

# Alejandro Bernal Collazos



## Interests

Software Engineering

DevOps

Web Development

Internet Of Things

Continuous Deployment

## Contact Information

<b>Phone</b>	(+0057 321 431 9162)
<b>Email</b>	alejandro@alejandro.bio
<b>Birth date</b>	September the 16th of 1986
<b>Website</b>	<a href="https://www.alejandro.bio">https://www.alejandro.bio</a>



## Professional Experience



**Position** Senior Cloud Automation Engineer

**Company** Upwork

**Start date** February the 18th of 2020

**Finish date** September the 03rd of 2024

### Infrastructure as Code with Terraform

- Design and implement infrastructure for development teams using Terraform Enterprise, adhering to industry best practices for modularity and scalability.
- Served as a Terraform Manager, overseeing code quality, maintaining standards, and guiding teams in adopting IaC methodologies.

### Database Infrastructure Support

- Assisted DBA teams in adhering to Operations Engineering standards for setting up and managing infrastructure (linux servers) with terraform and chef.
- Troubleshoot issues with Data Infrastructure in Staging and Prod environments.
- Troubleshoot issues with RDS PostgreSQL DBs for tools that use them like Sonarqube and Retool.

- Troubleshoot issues together with the Data infrastructure team related to the automations in place to extract huge volumes of data from the different DBs in the organization, transform such data and then load them into Snowflake.
- Database Benchmark, Design and Architecture for tools needed in the organization
- Analyze business requirements to design database structures (permissions, schemas, tables, indexes, etc.).
- Choose the appropriate database technology (e.g., SQL, NoSQL, relational, or distributed databases).
- Ensure the database architecture supports scalability and high availability.
- Diagnose and resolve performance bottlenecks related to database configurations or queries.
- Regularly analyze database logs and metrics to identify inefficiencies.
- Implement database backup and recovery solutions to safeguard against data loss.
- Create backup schedules (full, incremental, or differential) and verify backups regularly.
- Perform disaster recovery tests to ensure data can be restored in emergencies.
- Implement this on tools like
  - Sonarqube
  - Retool
  - Biticket
  - Rundeck

### **SonarQube Deployment & Automation**

- Investigated, Deploy, Test and Manage **SonarQube** for code quality analysis across more than 2,000 projects, integrating it into CI/CD pipelines.
- Deploy Sonarqube in Kubernetes and managed through ArgoCD.
- Automated configuration monitoring for SonarQube using its APIs to ensure system integrity and availability, through an scheduled tasks executed in Kubernetes, scripted in python and managed by ArgoCD.
- Support of sonarqube infrastructure, implementation and configuration.

### **Load Balancer & CDN Management**

- Deploy, manage, support and optimize **Nginx load balancers layers**, handling incoming requests from cloudflare and ensuring high availability.

- Manage and support **Cloudflare** to handle all requests, applying appropriate routing and forwarding policies to internal load balancers.
- Troubleshoot issues with the request handling in staging and production environments, together with development and engineering teams responsible for the AWS ECS microservices layers that hold most of the company application logic

### Configuration Management with Chef

- Deploy, support and managed **Chef** to standardize configuration management across all Linux servers, ensuring consistent environments and reducing manual intervention.
- Troubleshoot issues with the Chef recipes for the different types of tools used by many teams, like the Search Teams, Security Teams, Data Infrastructure Teams, etc.

### Monitoring & Automation

- Automated routine infrastructure tasks (setting up a new linux server in a OpenSearch cluster or a new node in the kafka cluster), including monitoring configurations and system health checks, leveraging Python scripts for enhanced visibility into the logging infrastructure.
- Proactively managed and resolved issues within the **logging pipeline**, which included Filebeat log extractor, Logstash Servers, **Kafka Clusters** and **OpenSearch**.
- Troubleshooting and Fix issues with the OpenSearch indexes

### Production Support & Troubleshooting:

- Diagnosed and resolved complex production issues related to request routing between external clients and internal infrastructure components.
- Improved system reliability by addressing critical bottlenecks and optimizing routing policies.

### AWS Cloud Management:

- Proficient in managing a wide range of **AWS services**, including **ECS, EC2, RDS, Lambda, IAM, CloudWatch**, and **OpenSearch**, ensuring scalability, security, and cost-efficiency across all cloud operations.

### Core Skills and Tools



- Terraform, Chef, SonarQube, Nginx, Cloudflare
- Docker, Kubernetes, Linux, ArgoCD, GitHub, Bitbucket
- PostgreSQL, MySQL, MongoDB, AWS Aurora
- Python scripting for automation and monitoring, Java, JavaScript
- Kafka, Filebeat, Logstash, OpenSearch
- AWS (ECS, EC2, RDS, Lambda, CloudWatch, IAM)
- Troubleshooting and resolving production infrastructure issues



**Position**      **DevOps and DBA**

**Company**      **IncluIT**

**Client**        **Naranja**

**Start date**    **June of 2019**

**Finish date**   **April of 2020**

## **Activities**

**Database Design and Architecture**

- Analyze business requirements to design database structures (schemas, tables, indexes, etc.).
- Choose the appropriate database technology (e.g., SQL, NoSQL, relational, or distributed databases).
- Ensure the database architecture supports scalability and high availability.

### **Database Deployment and Setup**

- Install, configure, and upgrade database management systems (e.g., MySQL, PostgreSQL, MongoDB, Oracle, SQL Server).
- Set up and configure database instances, clusters, and failover mechanisms.
- Implement database replication, sharding, and partitioning for large-scale deployments.

### **Performance Tuning and Optimization**

- Monitor and optimize database performance, including query tuning, indexing, and cache management.
- Diagnose and resolve performance bottlenecks related to database configurations or queries.
- Regularly analyze database logs and metrics to identify inefficiencies.

### **Backup and Recovery**

- Implement database backup and recovery solutions to safeguard against data loss.
- Create backup schedules (full, incremental, or differential) and verify backups regularly.
- Perform disaster recovery tests to ensure data can be restored in emergencies.

### **Security Management**

- Implement access control mechanisms to secure sensitive data.
- Apply database encryption techniques (e.g., at-rest and in-transit encryption).
- Monitor for and protect against database vulnerabilities and unauthorized access.

### **Maintenance and Monitoring**

- Monitor database health using tools like CloudWatch, Prometheus, Grafana, or custom scripts.
- Apply patches and upgrades to database software to keep it secure and up-to-date.
- Regularly perform database maintenance tasks like reindexing, vacuuming, and purging old data.

### **Troubleshooting and Support**

- Diagnose and resolve database-related issues, such as failed queries, replication lag, or



connectivity problems.

- Work with development teams to debug and fix database-related application issues.
- Provide on-call support for production database issues.

### **Automation and DevOps Integration**

- Automate routine tasks like backups, schema migrations, and scaling using scripts or tools (e.g., Terraform, Ansible).
- Collaborate with DevOps teams to integrate databases into CI/CD pipelines.
- Use infrastructure as code (IaC) to provision and manage database environments.

### **Capacity Planning**

- Monitor database storage usage and plan for future growth.
- Perform database scaling (vertical or horizontal) to handle increasing workloads.
- Forecast resource requirements to ensure database performance remains optimal over time.

### **Compliance and Auditing**

- Ensure databases comply with regulatory standards (e.g., GDPR, HIPAA, PCI-DSS).
- Implement auditing solutions to track data access and modifications.
- Provide audit reports to ensure accountability and compliance.

### **Key Tools and Technologies Used**

- Relational Databases: MySQL, PostgreSQL, Oracle, SQL Server.
- NoSQL Databases: MongoDB, Cassandra, DynamoDB, Redis.
- Monitoring Tools: CloudWatch, Nagios, Zabbix, Prometheus, Datadog.
- Scripting: Python, Shell scripting, PowerShell.
- Backup Tools: Percona, pgBackRest.
- Automation: Terraform, Ansible, Chef.

### **Infrastructure**

- Design service and application infrastructure
- Manages pipeline with Gitlab to deploy app/service into production
- Configures the monitor tools with DataDog
- Configure the kubernetes clusters for these environments
  - Develop
  - Staging
  - Production
- Create bash scripts to automate processes like backups, cleaning and notification



## Continuous Integration / Delivery / Deployment with Jenkins

- Implement GitLab to orchestrate these process
  - Continuous Integration
  - Continuous Delivery
  - Continuous Deployment
- Work side by side with development teams in order to standardize and create pipeline as code with Python for their components using Docker
  - FrontEnd
  - Backend
  - Database
- Setup, configure and maintain Docker Registry to store the images from the Continuous Delivery process
- Setup, configure and maintain Kubernetes Clusters for these environments
  - Development
  - Staging
  - Production

## Support

- Help the development with
  - How to perform deployments into production
  - How to request for development environments
  - Coach and teach about the architecture of the infrastructure
    - Development environments
    - Production environments

# ASYS

<b>Position</b>	<b>Teacher</b>
<b>Company</b>	<b>ASYS</b>
<b>Client</b>	<b>Santex - IncludIT - Mundos E</b>
<b>Project</b>	<b>Courses on DevOps</b>
<b>Start date</b>	<b>January of 2019</b>
<b>Finish date</b>	<b>July of 2021</b>

## Activities

### DevOps Course

#### 1. DevOps

**Objective:** Understand the concept of DevOps and its three fundamental principles through an integrative project.

##### 1.1 What is DevOps:

Software development culture based on:

- Flow
- Feedback
- Continuous Improvement

##### 1.2 Course Overview:

Objectives, structure, theory, integrative project, course mode, tools.

##### 1.3 Integrative Project:

Create an online résumé implementing DevOps principles.

##### 1.4 Best Practices:

- Scrum, Kanban, Behavior-Driven Development (BDD), Test-Driven Development (TDD), Gitflow, CI/CD, Infrastructure as Code, Logging standards.
- **1.5 Methodology:**
- 30% Theory / 70% Practice

## 2. Learning to Learn

## 3. Kanban

### 3.1 Kanban Overview:

Understand, design, and implement Toyota's Kanban method for support and operation-oriented projects.

### 3.2 Jira Configuration:

Use Jira to set up a Kanban project for software operation tasks.

## 4. Language

## 5. Scrum

### 5.1 Scrum Overview:

Design and work on an Agile-based project using Jira for software development.

## 6. Team Building

## 7. Software Architecture

**Goal:** Understand and implement design patterns for scalable, distributed, resilient applications.

### 7.1 Distributed Computing Design:

Decoupled application layers (e.g., FrontEnd, BackEnd, DataEnd).

### 7.2 Horizontal Scalability:

Design the résumé for scalable deployment across multiple machines.

### 7.3 Operations Design:

Manageability, debugging, monitoring, and traceability.

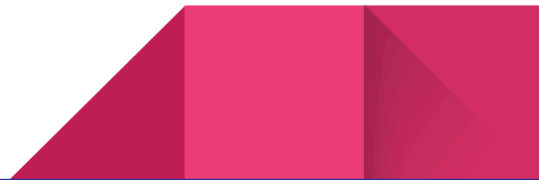
### 7.4 Resilience Design:

Ensure continuity when components fail.

## 8. Quality

Focus on quality practices like TDD, Unit Tests, Service Tests, and BDD.

### 8.1 Test-Driven Development



## 8.2 Behavior-Driven Development

## 8.3 Unit Tests

## 8.4 Service Tests

## 9. Feedback and Conversations

## 10. Networking, AWS, Linux, and CentOS

### 10.1 How the Internet Works:

Internet architecture, TCP/IP, DNS servers.

### 10.2 Amazon AWS:

Components and features of AWS.

### 10.3 Linux Overview:

Processes, services, networking, file systems, containers.

### 10.4 Virtual Private Servers:

Work with DigitalOcean VPS.

### 10.5 Nginx Installation:

Install and configure Nginx manually.

## 11. Infrastructure as Code

**11.1 Terraform:** Infrastructure creation with code.

**11.2 Ansible:** Configuration management with code.

## 12. Git and GitHub

**12.1 Git Basics:** Commands, branches, commits, etc.

**12.2 GitHub Repository:** Create and upload a repository.

**12.3 Gitflow:** Branching model.

## 13. Neuroscience and Emotional Intelligence

## 14. API Manager



**14.1 Installation:** Choose and install an API manager.

**14.2 Configuration:** Set up and test configurations.

## 15. Front-End Development

**15.1 Wiremock Design**

**15.2 CSS & HTML**

**15.3 ReactJS**

**15.4 Unit Tests**

**15.5 Caching**

## 16. Back-End Development

**16.1 RESTful APIs**

**16.2 NodeJS:** Logging, error handling, migrations, security.

**16.3 Dockerized APIs**

**16.4 Gitflow-Based API Repositories**

**16.5 Jenkins Pipelines**

**16.6 Continuous Delivery for APIs**

**16.7 Continuous Deployment for APIs**

**16.8 Caching Policies**

## 17. DataEnd

**17.1 MySQL:** Installation and usage.

**17.2 PostgreSQL:** Installation and usage.

**17.3 Data Pipeline:** Implement the data layer pipeline.

**17.4 Caching Policies for Databases**

## 18. Security

**18.1 Overview:** Authentication, authorization, attack prevention.

**18.2 Security Policies**



**18.3 OWASP:** Secure software development.

## **19. Docker**

**19.1 Overview:** Virtualization basics.

**19.2 Dockerized UI**

**19.3 Automated Tests with Selenium**

## **20. Kubernetes**

**20.1 Overview:** Nodes, services, pods, replica sets, deployments, stateful sets, kubectl.

## **21. Jenkins and On-Demand Environments**

**21.1 Jenkins Overview**

**21.2 Development Environments:** Development, staging, production.

**21.3 Automated Environment Creation**

## **22. CI/CD**

**22.1 Continuous Integration**

**22.2 Continuous Delivery**

**22.3 Continuous Deployment**

## **23. Front-End Pipeline**

## **24. Back-End Pipeline**

## **25. DataEnd Pipeline**

## **26. Monitoring Tools**

**26.1 ELK Stack:** Application, infrastructure monitoring, logs, metrics, telemetry.

## **27-30. Telemetry**

**27.1 Front-End Telemetry**

**28.1 Back-End Telemetry**



## **29.1 Business Telemetry**

## **30.1 Infrastructure Telemetry**

## **31. Operations**

### **31.1 Load Balancers**

### **31.2 Content Delivery Networks (CDNs)**

### **31.3 Deployment Types:** Green-Blue, Canary, Rolling Updates.

### **31.4 Stability Principles**

## **32. Continuous Improvement**

### **32.1 Developer Time Management**

### **32.2 Postmortems**

### **32.3 Resilience Mechanisms**

### **32.4 Organizational Culture:** Fear vs. honesty.

### **32.5 Experimentation (A/B Testing, Game Days)**



# santex



**Position**                      **Site Reliability Engineer (DevOps)**

**Company**                      **Santex**

**Client**                         **Grenzebach**

**Start date**                    **June of 2017**

**Finish date**                 **December of 2018**

## **Activities**

### **Database Design and Architecture**

- Analyze business requirements to design database structures (schemas, tables, indexes, etc.).
- Choose the appropriate database technology (e.g., SQL, NoSQL, relational, or distributed databases).
- Ensure the database architecture supports scalability and high availability.

### **Database Deployment and Setup**

- Install, configure, and upgrade database management systems (e.g., MySQL, PostgreSQL, MongoDB, Oracle, SQL Server).
- Set up and configure database instances, clusters, and failover mechanisms.
- Implement database replication, sharding, and partitioning for large-scale deployments.



## Performance Tuning and Optimization

- Monitor and optimize database performance, including query tuning, indexing, and cache management.
- Diagnose and resolve performance bottlenecks related to database configurations or queries.
- Regularly analyze database logs and metrics to identify inefficiencies.

## Backup and Recovery

- Implement database backup and recovery solutions to safeguard against data loss.
- Create backup schedules (full, incremental, or differential) and verify backups regularly.
- Perform disaster recovery tests to ensure data can be restored in emergencies.


## Security Management

- Implement access control mechanisms to secure sensitive data.
- Apply database encryption techniques (e.g., at-rest and in-transit encryption).
- Monitor for and protect against database vulnerabilities and unauthorized access.

## Maintenance and Monitoring

- Monitor database health using tools like CloudWatch, Prometheus, Grafana, or custom scripts.
- Apply patches and upgrades to database software to keep it secure and up-to-date.
- Regularly perform database maintenance tasks like reindexing, vacuuming, and purging old data.

## Troubleshooting and Support

- Diagnose and resolve database-related issues, such as failed queries, replication lag, or connectivity problems.
  - Work with development teams to debug and fix database-related application issues.
  - Provide on-call support for production database issues.
- 

## Automation and DevOps Integration

- Automate routine tasks like backups, schema migrations, and scaling using scripts or tools (e.g., Terraform, Ansible).
- Collaborate with DevOps teams to integrate databases into CI/CD pipelines.
- Use infrastructure as code (IaC) to provision and manage database environments.

## Capacity Planning

- Monitor database storage usage and plan for future growth.
- Perform database scaling (vertical or horizontal) to handle increasing workloads.
- Forecast resource requirements to ensure database performance remains optimal over time.

## Infrastructure

- Design service and application infrastructure
- Defines infrastructure with Ansible
- Manages pipeline with Jenkins to deploy app/service into production
- Configures the monitor tools with Elasticsearch Logstash and Kibana (ELK) for Monitoring
- Configure the kubernetes clusters for these environments
  - Develop
  - Staging
  - Production
- Create bash scripts to automate processes like backups, cleaning and notification

## Continuous Integration / Delivery / Deployment with Jenkins

- Implement Jenkins to orchestrate these process
  - Continuous Integration
  - Continuous Delivery
  - Continuous Deployment
- Work side by side with development teams in order to standardize and create pipeline as code with Groovy for their components using Docker
  - FrontEnd
  - Backend
  - Database

- Setup, configure and maintain Docker Registry to store the images from the Continuous Delivery process
- Setup, configure and maintain Kubernetes Clusters for these environments
  - Development
  - Staging
  - Production

## Support

- Help the development with
  - How to perform deployments into production
  - How to request for development environments
  - Coach and teach about the architecture of the infrastructure
    - Development environments
    - Production environments





<b>Position</b>	<b>DevOps</b>
<b>Company</b>	<b>InclUT</b>
<b>Client</b>	<b>McAfee</b>
<b>Start date</b>	<b>February of 2017</b>
<b>Finish date</b>	<b>June of 2017</b>

### **Activities**

#### Infrastructure

- Design service and application infrastructure
- Defines infrastructure within Terraform files
- Manages pipeline with TeamCity to deploy app/service into production
- Configure testing phases in the pipeline
- Configures the monitor tools with Elasticsearch Logstash and Kibana (ELK) for Monitoring
- Implements Continuous Integration, Continuous Delivery and Deployment in the pipeline
- Configure the kubernetes clusters for these environments
  - Develop
  - Staging
  - Production
- Manage AWS
  - Manage
    - Elastic Kubernetes Services
    - Elastic Container Registry

#### Support

- Support for the development team with the implementation of the pipeline
- Troubleshoot development environment issues
-



**Position**                      **Developer within a DevOps Culture**

**Client**                              **AppDirect.com**

**Start date**                        **June of 2016**

**Finish date**                      **February of 2017**

### **Activities**

#### **Development**

- Create new features made with **Spring**, like new services, and integrations with payment gateways.
- Create new features made with **Wicket** (an apache java Framework to develop web applications) like new windows that handle new processes and perform actions on the server side.
- Make **unit tests** for every part of the code that were modified.
- Make **automated tests** for every new feature with **Selenium**.
- Use **GitHub** and **Jenkins** in order to build the new code and integrate it into the master branch to be delivered into production.
- Use **Maven** to manage package dependencies, run the legacy application locally.
- Use **Docker** to run the legacy application within a container, to avoid modification in the local machine.
- Use **AWS machine instances EC2** in order to perform automated tests before delivering a new feature into production.
- Use **JPA** in order to persist data against the DB.
- Perform code review for modifications made by my colleagues.

#### **Bug Fix**

- Troubleshoot and solve issues with the legacy application (monolith), based on **Wicket**, **Spring**, **JavaScript** using **MySQL** database.
- Make unit tests for any change in order to cover such changes.
- Make automated tests for any new change in the existent application that impact the user interface such as the business logic

## DevOps

February 2014 -> June 2016



**Position** Developer and later System Administrator within a DevOps culture

**Client** LATAM.com

**Start date** February of 2014

**Finish date** June of 2016

### Activities

#### Development

- Support and Maintain the legacy website based on **Perl**
- Create backend services based on **Java** with the Spring Boot framework
- Create and setup **Docker** for the backend services
- Work with **Git** to perform version control and **Git Flow** as the branching model
- Work with **Jira** for work tracking
- Work with **Scrum** as the Agile practice to create new feature within the development team
- Work with **Artifactory** to manage legacy backend services based on Spring
- Work with **Splunk** for **Monitoring** to perform log aggregation and notifications

#### Automation - Configuration / Setup and Maintenance

- **Scripting** Create and maintain scripts (made in **Bash** and **Python** mainly) to make Deployments (of **java** artifacts from archives into the **tomcat** servers of a given machine)
- Use **SVN** and **GIT** to version control **Chef** cookbooks
- Request machines to the cloud provider **Softlayer** through a self service layer that developers can use by their own
- Create, configure and maintain **Linux** user accounts (**CentOS** for development and **RedHat** for production)
- Create, configure and maintain **Linux** machines (CentOS for development and RedHat for production environments)
- Install and Configure servers (**Apache** http, dns, ldap ,Haproxy, Tomcat)
- Configure Linux **Firewalls**
- Configure Linux **Tunnels**
- Setup **Cron** rules to execute jobs on given schedules and automatically
- Configure Apache HTTP server
  - Setup reverse proxy rules
  - Setup virtual hosts
- Create, Maintain and configure **MySQL DB cluster**
- Configure **Haproxy** server
  - In order to perform load Balance
  - To promote service discovery
- Configure **Chef**
  - Create and Maintain Cookbooks
- Configure **Rundeck Server**
  - Define Jobs to be executed on given machines, like restart a given set of machines defined by a role
- Setup and maintain **Nagios Server** for Infrastructure **Monitoring**
  - To perform monitoring in the development environments and production environments
- Version Control Server **SVN**
  - Administration

### Bug Fix and diagnosis

- Troubleshoot and solve issues on
  - Infrastructure
  - Application
  - Deployment
  - Java Web Services

- JavaScript applications

## Support

- Help the development teams in this tasks
  - How to perform deployments into production
  - How to request for development environments
  - Coach and teach about the architecture of the infrastructure
    - Development environments
    - Production environments

## User Accounts

- Configure LDAP servers in order to
  - Create - Modify and Delete
  - Groups
  - Users
  - Domains
  - Attributes







**Position**      **Developer**  
**Client**        **AA.com American Airlines**  
**Start date**    **February of 2012**  
**Finish date**   **February of 2014**  
**Activities**

### Development

- **Java** create web services based on spring that could interact via SOAP with clients and providers of information
- **Logging** we used Log4j in order to perform the logs of our web services
- **Unit Testing** with JUnit for the unit tests
  - Test for functions in the code
- **SonarQube** usage for static code analysis
  - Coverage of 80% or greater
  - Duplicated code
  - Syntax checks
- **Branching Model** we used a flavor of gitflow based on feature branches in order to implement the continuous integration process
- **Continuous Integration** using the branching model we established a set of steps to guarantee that the code (whether on feature branches, pull requests or new merges into the develop branch) works according to our functional policy
  - Code is able to build
  - All unit tests passed
  - The amount of coverage is within the established parameters ( $\geq 80\%$ )
  - The static code analysis passed properly
    - Check for code duplication
    - Cyclomatic complexity

- Syntax check
- **Systems Development Life Cycle** The requirements for new improvements or bug fixes came from tickets and according to their priority, some of them were done as hotfixes or some of them worked out using **Agile/Scrum** within a sprint of 2 weeks period and later delivered to **Artifactory** --- using the **semantic versioning** --- as packages,, for a provider (Verizon) in charge of the deployment in a production environment
- **Software Configuration Management** Although we did not have full control over the management of the services in production we had
  - A tool to check which specific version is installed in production and in which server
  - Within the release process we documented on the release notes which changes or tickets were included in a particular version in order to keep track of them
- **Regression Testing** with SOAP UI against the services layer, in order to avoid new changes to break existing functionality

## Support

- **Maintain** the web services already created as well as the new ones, this implies update the wsdl or service contract in order to match the new feature in the service layer
- **Talk and coordinate** with Development teams based in Dallas issues or usage of our web services
- **Talk and coordinate** with QA team based in India the testing phases on each one of the services

## Formal Education

National Technical University

January 2008 – January 2009

City Capital Federal, Buenos Aires. Argentina

Title "Systems Engineering"

Status **Not Finished**

## Certification

Oracle Certified Professional, Java SE 6 Programmer

January 2012



## Linux Foundation Certified SysAdmin **May 2017**



## AWS Certified Cloud Practitioner **September 2019**

**aws**  **CERTIFIED**

**Alejandro Bernal**

has successfully completed the AWS Certification  
 requirements and has achieved their:

**AWS Certified Cloud Practitioner**

**Issue Date**  
 Sep 08, 2019

**Expiration Date**  
 Sep 08, 2022

*Maureen Lonergan*

Maureen Lonergan  
 Director, Training and Certification

Validation Number H55GXTD2G2R4Q9GE  
 Validate at: <http://aws.amazon.com/verification>

## Languages

### English

Advanced Writing and Speaking

### Portuguese

Intermediate Writing and Speaking

### Spanish

Mother Language Advanced Writing and Speaking

## Technical Skills

### Programming and Scripting languages

- Java
- SQL
- Php
- Bash
- Perl
- Python
- JavaScript

### Operative Systems

- Linux

- Ubuntu
- Fedora
- CentOS
- Debian
- Windows
- Mac OS

## **Content Management Systems**

- Drupal

## **Databases**

- MySQL
- PostgreSQL

## **Web Servers and Web Application Servers**

- Apache
- Tomcat
- Nginx

## **Configuration Management Tools**

- Ansible
- Chef

## **Virtualization Technologies**

- Docker

## **Continuous Integration Servers**

- Jenkins
- TeamCity
- CodeShip

## **Containers Orchestrators**

- Kubernetes

## **Cloud Services for Infrastructure as a Service IaaS**



- AWS
  - Route53
  - API Gateway
  - IAM
  - Elastic Kubernetes Service
  - EC2
  - Elastic Load Balancer
  - Auto Scaling Groups
  - ECS
  - ECR
  - SNS - SQS - SES
  - Network Load Balancers
  - Cloud Formations
  - Cloud Front

## Courses

Portuguese	[CEPE : Center] March 2013 - January 2015
JAVA Enterprise Edition 5	[UTN Córdoba] July 2012 - January 2013
UML y UP : Design and Analysis	[Educacion IT] September 2011 - January 2011
Java Hibernate	[Educacion IT] September 2011 - January 2011
Java Web	[Educacion IT] September 2011 - January 2011
Java Programming	[Educacion IT] September 2011 - January 2011
PHP Programmer	[Educacion IT] September 2011 - January 2011
Linux System Administrator	[Educacion IT] September 2011 - January 2011