George Ma

☎ (661) 513-3350 | ✉ georgema2020@gmail.com | Github | LinkedIn | 🏠 Portfolio

EDUCATION

University of California, San Diego — **La Jolla, CA**

***B.S. in Computer Science**, Minor in Business Analytics; GPA: 3.9* — *Expected June 2027*

Relevant Courses: Data Structures & Algorithms, Object-Oriented Programming, Discrete Math, Data Science Theory

TECHNICAL SKILLS

Programming Languages: Python, Javascript/Typescript, SQL, Java, C++, C#, Bash, Groovy, HTML/CSS, YAML
Libraries/Frameworks: Flask, FastAPI, Express.js, React, Next.js, React Native, JUnit, PyTorch, scikit-learn, Pandas
Technologies & Tools: Node.js, REST API, MySQL, PostgreSQL, Git, Docker, AWS, Firebase, Gradle, CUDA

WORK EXPERIENCE

Praxie AI — **San Francisco, CA**

Software Engineer Intern — *April 2025 - Present*

- Developed 10+ reusable **React Native** pages and UI components in **TypeScript** serving 300+ youth golfers
- Designed 6 **sub-2s nested query algorithms** for paginated tournament search filtering from Firestore collections
- Optimized directory modularization of components and hooks, reducing onboarding time for features by 42.9%
- Architected a performant data model by creating and deploying **8 data migration scripts** using the **Firestore Admin SDK** to denormalize data structures with pre-computed fields, boosting data-fetching speeds by 20%

UCSD Alpha Kappa Psi ↗ — **La Jolla, CA**

Webmaster/Lead Developer — *December 2024 - Present*

- Spearheaded the migration of the chapter website from Wix to a Jamstack solution (**Next.js**, **Tailwind CSS**, **Supabase**), resulting in a scalable platform with **60% faster** page load times for over **600 monthly active users**
- Directed an **Agile** workflow for code reviews and issue tracking with Git to guide a team of 3 developers
- Designed an optimized **PostgreSQL** database schema, reducing data-fetching times for dynamic content to sub-1s
- Authored comprehensive documentation that enables future webmasters to easily manage and update the site

Data Science Student Society (DS3) @ UCSD — **La Jolla, CA**

Data Science Consultant (Client: Solana Center) — *March 2025 - June 2025*

- Co-developed a reusable Python data processing pipeline using **Pandas** and **NumPy** to clean, standardize, and impute values in datasets on the client's waste collection program, reducing manual data prep time by 76%
- Engineered a forecasting module in Python to predict participation trends, implementing **SARIMA**, **Prophet**, and **XGBoost** models to achieve 88.2% accuracy via automated cross-validation and residual diagnostic pipelines
- Built a **Streamlit** dashboard with **Matplotlib** to provide the client actionable insights on trends

PROJECTS

Watchdog | ***Demo*** ↗ — August 2025 - September 2025

- Architected an **AI-powered CI/CD** platform automating PR reviews, lint/format, and security for 6+ languages
- Engineered two serverless, **dockerized MCP servers** (FastAPI and Express.js) on **AWS ECS** and **Fargate** with a parallelized, language-aware chunking algorithm enabling accurate, in-depth LLM reviews of complex code diffs
- Orchestrated a multi-runtime CI environment with intelligent language detection to **conditionally execute 13+ polyglot toolchains** for dynamic, full-stack linting and formatting, reducing pipeline duration to **under 2 mins**
- Enhanced security and reliability by implementing graceful degradation and least-privilege token scoping

Spotify Mood Player | ***Demo*** ↗ | **Website** ↗ — April 2025 - August 2025

- Created a full-stack mood-based music categorization and playback app with a **React**/TypeScript frontend, **Flask**/Python REST API backend deployed via a **CI/CD pipeline** on **AWS Lambda**, and PostgreSQL Supabase DB
- Implemented Spotify **OAuth 2.0** flow with **session cookies** via a first-party proxy for cross-browser compatibility
- Achieved 92.6% accuracy in track classification by designing a dockerized, end-to-end **MCP pipeline** leveraging **OpenAI**, fine-tuned with lyrics from Genius API and audio features extracted from iTunes API using Librosa
- Optimized analysis runtime by parallelizing computations with a **ThreadPoolExecutor** per Gunicorn worker

Pokemon Generator | ***Demo*** ↗ | **Website** ↗ — April 2025 - July 2025

- Created a full-stack dockerized Flask web app with a Tailwind-styled UI that generates and displays Pokemon (image, stats, ability) based on user input via a PostgreSQL Supabase database using **SQLAlchemy ORM**
- Developed a **PyTorch Conditional GAN** featuring 6 convolutional layers (trained using **CUDA**) to create unique 256x256 pixel Pokemon images from user-defined condition vectors and random noise vectors via **REST API**
- Used **scikit-learn RandomForestRegressors** and a dictionary to predict stats and select ability based on type(s)