**Manual técnico Proyecto CAMBALACHES**

**Nicolas Pachon Villabon**

**Sergio Andrey Paez**

**Johan Sebastian Gonzales**

**Servicio nacional de aprendizaje**

**2558104 G1 Análisis y desarrollo de software**

**Marzo de 2024**

|  |  |  |
| --- | --- | --- |
| No | Descripción | Detalle |
| 1 | Alcance | **Este manual describe los detalles técnicos de la aplicación CAMBALACHES que permitirán al personal técnico suministrar el soporte de primer nivel sobre la aplicación** |
| 2 | Descripción de procesos | **Diagrama  Descripción generada automáticamenteEl sistema de Gestión CAMBALACHES soporta el caso de uso “gestión de compra y ventas” de acuerdo con el siguiente gráfico:** |
| 3 | Diagramas UML | **El diagrama de clases del sistema es el siguiente:**  **Diagrama  Descripción generada automáticamente**  **Clase usuario**    **Clase vendedor**    **Clase comprador**    **Clase administrador**    **Clase Roles      clase usuario\_has\_roles**    **Clase tienda**    **Clase productos**    **Clase tipo\_documento**    **Clase factura\_productos**    **Clase factura\_venta**    **clase factura**    **Clase categorías** |
| 4 | Modelo relacional de la base de datos | El modelo relacional de la base de datos cambalaches se muestra a continuación:  **Diagrama  Descripción generada automáticamente** |
| 5 | Descripción de la plataforma | Cambalaches es una aplicación Web basada en una aplicación de venta y compra de productos de vestir  la interfaz gráfica esta desarrollada HTML, diseños o plantillas basados en librerías de Bootstrap. API desarrollado en node.js, y frontend con librerías react.js.  para instalar el servidor de la aplicación se requiere de lo siguiente:   1. Sistema operativo Windows 10 o posteriores Linux karnel 3.4 en adelante o mac os v11 o posteriores. 2. Servidor apache versión 2 o posteriores 3. React.JS (recomiendo tener instalados paquetes de recursos react para mejor experiencia) npm install. 4. SQL (Mysql2), phpmyadmin para administrar movimientos. (XAMPP). 5. Node.js v20.9.0.   para el mantenimiento del software se requiere el siguiente ambiente. |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 6 | Documentación del código fuente. | 1. Descripción de la aplicación:   Presento a continuación los archivos, frontend:    Archivos API, backend:     1. Estructura de directorios:  |  |  | | --- | --- | | Directorio | Descripción | | Src/assets | Contiene las imágenes que se carguen dentro del programa, tanto visuales como diseños, y productos. | | Src/components | Contiene carpetas con archivos de funcionalidades, como categorías, encabezados (Navbar), productos de carrito, descripciones. | | Src/hook | Contiene las validaciones para los usuarios registrados que desean logearse. | | Src/modules | Contiene en general la estructura del proyecto como tal, tiene un home, inicio, registro, productos, logeo. | | Src/services | Contiene el URL de la API, es decir la conexión a la base de datos para el login. | | Src/App.js | Contiene rutas para la navegación. | | Src/index.js | Contiene las librerías react. | | Package.json | Contiene todos los componentes y versiones para el funcionamiento del sistema frontend. | |  |  |  1. Relación de programas: Hablare en general de como se comporta cada archivo y sus componentes.  |  |  | | --- | --- | | Directorio | Descripción | | router/login.js | Contiene una parte fundamental para dar el acceso a los usuarios ya registrado en la base de datos. | | router/usuariosRouter.js | Hace las consultas la base de datos y valida los registros, con todo lo que requiere el formulario. | | router/tipo\_documentoRouter.js | Aquí queremos dar la opción a todas las personas para su ingreso según los tipos de documentos admitidos en la base de datos. | | router/usuario\_has\_rolesRouter.js | Este archivo nos ayuda a diferenciar el rol de vendedor comprador con el administrador. | | router/administradorRouter.js | Como bien lo mencione arriba, este es el rol que define al administrador, para que tome acciones únicas sobre la web. | | router/productosRouter.js | Este archivo, esta diseñado para que el front llame a la carga de productos, y asi cumpla con la expectativa para los usuarios de vender y comprar productos. | | router/app.js | Contiene la version mysql2, base de datos y conexión con el front para el manejo de datos. | | router/index.js | Contiene, la ruta del puerto que usa para trabajar de la mano con el servidor front, también contiene las rutas establecidas para el funcionamiento que tiene cada rol y le permite hacer uso de los ítems. | |