“RESULTADO DE APRENDIZAJE”

UNIVERSIDAD TECNOLÓGICA DE SAN LUIS RIO COLORADO

JAVIER VELAZQUEZ

DESARROLLO WEB INTEGRAL

FIDEL OROZCO CASTRO

IDGS9-1

2 DE OCTUBRE DEL 2024

Documento: Plan de Desarrollo WEB Basado en Metodología Ágil

1. Plan del Proceso de Desarrollo WEB (Metodología Ágil Seleccionada: Scrum)

Objetivo: Desarrollar un sitio web de comercio para un comercio tipo colectivo haciendo uso de la metodología Scrum asegurando una entrega incremental y satisfactoria hacia las necesidades del cliente.

Duración del Proyecto: 16 semanas divididas en 8 sprints de 2 semanas cada uno.

Fases del Proyecto:

1. Inicio:
    - Reunión de inicio con el equipo y stakeholders.
    - Análisis de requerimientos (requisitos y características).
    - Estimación del esfuerzo y planificación del primer sprint.
2. Desarrollo:
    - Sprint Planning: Planificación de tareas del sprint, correspondientes al análisis de requerimientos.
    - Dayli Stand-ups: Reuniones diarias de aproximadamente 15 minutos para revisar avances y resolver problemas.
    - Sprint Review: Revision del trabajo finalizado con los stakeholders al final de cada sprint.
    - Sprint Retrospective: Evaluacion interna del equipo para mejorar el proceso en el siguiente ciclo.
3. Entrega Final:
    - Integración y testeo final.
    - Revisión de aceptación por parte del cliente.
    - Lanzamiento del producto.

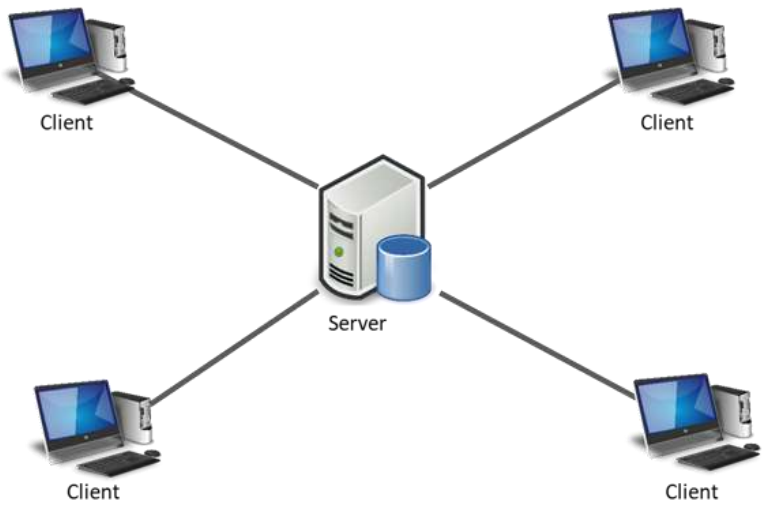
2. Justificación de la Arquitectura

Arquitectura Seleccionada: Cliente – Servidor

Justificacion:

- Escabilidad: Permite actualizar y ampliar ambos entornos por separado tanto en la parte del cliente como la parte del servidor, centralizando las necesidades de escalabilidad en el servidor donde tenemos más capacidad de intervención.
- Mantenimiento sencillo: Se maneja con mantenimientos individuales de las capas lo que mejora las posibilidades de actuación y las aísla en caso de problemas.
- Administracion de datos sencilla y segura: Los datos se encuentran en entornos de servidores completamente controlados.
- Flexible: Debido a su entorno de servidor puede ajustarse al trabajo con varios tipos de clientes.

3. Diagrama de la Arquitectura



4. Propuesta de los Patrones de Diseño

Patrones Propuestos:

1. Singleton: Para mejorar la conexión de la base de datos, asegurando que solo haya una instancia activa.
2. Factory: Para la creación de objetos no tan sencillos, facilitando la extensión y el mantenimiento.
3. Observer: Para las notificaciones, como lo son actualizaciones en el carrito al ser una pagina enfocada a la venta.

Justificación:

- Mantenibilidad: Los patrones seleccionados facilitan la organización del código y su extensibilidad a lo largo del tiempo.
- Reusabilidad: Permiten reutilizar código en diferentes partes de la aplicación.

5. Justificación de los Frameworks de Desarrollo WEB a Utilizar

Frontend: Vue.js

Justificación: Vue.js es un framework muy popular y utilizado en el desarrollo Frontend con características eficientes en la creación de interfaces y componentes que pueden ser reutilizados además de su facilidad en integración a otros proyectos.

Backend: Laravel

Justificación: Esta basado en lenguaje PHP lo cual lo hace sencillo en implementación de proyectos robustos. Ofrece distintas características esenciales para el proyecto como el manejo de sesiones, autenticación etc.

Base de Datos: MySQL

Justificación: El framework MySQL es uno maduro y ampliamente soportada. Basado en lenguaje SQL con sus respectivas relaciones que son indispensables dentro del proyecto.

API Gateway: Nginx

Justificación: Nginx se utiliza para manejar la carga de tráfico HTTP y como un balanceador de carga eficiente para los microservicios.

6. Esquema de Pruebas

Tipos de Pruebas:

1. Pruebas Unitarias: Asegurar que cada componente del sistema funcione correctamente de forma aislada. (PHPUnit para Laravel y Jest para Vue.js)
2. Pruebas de Integración: Verificar que los diferentes módulos y microservicios interactúen correctamente.
3. Pruebas de Sistema: Validar que el sistema completo cumple con los requisitos funcionales.
4. Pruebas de Aceptación: Realizadas por los usuarios finales para garantizar que el sistema cumple con sus expectativas.
5. Pruebas de Rendimiento: Evaluar la capacidad del sistema bajo carga utilizando herramientas como Apache JMeter.

Planificación de Pruebas:

- Las pruebas unitarias y de integración se ejecutarán en cada sprint.
- Las pruebas de sistema y de aceptación se realizarán al final de cada sprint y en la fase de entrega final.
- Las pruebas de rendimiento se realizarán antes de la liberación final.

Reporte: Configuracion del Entorno

Lista de herramientas Utilizadas:

- IDE: Visual Studio Code
- Versionamiento: Git y GitHub
- Framework Frontend: Vue.js
- Framework Backend: Laravel
- Base de Datos: MySQL
- Servidor Web: Nginx
- Sistema Operativo: Ubuntu 20.04 LTS (Servidores)
- Herramientas de prueba: PHPUnit, Jest
- Docker: Para la gestión de dependencias PHP
- Node.js & npm: Para la gestión de dependencias Frontend

Parametros de Confiuracion

Configuración del Entorno Local:

- PHP: Versión 8.1
- Node.js: Versión 16.x
- MySQL: Versión 8.0
- Servidor Web Local: Nginx 1.20.x

Configuración de Laravel:

- Archivo (.env) con las credenciales de base de datos.
- Configuración del servidor local apuntando al directorio public de Laravel.
- Configuración de caché y almacenamiento para desarrollo.

Configuración del Entorno de Producción:

- Servidor: DigitalOcean o AWS EC2
- Seguridad: Configuración de firewall, SSL/TLS a través de Let’s Encrypt.
- CI/CD: GitHub Actions configurado para desplegar automáticamente en producción.
- Escalabilidad: Configuración de balanceo de carga con Nginx y Docker Swarm/Kubernetes.