# POPCODE

## Método propuesto

## CASCADA

La metodología cascada cuenta con 5 faces que son fundamentales para el desarrollo del sistema web dividiendo el trabajo en fases secuenciales, en la que cada fase debe de completarse en su totalidad antes de que se pueda avanzar a lo siguiente.

## Fase 1

## Análisis

* Herramienta de recolección de información.
* Generar la propuesta.
* Definir los requisitos funcionales del sistema.
* Analizar el lector de códigos como protocolos soportados, configuración, documentación técnica.
* Establecer el flujo de trabajo en la línea de producción como el escaneo, Validación, Registro.
* Determinar roles y permisos (administrador, operador, supervisor).
* Analizar la infraestructura de red y servidor.
* Identificar riesgos y limitaciones.

## Fase 2

## Diseño

1. Crear diagramas de caso de uso
2. Crear diagramas secuenciales
3. Arquitectura cliente-servidor
4. Backend en PHP (Diagrama E-R)
5. Diagrama relacional
6. Frontend de la ventana de registro de usuarios
7. Frontend del Login
8. Frontend para el panel del administrador
9. Frontend para el panel del supervisor
10. Frontend para el panel del operador
11. Frontend para el inventario del administrador TI
12. Frontend para el inventario del administrador Calidad
13. Frontend para el inventario del administrador Mantenimiento
14. Frontend para el submenú de TI
15. Frontend para el submenú de vidrio, plástico, quebradizo del administrador Calidad
16. Frontend para el submenú de herramientas del administrador Calidad
17. Frontend para el submenú de manejo integral de plagas del administrador Calidad
18. Frontend para el submenú de utensilios detectables del administrador Calidad
19. Frontend para el submenú de cuchillos y afilados del administrador Calidad
20. Frontend para el submenú de refacciones del administrador de Mantenimiento
21. Frontend para visualizar los usuarios registrados de administración de TI
22. Frontend para visualizar los usuarios registrados de administración de Calidad
23. Frontend para visualizar los usuarios registrados de administración de Mantenimiento
24. Frontend para visualizar los productos registrados de administración de TI
25. Frontend para visualizar los productos registrados de administración de Calidad
26. Frontend para visualizar los productos registrados de administración de Mantenimiento
27. Frontend para actualizar productos y eliminar en el inventario de administración de TI
28. Frontend para actualizar productos y eliminar en el inventario de administración de Calidad
29. Frontend para actualizar productos y eliminar en el inventario de administración de Mantenimiento
30. Frontend para escanear productos de administración de TI
31. Frontend para escanear productos de administración de Calidad
32. Frontend para escanear productos de administración de Mantenimiento
33. Frontend para generar reportes del administrador de TI
34. Frontend para generar reportes de administrador de Calidad
35. Frontend para generar reportes de administrador de Mantenimiento
36. Frontend para inventarios del supervisor de TI
37. Frontend para inventarios del supervisor de Calidad
38. Frontend para inventarios del supervisor de Mantenimiento
39. Frontend para el submenú de periféricos de TI
40. Frontend para el registro, actualización y eliminación de productos del supervisor en el área de TI
41. Frontend para el registro, actualización y eliminación de productos del supervisor en el área de Calidad
42. Frontend para el registro, actualización y eliminación de productos del supervisor en el área de Mantenimiento
43. Frontend para el escaneo de productos del supervisor en el área de TI
44. Frontend para el escaneo de productos del supervisor en el área de Calidad
45. Frontend para el escaneo de productos del supervisor en el área de Mantenimiento
46. Frontend para generar reportes del supervisor de TI
47. Frontend para generar reportes del supervisor de Calidad
48. Frontend para generar reportes del supervisor de Mantenimiento
49. Frontend para escanear productos de operador
50. Diseñar la arquitectura del sistema

## Fase 3

## Implementación

* Elegir el SGBD a utilizar
* Codificación de la base de datos
* Configurar el lector Keyence para enviar datos al servidor.
* Programar el backend para recibir, validar y almacenar los datos.
* Programar el frontend para mostrar información al administrador.
* Configurar la base de datos en XAMPP.
* Integrar la lógica de activación de alarmas desde el lector.

## Fase 4

## Verificación

* Pruebas de la base de datos
* Pruebas unitarias de cada módulo (escaneo, validación, almacenamiento).
* Pruebas de integración del sistema completo
* Pruebas de rendimiento de la velocidad de respuesta, carga de datos.
* Pruebas de error
* Validación con usuarios

## Fase 5

## Mantenimiento

* Corrección de errores
* Hacer actualizaciones
* Manual de usuario
* Manual técnico

**Desarrollo**

## Fase 1

## Análisis

### Herramienta de recolección de información.

* Levantamiento de requerimientos

Para poder llevar a cabo el desarrollo del sistema web se necesitó hacer un levantamiento de requerimientos con el operador de la línea de producción de las palomitas SLIMPOP y con los gerentes de cada área para recabar la información fundamental de sus inventarios, revisando a si los manuales técnicos del lector de códigos de barras SICAR Checko2B que se consideran una opción a usar, verificando también su compatibilidad y las especificaciones de los mismos.

### Generar la propuesta

Se propone el desarrollo de un sistema web que automatice el proceso de escaneo de productos en la línea de producción y a si mismo integrar este módulo de escaneo en el inventariado de cada área, el sistema recibirá los datos del lector de códigos de barras SICAR Checko2B, validará los códigos escaneados y registrará la información en una base de datos, el administrador podrá acceder a un panel de control donde visualizará y editar los campos del inventario en general, el registro de los usuarios, las actualizaciones y eliminación de los productos, así como también podrá hacer escaneos, reportes detallados por lote, fecha, hora, código del producto, descripción y estado, por otra parte el Supervisor solo tendrá accesos limitados a los inventarios, como el registro de producto, actualización, escaneo y a generar reportes, finalmente el operador solo escaneara productos, esta solución busca reducir errores, mejorar y optimizar el tiempo.

### Definir los requisitos funcionales del sistema.

El sistema estará contando con requisitos específicos que son fundamentales para el funcionamiento.

* Escaneo de códigos de barras desde el lector.
* Validación automática del código escaneado.
* Registro de productos válidos y errores en la base de datos.
* Gestión de inventarios por cada área.
* Generación de reportes por fecha, usuario y tipo de producto.
* Control de acceso por roles definidos.

### Analizar el lector de códigos como protocolos soportados, configuración.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Nombre del lector** | **Lenguaje** | **Función** | **Compatibilidad** | **Componentes** | **Librerías** | **Interfaces compatibles** |
| SICAR Checko2B | JavaScript + Node.js | El lector actúa como si fuera un teclado cada vez que escanea un código, lo “escribe” en el campo activo del navegador. | Si es adecuado para el sistema en JavaScript solo se necesita configurar en modo teclado para que funcione sin complicaciones | Frontend (JavaScript)  Backend (PHP con XAMPP)  Lector Checko2B  Entrada de datos vía teclado USB o Bluetooth | Vanilla JS o React.js para la interfaz  SweetAlert2 para alertas visuales de éxito/error  Axios o Fetch API para enviar datos al backend. | Interfaz serie RS-232 – Teclado USB – Interfaz serie virtual USB – Bluetooth. |

### Establecer el flujo de trabajo en la línea de producción como el escaneo, Validación, Registro, Visualización.

El flujo de trabajo del sistema será el siguiente:

* El producto pasa frente al lector de códigos de barras SICAR Checko2B.
* El lector escanea el código de barras del producto.
* El lector envía el código escaneado al sistema web, actuando como teclado (USB/Bluetooth).
* JavaScript en el frontend captura el código y lo envía al servidor mediante una solicitud HTTP.
* El backend (PHP) recibe el código y lo valida contra la base de datos MySQL.
* Si el código es válido lo registra como exitoso y continua su proceso, pero si el código es inválido emite una alerta sonora y visual, la cual el programa marca como erróneo.
* El sistema guarda el registro del escaneo.
* El administrador accede al panel web para consultar registros históricos filtrar por fechas, productos o estados y generar reportes.

### Determinar roles y permisos (administrador, operador).

El sistema contará con un Login que permitirá el acceso a dos tipos de usuarios uno será el operador y el otro el administrador

Administrador: Usuario con acceso completo al sistema, podrá consultar todos los registros, actualizar y eliminar, generando reportes por fecha y producto, configurar parámetros del sistema, y gestionar usuarios.

Supervisor: Usuario con acceso limitado al sistema, podrá consultar registros de sus áreas y actualizarlas, generar reportes por fecha y producto, escanear el producto.

Operador: Usuario encargado de supervisar el escaneo en la línea de producción, tendrá acceso limitado a la interfaz web, donde podrá visualizar el estado de los escaneos en tiempo real, pero no podrá modificar ni consultar reportes históricos.

La autenticación se realizará mediante credenciales únicas, y se implementará control de acceso basado en roles para garantizar la seguridad de la información.

1. **Infraestructura de Red y Servidor**

El sistema funcionará en una red local (LAN) donde el lector SICAR Checko2B se conectará por USB o Bluetooth a una computadora o directamente al lector de escaneo, el servidor local estará configurado con XAMPP, el cual alojará tanto el sistema web como la base de datos, los dispositivos dentro de la red podrán acceder al sistema mediante la IP local asignada al servidor para garantizar mayor estabilidad y velocidad en la transmisión de datos, permitiendo el acceso multiusuario con roles definidos, lo que facilitará la gestión de permisos y tareas dentro de la plataforma, en caso de requerirse una expansión futura, la infraestructura está preparada para escalar a un entorno en la nube sin afectar la operatividad del sistema.

## Fase 2

## Diseño

### Crear diagramas de caso de uso

### Crear diagramas secuenciales

### PlantUML diagramArquitectura cliente-servidor

### Backend en PHP (Diagrama E-R)

### Diagrama relacional

### Frontend de la ventana de registro de usuarios

### Frontend del Login

### Frontend para el panel del administrador

### Frontend para el panel del supervisor

### Frontend para el panel del operador

### Frontend para el inventario del administrador TI

### Frontend para el inventario del administrador Calidad

### Frontend para el inventario del administrador Mantenimiento

### Frontend para el submenú de activos de material del administrador de TI

### Frontend para el submenú de equipos de computo del administrador de TI

### Frontend para el submenú de vidrio, plástico, quebradizo del administrador Calidad

### Frontend para el submenú de herramientas del administrador Calidad

### Frontend para el submenú de manejo integral de plagas del administrador Calidad

### Frontend para el submenú de utensilios detectables del administrador Calidad

### Frontend para el submenú de cuchillos y afilados del administrador Calidad

### Frontend para el submenú de refacciones del administrador de Mantenimiento

### Frontend para visualizar los usuarios registrados de administración de TI

### Frontend para visualizar los usuarios registrados de administración de Calidad

### Frontend para visualizar los usuarios registrados de administración de Mantenimiento

### Frontend para visualizar los productos registrados de administración de TI

### Frontend para visualizar los productos registrados de administración de Calidad

### Frontend para visualizar los productos registrados de administración de Mantenimiento

### Frontend para actualizar productos y eliminar en el inventario de administración de TI

### Frontend para actualizar productos y eliminar en el inventario de administración de Calidad

### Frontend para actualizar productos y eliminar en el inventario de administración de Mantenimiento

### Frontend para escanear productos de administración de TI

### Frontend para escanear productos de administración de Calidad

### Frontend para escanear productos de administración de Mantenimiento

### Frontend para generar reportes del administrador de TI

### Frontend para generar reportes de administrador de Calidad

### Frontend para generar reportes de administrador de Mantenimiento

### Frontend para inventarios del supervisor de TI

### Frontend para inventarios del supervisor de Calidad

### Frontend para inventarios del supervisor de Mantenimiento

### Frontend para el submenú de material TI de TI

### Frontend para el equipo de cómputo de TI

### Frontend para el registro, actualización y eliminación de productos del supervisor en el área de TI

### Frontend para el registro, actualización y eliminación de productos del supervisor en el área de Calidad

### Frontend para el registro, actualización y eliminación de productos del supervisor en el área de Mantenimiento

### Frontend para el escaneo de productos del supervisor en el área de TI

### Frontend para el escaneo de productos del supervisor en el área de Calidad

### Frontend para el escaneo de productos del supervisor en el área de Mantenimiento

### Frontend para generar reportes del supervisor de TI

### Frontend para generar reportes del supervisor de Calidad

### Frontend para generar reportes del supervisor de Mantenimiento

### Frontend para escanear productos de operador

### Diseñar la arquitectura del sistema

## Fase 3

## Implementación

### Elegir el SGBD a utilizar

Para el desarrollo del sistema se utilizará XAMPP, este siendo una distribución de código abierto que permite instalar fácilmente un servidor web, este paquete incluye componentes esenciales como Apache, MySQL y PHP, lo que lo convierte en una herramienta gratuita ideal para el desarrollo y prueba de aplicaciones web en sistemas operativos como Windows, Linux y macOS, además, incorpora phpMyAdmin, una interfaz gráfica que facilita la gestión de bases de datos de manera sencilla y eficiente.

### Codificación de la base de datos

Create database PopCode;

Create table usuarios (

id\_usuario INT Primary key not null,

nombre\_usuario varchar(45),

contraseña varchar(16),

rol varchar(20)

);

Create table productos (

codigo\_barras INT Primary key not null,

descripcion varchar(45) unique,

imagen varchar(255)

);

Create table productos\_escaneados (

id\_escaneo int primary key not null,

fecha\_escaneo date,

hora\_escaneo time,

codigo\_barras int not null,

descripcion\_producto varchar(45),

estado varchar(45),

constraint fk\_codigo\_barras

foreign key (codigo\_barras)

references productos(codigo\_barras),

constraint fk\_descripcion\_producto

foreign key (descripcion\_producto)

references productos(descripcion)

);

Insert into usuarios (id\_usuario, nombre\_usuario, contraseña, rol) values (1, ‘Admin1’, ‘admin1p’, ‘Administrador’), (2, ‘Oper1’, ‘Oper1p’, ‘Operador’);

### Configurar el lector X para enviar datos al servidor.

### Programar el backend para recibir, validar y almacenar los datos.

### Programar el frontend para mostrar información al administrador.

### Configurar la base de datos en XAMPP.

### Integrar la lógica de activación de alarmas desde el lector.

## Fase 4

## Verificación

### Pruebas de la base de datos

### Pruebas unitarias de cada módulo (escaneo, validación, almacenamiento).

### Pruebas de integración del sistema completo

### Pruebas de rendimiento de la velocidad de respuesta, carga de datos.

### Pruebas de error

### Validación con usuarios

## Fase 5

## Mantenimiento

### Corrección de errores

### Hacer actualizaciones

### Manual de usuario

### Manual técnico

Cronograma de actividades