

Plan de Proyecto: Despliegue de una Aplicación Web Escalable en AWS con IaC

Daniel Alejandro Castro Escobar

Laura Valentina Revelo Villareal

Katerine Valens Orejuela

Docente

Mario Germán Castillo Ramirez

Universidad ICESI

Ingeniería Telemática

2025

Resumen

Este proyecto presenta el diseño, construcción y despliegue de una librería virtual alojada en AWS, implementada mediante una arquitectura escalable, segura y automatizada siguiendo las buenas prácticas de infraestructura en la nube. La solución integra un frontend estático que permite visualizar libros, gestionarlos en un carrito y realizar el flujo de compra, junto con un backend desarrollado en Node.js y Express.js encargado de exponer las rutas API, gestionar usuarios, procesar operaciones del carrito y conectarse a una base de datos MySQL alojada en Amazon RDS.

La infraestructura fue definida utilizando AWS CloudFormation, lo que garantiza reproducibilidad y control de cambios. Se implementó una VPC con subredes públicas y privadas, un bastion host para administración segura, un Application Load Balancer, un Auto Scaling Group para las instancias EC2 que ejecutan la aplicación con PM2, y servicios de soporte como CloudWatch, SNS y CloudTrail. El despliegue automático emplea user data para configurar de forma autónoma el entorno, clonar el repositorio del proyecto, instalar dependencias y levantar la aplicación.

El resultado es un sistema funcional orientado a la venta de libros digitales, que demuestra la aplicación práctica de redes, seguridad, automatización y servicios administrados de AWS en el contexto de un proyecto académico.

Tabla de Contenido

| | |
|----------------------------------|----|
| Introducción..... | 4 |
| Objetivos..... | 5 |
| Objetivo General..... | 5 |
| Objetivos Específicos..... | 5 |
| Descripción del Problema..... | 6 |
| Alcance..... | 7 |
| Marco teórico..... | 8 |
| Cronograma..... | 13 |
| Recursos Técnicos..... | 15 |
| Servicios de AWS..... | 15 |
| Herramientas de desarrollo..... | 17 |
| Aprendizajes y dificultades..... | 19 |
| Conclusiones..... | 21 |
| Bibliografía..... | 22 |

Introducción

La adopción generalizada de servicios en la nube ha transformado la forma en que se desarrollan y despliegan aplicaciones web modernas. Actualmente, la creación de plataformas escalables, seguras y automatizadas es un requerimiento fundamental tanto en el sector empresarial como académico. En el marco de la asignatura Infraestructura III, este proyecto se orienta al diseño e implementación de una librería virtual, una plataforma web que permite la visualización de un catálogo de libros, la gestión de usuarios, operaciones de carrito y el flujo de compra, todo ello soportado por una arquitectura en AWS. El objetivo principal es construir una solución completa aplicando conceptos de redes, seguridad, automatización y computación en la nube mediante servicios como Amazon EC2, Amazon RDS, Elastic Load Balancing, Auto Scaling, VPC y CloudWatch. Además, se emplea AWS CloudFormation para definir la infraestructura como código, garantizando la consistencia del despliegue y facilitando su mantenimiento. La aplicación combina un frontend ligero basado en HTML, CSS y JavaScript con un backend en Node.js que interactúa con una base de datos MySQL. La aplicación implementa un flujo de compra completo, desde el catálogo y carrito hasta el pago en línea, integrando la pasarela de pagos Mercado Pago mediante su API..

Este documento presenta la planeación y justificación de la arquitectura seleccionada, las decisiones técnicas tomadas y el enfoque metodológico utilizado. A través de un análisis estructurado, se exponen los fundamentos que guiaron el diseño del proyecto y se contextualiza su importancia como ejercicio académico orientado al desarrollo de habilidades profesionales en entornos cloud y DevOps.

Objetivos

Objetivo General

Diseñar, implementar y desplegar una librería virtual en la nube de AWS utilizando principios de Infraestructura como Código (IaC), garantizando escalabilidad, seguridad, alta disponibilidad, automatización y monitoreo continuo mediante los servicios administrados de AWS.

Objetivos Específicos

- Implementar un entorno backend escalable utilizando EC2, Auto Scaling Group y Application Load Balancer para garantizar alta disponibilidad.
- Configurar una base de datos en Amazon RDS (MySQL) para la persistencia segura de usuarios, libros y transacciones.
- Desplegar el frontend estático de la librería virtual para ofrecer una interfaz funcional de catálogo, carrito y flujo de compra.
- Implementar monitoreo y notificaciones mediante Amazon CloudWatch y SNS

- Aplicar controles de seguridad usando VPC, subredes públicas y privadas, grupos de seguridad y bastion host para la administración segura.

Descripción del Problema

En el curso de Infraestructura III, se requiere desarrollar una solución web completa que permita poner en práctica los conocimientos adquiridos sobre redes, seguridad, automatización y servicios de computación en la nube. La problemática consiste en diseñar una infraestructura que soporte una aplicación de comercio electrónico, capaz de operar en un entorno real bajo condiciones como aumento de tráfico, fallos de servidores, riesgos de seguridad y requerimientos de monitoreo. Lo que implica integrar múltiples componentes de AWS, como VPC, subredes, bases de datos administradas, balanceadores de carga y grupos de autoescalado, manteniendo coherencia, seguridad y automatización.

Además, surge la necesidad de gestionar el despliegue de forma consistente en diferentes entornos. Por eso, la ausencia de infraestructura como código dificultaría la reproducibilidad, la evolución y el mantenimiento del sistema. Sin un monitoreo adecuado la aplicación no podría detectar fallos ni generar alertas oportunas y sin una arquitectura bien segmentada, la seguridad de los datos de usuarios y transacciones estaría comprometida. Este proyecto responde a dicha problemática desarrollando una solución que integra la aplicación web con una arquitectura escalable en AWS, automatizada mediante CloudFormation y equipada con todos los mecanismos de seguridad, monitoreo y resiliencia necesarios para su operación.

Alcance

El proyecto abarca el despliegue completo de una aplicación web de comercio electrónico en AWS, incluyendo la definición de toda la infraestructura mediante CloudFormation. El alcance contempla la creación de una VPC con subredes públicas y privadas, la implementación de un bastion host para administración segura, el despliegue del backend en instancias EC2 con Auto Scaling y Application Load Balancer, la configuración de una base de datos MySQL en Amazon RDS y la publicación del frontend estático. También incluye la integración de servicios de observabilidad como CloudWatch, SNS y CloudTrail para garantizar monitoreo, notificaciones y auditoría del sistema.

La aplicación logró implementar un flujo básico, pero completo, de comercio electrónico. Permite visualizar un catálogo de productos con su nombre, descripción y precio, gestionar un carrito de compras añadiendo y eliminando ítems mientras se calcula automáticamente el total, y ejecutar un proceso de pago simulado a través de un servicio de pago. Además, se incorporó el registro de usuarios con nombre de usuario y contraseña, dejando sentada la base para manejar sesiones y personalizar futuras funcionalidades.

No incluye:

Desarrollo de nuevas funcionalidades de la aplicación, diseño visual avanzado del frontend y automatización de CI/CD..

Marco Teórico

Infraestructura como Código (IaC) – AWS CloudFormation

La IaC consiste en definir toda la infraestructura mediante archivos declarativos, permitiendo reproducibilidad, control de versiones y automatización del despliegue. AWS CloudFormation permite describir VPCs, subredes, instancias, Roles IAM, Load Balancers, bases de datos y más, en un único archivo YAML. En este proyecto, CloudFormation fue esencial para crear la red completa (VPC, subredes públicas y privadas, tabla de rutas, NAT, IGW), desplegar un Auto Scaling Group con Launch Template, para crear un Application Load Balancer, proveer una base de datos RDS MySQL y para configurar monitoreo con CloudWatch, SNS y CloudTrail. El uso de IaC garantiza que cualquier persona pueda recrear la arquitectura automáticamente.

AWS CloudFormation

Es un servicio que permite modelar y desplegar infraestructura en AWS utilizando plantillas escritas en YAML o JSON. Estas plantillas describen recursos como redes, instancias, bases de datos y sistemas de seguridad, de forma que AWS puede crearlos, actualizarlos o eliminarlos automáticamente. CloudFormation facilita la estandarización, el versionamiento y la automatización de despliegues complejos, eliminando la necesidad de configuraciones manuales y promoviendo la coherencia entre entornos.

Amazon VPC (Virtual Private Cloud)

Es un servicio que permite crear redes virtuales aisladas dentro de AWS. El usuario define rangos de direcciones IP, subredes públicas y privadas, tablas de ruteo, gateways y políticas de seguridad. Las subredes públicas permiten acceso directo desde Internet, mientras que las privadas protegen recursos sensibles como bases de datos. Esta arquitectura otorga un control detallado sobre la red y aumenta significativamente la seguridad.

Internet Gateway y NAT Gateway

Es el componente de VPC que permite la comunicación entre subredes públicas e Internet. Por otro lado, el NAT Gateway permite que instancias ubicadas en subredes privadas puedan acceder a Internet para actualizaciones o descargas, sin exponerse públicamente. Ambos recursos trabajan de manera complementaria para garantizar conectividad segura y eficiente dentro de una arquitectura en la nube.

Amazon EC2 (Elastic Compute Cloud)

Es el servicio de máquinas virtuales de AWS que ofrece capacidad de cómputo escalable bajo demanda. Permite configurar el tipo de instancia, sistema operativo, almacenamiento, red y permisos, adaptándose a múltiples casos de uso, desde aplicaciones simples hasta arquitecturas empresariales. Su flexibilidad y modelo de pago por uso lo convierten en un componente fundamental en el despliegue de aplicaciones web.

Launch Template

Es un recurso que define la configuración base para lanzar instancias EC2 de manera estandarizada. Incluye parámetros como la AMI, el tipo de instancia, los grupos de seguridad, el

user data y el par de claves SSH. Al centralizar estos parámetros, se garantiza que todas las instancias se creen con las mismas características, facilitando la escalabilidad y reduciendo errores de configuración.

Auto Scaling Group (ASG)

Es un servicio que permite crear grupos de instancias EC2 que se ajustan automáticamente a la demanda. Puede incrementar o disminuir el número de instancias basándose en métricas como el uso de CPU o el estado de Health. Gracias a esto, las aplicaciones mantienen disponibilidad y rendimiento aun cuando el tráfico varía, al tiempo que optimizan costos.

Application Load Balancer (ALB)

Distribuye el tráfico HTTP y HTTPS entre múltiples instancias de servidor. Opera en la capa siete del modelo OSI, lo que le permite enrutar solicitudes según rutas, cabeceras o reglas de aplicación. Además, realiza monitoreo continuo mediante health checks, enviando tráfico únicamente a instancias saludables. Este servicio garantiza alta disponibilidad y equilibrio de carga en aplicaciones web.

Security Groups

Son firewalls virtuales que controlan el tráfico entrante y saliente de los recursos dentro de una VPC. Funcionan de manera stateful, permitiendo respuestas automáticas a conexiones autorizadas. La definición adecuada de reglas de seguridad es esencial para proteger aplicaciones frente a accesos no autorizados y minimizar vulnerabilidades.

Bastion Host

Es una instancia situada en una subred pública que funciona como punto de acceso seguro hacia recursos privados. Permite administrar infraestructura interna sin exponer directamente servidores sensibles a Internet. Su uso es una práctica recomendada para garantizar un acceso controlado, auditable y protegido en entornos de producción.

Amazon RDS (Relational Database Service)

Amazon RDS es un servicio administrado para bases de datos relacionales como MySQL, PostgreSQL o SQL Server. Automatiza tareas como copias de seguridad, parches, monitoreo y failover, reduciendo la carga operativa en comparación con administrar bases de datos manualmente. Además, permite ubicar las instancias en subredes privadas para incrementar la seguridad y estabilidad del sistema.

Amazon CloudWatch

Servicio de monitoreo y observabilidad de AWS. Recolecta métricas de rendimiento, logs y eventos de la infraestructura, lo que permite detectar anomalías, crear alarmas y realizar análisis de comportamiento del sistema. Su integración con otros servicios facilita la supervisión continua del entorno.

Amazon SNS (Simple Notification Service)

Amazon SNS es un servicio de mensajería que permite enviar notificaciones por correo electrónico, SMS o integraciones más avanzadas. En entornos de monitoreo, SNS se utiliza para

recibir alertas provenientes de CloudWatch, permitiendo que los administradores actúen rápidamente ante incidentes o fallos críticos.

AWS CloudTrail

AWS CloudTrail registra todas las acciones realizadas sobre los recursos de AWS. Este servicio permite auditar cambios, rastrear operaciones sospechosas, realizar análisis forense y cumplir con requisitos de seguridad. CloudTrail captura información detallada como usuarios, servicios afectados, horarios de ejecución y parámetros utilizados, convirtiéndose en un componente vital para el gobierno y control de la infraestructura.

Frontend y Backend

El frontend corresponde a la capa visual e interactiva de una aplicación, construida con tecnologías como HTML, CSS y JavaScript. Es la parte que ve y utiliza el usuario. El backend, por su parte, está compuesto por la lógica de negocio, las rutas, la autenticación y la interacción con la base de datos. Ambos componentes se comunican entre sí mediante APIs y forman la estructura esencial de aplicaciones web modernas, como las plataformas de comercio electrónico.

Cronograma

| Fase | Actividad Principal | Duración Estimada | Fecha |
|--|---|-------------------|-----------|
| Análisis y diseño inicial | Revisión del enunciado, definición de los requisitos, selección de servicios AWS y diseño preliminar de la arquitectura. | 1 día | Semana 14 |
| Configuración del backend | Desarrollo de la API REST, pruebas locales, estructura del proyecto y conexión inicial a la base de datos. | 2-3 días | Semana 14 |
| Configuración de la base de datos | Creación de la instancia RDS e intentar establecer conexión con backend. | 1-2 días | Semana 14 |
| Implementación IaC con CloudFormation | Construcción del VPC, subredes, rutas, balanceador, ASG, template del Launch Template, RDS, SNS, CloudWatch y CloudTrail. | 3 días | Semana 15 |
| Despliegue y pruebas | Ejecución del stack, corrección de errores, pruebas de instancias y verificación de conectividad. | 2 días | Semana 15 |
| Configuración de servidor y despliegue del backend | Ajuste del UserData, instalación manual en instancia de prueba, despliegue final en ASG y revisión del ALB. | 1 día | Semana 16 |

| | | | |
|------------------------------------|---|--------------------------|--------------|
| Integración del frontend | Ajuste de archivos estáticos, pruebas del flujo de compra y validación de API desde cliente web. | 1 día | Semana 16 |
| Pruebas finales | Validación completa del flujo del ecommerce, estado de instancias, health checks del ALB y rendimiento. | 1 día | Semana 16 |
| Documentación y presentación final | Redacción del informe, elaboración de diagramas, conclusiones y preparación de presentación. | Durante todo el proyecto | Semana 14-16 |

Recursos Técnicos

Servicios de AWS:

- *Amazon VPC (Virtual Private Cloud)*
 - Se empleó para crear la red principal del proyecto, definiendo subredes públicas y privadas, tablas de rutas, un Internet Gateway y un NAT Gateway. Esta configuración permitió aislar componentes críticos como la base de datos y garantizar el acceso seguro a las instancias.
- *Amazon EC2 (Elastic Compute Cloud)*
 - Se utilizaron instancias EC2 para alojar el backend de la aplicación, gestionado mediante un Launch Template y un Auto Scaling Group. El procesamiento se distribuyó a través de un Application Load Balancer, asegurando disponibilidad y tolerancia a fallos.
- *Amazon RDS (Relational Database Service)*
 - Se desplegó una instancia MySQL para la persistencia de datos, alojada en subredes privadas y protegida mediante un Security Group dedicado. RDS facilitó la administración automática de parches, backups y disponibilidad.
- *AWS CloudFormation*

- Se implementó Infraestructura como Código utilizando una plantilla integral que creó todos los recursos: VPC, subredes, EC2, ALB, ASG, RDS, CloudWatch, SNS y políticas de seguridad. Esto garantizó consistencia, repetibilidad y facilidad de mantenimiento.
- *Amazon CloudWatch*
 - Se configuraron métricas, alarmas y monitoreo de recursos críticos, como uso de CPU en instancias del ASG, estado del ALB y salud del Target Group. Además, se habilitó el registro de eventos para diagnóstico.
- *Amazon SNS (Simple Notification Service)*
 - Se empleó para enviar notificaciones automáticas en caso de alarmas activadas por CloudWatch, permitiendo alertas sobre fallos, tráfico anómalo o indisponibilidad del servicio.
- *AWS IAM (Identity and Access Management)*
 - Se definieron roles, políticas y permisos para garantizar un acceso seguro a los recursos. IAM se utilizó para el control de privilegios, acceso a servicios y seguridad en la ejecución del CloudFormation.
- *AWS S3 y CloudTrail*

- Aunque la aplicación no almacena archivos, S3 se utilizó para guardar artefactos del proyecto y como destino de logs generados por AWS CloudTrail. CloudTrail permitió habilitar auditorías sobre todas las operaciones realizadas en la cuenta.

Herramientas de Desarrollo:

Además de los servicios en la nube, se utilizaron diversas herramientas de programación y control de versiones para construir el backend y el frontend del ecommerce:

- *Node.js y Express.js*
 - Tecnologías utilizadas para desarrollar el backend, permitiendo la creación de una API REST eficiente para manejar productos, usuarios, autenticación y operaciones del carrito de compras.
- *HTML, CSS y JavaScript*
 - Lenguajes utilizados para el desarrollo del frontend estático, que brinda la interfaz gráfica al usuario.
- *GitHub*
 - Se utilizó como repositorio principal del proyecto, facilitando control de versiones, despliegue automático mediante el User Data y mantenimiento del código fuente.
- *PM2 y Nginx*

- PM2 se empleó como administrador de procesos para el backend Node.js, mientras que Nginx funcionó como servidor web y proxy reverso, garantizando rendimiento y estabilidad.

Aprendizajes y dificultades

A lo largo del desarrollo del proyecto se evidenció la importancia de comprender a profundidad el funcionamiento de cada componente dentro de la arquitectura de AWS. Uno de los mayores aprendizajes fue el manejo de CloudFormation, ya que su uso implicó afrontar múltiples errores relacionados con dependencias, propiedades inválidas y configuraciones que debían ajustarse para que la infraestructura pudiera desplegarse de manera correcta. Este proceso permitió fortalecer la habilidad de interpretar mensajes de error, analizar la secuencia de creación de recursos y comprender la relevancia de la automatización en entornos reales.

Otra lección significativa surgió con el despliegue del backend mediante User Data. Aunque se diseñó un script completamente funcional, algunos pasos no se ejecutaron debido a problemas con permisos, dependencias del sistema y rutas inexistentes. Esto llevó a realizar una configuración manual dentro de la instancia, lo cual proporcionó una visión más clara del funcionamiento de Node.js, PM2 y Nginx, así como de la necesidad de validar cada componente por separado antes de integrarlo en un script automatizado.

También resultó fundamental comprender la dinámica de conectividad dentro de la VPC. El uso de un Bastion Host, la conexión del backend hacia una base de datos en subred privada y la correcta definición de reglas en los Security Groups fueron aspectos que demostraron lo fácil que es bloquear o interrumpir el funcionamiento de la aplicación con una configuración mínima incorrecta. Esta experiencia reforzó la importancia del diseño cuidadoso de la red y la seguridad a nivel de infraestructura.

Además, el manejo del frontend permitió reconocer la necesidad de una estructura clara de archivos y rutas para garantizar que los recursos estáticos fueran servidos correctamente mediante Nginx. Ajustar la ubicación de imágenes, scripts y hojas de estilo evidenció la importancia de organizar adecuadamente los elementos del proyecto y probarlos constantemente.

Por último, el uso de CloudWatch y SNS permitió comprender la relevancia del monitoreo continuo. Las métricas del ALB, los estados de salud de las instancias y las alarmas configuradas demostraron que la observabilidad es un componente esencial para asegurar que el sistema responda adecuadamente y que los fallos puedan detectarse a tiempo. En conjunto, estas experiencias fortalecieron la comprensión de la arquitectura en la nube y resaltaron la importancia de integrar buenas prácticas desde el inicio del desarrollo.

Conclusiones

El desarrollo del proyecto permitió comprender de manera práctica cómo los servicios de AWS pueden integrarse para construir una aplicación web escalable, segura y altamente disponible. A través de la implementación de Infraestructura como Código con CloudFormation, fue posible desplegar una arquitectura que cumplió con los requisitos de un entorno de producción real, demostrando la eficacia de los modelos declarativos para la automatización y reproducibilidad de recursos.

La creación de la VPC, las subredes segmentadas, la base de datos RDS en ambiente privado y el backend distribuido en un Auto Scaling Group evidenciaron la importancia del diseño de redes y la correcta configuración de reglas de seguridad para garantizar un funcionamiento estable. El uso de herramientas como CloudWatch y SNS reforzó la necesidad del monitoreo activo y las alertas tempranas para asegurar la continuidad del servicio.

Asimismo, la integración del backend en Node.js, Nginx como proxy reverso y PM2 como gestor de procesos permitió validar el despliegue de una aplicación web real dentro de EC2. El frontend, publicado mediante el servidor web, completó la funcionalidad esperada de la librería virtual, demostrando que la arquitectura responde adecuadamente al flujo del usuario. En conjunto, el proyecto permitió consolidar conocimientos clave sobre infraestructura, automatización, seguridad y despliegue de aplicaciones.

Bibliografia

Amazon Web Services (AWS). (2025). Amazon EC2 Documentation. Amazon Web Services, Inc. <https://docs.aws.amazon.com/ec2/>

Amazon Web Services (AWS). (2025). AWS CloudFormation User Guide. Amazon Web Services, Inc. <https://docs.aws.amazon.com/cloudformation/>

Amazon Web Services. (2025). Amazon EC2 Auto Scaling User Guide. Amazon Web Services, Inc. <https://docs.aws.amazon.com/autoscaling/>