Elabora documentos técnicos y de usuario del software, manual de usuario, técnico, plan de capacitación y acta de entrega del proyecto de software.

Juan Luis Becquet Martínez

Servicio Nacional de Aprendizaje

(2721441)Análisis y desarrollo de software

Catherine Ramirez

17 de Noviembre de 2024

Sumario

Introducción

Proporcionar una guía clara y detallada para que los usuarios puedan comprender y utilizar de manera efectiva el sistema de gestión académica diseñado para colegios pequeños. Este manual tiene como objetivo facilitar el acceso y aprovechamiento de todas las funcionalidades del sistema, incluyendo la administración de estudiantes, gestión de calificaciones, registro de asistencia, comunicación con padres, y generación de reportes, garantizando así una experiencia de usuario óptima y eficiente.

Alcance

Este manual cubre todas las funcionalidades principales del sistema de gestión académica, incluyendo los procesos de inicio de sesión, gestión de estudiantes, registro de calificaciones, control de asistencia, comunicación con los padres y generación de reportes. Además, aborda aspectos administrativos como la configuración de usuarios y la personalización de materias y clases. El manual está diseñado para ser una herramienta de referencia tanto para el personal administrativo como para los docentes, facilitando el uso adecuado del sistema. No incluye procedimientos técnicos de instalación o mantenimiento avanzado del sistema, que son competencia del equipo de soporte técnico.

Proposito

Proposito Operacional

El propósito de este manual es capacitar a los usuarios del sistema de gestión académica en el uso eficiente de sus funcionalidades, asegurando que puedan realizar sus tareas diarias de manera autónoma y sin dificultad. El manual busca reducir dudas y errores comunes, mejorar la productividad y garantizar una implementación fluida del sistema en el entorno educativo. Asimismo, pretende estandarizar el uso del sistema para facilitar la gestión académica en colegios pequeños, promoviendo una administración más organizada, comunicativa y orientada al seguimiento académico de los estudiantes.

Proposito tecnico

El propósito técnico de este manual es guiar a los usuarios y administradores en la correcta instalación y configuración del sistema de gestión académica en los dispositivos de la institución. Incluye instrucciones detalladas para la instalación de programas requeridos, configuración del entorno del sistema y recomendaciones técnicas para garantizar un rendimiento óptimo. Este apartado técnico está orientado a los responsables de TI y personal técnico del colegio, proporcionando los pasos necesarios para asegurar que el sistema esté completamente operativo y listo para su uso por el personal administrativo y docente

Objetivo final

El objetivo final de este manual es garantizar que todos los usuarios, desde el personal docente y administrativo hasta el equipo técnico, puedan utilizar e implementar el sistema de gestión académica de manera efectiva y sin contratiempos. A través de instrucciones claras y detalladas, el manual busca facilitar la comprensión, instalación, configuración y uso de todas las funciones del sistema, promoviendo así una gestión académica eficiente, organizada y adaptada a las necesidades específicas de colegios pequeños.

Prerequisitos

Python:

- **Versión**: Python 3.12
- **Descarga**: Python 3.12
- **Verificación de instalación**: Una vez instalado, abre la terminal o símbolo del sistema y ejecuta python --version para confirmar que Python 3.12 está activo.
- **Gestor de Paquetes Pip**:
- Debe estar incluido con Python 3.12, pero puedes verificarlo ejecutando pip --version.
- Pip es necesario para instalar las dependencias del proyecto, como Flask y pymysql.

Framework Flask:

- **Instalación**: Ejecuta pip install Flask para instalar el framework.
- **Librería pymysql**:
- **Instalación**: Ejecuta pip install pymysql para instalar la librería que facilita la conexión entre Python y MySQL.
- **MySQL Workbench**:
- **Descarga**: MySQL Workbench
- Asegúrate de tener una instancia de MySQL configurada y de conocer las credenciales (usuario, contraseña, puerto, y nombre de la base de datos) para la conexión.
- **JavaScript y HTML**:
- **Navegador web**: Asegúrate de tener un navegador actualizado (Chrome, Firefox, Safari o Edge).
- **Editor de código**: Utiliza un editor de texto como Visual Studio Code, Sublime Text, o Atom para modificar archivos HTML y JavaScript si es necesario.
- **Entorno Virtual (Opcional pero Recomendado)**:
- **Configuración**: Puedes crear un entorno virtual para aislar las dependencias del proyecto. Ejecuta python -m venv nombre_entorno y luego activa el entorno según tu sistema operativo.

Diagrama de casos de uso

Los diagramas de casos de uso son una representación gráfica de cómo interactúan los usuarios (actores) con las funcionalidades principales del sistema. En este software de gestión académica, el **Administrador** es el único actor principal, y los diagramas reflejan las acciones que puede realizar en el sistema.

Propósito de los Diagramas de Casos de Uso:

1. **Definir las Funcionalidades del Sistema**:
    - Muestran claramente qué acciones o procesos permite realizar el software, como registrar estudiantes, asignar calificaciones, o crear grupos.
2. **Comunicación Efectiva**:
    - Facilitan la comunicación entre desarrolladores, analistas y clientes al describir de forma visual las expectativas y capacidades del sistema.
3. **Guía para el Desarrollo**:
    - Ayudan a priorizar y estructurar las funcionalidades durante el diseño y desarrollo del software.
4. **Documentación del Sistema**:
    - Sirven como parte de la documentación para futuros mantenimientos o actualizaciones.

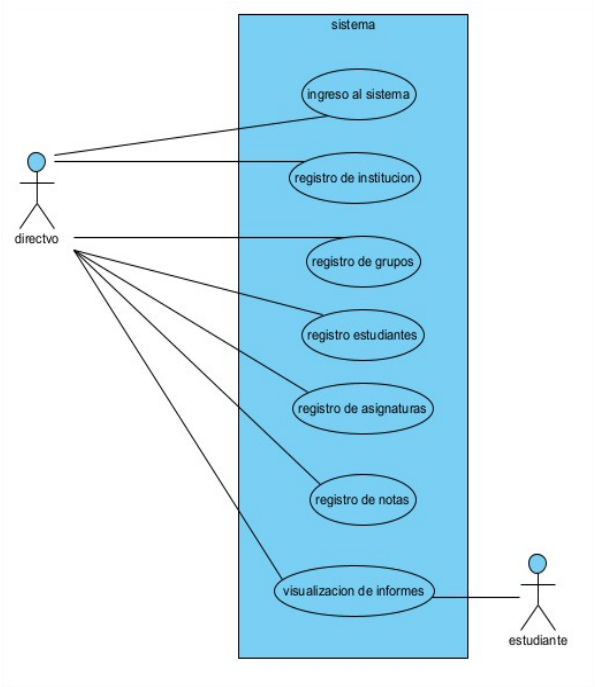


Diagrama de clases

Un diagrama de clases es una representación visual en UML que muestra las clases, interfaces, asociaciones y relaciones en un sistema.

funcion: Representar las clases, interfaces, atributos y métodos en un sistema orientado a objetos.

Beneficios: Facilita la comprensión de la estructura y la lógica del sistema, identifica las relaciones entre las clases y ayuda en el diseño de software

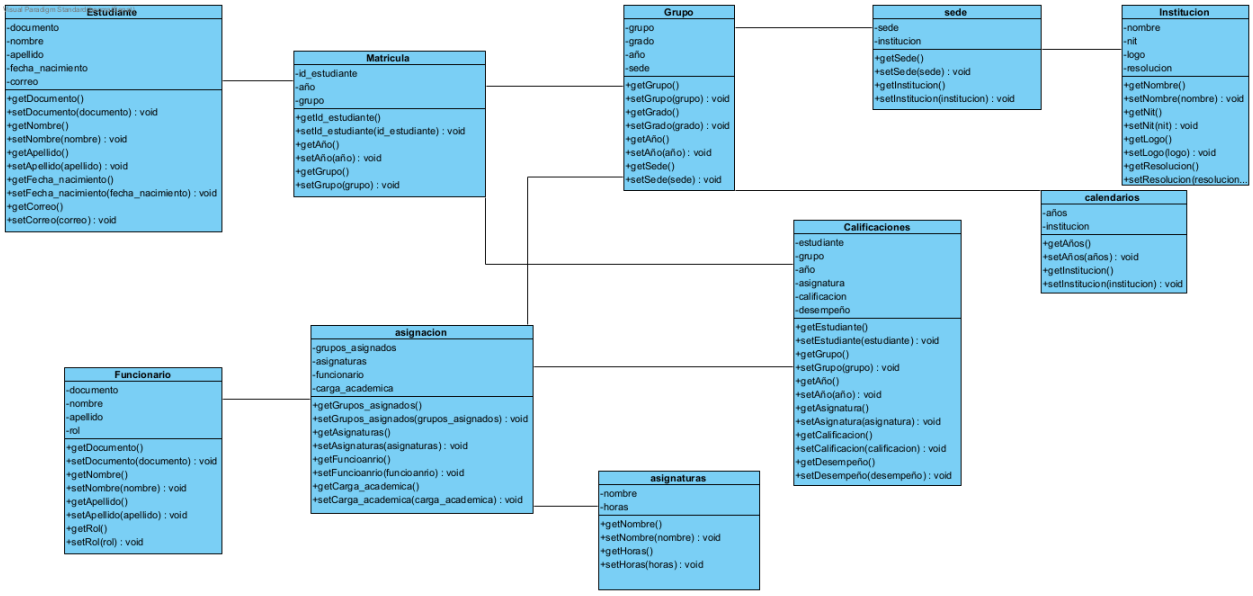


Diagrama de despliegue

Visualización de la Infraestructura:

- Representa los nodos físicos y virtuales (como servidores, bases de datos, clientes) y su interacción.
- **Planificación de la Instalación**:
- Ayuda a diseñar la configuración del sistema antes de implementarlo, asegurando que todos los componentes estén correctamente conectados.
- **Identificación de Dependencias**:
- Muestra las relaciones y dependencias entre el backend, frontend, y la base de datos, ayudando a detectar posibles puntos de fallo.
- **Optimización de Recursos**:
- Permite identificar oportunidades para mejorar el rendimiento o reducir costos en la infraestructura.

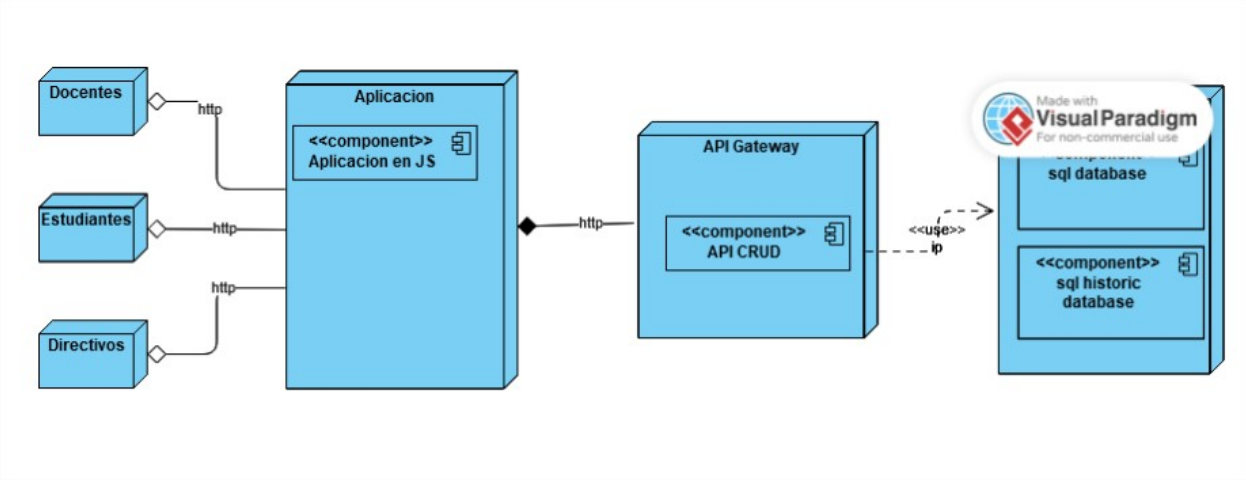


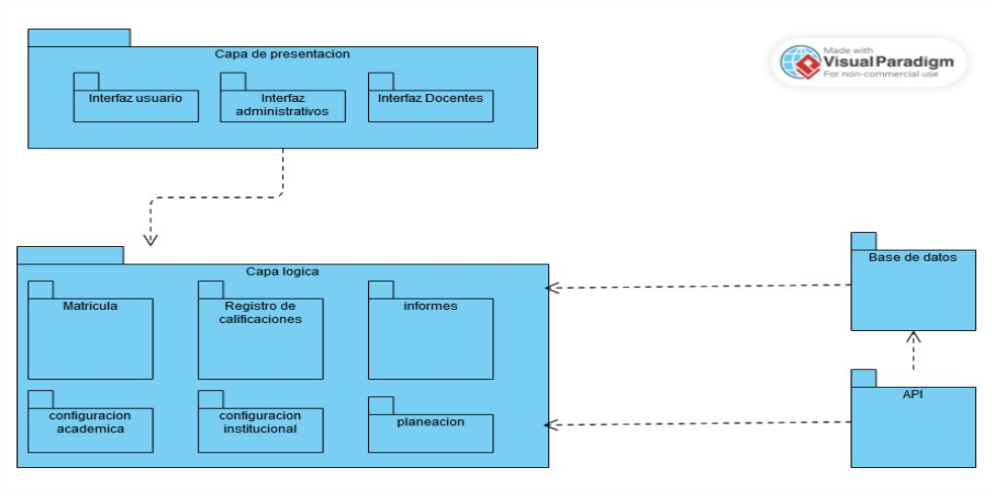
Diagrama de paquetes y componentes

El diagrama de componentes es un modelo UML que representa las partes físicas del sistema, como archivos, bibliotecas, módulos y sus relaciones. Muestra cómo las piezas del software interactúan entre sí para cumplir con los casos de uso.

Componentes Clave para este Software

Para el sistema de gestión académica, los principales componentes son:

1. **Flask Backend**:
   - `App.py`: Archivo principal que ejecuta el servidor Flask.
   - `Controladores`: Archivos que gestionan las rutas y solicitudes (`routes.py`, `calificaciones_controller.py`).
2. **Base de Datos**:
   - MySQL Workbench como sistema de almacenamiento.
   - Tablas para `Estudiantes`, `Grupos`, `Asignaturas`, `Calificaciones`, `Calendarios`.
3. **Frontend**:
   - HTML estático y JavaScript puro.
   - Archivos relacionados (`index.html`, `admin_dashboard.html`, `script.js`).
4. **Pymysql**:
   - Módulo que actúa como intermediario entre el backend y la base de datos.
5. **Módulos de Servicios**:
   - `services.py`: Contiene la lógica del negocio para operaciones como crear asignaturas o asignar calificaciones.



Componentes y estandares

Framework de Desarrollo:

HTML, Python: El sistema utiliza HTML5 para el desarrollo del front-end y python para el back-end. Esta combinación ofrece flexibilidad y escalabilidad en el desarrollo de aplicaciones web.

Estándares de Desarrollo:

HTML5: Se sigue el estándar HTML5 para la estructura y presentación de las páginas web. Esto garantiza la compatibilidad con navegadores modernos y el acceso a las últimas características web.

python Coding Standards:

Se adhieren a las convenciones y estándares de codificación recomendados para el lenguaje de programación python. Esto incluye prácticas de nomenclatura, organización de código y manejo de excepciones.

Modelo de Datos:

Se representa mediante diagramas de casos de uso, diagramas de componentes y diagramas de clases. Consulte la sección correspondiente para obtener detalles.

Instalacion

CONFIGURACIÓN

Instalación del Sistema Operativo:

Instala Windows Server 2019 en los servidores de aplicaciones.

Configuración del Hardware:

Asegúrate de cumplir con los requisitos mínimos y recomendados de hardware.

Instalación de Dependencias:

Instala python 3.12.

Instala MySQL 8.0.

Instala xampp

Configuración de MySQL:

ejecuta el script que se pone en la seccion de script de tablas

Despliegue de la Aplicación:

copia las carpetas que contienen la aplicación al fichero deseado luego ejecutas los siguientes comandos:

- Python -m venv venv esto creara un ambiente virtualesluego ejecutaras el comando
- venv/scripts/activate
- y finalmente pip install -r requirements.txt lo cual instalara las dependecias que se mencionan en el archivo de requirements
- **python app.py**

Configura el firewall para permitir el tráfico en los puertos utilizados por xampp

y MySQL.

Pruebas y Verificación:

Realiza pruebas para asegurarte de que la aplicación se despliega

correctamente y es accesible desde los navegadores compatibles.

Script de creacion de BD y tablas

```
-- -----------------------------------------------------
-- Schema proyecto
-- -----------------------------------------------------
-- -----------------------------------------------------
-- Schema proyecto
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `proyecto` DEFAULT CHARACTER SET utf8 ;
USE `proyecto` ;
-- -----------------------------------------------------
-- Table `proyecto`.`Alumnos`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `proyecto`.`Alumnos` (
`idAlumnos` INT NOT NULL AUTO_INCREMENT,
`Nombres` VARCHAR(45) NULL,
`Apellidos` VARCHAR(45) NULL,
`TipoDocumento` VARCHAR(45) NOT NULL,
`Documento` VARCHAR(45) NOT NULL,
`Sexo` VARCHAR(45) NULL,
`Nacionalidad` VARCHAR(45) NULL,
`Direccion` VARCHAR(45) NULL,
`foto` VARCHAR(45) NULL,
`NombreAcudiente` VARCHAR(45) NULL,
`DocumentoAcudiente` VARCHAR(45) NULL,
`FechaNacimiento` DATE NULL,
`Telefono` VARCHAR(45) NULL,
`CorreoElectronico` VARCHAR(45) NOT NULL,
PRIMARY KEY (`idAlumnos`))
ENGINE = InnoDB;
-- -----------------------------------------------------
-- Table `proyecto`.`Cursos`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `proyecto`.`Cursos` (
`idCurso` INT NOT NULL,
`NombreCurso` VARCHAR(45) NOT NULL,
`IdJornada` INT NOT NULL,
`anio` INT NOT NULL,4
```

```
PRIMARY KEY (`idCurso`))
ENGINE = InnoDB;
-- -----------------------------------------------------
-- Table `proyecto`.`Asignaturas`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `proyecto`.`Asignaturas` (
`idAsignaturas` INT NOT NULL,
`NombreAsignatura` VARCHAR(45) NOT NULL,
`Descripcion` VARCHAR(45) NULL,
`idNivel` INT NOT NULL,
`Creditos` INT NULL,
PRIMARY KEY (`idAsignaturas`))
ENGINE = InnoDB;
-- -----------------------------------------------------
-- Table `proyecto`.`Matriculas`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `proyecto`.`Matriculas` (
`idMatriculas` INT NOT NULL AUTO_INCREMENT,
`idCurso` INT NOT NULL,
`idAlumno` INT NOT NULL,
`anio` INT NOT NULL,
PRIMARY KEY (`idMatriculas`),
INDEX `fk_Matriculas_Alumnos_idx` (`idAlumno` ASC) VISIBLE,
INDEX `idCurso_idx` (`idCurso` ASC) VISIBLE,
CONSTRAINT `idAlumnos`
FOREIGN KEY (`idAlumno`)
REFERENCES `proyecto`.`Alumnos` (`idAlumnos`)
ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `idCurso`
FOREIGN KEY (`idCurso`)
REFERENCES `proyecto`.`Cursos` (`idCurso`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;
-- -----------------------------------------------------
-- Table `proyecto`.`Calificaciones`5
```

```
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `proyecto`.`Calificaciones` (
`idAlumno` INT NOT NULL,
`idCurso` INT NOT NULL,
`anio` INT NOT NULL,
`idAsignatura` INT NOT NULL,
`Nota` INT NULL,
`Desempenio` VARCHAR(45) NULL,
PRIMARY KEY (`idAlumno`),
INDEX `fk_Calificaciones_Cursos1_idx` (`idCurso` ASC) VISIBLE,
CONSTRAINT `fk_Calificaciones_Cursos1`
FOREIGN KEY (`idCurso`)
REFERENCES `proyecto`.`Cursos` (`idCurso`)
ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `fk_Calificaciones_idAlumno`
FOREIGN KEY (`idAlumno`)
REFERENCES `proyecto`.`Matriculas` (`idAlumno`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;
-- -----------------------------------------------------
-- Table `proyecto`.`Roles`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `proyecto`.`Roles` (
`idRol` INT NOT NULL,
`descripcion` VARCHAR(45) NULL,
PRIMARY KEY (`idRol`))
ENGINE = InnoDB;
-- -----------------------------------------------------
-- Table `proyecto`.`Docente`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `proyecto`.`Docente` (
`idDocente` INT NOT NULL,
`Nombre` VARCHAR(45) NULL,
`Apellidos` VARCHAR(45) NULL,
`Documento` VARCHAR(45) NOT NULL,
`TipoDocumento` VARCHAR(45) NOT NULL,
`Direccion` VARCHAR(45) NULL,6
```

```
`Email` VARCHAR(45) NOT NULL,
`Correo` VARCHAR(45) NULL,
`idRol` INT NOT NULL,
PRIMARY KEY (`idDocente`),
INDEX `fk_Docente_Roles1_idx` (`idRol` ASC) VISIBLE,
CONSTRAINT `fk_Docente_Roles1`
FOREIGN KEY (`idRol`)
REFERENCES `proyecto`.`Roles` (`idRol`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;
-- -----------------------------------------------------
-- Table `proyecto`.`AsignacionAcademica`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `proyecto`.`AsignacionAcademica` (
`Cursos_idCurso` INT NOT NULL,
`Asignaturas_idAsignaturas` INT NOT NULL,
`idDocente` INT NOT NULL,
`anio` INT NOT NULL,
`idCurso` INT NOT NULL,
PRIMARY KEY (`Cursos_idCurso`, `Asignaturas_idAsignaturas`),
INDEX `fk_Cursos_has_Asignaturas_Asignaturas1_idx` (`Asignaturas_idAsignaturas` ASC)
VISIBLE,
INDEX `fk_Cursos_has_Asignaturas_Cursos1_idx` (`Cursos_idCurso` ASC) VISIBLE,
INDEX `idDocente_idx` (`idDocente` ASC) VISIBLE,
CONSTRAINT `fk_Cursos_has_Asignaturas_Cursos1`
FOREIGN KEY (`Cursos_idCurso`)
REFERENCES `proyecto`.`Cursos` (`idCurso`)
ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `fk_Cursos_has_Asignaturas_Asignaturas1`
FOREIGN KEY (`Asignaturas_idAsignaturas`)
REFERENCES `proyecto`.`Asignaturas` (`idAsignaturas`)
ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `idDocente`
FOREIGN KEY (`idDocente`)
REFERENCES `proyecto`.`Docente` (`idDocente`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;7
```

```
-- -----------------------------------------------------
-- Table `proyecto`.`Jornadas`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `proyecto`.`Jornadas` (
`idJornada` INT NOT NULL,
`Jornadas` VARCHAR(45) NULL,
PRIMARY KEY (`idJornada`),
CONSTRAINT `fk_Jornadas_Cursos1`
FOREIGN KEY (`idJornada`)
REFERENCES `proyecto`.`Cursos` (`idCurso`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;
SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

Software base del sistema y preerequisitos

Requerimientos Mínimos y Recomendados de Hardware:
Disco Duro: Mínimo 20 GB, Recomendado 50 GB
Memoria RAM: Mínimo 8 GB, Recomendado 16 GB
Procesador: Mínimo Dual-core, Recomendado Quad-core
Requerimientos Mínimos de Software:
Sistema Operativo: Windows 10 o superior
Lenguajes de Programación: HTML5, python
Servidor Web: Apache HTTP Server 2.4
Navegadores Compatibles: Google Chrome, Mozilla Firefox, Microsoft Edge (Versiones Recientes)
Base de Datos: MySQL 8.0
Sistema Operativo de los Servidores:
Servidores de Aplicación y Base de Datos: Windows Server 2019
Servidores de Bases de Datos Admitidos:
MySQL 8.0
Servidores de Aplicación:
xampp
Navegadores Compatibles y su Versión:
Google Chrome (Versión Reciente)
Mozilla Firefox (Versión Reciente)
Microsoft Edge (Versión Reciente)
Lenguajes de Programación Utilizados en el Desarrollo:
HTML5 para el Front-end