

Nombre: Ángel Gabriel Martínez Moreno Ana Daniela Torrero Jasso Pablo Torres Doria Luis Mario Quintanilla Álvarez	Matrícula: 3004203 2953691 3003910 2886747
Nombre del curso: Proyecto integrador de la metodología DevOps	Profesor: Julio Antonio García Moreno
Módulo: 1	Fecha: 1 de enero de 2025
Bibliografía: <ol style="list-style-type: none"> 1. Sahin, K. (2025, 20 enero). Python Web scraping: full tutorial with examples (2025). <i>ScrapingBee</i>. https://www.scrapingbee.com/blog/web-scraping-101-with-python/ 2. Python For ETL: How to Build ETL Pipelines With Examples. (2024, 25 junio). Airbyte. https://airbyte.com/data-engineering-resources/python-etl 3. Lumigo. (2024, 21 agosto). How to Deploy AWS Lambda with Terraform. https://lumigo.io/aws-lambda-deployment/aws-lambda-terraform/ 4. <i>El comparador de autos nuevos más completo en México</i> my-car.mx. (s. f.). My Car México. https://my-car.mx/comparador 5. Autocosmos. (s. f.). Comparador de autos. Autocosmos.com. https://www.autocosmos.com.mx/catalogo/comparar 6. Calibraint. (2024, 13 junio). Benefits of DevOps In Web Development - A Deep Dive. Calibraint. https://www.calibraint.com/blog/benefits-of-devops-in-web-development 	

Fase I: Introspección.

Nombre del proyecto: CarWizard

Descripción: Página web de consulta de datos financieros, técnicos y de propiedades de los autos de las marcas más comunes a nivel mundial, donde podrás filtrar por marca, precio, financiamiento o datos específicos como consumo.

Proceso esperado:

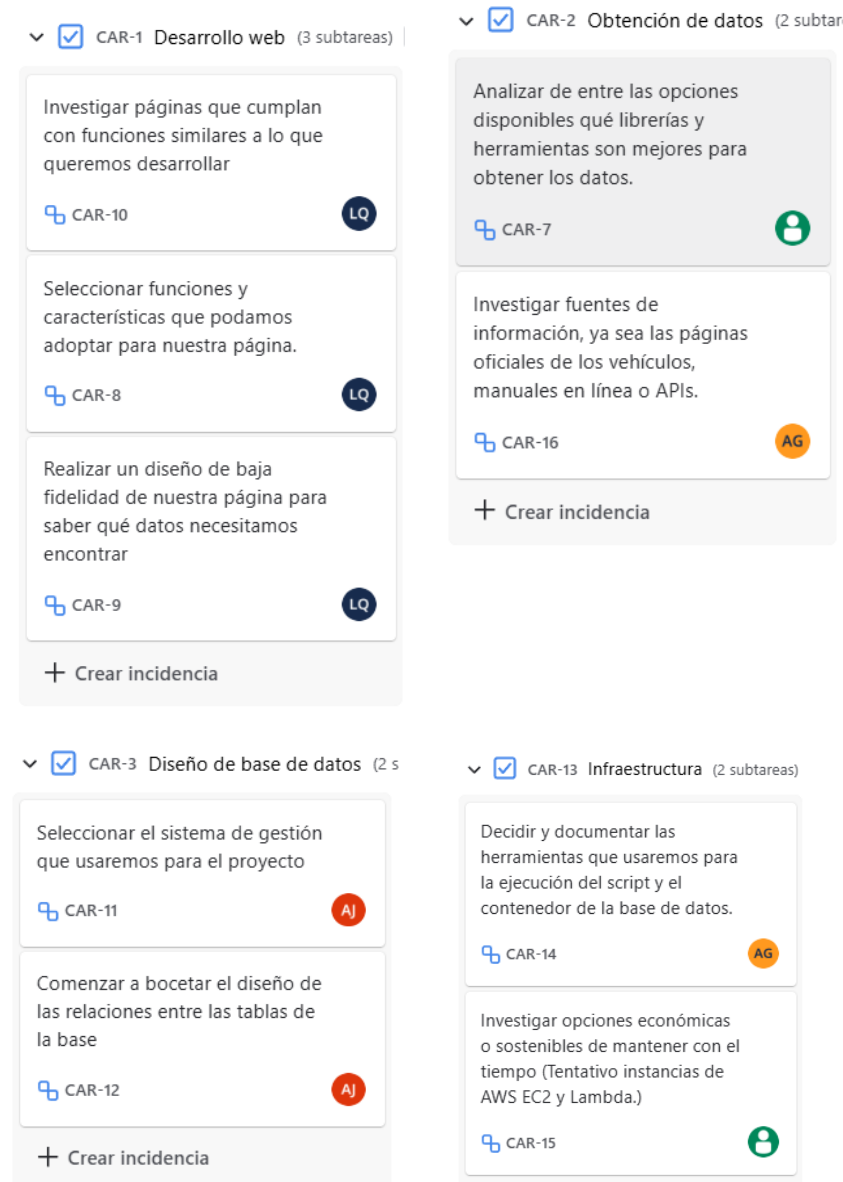
- **Obtención de datos:** Programar scripts de Python que encuentren los datos de las páginas oficiales de las marcas para obtener las especificaciones de sus modelos, por medio de consultas a una API o Web Scrapping.^{[1][2]}
- **Diseño de la base de datos:** Usar MySQL para diseñar e implementar una base de datos relacional que guarde los datos de los diferentes tipos de vehículos.
- Implementar una instancia de EC2 que contenga la base de datos.
- Usar AWS lambda para ejecutar el script de Python.
- Crear un script de Terraform que levante la infraestructura en la nube.^[2]
- **Página web:** Diseñar y montar una página web que muestre y haga interactiva la búsqueda y comparación de los coches.
- **IaC con Terraform:** Crear un script para levantar los servicios en la nube necesarios.

Páginas web que hacen cosas similares existen, pero nuestro principal diferenciador es que no queremos hacer un análisis superficial que meramente compare las características y datos crudos de los vehículos, si no aportar gráficas y visuales que permitan al usuario tener

datos interpretados que le ayuden a tomar la mejor decisión a la hora de adquirir una automóvil.[4][5]

1. Organice todas las tareas planificadas en un tablero Kanban (Trello, Jira).

La idea que tenemos de proyecto incluye múltiples fases y aspectos distintos, las tareas se enfocan en el diseño, desarrollo, pruebas y ejecución de cada componente individual de la página. Para la primera fase este es el enlace al tablero:



✓ ☒ CAR-1 Desarrollo web (3 subtasks)

Investigar páginas que cumplan con funciones similares a lo que queremos desarrollar

CAR-10 LQ

Seleccionar funciones y características que podamos adoptar para nuestra página.

CAR-8 LQ

Realizar un diseño de baja fidelidad de nuestra página para saber qué datos necesitamos encontrar

CAR-9 LQ

+ Crear incidencia

✓ ☒ CAR-2 Obtención de datos (2 subtasks)

Analizar de entre las opciones disponibles qué librerías y herramientas son mejores para obtener los datos.

CAR-7

Investigar fuentes de información, ya sea las páginas oficiales de los vehículos, manuales en línea o APIs.

CAR-16 AG

+ Crear incidencia

✓ ☒ CAR-3 Diseño de base de datos (2 subtasks)

Seleccionar el sistema de gestión que usaremos para el proyecto

CAR-11 AJ

Comenzar a bocetar el diseño de las relaciones entre las tablas de la base

CAR-12 AJ

+ Crear incidencia

✓ ☒ CAR-13 Infraestructura (2 subtasks)

Decidir y documentar las herramientas que usaremos para la ejecución del script y el contenedor de la base de datos.

CAR-14 AG

Investigar opciones económicas o sostenibles de mantener con el tiempo (Tentativo instancias de AWS EC2 y Lambda.)

CAR-15

Las actividades por realizar se dividen entre los 4 desarrolladores de la aplicación web, cada uno centrado en el desarrollo del objetivo específico de su área, actualmente se contemplan 4 áreas de desarrollo:

- Desarrollo web.
- Desarrollo de base de datos.

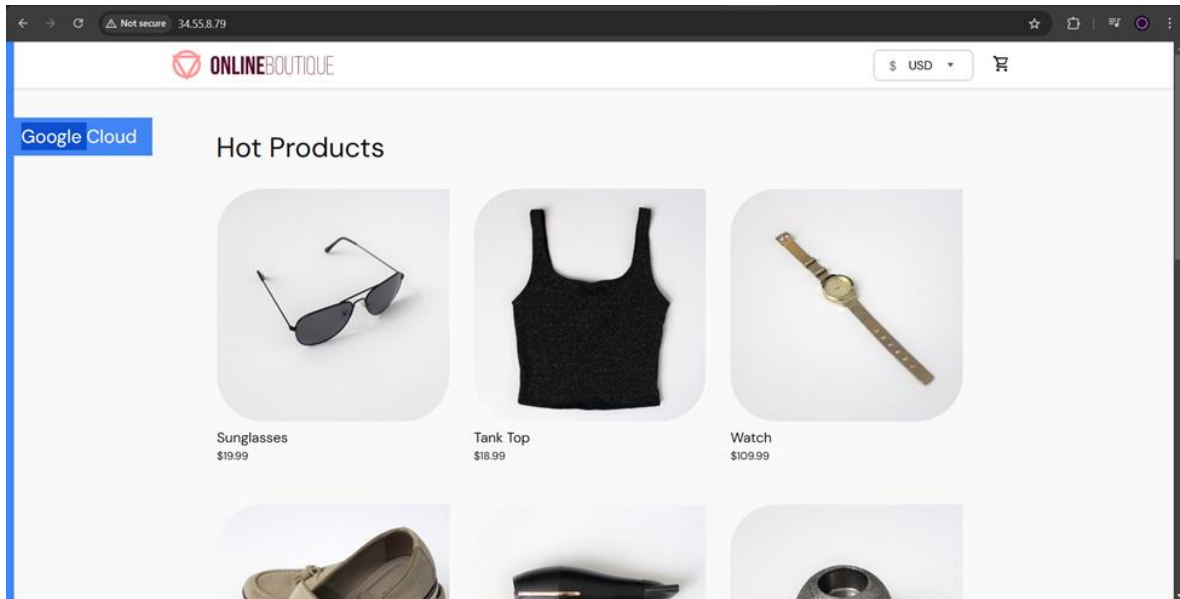
- Desarrollo de infraestructura en la nube.
- Desarrollo de scripts y captación de datos.

2. Use la aplicación *open source* “Online Boutique” desarrollada por Google.

```
C:\Users\luism\OneDrive\Documents\online-boutique\microservices-demo>kubectl apply -f ./release/kubernetes-manifests.yaml
deployment.apps/emailservice created
service/emailservice created
serviceaccount/emailservice created
deployment.apps/checkoutservice created
service/checkoutservice created
serviceaccount/checkoutservice created
deployment.apps/recommendationservice created
service/recommendationservice created
serviceaccount/recommendationservice created
deployment.apps/frontend created
service/frontend created
service/frontend-external created
serviceaccount/frontend created
deployment.apps/paymentservice created
service/paymentservice created
serviceaccount/paymentservice created
deployment.apps/productcatalogservice created
service/productcatalogservice created
serviceaccount/productcatalogservice created
deployment.apps/cartservice created
service/cartservice created
serviceaccount/cartservice created
deployment.apps/redis-cart created
service/redis-cart created
Warning: autopilot-default-resources-mutator:Autopilot updated Deployment default/loadgenerator: defaulted unspecified 'cpu' resource for containers [frontend-check] (see http://g.co/gke/autopilot-defaults).
deployment.apps/loadgenerator created
serviceaccount/loadgenerator created
deployment.apps/currencyservice created
service/currencyservice created
serviceaccount/currencyservice created
deployment.apps/shippingservice created
service/shippingservice created
serviceaccount/shippingservice created
deployment.apps/adservice created
service/adservice created
serviceaccount/adservice created
```

```
C:\Users\luism\OneDrive\Documents\online-boutique\microservices-demo>kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
adservice-5d5bbf6857-nnqfc          1/1     Running   1 (6m18s ago)    8m32s
cartservice-55b7f75449-nwgt2        1/1     Running   1 (6m10s ago)    8m36s
checkoutservice-86d7fdbd84-wdnm9    1/1     Running   0              8m39s
currencyservice-547f8cdb94-6xdzs    1/1     Running   1 (6m20s ago)    8m34s
emailservice-85cddc866d-ntws2       1/1     Running   4 (5m41s ago)    8m39s
frontend-f76df9b89-kdrtx            1/1     Running   0              8m38s
loadgenerator-b4c9ff8bd-p2fh2       1/1     Running   0              8m35s
paymentservice-77984ccb4d-75wvg     1/1     Running   2              8m37s
productcatalogservice-745f7776fb-fk25h 1/1     Running   0              8m36s
recommendationservice-8499974949-fbl2v 1/1     Running   4 (5m29s ago)    8m39s
redis-cart-68f9695b48-2nh97         1/1     Running   0              8m35s
shippingservice-58d578789b-tjlfw     1/1     Running   0              8m33s
```

```
luism@lquint MINGW64 ~/OneDrive/Documents/online-boutique/microservices-demo ((v0.10.2))
$ kubectl get service frontend-external | awk '{print $4}'
EXTERNAL-IP
34.55.8.79
```



3. Organice una propuesta comercial donde explique por qué esta aplicación debe tener la metodología DevOps para ser mejor.

CarWizard es una plataforma para la consulta y comparación de datos financieros, técnicos y de propiedades de automóviles. Para mejorar su rendimiento, escalabilidad y seguridad, la adopción de DevOps es clave.

Beneficios de DevOps en CarWizard^[6]

1. Automatización y Despliegue Continuo (CI/CD)

- Implementación de CI/CD con GitHub Actions, permitiendo despliegues automáticos y sin interrupciones.
- Reducción de errores humanos con Infraestructura como Código (IaC) usando Terraform para aprovisionamiento automatizado.

2. Escalabilidad y Alta Disponibilidad

- Uso de AWS Lambda para ejecutar dinámicamente los scripts de recolección de datos sin necesidad de servidores fijos.
- Implementación de AWS EC2 para alojar la base de datos MySQL y garantizar estabilidad en la plataforma.
- Uso de autoescalado en EC2 para manejar variaciones en el tráfico y optimizar costos.

3. Monitoreo y Observabilidad

- Implementación de AWS CloudWatch para monitoreo en tiempo real de los servicios y detección temprana de fallos.

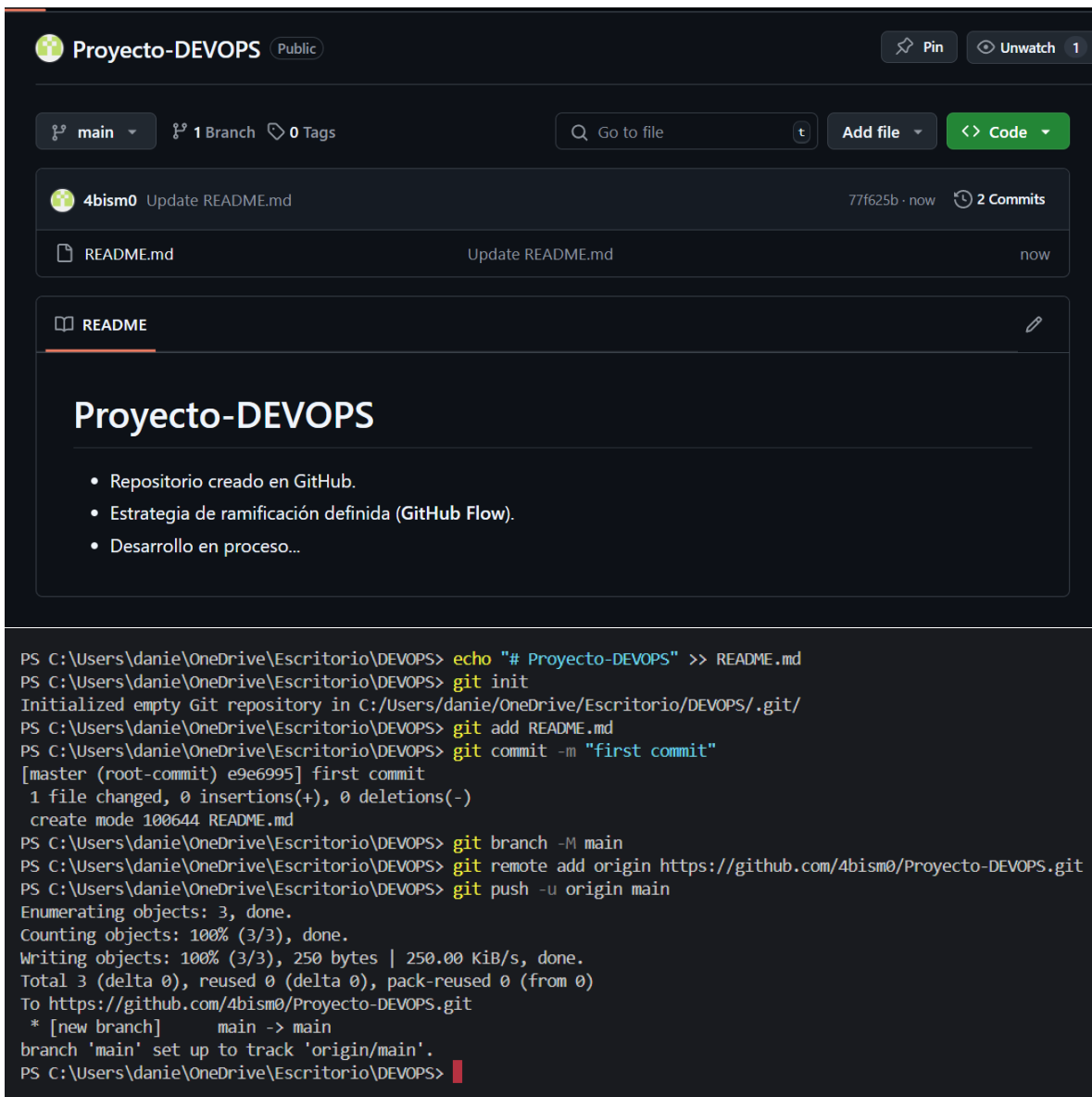
- Centralización de logs y métricas para mejorar la capacidad de respuesta ante incidentes.

4. Seguridad y Gestión de Configuración

- Aplicación de escaneo de vulnerabilidades en infraestructura con AWS Inspector.
- Gestión segura de accesos y credenciales con AWS Secrets Manager.

4. Cree el repositorio en Git donde trabajará y elija una estrategia de ramificación para aplicar en el proyecto.

Ya hemos creado el repositorio de GitHub con nuestro proyecto, el enlace es el siguiente:
<https://github.com/4bism0/Proyecto-DEVOPS.git>



The image shows a GitHub repository page for 'Proyecto-DEVOPS' and a terminal window with the following content:

GitHub Repository View:

- Repository: Proyecto-DEVOPS (Public)
- Branch: main (1 Branch)
- Tags: 0 Tags
- Search: Go to file
- Buttons: Add file, Code
- Commit: 4bism0 Update README.md (77f625b · now) 2 Commits
- File: README.md (Update README.md · now)
- README content:
 - Proyecto-DEVOPS
 - Repositorio creado en GitHub.
 - Estrategia de ramificación definida (GitHub Flow).
 - Desarrollo en proceso...

Terminal Output:

```
PS C:\Users\danie\OneDrive\Escritorio\DEVOPS> echo "# Proyecto-DEVOPS" >> README.md
PS C:\Users\danie\OneDrive\Escritorio\DEVOPS> git init
Initialized empty Git repository in C:/Users/danie/OneDrive/Escritorio/DEVOPS/.git/
PS C:\Users\danie\OneDrive\Escritorio\DEVOPS> git add README.md
PS C:\Users\danie\OneDrive\Escritorio\DEVOPS> git commit -m "first commit"
[master (root-commit) e9e6995] first commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 README.md
PS C:\Users\danie\OneDrive\Escritorio\DEVOPS> git branch -M main
PS C:\Users\danie\OneDrive\Escritorio\DEVOPS> git remote add origin https://github.com/4bism0/Proyecto-DEVOPS.git
PS C:\Users\danie\OneDrive\Escritorio\DEVOPS> git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 250 bytes | 250.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/4bism0/Proyecto-DEVOPS.git
* [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
PS C:\Users\danie\OneDrive\Escritorio\DEVOPS>
```

Para revisar el código fuente, así como más información acerca del proyecto, puede visitar el repositorio oficial de GitHub: <https://github.com/GoogleCloudPlatform/microservices-demo>