ARUNIT BAIDYA

Boston, MA, USA (Ready to Relocate)

Github LinkedIn +16179355115 baidyaarunit@gmail.com

Education

Northeasten University, Khoury College of Computer Sciences **Aug 2024 – April 2026**

Master of Science, Computer Science Boston, MA

- **Courses:** Program Design Paradigms, Computer Systems

Anna University, PSG College Of Technology **Aug 2018 – July 2022**

Bachelor of Engineering, Computer Science and Engineering Coimbatore, India

- **Courses:** Data Structures, Operating Systems, Discrete Mathematics, Database Management Systems

Skills

- **Technical Skills**: GDB, Valgrind, C, C++, Python, Java, Go, API Development, PostgresDB, RestAPIs, SQL, Spark, Scala, Delta Lake, Kubernetes, Argo Workflow, Docker, AWS S3, AWS SNS, AWS SQS, Git
- **Software Engineering:** Continuous Integration & Deployment, Agile Development, Test Driven Development, Code Reviews
- **Collaboration:** Excellent communication skills, cross-functional collaboration, context-driven and domain knowledge interest

Work Experience

Arcesium

Software Engineer **Jul 2022 - Jul 2024, Bangalore**

- **Met client SLOs** of maximum 10 minutes for time to output across different datasets(1000-100,000 output data records) by redesigning **SQL operations**, **debugging optimizations in Python & Java** code for **Spark**, ***improving process times by 40%*.**
- **Designed partitioning strategy** for constant read & merge write performance in Delta-Tables storing historical data.
- **Owned development** for implementing config driven data-filtering module via partition pruning for static & dynamic filters from conception to delivery, **reducing compute resource usage** and streamlining ***processing workflows by 30%.***
- **Optimized resource** provisioning by **bucketing system loads, benchmarking** executor resources (instances, cores, storage) using ArgoWorkflow(like AirFlow) on Kubernetes, and deploying to AWS, ***adopted by 10+ dataset flows***.
- Implemented **SQL** transformations to perform slowly changing dimensions type 2 merges which maintain **bi-temporality** of data, enabling clients to query historical records and draw key insights from the **state of data at specific points in time**.
- Wrote **LLD documentation** with business logic translation ***reducing inquiries by 50%*** from cross-functional teams.

Software Engineering Intern **Feb 2022 - May 2022, Remote**

- ***Increased ETL throughput by 70%*** via preprocessing stage that splits heterogeneous data, enabling parallel processing.
- **Built data ingestion module** supporting input types(Parquet, CSV, txt) with schema validation to ensure data consistency.
- **Increased ETL robustness** by engineering scalable Python module to extract erroneous records to global kickouts Delta-Table.
- **Improved ETL observability** by introducing efficient run history logging, storing pipeline execution statistics in Delta-Tables.

Software Engineering Intern **May 2021 - Jul 2021, Remote**

- **Automated PDF Parsing** in **python** to identify and parse tabular information using CascadeTabNet, OpenCV, Camelot.
- **Implemented CRON processing functionality** to parse daily incoming PDF data using stored filters and annotations, ensuring timely and accurate data processing, ***saving 4 hours of work per week*** for manually applying annotations to daily PDF files.

Projects

Image Processing Application (Java) (Course Project) **October 2024 – November 2024, Boston**

- Image processing application in **Java**, accepting inputs from CLI, txt script file, or GUI to process transformations on images. Project build **intended to demonstrate programming design principles** learnt as part of graduate course work.
- Following **SOLID principles** using passive MVC model, functional interfaces, higher order functions, factory & command design patterns, and test-driven development.

Qthreads – Cooperative user space thread library (C) (Course Project) **November 2024, Boston**

- A **user-space thread library** in **C**, like the POSIX threads library, implementing functions to create threads, yield threads, put threads to sleep, provide mutexes and condition variables for managing concurrency in a **cooperative threading** model.

Simple Linux Shell (C) (Course Project) **October 2024, Boston**

- Built a C-based Linux shell with command execution for internal & external commands, piping, & I/O redirection using **fork.**