Brian Dashnaw

Python Developer

brian.prop56@gmail.com
+1 (607) 527-5375
Orlando, FL 32828, USA

PROFESSIONAL SUMMARY:

I am a Senior Software Engineer and Python Developer with over 15 years of experience designing, developing, and automating **cloud-native applications** using **Python** and modern frameworks like **Flask**, **Django**, and **FastAPI**. With expertise in **RESTful APIs**, **microservices architectures**, and cloud platforms such as **AWS**, **Google Cloud**, and **Azure**, I build scalable and resilient backend systems. I specialize in optimizing system performance through **containerization** with **Docker** and **Kubernetes**, while streamlining deployment processes with **CI/CD pipelines** using **Jenkins** and **GitLab**. My background includes data processing with **Pandas** and **NumPy**, along with deploying **machine learning models** using **TensorFlow** and **scikit-learn**. I am proficient in implementing **infrastructure as code (IaC)** using **Terraform** and **Ansible**, driving automation, cost optimization, and high availability across environments

EXPERIENCE:

EY (Orlando, FL)
Senior Software Engineer / Python Developer (04/2020 – Present)

- Led the development of **microservices** using **Python** with frameworks like **Flask** and **Django**, improving modularity and increasing deployment speed by 40%.
- Developed and maintained **RESTful APIs** using **FastAPI** and **Flask**, improving system performance by reducing API response times by 30%.
- Designed and built **data pipelines** using **Pandas** and **NumPy** to automate data transformation and integration, improving efficiency by 50%.
- Automated infrastructure deployment using **Terraform** and **Ansible**, reducing manual configurations by 60% and ensuring consistent setup across environments.
- Built and deployed serverless applications using **AWS Lambda** and **API Gateway**, reducing operational overhead and improving performance for event-driven workloads.
- Integrated **CI/CD pipelines** using **Jenkins** and **GitLab**, automating build, testing, and deployment processes, reducing manual intervention by 50%.
- Developed **ETL pipelines** using **Python** and **PySpark**, improving real-time data processing capabilities and enabling large-scale data analytics.
- Implemented **containerization** for Python applications using **Docker**, enabling portability and reducing environment setup times by 40%.
- Integrated **Celery** with **RabbitMQ** to manage background tasks, improving the system's ability to handle high-volume asynchronous processes.
- Optimized **SQLAlchemy** queries for **PostgreSQL** and **MySQL**, reducing query execution times by 40% and improving database performance.
- Managed container orchestration using **Kubernetes**, ensuring scalability and high availability for Python-based microservices under high traffic conditions.

- Built interactive **data visualization dashboards** using **Plotly** and **Dash**, providing real-time insights into business metrics and system performance.
- Integrated **Prometheus** and **Grafana** for real-time monitoring and alerting, improving system reliability and reducing **MTTR** by 35%.
- Developed automated testing frameworks using **PyTest** and **Unittest**, increasing code coverage and reducing bugs in production by 30%.
- Built and managed a **data lake** using **AWS S3** and **Glue**, enabling efficient big data storage and analytics for business decision-making.
- Implemented **blue-green deployments** and **canary releases** using **Kubernetes**, achieving zero-downtime deployments and improving user experience.
- Automated security tasks using **Ansible**, handling tasks such as **IAM role assignments** and **VPC configurations**, ensuring compliance with security standards.
- Collaborated with data scientists to deploy **machine learning models** using **Flask** and **TensorFlow**, improving model inference times by 35%.
- Implemented **API rate limiting** and **throttling** mechanisms for **Flask** APIs, improving system stability during high-traffic periods.
- Managed **Kafka** integration for real-time data streaming, improving data pipeline performance and reducing latency in real-time processing by 25%.
- Built and maintained **machine learning pipelines** using **Python** and **scikit-learn**, deploying models efficiently in production environments.
- Played a key role in optimizing **Docker images** for Python applications, reducing image sizes by 50% and speeding up container deployment times.
- Improved cloud infrastructure management using **Terraform**, automating resource provisioning and reducing human errors during environment setup.
- Led the migration of on-premise applications to the cloud, using **Python** and **AWS** tools, improving system scalability and reducing maintenance overhead.
- Collaborated with frontend developers to build **GraphQL APIs** using **Graphene** in Python, enabling efficient data fetching and improving the user experience.

TCS (Tampa, NY)
Senior Software Engineer / Python Developer (05/2017 – 03/2020)

- Led the development of **Python-based microservices** using **Flask** and **FastAPI**, improving system scalability and reducing development cycles by 30%.
- Built and maintained **API-driven backends** using **Django** and **Flask**, enabling seamless integration with external systems and reducing API response times by 25%.
- Designed and implemented **ETL pipelines** using **Python**, **Pandas**, and **Airflow**, improving data transformation processes and increasing data accessibility by 40%.
- Automated cloud infrastructure provisioning using **Terraform** and **Ansible**, reducing setup times by 50% and improving deployment consistency across environments.
- Developed **serverless applications** using **AWS Lambda** and **Google Cloud Functions**, reducing operational costs and improving the performance of event-driven workloads.

- Designed and implemented **CI/CD pipelines** using **GitLab CI** and **Jenkins**, automating Python application testing and deployment, reducing release times by 35%.
- Optimized **PostgreSQL** and **MongoDB** queries using **SQLAlchemy**, reducing query execution times by 40% and improving overall system performance.
- Built real-time data processing pipelines using **Python** and **PySpark**, improving data analytics capabilities and reducing processing times by 30%.
- Implemented container orchestration using **Docker** and **Kubernetes**, improving scalability and reliability of Python-based applications in production environments.
- Integrated **unit testing** and **integration testing** using **PyTest** and **Unittest**, improving code quality and reducing bugs in production by 20%.
- Developed **real-time monitoring** solutions using **Prometheus** and **Grafana**, enabling proactive issue detection and reducing system downtime by 25%.
- Led the development of a **machine learning platform** using **Django** and **TensorFlow**, improving the deployment and management of ML models in production.
- Automated security configurations using **Ansible**, managing **AWS IAM roles** and **VPC security groups** to ensure cloud security and compliance.
- Built and optimized a **data warehouse** using **Amazon Redshift**, improving query performance for large datasets and reducing data retrieval times by 30%.
- Integrated **Kafka** for real-time data streaming, improving pipeline performance and enabling faster decision-making through real-time analytics.
- Developed **custom Python scripts** for system monitoring, backup automation, and data migration, improving operational efficiency and reducing manual intervention.
- Enhanced API security by implementing **OAuth2** and **JWT authentication** in Python-based APIs, improving data protection and access control.
- Collaborated with frontend teams to build **GraphQL APIs** using **Django** and **Graphene**, improving the efficiency of data fetching and enhancing the user experience.
- Developed **asynchronous tasks** using **Celery** and **RabbitMQ**, improving system performance by enabling high-volume asynchronous processing.
- Implemented **blue-green deployments** using **Kubernetes**, ensuring smooth rollouts of updates with minimal downtime for end-users.
- Managed **multi-region deployments** of Python services using **AWS**, ensuring high availability and disaster recovery capabilities across data centers.
- Optimized Python-based machine learning pipelines, integrating **scikit-learn** and **TensorFlow**, improving model training and inference speeds.
- Developed a centralized logging solution using **ELK Stack**, improving error detection and troubleshooting capabilities across environments.
- Played a critical role in reducing infrastructure costs by 20% by optimizing the use of **EC2 Spot Instances** and **auto-scaling groups**.
- Built and maintained real-time **data visualization dashboards** using **Dash** and **Plotly**, providing business stakeholders with actionable insights.

Atlassian (New York, NY)
Senior Software Engineer / Python Developer (04/2015 – 04/2017)

- Developed high-performance **RESTful APIs** using **Flask** and **Django**, supporting mission-critical applications with high traffic, reducing latency by 30%.
- Led the migration of legacy applications to Python-based **microservices** using **Django** and **Flask**, improving system scalability and reducing maintenance overhead.
- Automated infrastructure deployment using **Terraform** and **Ansible**, improving infrastructure consistency and reducing manual configuration errors by 50%.
- Built and managed containerized Python applications using **Docker**, improving deployment speed and scalability across production environments.
- Integrated **CI/CD pipelines** using **Jenkins** and **GitLab**, automating testing and deployment workflows, improving deployment frequency by 40%.
- Designed **data pipelines** using **Python** and **Pandas**, automating data transformation and improving data analysis efficiency for business teams.
- Implemented real-time **monitoring** solutions using **Prometheus** and **Grafana**, improving system uptime and reducing downtime by 20%.
- Developed automated **testing frameworks** using **PyTest** and **Unittest**, improving code quality and reducing bugs in production by 30%.
- Designed and implemented **ETL pipelines** using **Airflow**, enabling efficient data extraction, transformation, and loading for large-scale data processing.
- Collaborated with data scientists to deploy **machine learning models** in production using **Flask** and **TensorFlow**, reducing model inference times by 30%.
- Built **data visualization tools** using **Dash** and **Plotly**, providing business stakeholders with real-time insights into system performance metrics.
- Implemented **blue-green deployments** and **canary releases** using **Kubernetes**, reducing downtime during production deployments by 25%.
- Improved database performance by optimizing **SQLAlchemy** queries for **PostgreSQL** and **MySQL**, reducing query times by 35%.
- Automated cloud resource provisioning using **Terraform**, ensuring scalable and reliable infrastructure across multiple regions for Python-based services.
- Developed **real-time data streaming** solutions using **Kafka** and **Python**, improving data processing latency and supporting event-driven architectures.
- Integrated **OAuth2** and **JWT authentication** into Python APIs, improving security protocols for internal and external applications.
- Managed multi-region deployments for Python services using **AWS**, ensuring high availability and reducing downtime during disaster recovery.
- Reduced the size of **Docker images** for Python applications by 40%, improving deployment times and reducing resource consumption.

- Led the optimization of **Python web applications**, reducing page load times and improving user experience through caching and database query optimization.
- Built custom Python-based scripts for **system monitoring**, task automation, and data migration, improving operational efficiency across the team.
- Implemented **machine learning pipelines** with **Django** and **scikit-learn**, enabling real-time model training and inference for business applications.
- Integrated **Redis** as a caching layer for Python applications, reducing load times and improving system responsiveness by 40%.
- Improved cloud infrastructure automation with **Ansible**, reducing the time required to provision resources and improving operational efficiency.
- Developed and maintained **unit tests** using **PyTest**, ensuring high test coverage and reducing defects in production by 30%.

TECHNICAL SKILLS:

Primary Stack:

- Java 10+, Spring Boot, Hibernate, Microservices, REST APIs, GraphQL, Maven, Gradle

Frameworks/Libraries:

- Spring (Spring Boot, Spring MVC, Spring Security, Spring Data), React, Angular, Vue.js, JPA

Databases:

- PostgreSQL, MySQL, MongoDB, SQL Server, Oracle Database, SQLite, Redis, DynamoDB, Cassandra

Cloud & DevOps:

- AWS (Lambda, S3, EC2, RDS, CloudFormation, EKS), Google Cloud Platform, Azure
- **Terraform, Ansible, AWS CloudFormation** for **Infrastructure as Code (IaC)**
- **CI/CD tools:** Jenkins, GitLab CI, AWS CodePipeline
- **Containerization:** Docker, Kubernetes, AWS Fargate
- **Serverless computing:** AWS Lambda, API Gateway, Google Cloud Functions
- **Multi-cloud orchestration** using **Kubernetes** and Terraform
- **Monitoring & Logging:** Prometheus, Grafana, Splunk, CloudWatch, ELK Stack
- **Cloud cost optimization:** EC2 Auto Scaling, Spot Instances, AWS Cost Explorer
- **Container security:** Kubernetes RBAC, Open Policy Agent (OPA), and Aqua Security
- **API management:** API Gateway, NGINX, Kong API Gateway
- **Load balancing & Traffic management:** NGINX, AWS Elastic Load Balancer (ELB), and Traefik
- **Automated testing:** Selenium, JUnit, PyTest, Cucumber for continuous testing in CI/CD
- **Automated infrastructure scaling:** Kubernetes Horizontal Pod Autoscaler, AWS Auto Scaling
- **Secrets management:** AWS Secrets Manager, HashiCorp Vault
- **Disaster recovery:** AWS Backup, S3 Versioning, and Cross-Region Replication
- **Orchestration frameworks:** Helm for Kubernetes, Docker Compose for multi-container Docker applications
- **Security compliance automation:** AWS Config, Ansible Security Playbooks

- **GitOps workflows:** FluxCD, ArgoCD for declarative Kubernetes management
- **Incident management:** PagerDuty, OpsGenie for automated alerting and incident response
- **Network & security monitoring:** AWS VPC Flow Logs, Suricata for intrusion detection
- **Cloud network architecture:** AWS VPC Peering, Transit Gateway, CloudFront for CDN and DDoS protection
- **Performance optimization tools:** Gatling, Apache JMeter for load testing and benchmarking
- **Service mesh implementation:** Istio for managing microservices communication across Kubernetes clusters
- **Backup & Restore automation:** AWS Backup, Velero for Kubernetes cluster backups
- **Compliance and governance tools:** AWS Organizations, AWS Service Control Policies (SCP)

Primary Stack:

- Python (Flask, Django, FastAPI), Pandas, NumPy, SQLAlchemy, PySpark, Celery

Frameworks/Libraries:

- Django, Flask, FastAPI, GraphQL (Graphene), TensorFlow, scikit-learn, PyTest, Unittest

Databases:

- PostgreSQL, MySQL, MongoDB, DynamoDB, SQLite, Redis

Cloud & DevOps:

- AWS (Lambda, EC2, S3, RDS, CloudFormation, EKS), Google Cloud Platform, Azure
- Terraform, Ansible, Docker, Kubernetes, AWS Fargate

Data Processing & Analytics:

- Pandas, NumPy, PySpark, Airflow, Plotly, Dash, Jupyter Notebooks

CI/CD & Automation:

- Jenkins, GitLab CI, AWS CodePipeline, CircleCI, Travis CI

Monitoring & Logging:

- Prometheus, Grafana, Splunk, CloudWatch, ELK Stack (Elasticsearch, Logstash, Kibana)

Machine Learning & Data Science:

- TensorFlow, scikit-learn, Keras, PyTorch, Jupyter

API Development & Security:

- RESTful APIs, GraphQL APIs, OAuth2, JWT Authentication

Testing & Quality Assurance:

- PyTest, Unittest, Selenium

EDUCATION:

Bachelors of Computer Science

Standford University **01/2007 – 03/2011**