Arjun Aggarwal

arjunaggarwal173@gmail.com | linkedin.com/in/arjunaggarwal1/ | github.com/arjunaggarwal03 | arjunaggarwal.dev

Education

University of Maryland Expected Graduation: May 2025
Bachelors of Science in Computer Science (Honors); Minors in Applied Math & Data Science *College Park, MD*

- **Achievement**: F24 Batch YCombinator Interviewee (top 6% of applicants out of roughly 10,000)
- **Relevant Courses**: Algorithms, Advanced Algorithms, Data Structures, Discrete Math, Intro to Machine Learning

Experience

Amazon Web Services May 2024 – August 2024
Software Development Engineer Intern *Seattle, WA*

- Achieved **100%** ledger reporting accuracy (**10M+** monthly financial events) by designing end-to-end auditing pipeline.
- Implemented **real-time** event tracking across services using **SNS-SQS messaging**, providing detailed monitoring.
- Provisioned a **Lambda** + **DynamoDB** CRUD application via **CloudFormation**, giving **eight teams** live event state visibility.
- Automated integrity logging (**EventBridge** to **S3**) and **CloudWatch** alarms, surfacing anomalies and preventing support tickets.
- Delivered a **React** dashboard with on-demand DynamoDB look-ups via **Lambda**, slashing root-cause **triage time to seconds**.

Bank of America June 2023 – August 2023
Software Engineering Intern *Jersey City, NJ*

- Automated risk data testing with **Python** and **SQL**, replacing an older Alteryx workflow and reducing run time by roughly **85%**.
- Integrated Bitbucket API with workflow tools, reducing manual **30+ minute** data check-in time to **seconds** for **750+ analysts**.
- Designed a test info microservice using **Java** and **Spring Boot**, containerized with **Docker**, replacing inefficient legacy scripts.

Capital One January 2023 – April 2023
Machine Learning Engineering Intern *College Park, MD*

- Applied **Spark**'s optimized distributed querying to the Card transaction graph (**900M edges**), enabling faster node info retrieval.
- Utilized Spark GraphFrames and **motif queries (DSL)** for filtered node searches, leading to median **6x faster** graph querying.
- Conducted **80 cloud-based trials** with varying RAM/storage metrics to validate results; presented metrics to stakeholders.

Projects

Music Similarity Search | *Backend: Python, Django, CLMR, Pinecone, Docker* | *Frontend: TypeScript, Svelte*

- Created a music similarity search engine using **CLMR** (Contrastive Learning of Musical Representations) for audio embeddings.
- Implemented audio processing pipeline with **Django REST API** to handle file uploads, normalization, and feature extraction.
- Leveraged **Pinecone vector DB** to store and query high-dimensional audio embeddings, enabling efficient similarity search.
- Containerized application using **Docker** for consistent deployment, with separate services for API and frontend components.

CryptoArb Engine | *Python, FastAPI, Kafka, Spark, Redis, Cassandra*

- Engineered a real-time cryptocurrency arbitrage detection system using **Apache Spark** Structured Streaming and **Kafka**, processing live price data from exchanges (Coinbase, Binance, and Kraken) to identify profitable trading opportunities.
- Designed a distributed system architecture using **Cassandra** for storage and **Redis** for caching, enabling sub-millisecond access.
- Exposed the database via **FastAPI** for trading metrics and historical data, with monitoring endpoints for system reliability.

Hermes | *Backend: Python, FastAPI, MongoDB, BERT, Pinecone, AWS EC2* | *Frontend: TypeScript, ReactJS*

- Designed a CLI tool allowing developers to message code snippets and communicate via the terminal, expediting development.
- Implemented user messaging via **FastAPI WebSockets** with **MongoDB** to store chat data; deployed API to **AWS EC2** instance.
- Stored **BERT embeddings** of messages in a **Pinecone vector database**, providing users with semantic search for chat logs.
- Utilized **$1K** award in credits from AWS Activate to host API and website built using **React** and **TypeScript**.

YOLOv3-based Vehicle Parking Pass Detector | *Backend: Python, Flask, YOLOv3, Google OCR* | *Frontend: HTML/CSS, JavaScript*

- Achieved **96% accuracy** in detecting vehicle parking passes with custom-labeled training data via **YOLO** real-time detection.
- Integrated **Google Cloud AI** optical character recognition to detect pass identification numbers, automating record-keeping.

Skills

Languages: Python, Java, JavaScript/TypeScript, C/C++, Ruby, OCaml, SQL
Frameworks: Django, Flask, FastAPI, Apache Spark, Kafka, ReactJS
Tools & Technologies: Git, Linux/Unix, AWS Tools (S3, EC2, DynamoDB, Lambda, SNS/SQS), MongoDB, Pinecone