Jainish Parikh

San Jose, CA | (408)-828-4888 | jainishparikh97@gmail.com | linkedin.com/in/jainish-parikh | github.com/jainishsjsu

Summary:-

During Bachelor's most of my work was focused on Machine Learning, but specialization in Cloud computing and Virtualization in Masters have broadened my horizons. Having been exposed to Artificial Intelligence, Cloud Computing I would like to work in a place where I get to enhance my knowledge in them and help the community build a better product utilizing my skills.

Skills:-

Languages - Python, Java, C, Golang **Frontend -** HTML,CSS, JavaScript
Database - MySQL, MongoDB **Backend -** Django, Flask, NodeJS, ExpressJS, PHP
Cloud Technologies/Services - Docker, Kubernetes(GKE, EKS), Terraform, Jenkins, AWS, GCP, API Gateways, Git
Data Science - Machine Learning(Keras, NumPy, Pandas, Scikit-Learn), Deep Learning(RNN, LSTM, GRU), Natural Language Processing(Word Embeddings, Word2Vec)

Education:-

Master of Science in Software Engineering ***August 2019 - (Expected) May 2021***
San Jose State University, San Jose, CA, United States
Bachelor of Engineering in Information technology ***August 2015 - May 2019***
L.D. College of Engineering, Ahmedabad, India

Experience:-

Juniper Networks, Sunnyvale, CA, US ***May 2020 - Present***

Site Reliability Engineer Intern

- Created Dashboards, Metrics, and generated alerts for various services in monitoring tools such as SignalFx and AWS CloudWatch using Terraform.
- Used Terraform to create and manage resources in AWS environments.
- Documented monitoring so that any issues in production can be resolved as quickly as possible.

Solusoft Technologies, Ahmedabad, India ***March 2019 - May 2019***

Python Developer

- Conceptualized and successfully built a sentiment classifier using Keras which categorized emails according to severity.
- Built an automated pipeline that fetches emails periodically and updates the data and visualization charts accordingly.
- Applied Word embeddings using Word2Vec to process textual data from emails and Integrated Named Entity Recognition to determine the various entities from the email.

Projects:-

BookShow (AWS, Golang, Docker, MongoDB, Kong API Gateway)

- Developed a multi-cloud Service application(SaaS) which served as a movie booking web interface.
- Built a containerized RESTful API for User registration in Golang.
- Designed and managed an independent sharded MongoDB cluster in private VPC in AWS.
- Implemented frontend using ReactJS and it was hosted on Google Cloud Platform along with Kong API gateway.

AiSight (Python, Flask, Docker, Machine Learning):

- AiSight is a portable device that helped visually impaired people better comprehend their surroundings using Computer Vision and Natural Language Processing.
- Utilized a machine learning model for image captioning which described images instead of objects in the image.
- Containerized model using Docker and deployed it on cloud(AWS). Flask was used to synchronize multiple requests.
- Managed multiple requests simultaneously by AWS Load balancing and Auto-Scaling.

Replicated Key-Value Pair Database (Java, Restlet, Docker)

- Developed a peer-to-peer always Available, Partition Tolerant and eventually Consistent (CAP) database system.
- The Decentralized system consisted of 5 nodes which incorporated a broadcasting mechanism for peer-to-peer communication.
- Managed conflicts after partitions by the implementation of Vector clocks.
- Wrote a RESTful API and implemented an algorithm to support functioning during partitions.

Opinion Miner (NodeJS, REST APIs, Jenkins, Docker, Kubernetes, AWS, MongoDB)

- Data from twitter was mined using Twitter API and analyzed so that businesses can make better strategic decisions.
- Developed REST APIs in NodeJS and ExpressJS to interact with the database and serve front-end. Integrated Twitter APIs to fetch and process tweets before storing them in the database.
- Jenkins and Docker were used for Continuous Integration and Continuous Deployment along with Kubernetes for cluster management. Visualization of results in the front-end was done using jQuery, HTML, CSS, and D3.js