Duc Le

(616) 290-5762 | Fort Worth, TX | duc.anh.le@tcu.edu | linkedin.com/in/duc-le | github.com/DucLe-2005

Education

Texas Christian University, College of Science & Engineering Expected May 2028

B.S. in Computer Science and B.S. in Mathematics *Cumulative GPA: 3.84/4.0*

- **Honors & Awards:** Dean's Honor List (All Semesters), Google IT Support Professional Certificate (2024).
- **Relevant Coursework:** Data Structures and Algorithms, Operating Systems, Object-Oriented Programming, Computer Architecture, Programming Languages, Unix/Linux System Administration, Linear Algebra, Statistics, Probability.

Experience

Finbud Mar. 2025 – Aug. 2025

Software Engineer Intern *St. Paul, MN*

- Architected and deployed an **end-to-end RAG (Retrieval-Augmented Generation)** pipeline using **Python** and **Selenium**, processing and vectorizing 10,000+ financial documents and market and newsfeed data, achieving **95%** accuracy.
- Managed and optimized a **Qdrant vector database** to store document embeddings, implementing indexing strategies that **accelerated similarity search speeds by 40%** for the RAG pipeline.
- Optimized Azure VM and container lifecycles with automation tools such as **PowerShell and Azure Automation**, creating a daily start/stop schedule that reduced cloud costs by **50%**.
- Containerized the RAG pipeline with **Docker** and deployed on **Azure VMs**, cutting deployment time by **80%**.
- Built a user-friendly **Vue.js chat UI** integrated with the RAG pipeline, adding real-time feedback that **increased engagement by 30%**.
- Presented live technical demos to **7 internal stakeholders** (CEO, mentor, and engineers), walking through the RAG pipeline retrieval flow and chat integration, directly shaping the product narrative for **Y Combinator demo day**.

British Airways Reviews Dashboard Jan. 2025 – Present

Software Engineer *Fort Worth, TX*

- Collaborated on an **agile** team of 4 to deliver a real-time analytics dashboard processing **3,000+** passenger reviews via a **TypeScript/Next.js** frontend with 15+ charts rendering in under 1 second.
- Streamlined the data pipeline within an **AWS cloud computing** environment using **Snowflake**, **EventBridge**, and **Lambda** to pre-process review data (percentages, counts, and aggregates) and cache the results as JSON objects in **S3**, reducing latency by **99%**.
- Engineered a RESTful API using **Node.js and Express.js** to serve aggregated review data from S3 to the frontend, implementing pagination and filtering to handle large data loads efficiently.
- Configured **AWS CloudWatch** alarms and dashboards to monitor ECS services and Lambda functions, enabling proactive troubleshooting and maintaining **99.9% uptime** across critical cloud infrastructure.
- Implemented a **CI/CD pipeline** with **GitHub Actions**, integrating version control and automated **Jest** tests to reach **90%** coverage.

Projects

EPL Player Stats | *Java, Spring Boot, PostgreSQL, Docker, SQL, React.js, CSS* GitHub

- Implemented an end-to-end **Spring Boot** backend with RESTful web services to query English Premier League player statistics via **Spring Data JPA** and **PostgreSQL**, adding query filtering and pagination for efficient stat retrieval by position, nationality, or club.
- **Created** a responsive user interface with **React.js** and **CSS**, including interactive player profile cards and sortable stat tables to explore stats for **500+ EPL players**, improving usability for end users.
- Automated infrastructure deployment with **Terraform**, using **AWS EC2** and **RDS (PostgreSQL)** to cut setup time by 80%.
- Secured APIs with **Spring Security** and **JWTs**, adding custom login flows and email verification to block unauthorized access.
- Established a robust testing suite using **JUnit, Mockito, and Testcontainers**, achieving **85% test coverage**, and integrated into a **GitHub Actions** CI pipeline to automate all testing and ensure code reliability before deployment.

Go Database Internals | *Go* GitHub

- Built a lightweight database engine in **Go** with B-tree indexing, implementing node splitting, balanced storage, and efficient key lookups.
- Strengthened reliability by applying database best practices such as **copy-on-write** and **double-write** crash recovery techniques.
- Optimized page layout and memory usage with freelists and offsets while benchmarking **LSM-trees** for write efficiency and query speed.

Forever | *React.js, Node.js, Express, HTML, CSS, MongoDB* GitHub

- Built a scalable full-stack e-commerce platform using the **MERN stack**, engineering **15+ reusable React components** and a persistent shopping cart with the **React Context** for global state management.
- Structured a non-relational **MongoDB** schema to store products, users, and orders, optimizing it with proper indexing to ensure fast query responses and scalability for **10,000+ products**.
- Crafted a responsive component system with **Tailwind CSS** (grid/flex utilities) and semantic **HTML**, minimizing layout shifts.

Technical Skills

Languages & Frameworks: Python, Java, JavaScript/TypeScript, Go, SQL, React.js, Vue.js, FastAPI, Spring Boot, Node.js, HTML/CSS
Databases: PostgreSQL, MongoDB, Qdrant, AWS RDS, NoSQL
Libraries & Tools: Git, GitHub, Vercel, Docker, AWS, Azure, Terraform, JUnit, Jest, Mockito, Testcontainers, Selenium