Shyam Patel

Provided on Request | Provided on Request | shyamp99.github.io | shyamp99 | shyamp99

Experience

Meta
Feb 2024 - Present

Software Engineer

- Primarily work on **unreleased devices** and other devices like **Orion**, **Ray Ban Meta** under the Wearables Audio Systems Team
- Build highly optimized features like audio state crash recovery, and adaptive volume control to meet strict requirements due to limited hardware capabilities (i.e code that lives on devices)
- The Wearables audio team owns the underlying firmware for audio systems spanning from hardware level to just below user space
- Built dashboard for displaying daily MCU and DSP memory usage analysis on **Ray Ban Meta** and other unreleased products using **plotly**
- Primarily work with **C++**, **C**, **Java**, Python, **Plotly** and SQL

Cardinal Capital Management (acquired by IMC)
Nov 2022 - July 2023

Software Developer

- Collaborated with traders to build and design features for the prop shop's systems to improve both fundamental and high frequency strategies. Features/improvements would have to be capable of handling trades for strategies that can span days or nanoseconds
- Built daily report with interactive dashboards for algo and FPGA analysis. Report analyzed **100s of market updates for each of the algos' +175,000 daily attempted CBOE or CME trades**. Utilized parallel processing, to ensure completion within an hour after market close.
- Improved performance for accepting and persisting CBOE floor trades by implementing **multithreading with immutable message passing**
- Primarily worked with **Java**, **Python**, **Pandas**, **Numpy**, **Plotly** and **C++**

BNY Mellon
Aug 2021 - Oct 2022

Software Engineer - Infrastructure

- Worked on the Kafka Dev team which owned the bank's internal **Apache Kafka infrastructure as a service platform**
- Built and maintained new features while aiding in a highly reliable platform to help deploy applications handling millions of dollars
- Worked with **Apache Kafka**, **Java**, Ansible, Junit, Cucumber/Gherkin, Bash Scripting as well as other frameworks and technologies

Rutgers University
May 2020 - May 2021 (TA)
Sept. 2019 - May 2021 (I-Lab)

Teaching Assistant and I-lab Assistant

- As TA: taught grad students the fundamentals of data science in **Python** and how to use libraries like **Matplotlib**, **Pandas** and **Numpy**
- As I-Lab assistant: taught and aided students in a myriad of topics. Some include: **Algorithms**, **Operating Systems**, **Artificial Intelligence**

LEFTE Lab
Jul. 2020 - Sept. 2020

Research Assistant

- Designed, implemented and optimized both concurrent and asynchronous color image processing for CV driven drone navigation at 30 fps (the maximum frame rate of the camera on drone) using **Python**, and OpenCV

Skills

Languages	C++, Java, C, Python
Frameworks/Libraries	Numpy, Pandas, Plotly, MySQL
Technologies	Git, Docker, Linux (CentOS, Ubuntu)

Education

Rutgers University
Sept. 2017 - May. 2021

B.S. in Computer Science — *New Brunswick, NJ*

- Relevant Coursework: **Operating Systems**, **Algorithms**, **Internet Technology**, **Graph Theory**, **Differential Equations**, Brain Inspired Computing (Graduate), Deep Learning, Introduction to Artificial Intelligence (Graduate), Systems Programming, Discrete Mathematics and Probability

Projects

BERT-CNN-Toxic-Speech-Classifier
Dec. 2020

repo: Shyamp99/BERT-CNN-Toxic-Speech-Classifier — *Group*

- Employed Google's **Bidirectional Encoder Transformation for Transformers (BERT)** architecture with a **Convolutional Neural Network** to classify online comments across 6 toxic labels: Toxic, Severe Toxic, Obscene, Threat, Insult and Identity Hate
- The model was trained using a **159,571 datapoint dataset** and achieved an **average 96.8% ROC-AUC for all labels (all labels were >93.9%)**
- Built using: **Python**, **Pytorch (with Cuda)**, **Plotly**, **Numpy**, **Pandas**, **Scikit-Learn** and Hugging Face

User Level Memory Management Simulator
Apr. 2020

Repo: Shyamp99/Virtual-Memory — *Partner*

- Designed and implemented user level memory management with a Translation Lookaside Buffer using **C Standard Library**
- Handles address translation from virtual to physical addresses, fragmentation in both virtual and physical memory and malloc/free operations