**Manual DevOps**

**Nombre del**

**proyecto:**

**CampusDocs**

**INTEGRANTES:**

**ESTRADA FLORES ALAN ALBERTO**

**VILLANUEVA RODRIGUEZ ALAN ISAY**

**ROMERO HERNÁNDEZ MIGUEL ÁNGEL**

**PIAGA REMENTERIA ROGELIO**

**PROFESOR:**

**Omar Téllez Barrientos**

**ASIGNATURA**

**:**

**Gestión del Proceso de Desarrollo de Software**



# Índice de Contenido

Contenido

[Índice de Contenido 2](#_Toc169713853)

[Introducción del proyecto 3](#_Toc169713854)

[Introducción a DevOps 4](#_Toc169713855)

[Definición 4](#_Toc169713856)

[Beneficios 4](#_Toc169713857)

[Principios 4](#_Toc169713858)

[Etapas del DevOps 5](#_Toc169713859)

[Planificación 5](#_Toc169713860)

[Desarrollo 6](#_Toc169713861)

[Integración continua 7](#_Toc169713862)

[Entrega continua 8](#_Toc169713863)

[Despliegue 9](#_Toc169713864)

[Operación 10](#_Toc169713865)

[Monitoreo y Feedback 11](#_Toc169713866)

[Aplicación de DevOps al proyecto 12](#_Toc169713867)

[Estructura de la Aplicación: 12](#_Toc169713868)

[Instalación de Laravel / Laravel Breeze: 12](#_Toc169713869)

[Arquitectura de la Aplicación 12](#_Toc169713870)

[Migraciones para la base de datos 13](#_Toc169713871)

[Arquitectura de la aplicación 17](#_Toc169713872)

[API Laravel / Modelo MVC 17](#_Toc169713873)

[Sistema Login 22](#_Toc169713874)

[Conclusiones 26](#_Toc169713875)

# Introducción del proyecto

Un programa dirigido a los docentes y estudiantes en el que se brinda un apoyo ágil y eficiente a la gestión de reportes técnicos de estadías y comunicación entre el estudiante, asesor industrial y académico.

El proyecto se centra en la creación e implementación de un sistema integral para la gestión eficiente de los reportes técnicos de estadías, con el propósito de facilitar la revisión y posteriormente las validaciones por parte de la universidad. Este sistema abordara desde la recopilación de información hasta el seguimiento post-validación.

El proyecto abarca varias áreas clave:

* Registro de Información Estudiantil
* Detalles de la Estadía
* Contenido del Reporte Técnico
* Normativas y Estándares
* Proceso de Validación
* Sistema de Seguimiento

El propósito principal del proyecto es mejorar la calidad y agilizar el proceso de revisión y validación de los reportes técnicos de estadías, asegurando el cumplimiento de los requisitos académicos.

# Introducción a DevOps

## Definición

DevOps es una práctica de ingeniería de software que tiene como objetivo unificar el desarrollo de software (Dev) y las operaciones de IT (Ops). Su objetivo es acortar el ciclo de vida del desarrollo del sistema y proporcionar una entrega continua con alta calidad de software.

## Beneficios

Los principales beneficios de implementar DevOps incluyen:

* Mayor frecuencia de despliegue
* Tiempos de lanzamiento más rápidos
* Mejor calidad de producto
* Mayor capacidad para resolver problemas rápidamente
* Mejor comunicación y colaboración entre equipos

## Principios

Los principios fundamentales de DevOps incluyen:

* Integración Continua
* Entrega Continua
* Infraestructura como Código
* Monitorización y Logging Continuos

# Etapas del DevOps

## Planificación

Esta etapa implica definir los requisitos, crear historias de usuario y planificar las tareas y recursos necesarios. Se utilizan herramientas de gestión de proyectos y colaboración como Github.

Objetivos:

Definir requisitos

Crear historias de usuario

Planificar tareas y recursos necesarios

Herramientas:

Jira, Trello, GitHub Projects.

Pasos:

Recopilación de Requisitos: Identificar y documentar las necesidades del proyecto.

Creación de Historias de Usuario: Dividir los requisitos en historias de usuario manejables.

Planificación de Sprints: Organizar el trabajo en sprints, asignando tareas específicas a cada miembro del equipo.

## Desarrollo

Se llevará a cabo la codificación del software. Se utilizan entornos de desarrollo integrados (IDE: Laravel), como base de datos: MySQL, sistemas de control de versiones (como Git) y técnicas de desarrollo ágil.

Objetivos:

Codificación del software

Gestión de versiones

Pruebas unitarias

Herramientas:

Visual Studio Code, IntelliJ IDEA, Git, GitHub

Pasos:

Desarrollo del Código: Escribir código siguiendo las mejores prácticas.

Gestión de Versiones: Usar Git para el control de versiones y la colaboración.

Pruebas Unitarias: Implementar pruebas para asegurar la calidad del código.

## Integración continua

El código desarrollado se integra regularmente en un repositorio compartido. Se ejecutan pruebas automatizadas para asegurar que los cambios no rompan la funcionalidad existente. Herramientas como Jenkins, Travis CI y CircleCI son comunes en esta etapa.

Objetivos:

Integrar cambios de código frecuentemente

Ejecutar pruebas automáticas

Herramientas:

Jenkins, Travis CI, CircleCI

Pasos:

Configuración de CI: Configurar un servidor de CI para automatizar las pruebas y la integración.

Automatización de Pruebas: Escribir y configurar pruebas automatizadas que se ejecuten con cada commit.

Integración Frecuente: Integrar código en el repositorio compartido varias veces al día.

## Entrega continua

Después de la integración, el código se despliega en un entorno de prueba o preproducción. Este proceso asegura que el software puede ser lanzado a producción en cualquier momento. Las herramientas de orquestación y automatización de despliegues, como Ansible, Chef y Puppet.

Objetivos:

Desplegar código en entornos de prueba y preproducción

Herramientas:

Docker, Kubernetes, Ansible, Puppet

Pasos:

Despliegue Automatizado: Configurar scripts para el despliegue automatizado en entornos de prueba.

Pruebas de Integración: Ejecutar pruebas de integración para asegurar que los cambios funcionan en conjunto.

Aprobación para Producción: Preparar el código para su despliegue en producción.

## Despliegue

La aplicación se lanza a un entorno de producción. Este proceso se realiza de manera automatizada y controlada para minimizar los riesgos. Docker y otras herramientas de contenedorización y orquestación son frecuentemente utilizadas.

Objetivos:

Desplegar la aplicación en producción de manera segura y eficiente

Herramientas:

Docker, Kubernetes, Terraform

Pasos:

Automatización del Despliegue: Usar herramientas como Docker y Kubernetes para automatizar el despliegue.

Despliegue Controlado: Implementar estrategias como despliegues canarios o blue-green deployment para minimizar riesgos.

Verificación Post-Despliegue: Verificar que el despliegue fue exitoso y que la aplicación funciona correctamente en producción.

## Operación

En esta etapa, se monitoriza la aplicación en producción para asegurar su funcionamiento correcto y eficiente. Se utilizan herramientas de monitoreo y logging com Grafana.

Objetivos:

Monitorizar el rendimiento y funcionamiento de la aplicación en producción

Herramientas:

Grafana, Prometheus, ELK Stack (Elasticsearch, Logstash, Kibana)

Pasos:

Configuración de Monitoreo: Configurar herramientas de monitoreo para supervisar el rendimiento y la disponibilidad de la aplicación.

Logging: Implementar soluciones de logging para registrar eventos y transacciones.

Alertas y Notificaciones: Configurar alertas para notificar al equipo de cualquier problema o anomalía.

## Monitoreo y Feedback

Se recopilan datos sobre el rendimiento y el uso de la aplicación, así como feedback de los usuarios finales. Esta información es crucial para identificar problemas, realizar mejoras y planificar futuras versiones.

Objetivos:

Recopilar y analizar datos sobre el uso y rendimiento de la aplicación

Herramientas:

Google Analytics, New Relic, Sentry

Pasos:

Recopilación de Datos: Obtener datos de rendimiento y uso mediante herramientas de monitoreo y análisis.

Análisis de Feedback: Analizar los datos y la retroalimentación de los usuarios para identificar áreas de mejora.

Mejoras Continuas: Planificar y priorizar mejoras basadas en el feedback y el análisis de datos.

# Aplicación de DevOps al proyecto

## Estructura de la Aplicación:

Framework: Laravel

Base de Datos: MySQL

## Instalación de Laravel / Laravel Open Admin:

Código:

composer require laravel/OpenAdmin –dev

Php artisan admin:install

Npm install && npm run dev

Php artisan migrate

## Arquitectura de la Aplicación

Modelo MVC (Modelo-Vista-Controlador):

Modelos: Crear modelos para manejar datos (por ejemplo, Comentario, Roles, Usuarios, Reportes).

Controladores: Crear controladores para manejar la lógica de la aplicación.

Vistas: Crear vistas para la interfaz de usuario, como dashboards para cada rol.

Sistema de Login

Implementación de Autenticación: Configurar el sistema de login basado en roles utilizando Laravel Breeze.

## Migraciones para la base de datos

A continuación, se muestran los códigos necesarios para las migraciones necesarias para la base de datos “campusdocs” en PostgreSQL, para después, utilizar la misma y proceder con las funcionalidades, una vez generada de forma correcta la base de datos:

**Migración para la tabla “roles”**

**Captura de pantalla de computadora

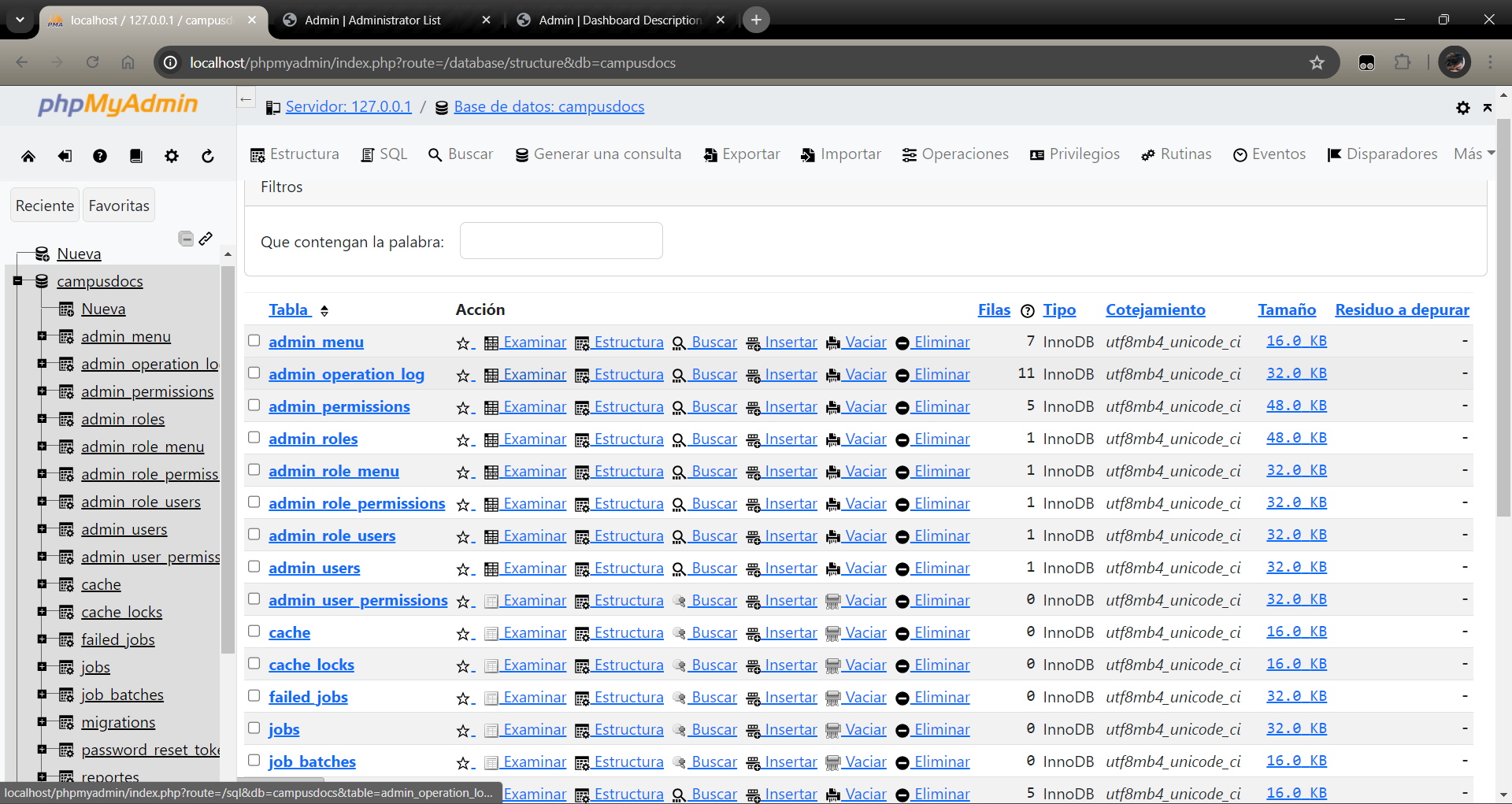
Descripción generada automáticamente**

**Migración para la tabla “reportes”**

**Captura de pantalla de computadora

Descripción generada automáticamente**

Una vez generado las migraciones, se procede a utilizar el comando “php artisan migrate” para realizar la migración a la base de datos, siempre y cuando se hayan definido las credenciales correctas y hacia que base de datos se aplica.



# Arquitectura de la aplicación

El proyecto se rige bajo el modelo MVC (Modelo – Vista – Controlador), en los cuales se integran todas las funcionalidades para la aplicación. Después de haber realizado las migraciones necesarias para las tablas especificas de la base de datos, se procede con la codificación en diferentes secciones:

## API Laravel / Modelo MVC

1. Creación de Modelos (Comentario, Roles, Usuarios, Reportes) bajo el comando “php artisan make:model -mcr”:

Captura de pantalla de computadora

Descripción generada automáticamente

1. Creación de Controladores bajo el comando “php artisan make:controller”

Captura de pantalla de computadora

Descripción generada automáticamente

1. Creación de Vistas (Dashboards) de acuerdo con cada rol y su funcionalidad:

Captura de pantalla de computadora

Descripción generada automáticamente

Captura de pantalla de computadora

Descripción generada automáticamente

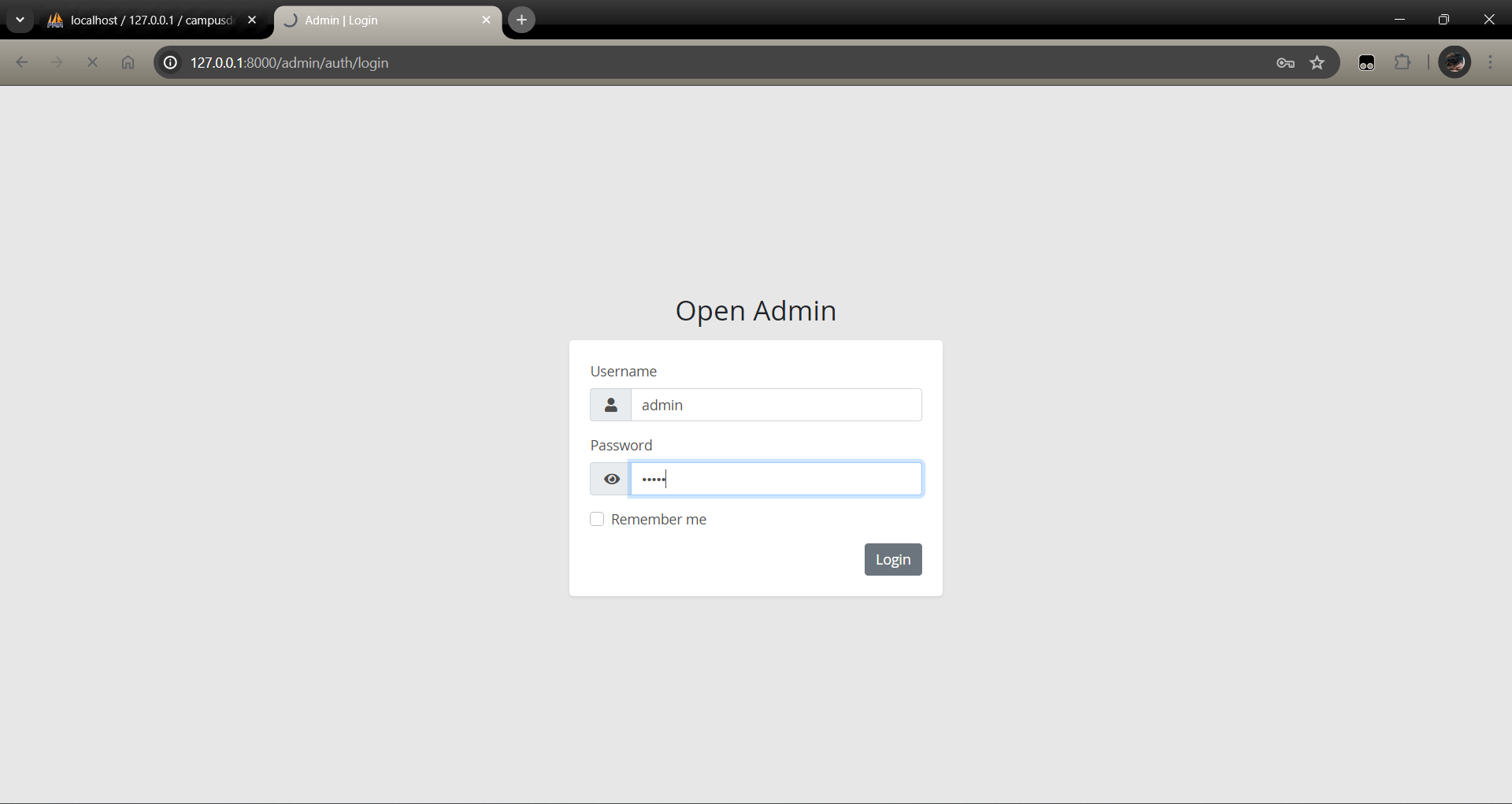
1. Creación del Middleware para distinguir cada rol y redireccionar a su dashboard respectivo:

Texto

Descripción generada automáticamente

## Sistema Login

Se realizan pruebas para el correcto funcionamiento de la API y el sistema Login:



Usuarios con protección en sus contraseñas.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

## Conclusiones

La implementación de DevOps proporciona un marco robusto para mejorar la eficiencia, calidad y velocidad del desarrollo de aplicaciones web. Siguiendo las etapas y prácticas descritas, los equipos de desarrollo y operaciones pueden colaborar más efectivamente, logrando un ciclo de vida de software más ágil y productivo.