facebook Messenger Web, **React & JavaScript**) **June 2025 – September 2025**

- Owned end-to-end the delivery of persistent draft messages and filters features for Facebook Messenger web, shipped to 1B+ users; A/B experiments showed a **2% increase in message sends**.
- Architected APIs over a client-side persistent store, **cutting read/write latency by 20%.**
- Diagnosed and fixed production race conditions in chat creation, flickers and bundle size issues, improving performance, reliability and maintainability.
- Led cross-functional collaboration between architecture, QA and privacy teams to navigate the complex compliance requirements and begin feature roll-out to users.
- Used my initiative to create unit and end-to-end tests for critical Messenger features, such as chat previews, which became useful for several engineers during my time as they introduced breaking changes.

OncoFlow (AI Health tech startup) – Full-stack engineer **June 2024 – September 2024**

- Joined as **first full-stack engineer**; independently **owned the company's core product end-to-end**: web app + REST API backend + AI models (with **React, Typescript, Tailwind** & **Python** FastAPI).
- Built a dashboard platform enabling NHS workers to log in, interface with and run AI models, connect to medical databases with cancer treatment information, and generate and edit AI-produced medical reports.
- Developed a demo environment to simulate the data environment used in the NHS for testing before integration.
- **Cut costs and latency** of using AI models with client- and server-side caching.
- 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.

PROJECTS

Integrating and **improving image generation models (Flow Matching)** to model the probability of death for patients treated for sepsis **(Python, PyTorch, Distributional Reinforcement Learning)**

- Computed the uncertainty of the distribution of returns learnt by the model - used to improve downstream results.
- Developed and proved a shortcut that **cut inference time** of the image-generation model by **20x.**
- Achieved **state-of-the-art** sepsis mortality detection: **90% of at-risk patients** identified vs. **80% prior best**.

(IC HACK Winner) VisuMath (Worked on Backend - **Python**) (https://devpost.com/software/intellilearn-kjxv19):

- A gallery for educational math videos, search for videos with a prompt and generate new videos with an AI Agent.
- Designed the overall system architecture and was responsible for building a **fault-tolerant, horizontally scalable backend** that decouples request handling from long-running agentic jobs.
- The backend was made of 3 layers: an API server (using Python Fast API), a Redis queue and a Celery Worker (a Python process handling long-running tasks, queued in the Redis queue).
- Used AWS S3 (video storage), DynamoDB (metadata) and Pinecone (vector search) to enable semantic retrieval.

Snipr (Tauri desktop app, frontend – **React** & **Typescript**, backend – **Rust**) (https://github.com/jaym-01/snipr):

- Built a production-ready desktop app that removes silences and provides transcripts for audio files, with users.
- Implemented a CI-CD pipeline using GitHub Actions to build the app and provide automatic updates for users.
- Implemented decoding and encoding audio files to PCM samples with FFMPEG, an algorithm to remove silences, and provided a responsive UI by spawning new threads to process the audio file, all in Rust.
- Developed a Rust-Python integration using PyO3 to call OpenAI's Whisper model to provide audio transcriptions.

SKILLS

Programming Languages/Frameworks: (Proficient): **Python**, **TypeScript, JavaScript, C++, PyTorch, NumPy, Node.js, React.js, REST APIs**; (Intermediate): **C, SQL, Jest, Rust, Verilog, AWS.**

Proficient with **Git, GitHub & UI design with Figma.** Familiar with **Unix terminal, Bash Scripting, Docker**.

Proficient with and without coding tools (such as Claude Code)