Universidad Mexiquense del Bicentenario Almoloya de Alquisiras

Ingeniería en Sistemas Computacionales

Manual Técnico SchoolWallet

Equipo: Nexora

Integrantes:

Adriel Larios Hernandez

Danna Yahzee Rosas Hinojosa

Juan Carlos Sanchez Avila

Alex Castillo Reynoso

Docente: Ángel Hernández Hernández

18/09/2025

**Indice**

[Introducción 4](#_Toc209072779)

[1.1 Descripción general del proyecto 5](#_Toc209072780)

[1.1.1 Nombre del proyecto: SchoolWallet v1.0 5](#_Toc209072781)

[1.1.2 Funciones principales 5](#_Toc209072782)

[1.1.3 Requerimientos previos 5](#_Toc209072783)

[1.1.3.1 Hardware 5](#_Toc209072784)

[1.1.4 Conocimientos previos 6](#_Toc209072785)

[**2. Instrucciones de uso / Procedimientos** 6](#_Toc209072786)

[2.1 Crear un Wallet en Open Pyments 6](#_Toc209072787)

[**2.2 Instalación de Node.js** *2.2.1 Instalación de Node.js* 8](#_Toc209072788)

[*2.2.2 Verificación de versiones instaladas* **2.3 Instalación de MySQL / PostgreSQL** *2.3.1 Configuración inicial de la base de datos* 8](#_Toc209072789)

[3. Descarga del repositorio o proyecto base 8](#_Toc209072790)

[3. Creación y configuración de cuentas en Interledger 8](#_Toc209072791)

[4. Integración de wallets con SchoolWallet 8](#_Toc209072792)

[5. Uso del sistema SchoolWallet 9](#_Toc209072793)

[6. Mantenimiento y resolución de problemas 9](#_Toc209072794)

[6.1. Indicaciones para dar soporte o mantenimiento 9](#_Toc209072795)

[6.3. Buenas prácticas de soporte 11](#_Toc209072796)

[Glosario 11](#_Toc209072797)

[Referencias 11](#_Toc209072798)

[Control de versiones 11](#_Toc209072799)

# Introducción

El presente manual técnico tiene como objetivo proporcionar una guía clara, estructurada y accesible sobre el funcionamiento, uso y mantenimiento de la plataforma SchoolWallet, un sistema innovador diseñado para optimizar los procesos de pago en instituciones educativas y pequeños comercios. A través de esta herramienta, se busca facilitar transacciones rápidas, seguras y transparentes, eliminando el uso de efectivo y reduciendo la dependencia de intermediarios costosos como bancos o terminales físicas. Asimismo, este manual servirá como referencia tanto para la instalación inicial y configuración del sistema, como para la capacitación de los usuarios encargados de su operación cotidiana.

El alcance de este documento se centra en describir los aspectos técnicos y operativos fundamentales de SchoolWallet, incluyendo su arquitectura tecnológica, los procedimientos de implementación, el uso de la interfaz web y la interacción con la API de pagos mediante el estándar Open Payments. No obstante, cabe aclarar que este manual no abarca procesos administrativos internos de las instituciones, ni se enfoca en la capacitación detallada de usuarios finales como estudiantes o clientes. Su orientación es eminentemente técnica, dirigida a administradores, desarrolladores y personal de soporte, quienes serán los responsables de garantizar el correcto funcionamiento de la plataforma.

En cuanto al público objetivo, este manual está dirigido principalmente a tres grupos:

Administradores de sistemas: responsables de la configuración inicial, el mantenimiento preventivo y correctivo, así como de la seguridad de los datos.

Desarrolladores e integradores: encargados de adaptar, escalar o personalizar la solución de acuerdo con las necesidades de cada institución o comercio.

Personal operativo de soporte: quienes requieren instrucciones prácticas para monitorear la plataforma, resolver incidencias y garantizar la continuidad del servicio.

Finalmente, el contenido de este manual se organiza de manera progresiva para facilitar su comprensión. Se inicia con una descripción general de la problemática y la propuesta de valor de SchoolWallet, seguida por una explicación de la solución técnica a alto nivel. Posteriormente, se incluyen las instrucciones de uso y procedimientos operativos, acompañados de diagramas y ejemplos prácticos. Además, se dedica un apartado a la resolución de problemas comunes, así como a las recomendaciones de seguridad y mantenimiento. El manual concluye con anexos, glosario de términos y referencias, lo que permitirá al lector contar con un panorama integral para el uso eficaz de la plataforma.

## Descripción general del proyecto

* + 1. Nombre del proyecto: SchoolWallet v1.0  
       El sistema se denomina **SchoolWallet v1.0**, una plataforma web desarrollada con tecnologías modernas y basada en el estándar de pagos **Open Payments**. Su diseño modular y escalable permite atender las necesidades de instituciones educativas y pequeños comercios, brindando una solución confiable y adaptable a diferentes entornos.

### Funciones principales

* SchoolWallet ha sido concebido como un sistema integral de pagos digitales que cubre las siguientes funciones:
* Gestión de transacciones seguras y rápidas, permitiendo pagos digitales sin necesidad de efectivo ni terminales físicas.
* Interoperabilidad con múltiples billeteras digitales, gracias a la implementación del estándar Open Payments.
* Administración de usuarios, productos e inventarios, con actualización en tiempo real.
* Generación de reportes y notificaciones para administradores, padres de familia o responsables de comercio.
* Escalabilidad en servicios, integrando procesos de pago en cafeterías, papelerías, uniformes y eventos escolares.
* Trazabilidad y transparencia, registrando cada transacción en la base de datos para facilitar auditorías y control financiero.
* Seguridad avanzada, con autenticación de usuarios y cifrado de datos durante las operaciones.

### Requerimientos previos

### Hardware

* Servidor o computadora con procesador de al menos 2 núcleos (Intel i5 o equivalente).
* 8 GB de memoria RAM mínima.
* 100 GB de almacenamiento disponible en disco.
* Conexión estable a internet (mínimo 20 Mbps).

### Software

* Sistema operativo compatible: Windows 10/11, Linux (Ubuntu Server recomendado) o macOS.
* Entorno de ejecución Node.js para el backend.
* Gestor de base de datos MySQL o PostgreSQL.
* Navegador web actualizado (Chrome, Firefox, Edge o Safari) para acceso al frontend.
* Certificado SSL para garantizar la seguridad de las transacciones (HTTPS).

#### Software

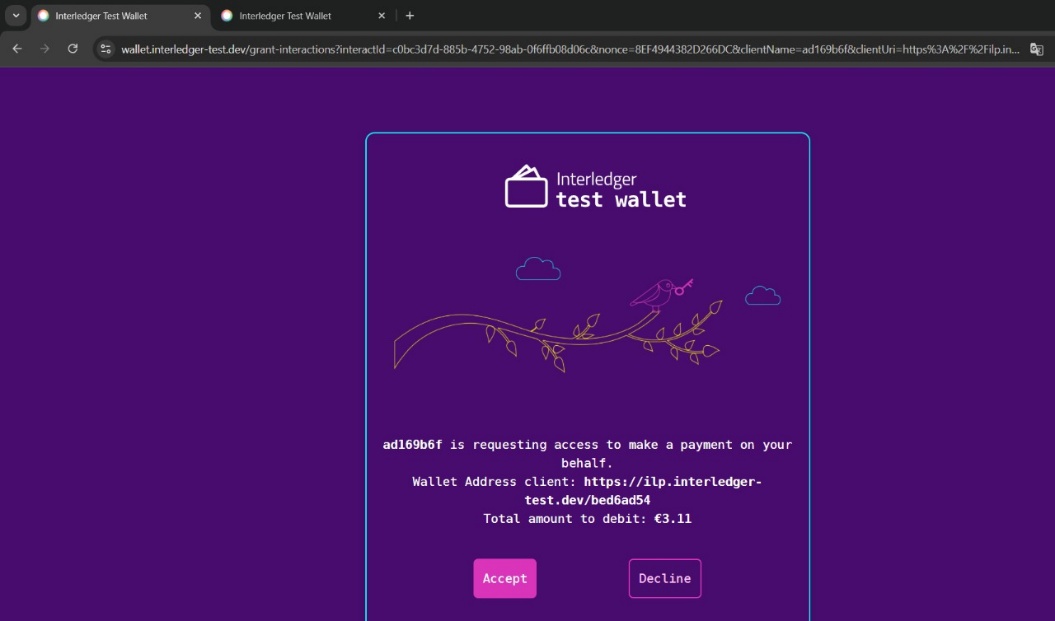
* Equipo de cómputo para administradores y personal operativo.
* Dispositivos móviles o computadoras con acceso a internet para los usuarios finales.

### Conocimientos previos

* Conocimientos básicos de administración de sistemas y redes.
* Manejo de bases de datos relacionales.
* Familiaridad con conceptos de seguridad informática y gestión de accesos.
* Comprensión general de plataformas de pago digital e integración de APIs.

## **2. Instrucciones de uso / Procedimientos**

### 2.1 Crear un Wallet en Open Pyments



* **Registrarse como usuario wallet / creador de wallet**

1. Accede al portal correspondiente que implemente Open Payments (por ejemplo, un proveedor de wallet compatible).
2. Crear una cuenta nueva: nombre de usuario, correo electrónico, contraseña, etc.
3. Verificar tu identidad si el sistema lo exige (correo de confirmación, verificación de número, etc.).

* **Obtener los credenciales necesarios**

1. Al registrarte, te deben dar un *wallet address* (dirección de cartera) compatible con Open Payments.
2. Puede que también te proporcionen tokens de acceso (API key / secret) si tu wallet se va a usar programáticamente.

* **Configurar variables de entorno / parámetros de conexión**

En la configuración de tu aplicación (por ejemplo, tu backend), guarda los credenciales recibidas:

* + Dirección de la wallet
  + API key / secret
  + Endpoint base de la API de Open Payments que use tu wallet provider
* **Realizar una prueba de conexión / creación**

1. Usa la API para consultar tu wallet recién creada, asegurarte de que responde (por ejemplo, hacer “GET /wallets/{tu-wallet-address}”).
2. Probar saldo si aplica, o verificar que se pueden recibir solicitudes de pago.

* **Configurar permisos o políticas de seguridad**

1. Definir límites (si la wallet permite límites de transferencia, límites diarios, etc.).
2. Verificar que solo usuarios autorizados puedan usar la wallet.
3. Asegurar que las transacciones estén cifradas y que la comunicación sea sobre HTTPS.

* **Registrar la wallet en tu sistema (SchoolWallet u otro)**

1. En tu aplicación, almacenar la información relevante de la wallet: dirección, credenciales, tipo de wallet.
2. Permitir que desde el sistema se puedan enviar solicitudes de pago y recibir confirmaciones, usando la wallet.

* **Pruebas funcionales finales**

1. Enviar una solicitud de pago de prueba para ver si la wallet la recibe.
2. Hacer la simulación de aceptación de pago.
3. Verificar que las notificaciones / webhooks / callbacks llegan si tu wallet los soporta.

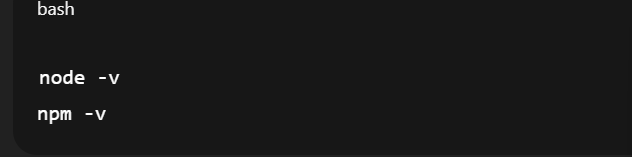
* **Registro de mantenimiento y monitoreo**

1. Guardar logs de las operaciones relacionadas con la wallet.
2. Monitorear usos inusuales, errores o fallos de conexión.
3. Tener alertas si algo falla.

### 2.2 **Instalación de Node.js**

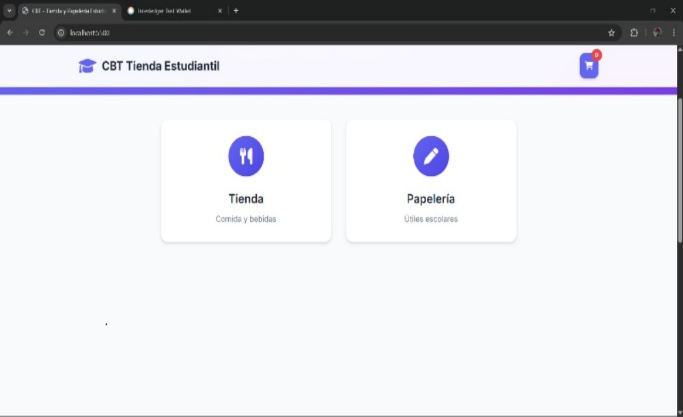
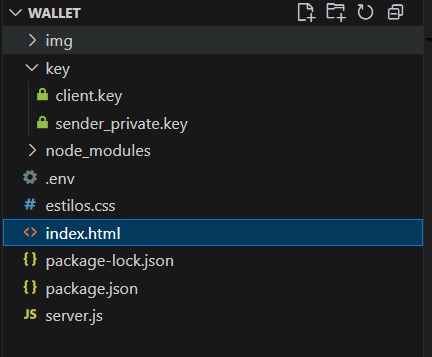
### En ****Windows****

1. Entra a 👉 <https://nodejs.org/es>
2. Descarga la versión **LTS (recomendada)**.
3. Abre el instalador y da **Siguiente → Aceptar → Instalar**.
4. Al terminar, abre **CMD** y escribe:



2.3 Instalación de Visual Studio Code

2.4 Creacion de la Pagina Web



# **3. Indicaciones para dar soporte o mantenimiento**

El sistema **SchoolWallet** requiere un mantenimiento preventivo y correctivo que asegure su correcto funcionamiento a lo largo del tiempo. A continuación, se presentan las principales actividades recomendadas:

**a) Mantenimiento preventivo**

* **Actualizaciones periódicas**: Verificar que la versión de **Node.js**, dependencias (npm update) y base de datos estén actualizadas para evitar vulnerabilidades.
* **Respaldo de datos**: Realizar copias de seguridad automáticas de la base de datos (MySQL/PostgreSQL) al menos una vez por semana.
* **Monitoreo de logs**: Revisar los registros de errores generados por el backend (logs/error.log) para detectar fallos tempranos.
* **Seguridad**: Renovar certificados SSL antes de su vencimiento y mantener configuradas las reglas de firewall y autenticación segura.
* **Espacio en disco**: Comprobar periódicamente que el servidor tenga suficiente almacenamiento para transacciones y reportes.

**b) Mantenimiento correctivo**

* **Errores en el backend**: Reiniciar el servidor con npm start o usar un gestor de procesos como **PM2** para garantizar disponibilidad.
* **Errores en la base de datos**: Ejecutar herramientas de reparación y optimización (mysqlcheck o comandos de PostgreSQL).
* **Fallas en la integración con Open Payments**: Validar credenciales en el archivo .env y regenerar tokens de autenticación en caso de expiración.
* **Problemas de acceso de usuarios**: Revisar permisos en la base de datos y credenciales almacenadas en el sistema.

## **3.1 Resolución de problemas comunes**

**Problema 1: El servidor no arranca**

* Verificar que Node.js esté correctamente instalado con node -v.
* Comprobar que todas las dependencias estén instaladas: npm install.
* Revisar el archivo .env para asegurarse de que las variables de conexión estén completas.

**Problema 2: No hay conexión con la base de datos**

* Confirmar que el servicio de MySQL/PostgreSQL esté en ejecución.
* Validar usuario, contraseña, host y puerto configurados.
* Probar la conexión con una herramienta externa (ejemplo: MySQL Workbench).

**Problema 3: Fallo al procesar un pago**

* Revisar conexión con la API de **Open Payments**.
* Validar que la wallet digital utilizada sea compatible.
* Consultar los logs para identificar el código de error devuelto por la API.

**Problema 4: La página web no carga**

* Revisar que el frontend esté en ejecución y accesible desde el navegador.
* Limpiar caché del navegador y recargar.
* Verificar que el puerto del servidor no esté ocupado por otra aplicación.

## **3.2. Buenas prácticas de soporte**

* Documentar cada incidencia resuelta en un registro de soporte.
* Mantener una lista de credenciales y configuraciones en un repositorio seguro.
* Establecer un plan de escalamiento: el personal de soporte atiende problemas básicos y, si no se resuelven, se escalan al área de desarrollo.
* Programar mantenimientos fuera del horario de uso intensivo (ejemplo: fines de semana).

# Glosario

# Glosario

IDE (Integrated Development Environment)

Definición: Entorno de desarrollo integrado que proporciona herramientas para programar, como editor de código, depurador y control de versiones, todo en un mismo software. VS Code funciona como un IDE ligero y extensible.

Extensión (Extension)

Definición: Complemento que se instala en VS Code para agregar funcionalidades adicionales, como soporte para un lenguaje de programación, herramientas de depuración o integración con Git.

Depurador (Debugger)

Definición: Herramienta que permite ejecutar un programa paso a paso, inspeccionar variables y encontrar errores de manera más eficiente.

Terminal Integrada (Integrated Terminal)

Definición: Consola de comandos incorporada dentro de VS Code que permite ejecutar scripts, compilar programas o usar control de versiones sin salir del editor.

Workspace (Espacio de trabajo)

Definición: Carpeta o conjunto de carpetas abiertas en VS Code, donde se organizan proyectos, archivos y configuraciones específicas de desarrollo.

# Referencias

Link para la creación de la Wallet: <https://wallet.interledger-test.dev/>

Link para la descarga de Visual Studio Code: <https://code.visualstudio.com/>

Link para la descarga de Node.js: <https://nodejs.org/es/download/>

Canvas Open Payments API: <https://www.canva.com/design/DAGyFc1JYig/HH-bJQRxg_SPv8uIkjcJlw/edit>

Link de Platzi: <https://platzi.com/home/>

# Control de versiones

Versión: Beta del Software **SchoolWallet v1.0**,

**Introduction**  
This technical manual aims to provide a clear, structured, and accessible guide on the operation, use, and maintenance of the SchoolWallet platform, an innovative system designed to optimize payment processes in educational institutions and small businesses. Through this tool, it seeks to facilitate fast, secure, and transparent transactions, eliminating the use of cash and reducing reliance on costly intermediaries such as banks or physical terminals. Likewise, this manual will serve as a reference both for the initial installation and configuration of the system, as well as for the training of users responsible for its daily operation.

The scope of this document focuses on describing the fundamental technical and operational aspects of SchoolWallet, including its technological architecture, implementation procedures, the use of the web interface, and interaction with the payment API through the Open Payments standard. However, it should be clarified that this manual does not cover internal administrative processes of the institutions, nor does it focus on detailed training for end users such as students or customers. Its orientation is eminently technical, directed at administrators, developers, and support staff, who will be responsible for ensuring the correct functioning of the platform.

Regarding the target audience, this manual is primarily aimed at three groups:

* **System administrators:** responsible for the initial configuration, preventive and corrective maintenance, as well as data security.
* **Developers and integrators:** responsible for adapting, scaling, or customizing the solution according to the needs of each institution or business.
* **Operational support staff:** who require practical instructions to monitor the platform, resolve incidents, and ensure service continuity.

Finally, the content of this manual is organized progressively to facilitate understanding. It begins with a general description of the problem and the value proposition of SchoolWallet, followed by an explanation of the technical solution at a high level. Subsequently, usage instructions and operational procedures are included, accompanied by diagrams and practical examples. Additionally, a section is dedicated to the resolution of common problems, as well as security and maintenance recommendations. The manual concludes with appendices, a glossary of terms, and references, allowing the reader to have a comprehensive overview for the effective use of the platform.

**1.1 Project Overview**  
**1.1.1 Project Name: SchoolWallet v1.0**  
The system is called **SchoolWallet v1.0**, a web platform developed with modern technologies and based on the Open Payments standard. Its modular and scalable design allows it to meet the needs of educational institutions and small businesses, providing a reliable solution that can adapt to different environments.

**1.1.2 Main Functions**

* SchoolWallet has been conceived as a comprehensive digital payment system covering the following functions:
* Management of secure and fast transactions, allowing digital payments without the need for cash or physical terminals.
* Interoperability with multiple digital wallets, thanks to the implementation of the Open Payments standard.
* Administration of users, products, and inventories, with real-time updates.
* Generation of reports and notifications for administrators, parents, or business managers.
* Scalability in services, integrating payment processes in cafeterias, stationery shops, uniforms, and school events.
* Traceability and transparency, recording every transaction in the database to facilitate audits and financial control.
* Advanced security, with user authentication and data encryption during operations.

**Operations**  
**1.1.3 Prerequisites**

**1.1.3.1 Hardware**

* Server or computer with at least a dual-core processor (Intel i5 or equivalent).
* Minimum 8 GB of RAM.
* 100 GB of available disk storage.
* Stable internet connection (minimum 20 Mbps).

**1.1.3.2 Software**

* Compatible operating system: Windows 10/11, Linux (Ubuntu Server recommended), or macOS.
* Node.js runtime environment for the backend.
* MySQL or PostgreSQL database management system.
* Updated web browser (Chrome, Firefox, Edge, or Safari) for frontend access.
* SSL certificate to ensure transaction security (HTTPS).

**Software for users**

* Computers for administrators and operational staff.
* Mobile devices or computers with internet access for end users.

**1.1.4 Previous Knowledge**

* Basic knowledge of system and network administration.
* Experience with relational database management.
* Familiarity with IT security concepts and access control.
* General understanding of digital payment platforms and API integration.

**2. Instructions for Use / Procedures**

**2.1 Create a Wallet in Open Payments**

**Register as a wallet user / wallet creator**

1. Access the portal provided by a compatible Open Payments provider.
2. Create a new account: username, email, password, etc.
3. Verify your identity if required (confirmation email, phone verification, etc.).

**Obtain the necessary credentials**

1. Upon registration, you should receive a wallet address compatible with Open Payments.
2. You may also be provided with access tokens (API key / secret) if your wallet will be used programmatically.

**Configure environment variables / connection parameters**  
In your application configuration (e.g., backend), store the received credentials:

* Wallet address
* API key / secret
* Base API endpoint of the Open Payments provider used

**Perform a connection / creation test**

1. Use the API to query your newly created wallet and ensure it responds (e.g., “GET /wallets/{your-wallet-address}”).
2. Test the balance if applicable, or verify that payment requests can be received.

**Configure permissions or security policies**

1. Define limits (if the wallet allows transfer limits, daily limits, etc.).
2. Ensure that only authorized users can use the wallet.
3. Ensure transactions are encrypted and communications are over HTTPS.

**Register the wallet in your system (SchoolWallet or other)**

1. Store relevant wallet information in your application: address, credentials, wallet type.
2. Allow sending payment requests and receiving confirmations through the system using the wallet.

**Final functional tests**

1. Send a test payment request to see if the wallet receives it.
2. Simulate payment acceptance.
3. Verify that notifications / webhooks / callbacks are received if supported by your wallet.

**Maintenance and monitoring log**

1. Save logs of wallet-related operations.
2. Monitor unusual usage, errors, or connection failures.
3. Set alerts if something fails.

**2.2 Node.js Installation**

**On Windows**

1. Go to 👉 https://nodejs.org/en
2. Download the LTS version (recommended).
3. Open the installer and click Next → Accept → Install.
4. When finished, open CMD and type:

**2.3 Visual Studio Code Installation**

**2.4 Web Page Creation**

**3. Guidelines for Support or Maintenance**

The SchoolWallet system requires preventive and corrective maintenance to ensure its proper functioning over time. Below are the main recommended activities:

**a) Preventive Maintenance**

* **Periodic updates:** Verify that Node.js, dependencies (npm update), and the database are up to date to avoid vulnerabilities.
* **Data backup:** Perform automatic backups of the database (MySQL/PostgreSQL) at least once a week.
* **Log monitoring:** Review backend error logs (logs/error.log) to detect early failures.
* **Security:** Renew SSL certificates before expiration and maintain configured firewall rules and secure authentication.
* **Disk space:** Periodically check that the server has enough storage for transactions and reports.

**b) Corrective Maintenance**

* **Backend errors:** Restart the server using npm start or use a process manager like PM2 to ensure availability.
* **Database errors:** Run repair and optimization tools (mysqlcheck or PostgreSQL commands).
* **Integration failures with Open Payments:** Validate credentials in the .env file and regenerate authentication tokens if expired.
* **User access problems:** Review permissions in the database and credentials stored in the system.

**3.1 Common Problem Resolution**

**Problem 1: Server does not start**

* Verify that Node.js is correctly installed using node -v.
* Ensure all dependencies are installed: npm install.
* Check the .env file to ensure connection variables are complete.

**Problem 2: No connection to the database**

* Confirm that the MySQL/PostgreSQL service is running.
* Validate configured username, password, host, and port.
* Test the connection using an external tool (e.g., MySQL Workbench).

**Problem 3: Payment processing fails**

* Check the connection with the Open Payments API.
* Validate that the digital wallet being used is compatible.
* Consult logs to identify the error code returned by the API.

**Problem 4: Web page does not load**

* Ensure that the frontend is running and accessible from the browser.
* Clear the browser cache and reload.
* Verify that the server port is not occupied by another application.

**3.2 Best Support Practices**

* Document each resolved incident in a support log.
* Maintain a list of credentials and configurations in a secure repository.
* Establish an escalation plan: support staff handle basic issues, and if they are not resolved, escalate them to the development team.
* Schedule maintenance outside peak usage hours (e.g., weekends).

**Glossary**

* Definition of technical terms used in the manual.