**1. Documentación de usuario final:**

**1.1. Guía de usuario**

*Ayuda: Incluye instrucciones sobre cómo utilizar el software. Usa capturas de pantalla, diagramas y ejemplos para hacerla más accesible.*

**Introducción**

Explicar que aplicacion es y para que sirve

**Requisitos del Sistema**

* **Navegador web**:
* **Resolución recomendada**:
* **Conexión a internet**:

**Acceso**

Explicamos como acceder a ella. De momento está en el localhost, pero si estuviera en un servidor web orperativa solo seria acceder mediante el enlace en el navegador

**Interfaz de Usuario**

Capturas de pantalla del login, escritorio y supercontrolador con las tablas.

Explicamos un poco por encima cada una de ellas, html y colores

**Pasos para Usar la Herramienta**

Explicamos el login, como funciona, como accedemos luego a las diferentes aplicaciones que hay y como usamos la vista del supercontrolador. Explicamos las diferentes tablas, el tooltip, el modal para meter datos y como se eliminan los datos.

**Consejos de Uso**

* **Mantenga su navegador actualizado:** Para asegurar la compatibilidad con las últimas funcionalidades.
* **Siga las instrucciones en pantalla:** Evite recargar la página durante el proceso.
* **Resolución de problemas:**
  + Si no puede acceder a la página, consulte con el administrador.
  + Revise la conexión a internet y los permisos del navegador.

**1.2. FAQ (Preguntas frecuentes)**

*Ayuda: Responde preguntas comunes para facilitar el soporte y resolver dudas rápidamente.*

**1. ¿Qué es?**Explicamos que es la app y para que sirve

**2. ¿Cómo accedo a la aplicación?**Cómo acceder a ella

**3. ¿Qué hago si no puedo acceder a la aplicación?**

* Verifica que la URL sea correcta.
* Asegúrate de estar conectado a internet.
* Si el problema persiste, informa al administrador.

**4. ¿Que aplicaciones puedo ver?  
5. ¿Como puedo ingresar un nuevo cliente?  
6. ¿Como puedo eliminar un cliente?  
7. ¿como puedo hacer una busqueda?  
8. ¿Qué hago si no aparecen los clientes?  
9.  ¿A quién contacto si tengo problemas técnicos?**Comunícate con el administrador del sistema y proporciona detalles como la URL del panel, la sala afectada y una descripción del problema.

**1.3. Solución de problemas comunes**

*Ayuda: Describe errores comunes que puedan surgir y sus posibles soluciones.*

**1. No puedo acceder a la aplicación.**

* **Causa posible:** La URL ingresada es incorrecta o el servidor no está disponible.
* **Solución:**
  + Verifica que la URL sea correcta.
  + Confirma que tienes conexión a internet.
  + Intenta acceder desde otro navegador.
  + Informa al administrador si el problema persiste.

**2. La aplicación no responde o se queda cargando.**

* **Causa posible:** Problemas de conexión o un error en el servidor.
* **Solución:**
  + Actualiza la página (F5 o botón de recargar).
  + Cierra otras aplicaciones que consuman tu conexión a internet.
  + Intenta acceder más tarde e informa al profesor.

**3. Los datos de los clientes introducidos no se actualizan.**

* **Causa posible:** Tu conexión a internet es inestable o hay un error en el sistema.
* **Solución:**
  + Asegúrate de no cerrar la página mientras trabajas.
  + Verifica tu conexión a internet.
  + Notifica al profesor para que revise el sistema.

**7. La ventana modal no se abre al hacer clic en una imagen.**

* **Causa posible:** Problemas con el código de JavaScript.
* **Solución:**
  + Recarga la página y asegúrate de que JavaScript esté habilitado en el navegador.
  + Si el problema persiste, contacta al administrador para una revisión técnica.

**2. Documentación técnica para desarrolladores**

**2.1. Introducción al software**

*Ayuda: Explica de qué se trata el software, su propósito, características principales y tecnologías utilizadas.*

**Descripción General**

Explicación de las  partes que conforman el software.

**Objetivos Principales**

El objetivo es poder realizar tareas relacionadas con la gestión de una empresa como control de clientes, pedidos,

**Arquitectura del Software**

1. **Frontend: supercontrolador**
2. **Backend: carpeta servidor**
3. **Base de Datos y Almacenamiento: base de datos de mysql en apache**

**Flujo de Trabajo**

Acceso a travé del login, acceso al escritorio donde selecciona una app y acceso al supercontrolador donde puede visualizarse cada una de las tablas con los datos de clientes usuarios pedidos etc

**Tecnologías Utilizadas**

1. **Frontend:**
   * HTML5 y CSS3 para la estructura y diseño.
   * JavaScript para funcionalidades interactivas.
2. **Backend:**
   * PHP para la lógica del servidor y procesamiento de solicitudes.
   * APIs internas para el manejo de datos dinámicos y multimedia.
3. **Otros recursos:**
   * Fuentes de Google (p. ej., Roboto, Ubuntu) para un diseño moderno.
   * Estilo responsivo y centrado en la experiencia de usuario.
   * tooltip para dar informacion

**Público Objetivo**

Administrativos y personal relacionado con la administracion y gestion de la empresa

**2.2. Arquitectura del sistema**

*Ayuda: Incluye diagramas de arquitectura y una descripción de los componentes principales (como bases de datos, servidores, API, etc.) y cómo interactúan entre sí.*

La arquitectura del sistema sigue un modelo **cliente-servidor**, donde el cliente se encarga de la interacción con el usuario y el servidor maneja las operaciones de la base de datos y la lógica de negocio.

**1. Estructura General**

El sistema se divide en dos componentes principales:

* **Frontend**: Interfaz para estudiantes y profesores.
* **Backend**: Servidor que gestiona la lógica de negocio, almacenamiento y comunicación con el cliente.

**2. Componentes de la Arquitectura**

**1. Frontend (Cliente)**

**Descripción:**

El frontend es una aplicación web construida con tecnologías HTML, CSS y JavaScript. Su objetivo es proporcionar una interfaz interactiva y dinámica para la gestión de datos

**Componentes:**

1. **HTML**:
   * Define la estructura básica de la interfaz.
   * Carga las bibliotecas necesarias y los estilos personalizados​(index).
2. **CSS**:
   * estilo.css: Proporciona el diseño visual principal, incluyendo el uso de flexbox para la disposición de elementos y estilos de tablas​(estilo).
   * Librerías específicas:
     + jvampliador.css: Herramientas para accesibilidad.
     + jvtooltip.css: Estilos de tooltips.
     + selectjv.css: Personalización de elementos <select>​(jvampliador)​(jvtooltip)​(selectjv).
3. **JavaScript**:
   * comportamiento.js: Gestiona la inicialización del programa, la apertura/cierre de modales y las interacciones con el servidor​(comportamiento).
   * pueblaTabla.js: Pobla dinámicamente las tablas con datos obtenidos del servidor, permitiendo edición y eliminación de registros​(pueblaTabla).
   * cargaDatosTabla.js: Se encarga de cargar la estructura de columnas y datos de las tablas seleccionadas desde el servidor​(cargaDatosTabla).
   * convierteTipoDato.js: Convierte los tipos de datos de la base de datos en tipos de entrada HTML​(convierteTipoDato).
   * Librerías específicas:
     + selectjv.js: Crea selectores personalizados con funcionalidad de búsqueda​(selectjv).
     + jvampliador.js: Añade botones de accesibilidad para ajustar fuentes y contraste​(jvampliador).
     + jvtooltip.js: Implementa tooltips dinámicos para mejorar la experiencia del usuario​(jvtooltip).

**Flujo de Datos en el Frontend:**

1. El usuario selecciona una tabla desde el menú lateral.
2. Se realizan llamadas al servidor para obtener columnas y datos relacionados.
3. Los datos se muestran en la tabla central, con herramientas de edición y eliminación.
4. Modales dinámicos permiten insertar nuevos registros.

**2. Backend (Servidor)**

**Descripción:**

El backend expone una serie de endpoints que permiten realizar operaciones CRUD (crear, leer, actualizar, eliminar) sobre las tablas de la base de datos.

**Endpoints Principales:**

1. **Listar tablas**:
   * URL: ../../servidor/?o=listatablas
   * Método: GET
   * Devuelve un listado de tablas disponibles en la base de datos.
2. **Cargar columnas**:
   * URL: ../../servidor/?o=columnastabla&tabla={nombre\_tabla}
   * Método: GET
   * Devuelve la estructura de columnas de una tabla específica.
3. **Cargar datos**:
   * URL: ../../servidor/?o=tabla&tabla={nombre\_tabla}
   * Método: GET
   * Devuelve los registros de una tabla específica.
4. **Actualizar datos**:
   * URL: ../../servidor/?o=actualizar
   * Método: POST
   * Recibe datos en formato JSON y actualiza registros en la base de datos.
5. **Eliminar datos**:
   * URL: ../../servidor/?o=eliminar&tabla={nombre\_tabla}&id={id}
   * Método: GET
   * Elimina un registro específico en la base de datos.
6. **Insertar datos**:
   * URL: ../../servidor/?o=insertar&tabla={nombre\_tabla}
   * Método: POST
   * Recibe datos de un formulario y los inserta en la base de datos.

**Flujo de Datos en el Backend:**

1. El cliente realiza solicitudes a los endpoints según las acciones del usuario.
2. El backend consulta la base de datos y devuelve respuestas en formato JSON.
3. Estas respuestas son procesadas y renderizadas en el frontend.

**3. Base de Datos**

**Descripción:**

La base de datos almacena los registros y metadatos necesarios para el funcionamiento del sistema.

**Estructura:**

* **Tablas Dinámicas**:
  + Las tablas son identificadas dinámicamente a través de los endpoints del backend.
  + Cada tabla tiene una clave primaria identificada como PRI, utilizada para operaciones específicas como actualización y eliminación.

**Relaciones:**

* Se soportan claves foráneas (MUL) que son manejadas como elementos <select> en el frontend.

**4. Comunicación Cliente-Servidor**

**Descripción:**

El sistema utiliza la API Fetch para realizar solicitudes HTTP entre el cliente y el servidor.

**Características:**

* Uso de métodos GET para recuperar datos y POST para enviar actualizaciones o inserciones.
* Respuestas en formato JSON para facilitar el manejo dinámico de datos en el frontend.

**5. Accesibilidad y Usabilidad**

* **Modales Dinámicos**:
  + Formularios personalizados se generan dinámicamente para cada tabla seleccionada.
* **Accesibilidad Mejorada**:
  + Herramientas para cambiar tamaño de fuente y contraste visual.
  + Tooltips dinámicos que describen las acciones disponibles.

**Diagramas**

**1. Flujo General del Sistema**

scss

Copiar código

Usuario ↔ Navegador (Frontend) ↔ Servidor (Backend) ↔ Base de Datos

**2. Componentes del Frontend**

scss

Copiar código

HTML (Estructura) ↔ CSS (Estilo) ↔ JS (Interactividad)

**3. Comunicación con Backend**

javascript

Copiar código

Frontend → Fetch → Endpoint Backend → Operaciones en BD → Respuesta JSON

**2.3. Estructura del código**

*Ayuda: Explica cómo está estructurado el código. Incluye una descripción de directorios y archivos, así como los principales módulos y sus responsabilidades.*

El código del sistema está organizado en múltiples archivos, cada uno con responsabilidades específicas. A continuación, se detalla la estructura del código para los principales archivos extraídos:

**1. Archivo**

**Propósito:**

**Estructura del Código:**

* **HTML**
* **Estilos CSS:**
* **Interacción:**

**Relación entre los Archivos**

**Organización de Archivos**

* **Frontend:**
* **Backend:**

**Ventajas de la Estructura**

1. **Modularidad:**Cada archivo tiene responsabilidades específicas, facilitando el mantenimiento.
2. **Flexibilidad:**Permite agregar nuevas funcionalidades sin afectar la estructura general.
3. **Simplicidad:**Divide las tareas del frontend y el backend claramente.

**2.4. Especificaciones de API**

*Ayuda: Describe las APIs o endpoints utilizados, con detalles de entrada y salida, ejemplos de solicitud y respuesta, y posibles códigos de error.*

[Inserta aquí tu texto]

**2.5. Dependencias y configuraciones**

*Ayuda: Enumera las dependencias del software y cómo instalarlas. Proporciona detalles de configuración, como archivos de configuración, variables de entorno y bases de datos.*

[Inserta aquí tu texto]

**1. Dependencias del Proyecto**

El sistema utiliza diversas librerías y archivos para proporcionar funcionalidades específicas en el frontend. Estas dependencias deben estar correctamente configuradas para asegurar el funcionamiento del software.

**1.1. Librerías y Archivos Internos**

1. **Librerías Personalizadas:**
   * selectjv.js y selectjv.css:
     + Propósito: Personalización de elementos <select> con opciones dinámicas y búsqueda interactiva.
     + Ubicación: /lib/selectjv/​(selectjv)​(selectjv).
   * jvampliador.js y jvampliador.css:
     + Propósito: Herramientas de accesibilidad como cambio de fuente, contraste y tamaño de texto.
     + Ubicación: /lib/jvampliador/​(jvampliador)​(jvampliador).
   * jvtooltip.js y jvtooltip.css:
     + Propósito: Tooltips dinámicos que muestran información al pasar el cursor sobre elementos.
     + Ubicación: /lib/jvtooltip/​(jvtooltip)​(jvtooltip).
2. **Archivos Específicos del Proyecto:**
   * convierteTipoDato.js: Conversión de tipos de datos de la base de datos a tipos de entrada HTML​(convierteTipoDato).
   * pueblaTabla.js: Población dinámica de tablas en el frontend​(pueblaTabla).
   * cargaDatosTabla.js: Carga de datos, columnas y formularios de inserción desde el servidor​(cargaDatosTabla).
   * comportamiento.js: Inicialización y manejo de eventos en la aplicación​(comportamiento).
3. **Recursos Estáticos:**
   * Fuentes personalizadas:
     + Archivo personalizada.ttf, registrado en jvampliador.css​(jvampliador).

**1.2. Dependencias del Servidor**

1. **Backend:**
   * Requiere un servidor con soporte para:
     + APIs RESTful.
     + Formato de respuesta en JSON.
     + Métodos HTTP (GET, POST).
2. **Base de Datos:**
   * El servidor debe conectarse a una base de datos relacional compatible con SQL.
   * Debe permitir consultas dinámicas para tablas especificadas en tiempo de ejecución.
3. **Scripts del Servidor:**
   * Los endpoints deben estar configurados para manejar operaciones como:
     + Listado de tablas (listatablas).
     + Carga de columnas (columnastabla).
     + Carga de datos (tabla).
     + Inserción (insertar).
     + Actualización (actualizar).
     + Eliminación (eliminar).

**2. Configuraciones del Proyecto**

**2.1. Configuración del Frontend**

1. **Archivos Incluidos en index.html**:

Scripts y estilos deben incluirse en el orden correcto:  
html  
Copiar código  
<!-- Librerías Personalizadas -->

<script src="lib/selectjv/selectjv.js"></script>

<link rel="stylesheet" href="lib/selectjv/selectjv.css">

<script src="lib/jvampliador/jvampliador.js"></script>

<link rel="stylesheet" href="lib/jvampliador/jvampliador.css">

<script src="lib/jvtooltip/jvtooltip.js"></script>

<link rel="stylesheet" href="lib/jvtooltip/jvtooltip.css">

<!-- Archivos Principales del Proyecto -->

<script src="js/convierteTipoDato.js"></script>

<script src="js/pueblaTabla.js"></script>

<script src="js/cargaDatosTabla.js"></script>

<script src="comportamiento.js"></script>

<link rel="stylesheet" href="estilo.css">

* **Prioridad en la Carga**:
  + Las librerías deben cargarse antes de los scripts del proyecto.

1. **Configuración de Fuentes Personalizadas**:
   * Las fuentes deben estar disponibles en la ruta configurada en estilo.css y jvampliador.css.
2. **Dependencias de Rutas Relativas**:
   * Verifica que las rutas especificadas en el HTML y CSS apunten correctamente a los archivos.

**2.2. Configuración del Backend**

1. **Conexión a la Base de Datos**:
   * La base de datos debe configurarse para aceptar conexiones del servidor.
   * Los nombres de las tablas deben coincidir con las solicitudes del frontend.
2. **Configuración de Endpoints**:
   * Asegúrate de que los endpoints definidos en el servidor estén alineados con las solicitudes del frontend, por ejemplo:
     + /servidor/?o=listatablas
     + /servidor/?o=columnastabla&tabla={nombre\_tabla}
   * Los métodos permitidos deben coincidir (GET, POST).
3. **Permisos**:
   * Configura permisos para asegurar que las operaciones de inserción, edición y eliminación sean seguras.
   * Implementa autenticación si es necesario.

**3. Guías de instalación y despliegue**

**3.1. Requisitos previos**

*Ayuda: Lista los requisitos del sistema, como versiones específicas de software, hardware, y permisos necesarios.*

El sistema utiliza una combinación de tecnologías de front-end y back-end, además de configuraciones en el servidor para asegurar su correcto funcionamiento. A continuación, se describen las dependencias y configuraciones necesarias para implementar y operar este software.

**Instalación y Configuración Inicial**

**Requisitos del Entorno**

**3.2. Instrucciones de instalación**

*Ayuda: Proporciona una guía paso a paso para instalar el software en diferentes entornos (desarrollo, producción, etc.).*

A continuación, se describen los pasos necesarios para instalar y configurar el sistema en un servidor local o remoto.

**Requisitos Previos**

1. **Servidor Web**:
   * Apache o Nginx instalado.
   * PHP configurado y habilitado.
   * Acceso a una terminal de comandos (si se realiza la instalación en un servidor).
2. **PHP**:
   * Versiones compatibles: PHP 7.4 o superior.
   * Extensiones habilitadas:
     + json
     + gd o imagick (opcional para manipulación de imágenes).
3. **Sistema Operativo**:
   * Linux (Ubuntu/Debian recomendado), Windows o macOS.
4. **Base de Datos (opcional)**:
   * MySQL o SQLite si se necesita almacenar datos adicionales (usuarios, metadatos de capturas).
5. **Herramientas necesarias**:
   * Un editor de texto para configurar archivos (Nano, Vim, VS Code, etc.).
   * Acceso al sistema de archivos para copiar y mover archivos.

**Pasos de Instalación**

**1. Descargar y Preparar los Archivos**

1. Descomprime el archivo ZIP proporcionado (por ejemplo, examen.zip).
2. Copia todos los archivos (index.php, admin.php, etc.) en el directorio raíz de tu servidor web.
   * En Apache: /var/www/html/
   * En XAMPP (Windows): C:\xampp\htdocs\

**2. Configurar el Servidor Web**

**3. Configurar Permisos de Archivos**

**5. Probar la Instalación**

1. Accede a las URLs principales desde el navegador:
   * Página del estudiante: http://localhost/index.php
   * Panel de administración: http://localhost/admin.php
2. Realiza pruebas:
   * Desde index.php, intenta capturar y enviar una pantalla.
   * Verifica que la captura sea visible en el panel de administración (admin.php).

**Estructura de Archivos**

**Configuraciones Opcionales**

**Resolución de Problemas**

1. **El servidor no responde:**
2. **Error 404 en URLs:**

**3.3. Despliegue**

*Ayuda: Explica cómo desplegar el software, ya sea en servidores locales o en la nube. Agrega procedimientos de actualización y rollback.*

[Inserta aquí tu texto]

**3.4. Pruebas y verificación**

*Ayuda: Detalla cómo ejecutar pruebas para asegurarse de que el software está funcionando correctamente después de la instalación.*

Después de instalar el sistema, es crucial realizar una serie de pruebas para garantizar que cada componente funcione correctamente. A continuación, se describen los pasos para ejecutar estas pruebas.

**1. Verificación Básica del Servidor**

Antes de probar las funcionalidades específicas del sistema, asegúrate de que el servidor y PHP estén configurados correctamente.

**Pasos:**

1. Accede a tu navegador y abre:
2. Si ves las interfaces correspondientes sin errores, el servidor está funcionando.

Si aparece un error 500, revisa los logs de Apache

**2. Pruebas de Funcionalidades**

**Objetivo:**

Asegurarse de que los estudiantes puedan interactuar correctamente con el sistema.

**Pruebas:**

**5. Validación de Errores**

**Objetivo:**

Simular situaciones comunes de error para verificar que el sistema las maneja adecuadamente.

**Pruebas:**

**6. Pruebas Finales de Usuario**

**Objetivo:**

Simular un entorno de uso real

**Pasos:**

**7. Resultado Esperado**

Si las pruebas son exitosas:

* Las páginas deben ser accesibles y funcionales.
* Las capturas de pantalla deben guardarse correctamente y ser visibles en el panel de administración.
* No deben aparecer errores en el navegador ni en los logs del servidor.

**4. Mantenimiento y soporte**

**4.1. Manejo de errores**

*Ayuda: Indica cómo capturar y manejar errores, incluyendo una lista de códigos de error y mensajes explicativos.*

Un sistema robusto debe ser capaz de identificar, capturar y manejar errores de manera que facilite su resolución. A continuación, se describe cómo manejar errores en este sistema, incluyendo una lista de códigos de error y sus mensajes explicativos.

**1. Captura de Errores**

**2. Manejo de Errores Comunes**

**3. Mensajes Explicativos**

**4. Ejemplo de Código con Manejo de Errores**

**5. Pruebas de Manejo de Errores**

1. **Errores Simulados:**
   * Envía datos incompletos desde el frontend y verifica los mensajes de error.
   * Desactiva permisos de escritura en el servidor y revisa los logs para el manejo de errores de almacenamiento.
2. **Errores Reales:**
   * Simula un corte de conexión en el navegador para verificar que el sistema maneje adecuadamente errores de red.
   * Intenta acceder a un archivo inexistente para confirmar el manejo del error 404.

**4.2. Registro de cambios (changelog)**

*Ayuda: Detalla los cambios realizados en cada versión, incluyendo nuevas características, correcciones y mejoras.*

A continuación, se documentan los cambios realizados en el desarrollo del software en forma de changelog. Incluye las nuevas características implementadas, correcciones realizadas, y mejoras aplicadas desde el inicio del desarrollo.

**4.3. Soporte y contacto**

*Ayuda: Proporciona información sobre cómo obtener soporte técnico, ya sea por correo, teléfono, o sistemas de tickets.*

El soporte técnico está disponible para resolver dudas, problemas técnicos, y proporcionar asistencia en el uso del sistema. A continuación, se detallan los canales de soporte disponibles.

**1. Canales de Soporte**

**Correo Electrónico**

* **Correo para Soporte Técnico General:**
  + Dirección: soporte@sistemaedu.com
  + Tiempo de respuesta: 24-48 horas hábiles.
  + Uso:
    - Reportar errores en el sistema.
    - Solicitar ayuda para la instalación o configuración.
    - Realizar consultas generales sobre el uso del software.

**Teléfono**

* **Línea de Soporte Técnico:**
  + Número: +34 600 123 456
  + Horario de atención: Lunes a Viernes, 9:00 a 18:00 (hora local).
  + Uso:
    - Consultas urgentes relacionadas con fallos críticos del sistema.
    - Asistencia para problemas que impidan el uso inmediato del software.

**Sistema de Tickets**

* **Plataforma de Tickets:**
  + URL: https://soporte.sistemaedu.com
  + Horario de atención: 24/7 (registro de incidencias).
  + Uso:
    - Reportar problemas específicos.
    - Realizar un seguimiento detallado del estado de una solicitud.
    - Enviar solicitudes de nuevas características.

**2. Información que Debes Proporcionar**

Para agilizar la resolución de problemas, incluye la siguiente información al contactar soporte:

1. **Detalles del problema:**
   * Descripción clara del problema.
   * Pasos para reproducir el error (si aplica).
2. **Capturas de pantalla o registros:**
   * Capturas de los errores visibles.
   * Logs del servidor (si se trata de un error en el backend).
3. **Entorno técnico:**
   * Versión del sistema operativo del servidor.
   * Versión de PHP.
   * Navegador utilizado (si aplica).
4. **Información de contacto:**
   * Nombre completo.
   * Dirección de correo electrónico o número telefónico.

**3. Escalación de Problemas**

Si el problema no puede ser resuelto en los tiempos estimados o requiere atención inmediata, solicita que tu caso sea escalado al **equipo de soporte avanzado** mediante cualquiera de los canales mencionados.

**4. Documentación y Recursos de Ayuda**

Antes de contactar al soporte, revisa la documentación técnica y las guías de usuario incluidas con el software:

* **Documentación Técnica:** Incluye soluciones a problemas comunes, manejo de errores y detalles de configuración.