



UNIVERSIDAD
AUTÓNOMA DE
ENCARNACIÓN | SEDE
COLONIAS
UNIDAS



AÑO INTERNACIONAL DE LA
EDUCACIÓN
para el desarrollo de la
CIENCIA Y LA TECNOLOGÍA
UNA E - IREDE 2015



UNIVERSIDAD
AUTÓNOMA DE
ENCARNACIÓN



Universidad Autónoma de Encarnación

Sede Colonias Unidas

Licenciatura en Análisis de Sistemas Informáticos

Informe Final de 200 horas de Pasantía Profesional en la empresa

Eno Bronstrup S.A.

Alumno: Luis Mario Morinigo Bracho

Profesor: Lic. Pedro Danieli

3°Curso – 5° Semestre

2025





Índice

Índice de ilustraciones	5
Índice de tablas	7
Introducción	8
1. Descripción de la empresa	9
1.1. Nombre de la empresa	9
1.2. Ubicación geográfica	9
1.3. Descripción de la empresa	9
1.4. Reseña Histórica	9
1.5. Misión	10
1.6. Visión	10
1.7. Estructura organizacional	10
2. Descripción del área de pasantía	11
2.1. Área de pasantía seleccionada	11
2.2. Análisis de ventajas y desventajas	11
2.3. Descripción del problema	12
3. Objetivos y metas de la pasantía	13
3.1. Objetivo general	13
3.2. Objetivos específicos	13
4. Actividades desarrolladas	14
5. Marco metodológico	16
5.1. Diseño de la investigación	16
5.2. Herramientas de investigación	16
5.3. Solución planteada	17
5.4. Alcances del proyecto	17
5.5. Limitaciones del proyecto	17
5.6. Selección del proyecto de desarrollo	18
5.7. Cronograma general del proyecto	18
6. Entorno de desarrollo	19
6.1. Requerimientos técnicos	19
6.2. Herramientas para el plan de desarrollo	19
6.2.1. Gestión de proyecto	19
6.2.2. Metodología de desarrollo	19



6.3.	Selección del entorno de desarrollo	20
6.4.	Costo del proyecto.....	21
6.5.	Tipo de desarrollo seleccionado.....	21
6.6.	Requisito del desarrollo	23
6.6.1.	Diseño de interfaces o prototipos.....	23
6.6.2.	Modelos de datos	34
6.7.	Código Fuente.....	38
6.7.1.	HTML	38
6.7.2.	Python	62
6.7.3.	JavaScript	74
6.	Estructura general del proyecto	75
6.7.	Inicio de sesión	75
6.8.	Pantalla Principal	76
6.9.	Pantalla de Marcación	76
6.10.	Pantalla de Dispositivos	77
6.11.	Pantalla de Agregar Dispositivo	77
6.12.	Pantalla Editar Dispositivo	78
6.13.	Pantalla Empleados	78
6.14.	Pantalla de Agregar Empleados.....	79
6.15.	Pantalla Editar Empleado.....	79
6.16.	Pantalla Usuarios.....	80
6.17.	Pantalla Agregar Usuario	81
6.18.	Pantalla Editar Usuario	81
6.19.	Pantalla Departamentos.....	82
6.20.	Pantalla Agregar Departamento.....	82
6.21.	Pantalla Editar Departamento	83
6.22.	Pantalla Horarios	83
6.23.	Pantalla Agregar Horario	84
6.24.	Pantalla Editar Horario.....	84
6.25.	Cerrar Sesión	85
7.	Conclusión	86
8.	Anexos	87
	Certificado de aprobación	87



Formulario información de la empresa.....	88
Nota de solicitud de pasantía.....	89
Carta compromiso de pasantía	90
Plan de trabajo pasantía.....	91
Formulario de evaluación (tutor externo).....	92
Formulario de evaluación (tutor académico).....	93
Dispositivo ZKteco	94
Github.....	94
Bibliografía.....	95



Índice de ilustraciones

Ilustración 1 Ubicación Geográfica de la empresa	9
Ilustración 2 Estructura Organizacional de EBSA	10
Ilustración 3 Cronograma de Actividades desarrolladas p1	14
Ilustración 4 Cronograma de Actividades desarrolladas p2	15
Ilustración 5 Prototipo de pantalla de iniciar sesión.....	23
Ilustración 6 Prototipo de pantalla principal del sistema	24
Ilustración 7 Prototipo de pantalla de marcaciones	25
Ilustración 8 Prototipo de pantalla de dispositivos.....	26
Ilustración 9 Prototipo de pantalla de empleados.....	27
Ilustración 10 Prototipo de pantalla de departamentos.....	28
Ilustración 11 Prototipo de pantalla de horarios.....	29
Ilustración 12 Prototipo de pantalla de usuarios.....	30
Ilustración 13 Prototipo de pantalla de reportes	31
Ilustración 14 Prototipo de pantalla de reportes marcaciones	32
Ilustración 15 Prototipo de pantalla de reportes empleados	33
Ilustración 16 Diagrama de casos de uso	34
Ilustración 17 Diagrama de flujos de datos	35
Ilustración 18 Diagrama entidad-relación.....	36
Ilustración 19 Estructura de la Base de datos	36
Ilustración 20 Tabla departamentos de la BD	36
Ilustración 21 Tabla de dispositivos de la BD	37
Ilustración 22 Tabla de empleados de la BD	37
Ilustración 23 Tabla de horarios de la BD.....	37
Ilustración 24 Tabla marcaciones de la BD	37
Ilustración 25 Tabla usuarios de la BD	38
Ilustración 26 Pantalla Iniciar Sesión del sistema AsistOK	75
Ilustración 27 Pantalla Principal del sistema AsistOK.....	76
Ilustración 28 Pantalla de Marcacion del sistema AsistOK	76
Ilustración 29 Pantalla de Dispositivos del sistema AsistOK.....	77
Ilustración 30 Pantalla de Agregar Dispositivos del sistema AsistOK	77
Ilustración 31 Pantalla Editar Dispositivo del sistema AsistOK	78
Ilustración 32 Pantalla Empleados del sistema AsistOK.....	78
Ilustración 33 Pantalla de Agregar Empleados del sistema AsistOK	79
Ilustración 34 Pantalla Editar Empleado del sistema AsistOK	79
Ilustración 35 Pantalla Usuarios con Admin del sistema AsistOK.....	80
Ilustración 36 Pantalla Usuarios con Usuario del sistema AsistOK.....	80
Ilustración 37 Pantalla Agregar Usuario del sistema AsistOK.....	81
Ilustración 38 Pantalla Editar Usuario del sistema AsistOK	81
Ilustración 39 Pantalla Departamentos del sistema AsistOK	82
Ilustración 40 Pantalla Agregar Departamento del sistema AsistOK.....	82
Ilustración 41 Pantalla Editar Departamento del sistema AsistOK.....	83
Ilustración 42 Pantalla Horarios del sistema AsistOK	83
Ilustración 43 Pantalla Agregar Horario del sistema AsistOK	84
Ilustración 44 Pantalla Editar Horario del sistema AsistOK	84
Ilustración 45 Pantalla Cerrar Sesión del sistema AsistOK	85
Ilustración 46 Certificado de aprobación	87



Ilustración 47 Formulario información de la empresa	88
Ilustración 48 Nota de solicitud de pasantía	89
Ilustración 49 Carta compromiso de pasantía	90
Ilustración 50 Plan de trabajo pasantía	91
Ilustración 51 Formulario de evaluación (tutor externo).....	92
Ilustración 52 Formulario de evaluación (tutor académico)	93
Ilustración 53 Dispositivo ZKteco	94



Índice de tablas

Tabla 1 Cronograma general del proyecto	18
Tabla 2 Tabla costos del proyecto.....	21



Introducción

El presente informe detalla la experiencia de una pasantía de 200 horas realizada en Eno Bronstrup S.A., una destacada empresa en Bella Vista, Departamento de Itapúa, Paraguay. Durante este periodo, se profundizó en la dinámica operativa del departamento de técnicos informáticos, lo que permitió no solo la identificación de una necesidad crítica sino también el inicio y avance significativo en el desarrollo de una solución tecnológica.

El objetivo central de esta pasantía fue aplicar de manera práctica los conocimientos técnicos y de gestión de proyectos, contribuyendo a la optimización de los procesos internos de la empresa. Este documento abarca desde la descripción general de la institución y el análisis del área de pasantía, hasta el detalle del marco metodológico, las actividades desarrolladas, el entorno de desarrollo y la estructura del sistema propuesto.

Se espera que el trabajo realizado sienta las bases para una implementación exitosa del Sistema de Marcación para el Control de Asistencia, demostrando el valor de la automatización y la eficiencia en la gestión de recursos humanos.

1. Descripción de la empresa

1.1. Nombre de la empresa

La empresa seleccionada para la realización de la pasantía se denomina Eno Bronstrup S.A.

1.2. Ubicación geográfica

La empresa Eno Bronstrup, tiene ubicado su sede en la Avenida marcial Samaniego km 4.5. Bella Vista, Departamento de Itapúa, Paraguay.



Ilustración 1 Ubicación Geográfica de la empresa

1.3. Descripción de la empresa

Eno Bronstrup S.A. es una empresa establecida en Bella Vista, dedicada principalmente a la venta de bienes y servicios, mayormente de Yerba Mate. Su operación se enfoca en satisfacer las necesidades de sus clientes a través de sus diversos productos. La empresa se distingue por la calidad de sus productos. (Selecta, s.f.)

1.4. Reseña Histórica

Eno Bronstrup, los primeros pasos que dio la empresa fue en el año 1942, cuando Don Reinaldo y esposa (descendientes europeos) decidieron producir Yerba Mate. (S.A), s.f.) Inicialmente se dedicaron a la plantación, posteriormente adquirieron una barbacuá y en el año 1.950 resolvieron completar el ciclo de producción lanzando al mercado la yerba elaborada con la marca Selecta. (Selecta, s.f.)



2. Descripción del área de pasantía

2.1. Área de pasantía seleccionada

La pasantía se llevó a cabo en el área administrativa, específicamente dentro del departamento Técnico Informático de Eno Bronstrup S.A. Este departamento es el pilar tecnológico de la empresa, brindando soporte esencial para la infraestructura informática, los sistemas de comunicación y las aplicaciones de software que sustentan todas las operaciones. Las responsabilidades del equipo incluyen la gestión de redes, el mantenimiento de hardware y software, la seguridad de la información, el soporte técnico a usuarios finales y el desarrollo o adaptación de soluciones tecnológicas internas. La interacción constante con otras áreas de la empresa permitió una visión integral de sus necesidades tecnológicas.

2.2. Análisis de ventajas y desventajas

A Partir de las observaciones realizadas durante la pasantía en el departamento técnicos informáticos se elaboró un análisis FODA para identificar las fortalezas, debilidades, oportunidades y amenazas del área:

Fortalezas:

- Ambiente de trabajo productivo y eficiente.
- Colaboración y ayuda mutua entre los miembros del equipo.
- Disposición de los profesionales a compartir conocimientos.
- Conocimiento técnico del personal en diversas áreas de la informática.

Debilidades:

- Procesos de comunicación interna susceptible de mejora.
- Oportunidad de implementar una capacitación más estructurada para nuevos integrantes.
- Dependencia de conocimientos tecitos en algunos procedimientos.

Oportunidades:

- Implementación de sistemas para automatizar tareas y mejorar la eficiencia.
- Adopción de nuevas tecnologías para optimizar la gestión de la infraestructura informática.





- Formalización de la documentación de procedimientos para facilitar la capacitación y la continuidad operativa.

Amenazas:

- Riesgos de seguridad informática si los sistemas no mantienen actualizados.
- Pérdida de conocimiento crítico si no se documentan los procedimientos clave.
- Dificultad para escalar el soporte técnico si la empresa crece sin una planificación adecuada.
- Descripción del problema principal detectado en el área de pasantía.

2.3. Descripción del problema

El problema principal identificado durante la pasantía, y que fue el foco del proyecto de desarrollo, se centra en la gestión semiautomatizada pero altamente ineficiente del control de asistencia de los empleados. Si bien la empresa Eno Bronstrup S.A. ya utilizaba un dispositivo para la captura de marcaciones de entrada y salida, el proceso de gestión de estos datos presentaba deficiencias significativas:

- Procesamiento Manual y Repetitivo de Datos: Los datos crudos capturados del dispositivo de marcación debían ser extraídos y convertidos manualmente a un formato de tabla en Excel. Este proceso es repetitivo, consume una cantidad considerable de tiempo administrativo y desvía recursos valiosos de tareas más estratégicas.
- Susceptibilidad a Errores en la Conversión y Manipulación: Durante la extracción y transformación de los datos del dispositivo a Excel, existe un alto riesgo de introducir errores humanos, ya sea por omisión, duplicación o transcripción incorrecta. Esto puede llevar a inexactitudes en los registros de asistencia.
- Dificultad en la Generación de Informes Avanzados: La generación de informes específicos se convierte en una tarea tediosa y propensa a errores al tener que manipular manualmente grandes volúmenes de datos en hojas de cálculo. Cada informe requeriría un procesamiento manual adicional.

Esta problemática afecta directamente la eficiencia del departamento de Recursos Humanos y la precisión en el cálculo de la nómina, lo que justifica la necesidad urgente de un sistema integral que automatice no solo la captura de datos, sino también su almacenamiento, gestión y la generación de informes.



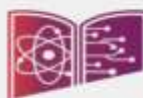
3. Objetivos y metas de la pasantía

3.1. Objetivo general

Aplicar los conocimientos teóricos y prácticos en el desarrollo de software y gestión de proyectos, contribuyendo a la optimización de los procesos de control de asistencia para la empresa. mediante el diseño, desarrollo parcial y propuesta de implementación de un sistema de marcación automatizado.

3.2. Objetivos específicos

- Identificar y analizar las necesidades específicas de la empresa en cuanto al control de asistencia.
- Diseñar la arquitectura y los componentes principales de un sistema de marcación de asistencia.
- Desarrollar prototipos funcionales de los módulos clave del sistema (ej. módulo de marcación, módulo de gestión de empleados).
- Establecer la estructura de la base de datos para el almacenamiento eficiente de la información de asistencia.



4. Actividades desarrolladas





		UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN		SEDE COLONIAS UNIDAS		AÑO INTERNACIONAL DE LA EDUCACIÓN para el desarrollo de la CIENCIA Y LA TECNOLOGÍA UNAE - IREDE 2015
Cronograma de Actividades de Pasantía						
Fecha	Hora inicio	Hora salida	Total horas	Actividades diarias	Responsable	
24/03/25	13:00:00	17:30:00	04:30:00	Realización de consultas sql y análisis del área	 Fernando Ferreira	
25/03/25	13:00:00	17:30:00	04:30:00	Realización de consultas y bloqueos SQL		
26/03/25	13:00:00	17:30:00	04:30:00	Se realizó mantenimiento de swich de un sector		
27/03/25	13:00:00	17:30:00	04:30:00	Se realizó mantenimiento de swich de un sector		
28/03/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL		
31/03/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL		
01/04/25	13:00:00	17:30:00	04:30:00	Se realizó mantenimiento a impresoras de un sector y se realizo consultas y bloqueos SQL		
02/04/25	13:00:00	17:30:00	04:30:00	Se realizó mantenimiento en el depósito de informática		
03/04/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL		
04/04/25	13:00:00	17:30:00	04:30:00	Se realizó documentación de informes		
07/04/25	13:00:00	17:30:00	04:30:00	Se realizó mantenimiento de laptops de un sector		
08/04/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL		
09/04/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL		
10/04/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL		
11/04/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL		
14/04/25	13:00:00	17:30:00	04:30:00	Se realizo mantenimiento en el depósito de informática		
15/04/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL		
21/04/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL		
22/04/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL		
23/04/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL y hubo una charla sobre seguridad informática por parte de TIGO Business		
24/04/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL		
25/04/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL		
28/04/25	13:00:00	17:30:00	04:30:00	Se realizó mantenimiento y limpieza de deposito de informática y laptops		
29/04/25	13:00:00	17:30:00	04:30:00	Se realizó mantenimiento y limpieza de deposito de informática		
30/04/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL y mantenimiento en el depósito de informática		
02/05/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL		
 UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN Sede Colonias Unidas: Guillermo Closs c/ Aviadores del Chaco, Hohenau, Paraguay ☎0775 232 608 ☎(0983) 796 259 🌐www.unae.edu.py						

Ilustración 3 Cronograma de Actividades desarrolladas p1





05/05/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL
06/05/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL
07/05/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL
08/05/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL y diseño del sistema del proyecto
09/05/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL y diseño del sistema del proyecto
12/05/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL y diseño del sistema del proyecto
13/05/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL y diseño del sistema del proyecto
16/05/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL y diseño de módulos del sistema del proyecto
19/05/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL y diseño de módulos del sistema del proyecto
20/05/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL y diseño de módulos del sistema del proyecto
21/05/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL y diseño de módulos del sistema del proyecto
22/05/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL y diseño de módulos del sistema del proyecto
23/05/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL y diseño de módulos del sistema del proyecto
26/05/25	13:00:00	17:30:00	04:30:00	Se realizo Se realizo consultas y bloqueos SQL y diseño de módulos del sistema del proyecto y mantenimiento de switch de un sector
27/05/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL y diseño de módulos del sistema del proyecto
28/05/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL y diseño de módulos del sistema del proyecto
29/05/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL y diseño de módulos del sistema del proyecto
30/05/25	13:00:00	17:30:00	04:30:00	Se realizo consultas y bloqueos SQL y diseño de módulos del sistema del proyecto

Guillermo Closs



Ilustración 4 Cronograma de Actividades desarrolladas p2

5. Marco metodológico

5.1. Diseño de la investigación

El diseño de la investigación para el presente proyecto de pasantía se enmarcó dentro de un enfoque cualitativo, basado específicamente en un diseño de observación descriptiva. Este enfoque fue crucial para comprender en profundidad los procesos actuales de control de asistencia y las necesidades específicas del departamento.

Se recopiló información valiosa mediante la observación directa de los procesos existentes, prestando especial atención a cómo se capturaban los datos del dispositivo de marcación y cómo se transformaban en tablas de Excel. Asimismo, la participación activa en las actividades diarias del departamento Técnico Informático y la interacción con el área de operaciones permitieron identificar de primera mano los desafíos operativos y los puntos críticos relacionados con la gestión de la asistencia.

Adicionalmente, se realizaron entrevistas informales y consultas directas con supervisores y compañeros de trabajo. Estas interacciones fueron fundamentales para obtener una comprensión más profunda de las prácticas actuales, las expectativas sobre una solución automatizada y los desafíos específicos que enfrenta el personal en la entrega de servicios y la gestión de la información de asistencia. Esta combinación de observación y consulta permitió una visión holística del problema y fundamentó la propuesta de solución tecnológica.

5.2. Herramientas de investigación

Para la recopilación de información y el entendimiento profundo del problema, se utilizaron las siguientes herramientas de investigación:

- Entrevistas Semiestructuradas: Conversaciones con personal clave de los departamentos de Recursos Humanos y Administración, así como con los técnicos informáticos, para entender a fondo los procesos actuales, los puntos débiles y las expectativas sobre una solución automatizada.
- Observación Participante: El pasante se integró en la rutina del departamento de TI, observando directamente cómo se gestionaba la asistencia y los desafíos diarios.
- Análisis Documental: Revisión de informes existentes y documentos relacionados con la gestión de personal para comprender el flujo de información.
- Lluvia de Ideas: Sesiones informales con el tutor de la pasantía y otros colegas para generar ideas sobre posibles soluciones y características del sistema.

5.3. Solución planteada

La solución propuesta es el desarrollo e implementación de un Sistema de Marcación para el Control de Asistencia. Este sistema busca reemplazar los métodos manuales por un proceso digitalizado y eficiente, con los siguientes componentes principales:

- Base de Datos Relacional (MySQL): Un repositorio centralizado para almacenar de manera segura y estructurada todas las marcaciones, datos de empleados y configuraciones del sistema.
- Aplicación Web (Python/Flask): Una plataforma accesible desde cualquier navegador web para la administración completa del sistema. Esta incluirá módulos para:
 - Gestión de empleados.
 - Visualización y auditoría de marcaciones.
 - Generación de informes de asistencia (horas trabajadas, tardanzas, ausencias).
 - La capacidad de importar Informes a formato PDF
 - Configuración de parámetros del sistema.
 - Gestión de usuarios y roles de acceso a la aplicación.

5.4. Alcances del proyecto

El alcance inicial de este proyecto se centraría en:

- Desarrollo de los módulos de software esenciales (marcación, gestión de empleados, gestión de marcaciones, gestión de horarios, informes básicos).
- Implementación del sistema en un área piloto de la empresa para pruebas y ajustes.
- Capacitación básica del personal involucrado en el uso del sistema.
- Generación de informes de asistencia para el cálculo de horas trabajadas, las tardanzas y ausencias.

5.5. Limitaciones del proyecto

El sistema en cuestión está limitado a ciertos factores excluyendo las siguientes funcionalidades avanzadas que podrían considerarse en futuras etapas:

- Gestión de Horas Extras o Vacaciones Complejas: Aunque se registrarán las horas, el cálculo automatizado y complejo de horas extras con diferentes tarifas o la gestión de solicitudes y aprobaciones de vacaciones está fuera de este alcance.
- Control de Acceso Físico a Instalaciones: El sistema se enfoca en el control de asistencia laboral, no en la apertura de puertas o control de acceso a áreas restringidas de la empresa.

- Geolocalización o Seguimiento Móvil: La solución se basa en marcadores físicos en las instalaciones, sin funcionalidades de geolocalización o seguimiento de empleados a través de dispositivos móviles.
- Personalización Extrema de Informes: Los informes serán configurables por filtros (fecha, empleado, departamento), pero no permitirán la creación de informes ad-hoc o con lógica de negocio compleja a través de la interfaz de usuario.

5.6. Selección del proyecto de desarrollo

El proyecto de desarrollo de un sistema de marcación para el control de asistencia fue seleccionado por las siguientes razones:

- Relevancia para la Empresa: Responde a una necesidad operativa real y tangible que impacta directamente en la eficiencia administrativa.
- Impacto Potencial: Una solución exitosa puede generar ahorros de tiempo, reducir errores y mejorar la precisión de los datos.
- Recursos Disponibles: Los recursos (hardware básico, software de desarrollo de código abierto) son accesibles para llevar a cabo la fase de prototipado.

5.7. Cronograma general del proyecto

Tabla 1 Cronograma general del proyecto

Fecha	Actividad
15 de abril – 30 abril	<ul style="list-style-type: none">- Definición de los requisitos- Desarrollo del Diagrama Casos de Usos- Desarrollo del Diagrama Entidad-Relación- Investigación de tecnologías a utilizar
01 de mayo – 30 de mayo	<ul style="list-style-type: none">- Desarrollo de los módulos del sistema- Desarrollo de la BD- Diseño de la Interfaz de Usuario
01 de junio – 15 de junio	<ul style="list-style-type: none">- Pruebas Unitarias del sistema- Identificación de errores- Depuración de errores y optimización
16 de junio – 18 de junio	<ul style="list-style-type: none">- Entrega del proyecto finalizado y documentación- Presentación de los resultados obtenidos



6. Entorno de desarrollo

6.1. Requerimientos técnicos

Equipos informativos: Laptop o computador de escritorio.

Sistema Operativo: Windows 8 o versiones posteriores.

Servidor: XAMPP.

Red: Tener conexión a internet.

Hardware de Marcación: Disponer de un dispositivo ZKteco. (zkteco, s.f.)

6.2. Herramientas para el plan de desarrollo

6.2.1. Gestión de proyecto

Para la gestión del proyecto de desarrollo del Sistema de Marcación, se utilizaron diversas herramientas y enfoques:

- Trello: Como herramienta visual de gestión de proyectos ágiles, Trello permitió organizar las tareas en listas (pendientes, en progreso, en pruebas, completadas), asignar responsabilidades, establecer fechas de vencimiento y añadir descripciones detalladas con checklists y adjuntos. (trello, s.f.)
- Git: Git fue la herramienta elegida para el control de versiones del código fuente. Esto permite gestionar los cambios, colaborar y revertir a versiones anteriores si es necesario. (git, s.f.)
- Reuniones de Seguimiento: Sesiones periódicas con el tutor de la pasantía y el equipo de TI para revisar avances, discutir desafíos y realinear objetivos.

6.2.2. Metodología de desarrollo

Para el desarrollo del sistema, se adoptó una metodología ágil, con un enfoque iterativo e incremental. Esto permitió flexibilidad para adaptarse a los requisitos emergentes y entregar prototipos funcionales en etapas tempranas. Aunque no se implementó un marco Scrum completo debido a la naturaleza individual de la pasantía, se aplicaron principios como:

- Desarrollo Iterativo: El proyecto se dividió en pequeñas iteraciones (semanas o quincenas), donde se planificaban, desarrollaban y probaban pequeñas porciones de funcionalidad.



- Entrega Incremental: Cada iteración buscaba entregar un incremento funcional del sistema.
- Feedback Continuo: Se buscó retroalimentación constante del tutor y de los usuarios clave para validar los avances y realizar ajustes.
- Documentación Just-in-Time: La documentación se generó a medida que el proyecto avanzaba, priorizando la claridad y la relevancia para la implementación.

6.3. Selección del entorno de desarrollo

Hardware: laptop Acer Nitro 5, Windows 11

Lenguaje Backend: Python: es uno de los lenguajes de programación más demandados en todo el mundo, debido a su excelente combinación entre sencillez y versatilidad. (Godaddy, s.f.)

Balsamiq Mockup: Es una solución de wireframing rápida que permite a los equipos colaborar, hacer maquetas, versiones de control, y ejecutar la prueba de usuario. (appvizer, s.f.)

Framework Web Backend: Flask: Es un framework para el desarrollo de proyectos web en Python. (arsys, arsys.es, s.f.)

Base de Datos: MySQL Workbench: Es una herramienta visual unificada para arquitectos de bases de datos, desarrolladores y administradores de bases de datos. (mysql, s.f.)

Servidor: XAMPP: es una distribución de Apache completamente gratuita y fácil de instalar que contiene MariaDB, PHP y Perl. (apachefriends, s.f.)

SQL: es un lenguaje de programación estandarizado y específico de dominio que sobresale en el manejo de relaciones de datos. (IBM, s.f.)

Frontend: HTML: Es el lenguaje estándar para crear y diseñar páginas web. Proporciona la estructura y disposición del contenido, lo que incluye texto, imágenes, enlaces y multimedia. (wix, s.f.)

Frontend: CSS: Es un lenguaje que se usa para estilizar elementos escritos en un lenguaje de marcado como HTML. (hostinger, s.f.)

Frontend: Bootstrap: Es un framework de CSS para estilos y componentes responsivos. (bootstrap, s.f.)



Frontend: JavaScript: Es un lenguaje de programación de alto nivel que se usa para hacer que las páginas web sean interactivas y dinámicas. (unir.net, s.f.)

Entorno de Desarrollo Integrado (IDE): Visual Studio Code: es un editor de código para programadores gratuito, de código abierto y multiplataforma. (arsys, arsys.es, s.f.)

6.4. Costo del proyecto

Tabla 2 Tabla costos del proyecto

Equipos de informática	5.000.000 Gs.
Sistemas Operativo	Incluido al equipo informático
Software de desarrollo	Licencia gratuita
Costo de traslado	100.000 Gs.
Viáticos	500.000 Gs.
Desarrollador	3.000.000 Gs.
Mantenimiento y Capacitación	150.000 Gs al mes
TOTAL	8.750.000 Gs.

6.5. Tipo de desarrollo seleccionado

Para abordar la problemática de la gestión de asistencia en Eno Bronstrup S.A., se optó por el desarrollo de una aplicación web a medida, denominada AsistOK. Este sistema está diseñado específicamente para automatizar y optimizar el control de asistencia de los empleados, funcionando como una plataforma centralizada y dinámica para el registro, la gestión y la consulta de las marcaciones. Permite, de manera eficiente, registrar entradas y salidas, administrar la información de los empleados, y generar informes detallados y precisos de asistencia, superando las limitaciones del proceso manual actual de conversión a Excel.

El objetivo principal de AsistOK es solucionar los problemas de ineficiencia, propensión a errores y falta de acceso rápido a datos que se presentan en los métodos tradicionales de control de asistencia. Con el desarrollo de este programa, se busca establecer un sistema estándar, completo y adaptable a las necesidades de la empresa, eliminando la manipulación manual de datos y proporcionando una fuente de información confiable para la toma de decisiones.



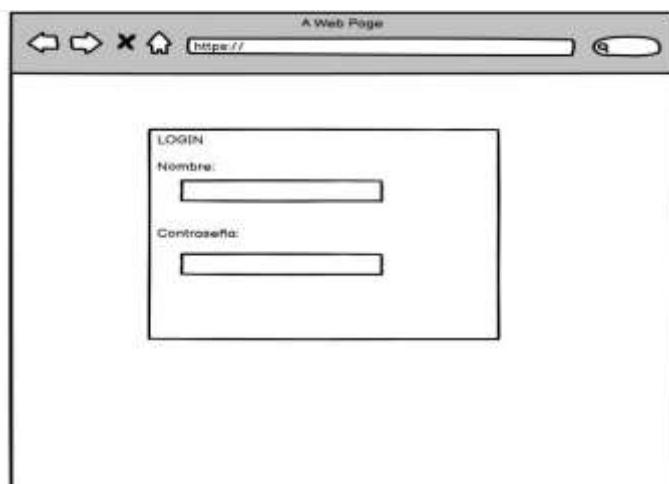
La elección de una aplicación web se basa en su inherente flexibilidad y accesibilidad, permitiendo que el personal administrativo y los supervisores accedan a los registros de asistencia desde cualquier dispositivo con conexión a internet, sin la necesidad de instalaciones complejas en cada puesto de trabajo. Utilizando tecnologías robustas como Python (Flask) para el backend y HTML, JavaScript, y CSS para el frontend, se garantiza un diseño atractivo, moderno y fácil de usar. Esto no solo mejora la experiencia del usuario, sino que también asegura que AsistOK sea una herramienta verdaderamente útil y eficiente para la gestión diaria del control de asistencia en la empresa.

6.6. Requisito del desarrollo

6.6.1. Diseño de interfaces o prototipos

Modulo LOGIN:

El usuario se conecta e ingresa a la pantalla de login al conectarse se le asocia con el usuario de la BD del sistema y dependiendo de su rol



A Web Page

https://

LOGIN

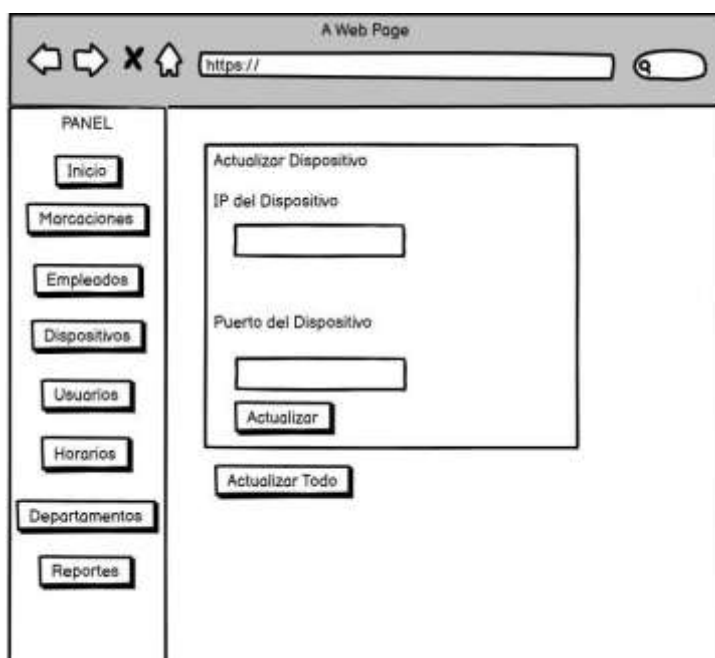
Nombre:

Contraseña:

Ilustración 5 Prototipo de pantalla de iniciar sesión

Modulo HOME:

Al logearse, se tendrá acceso a la barra lateral la cual le dará múltiples opciones como: marcaciones, dispositivos, empleados, usuarios (solo acceso el admin), departamentos, horarios y reportes. Para tener control sobre las siguientes pantallas al logearse y estar en el Home, va a existir un mecanismo(formulario) al interactuar que haga un proceso automático para que pase por cada dispositivo y capture los datos o elegir un dispositivo específico y los almacene en la BD o capturar los datos nuevos hacer que una vez capturada todos los datos, que después solo ingrese los nuevos y los agregar los nuevos y después seguir actualizando la tabla con nuevos datos y que cada dato que almacene en la tabla que lo asocie con el dispositivo que se capturo ese dato.



The image shows a web browser window titled "A Web Page" with a URL bar containing "https://". The main content area is divided into two sections. On the left is a vertical sidebar labeled "PANEL" containing a list of buttons: "Inicio", "Marcaciones", "Empleados", "Dispositivos", "Usuarios", "Horarios", "Departamentos", and "Reportes". The "Dispositivos" button is highlighted. On the right is a form titled "Actualizar Dispositivo". It contains two input fields: "IP del Dispositivo" and "Puerto del Dispositivo". Below these fields are two buttons: "Actualizar" and "Actualizar Todo".

Ilustración 6 Prototipo de pantalla principal del sistema

Modulo Marcaciones:

Esta pantalla va a mostrar todas las marcaciones recopiladas que están almacenado en la tabla de marcaciones de la BD, cuyos datos mostrara:

- id
- marcación
- tipo (Entrada o salida)
- a que empleado se asocio
- a que dispositivo va asociado

Se podrá filtrar

- por empleado, departamento, x tiempo.

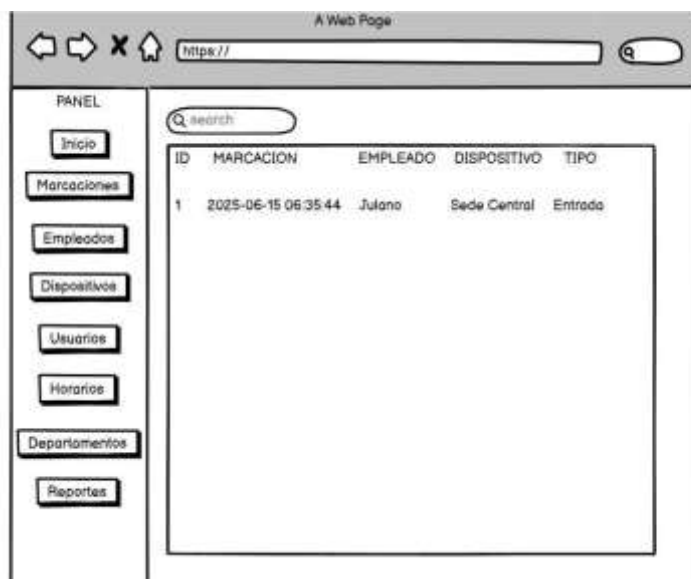


Ilustración 7 Prototipo de pantalla de marcaciones

Modulo Dispositivos:

Esta pantalla se utiliza para controlar los dispositivos, muestra un listado de los mismos de la tabla dispositivos de la BD, cuyos campos son: id, descripción, nombre, activo (si el dispositivo se encuentra en uso o no), puerto, IP.

También la opción (Botón) para acceder a las siguientes pantallas: editar, agregar.

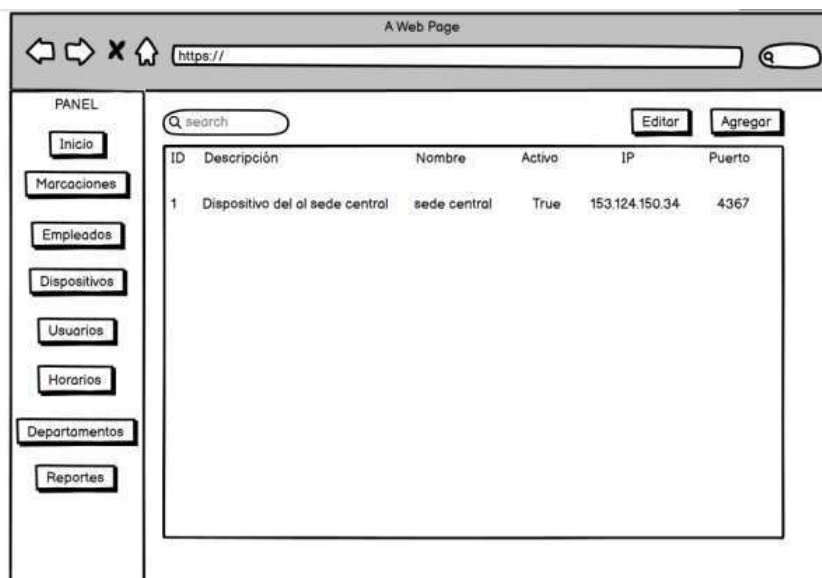


Ilustración 8 Prototipo de pantalla de dispositivos

Modulo Empleados:

Esta pantalla mostrara los datos en una tabla de los empleados almacenados en la tabla empleados de la BD, cuyos campos son: id, id (id del empleado en la marcación), nombre, activo, FK_departamento(nombre del departamento que este asignado) tendrá la opción de acceso a la pantalla de editar que podrá tener filtros de búsqueda como:

- filtrar por nombre, id, departamento.

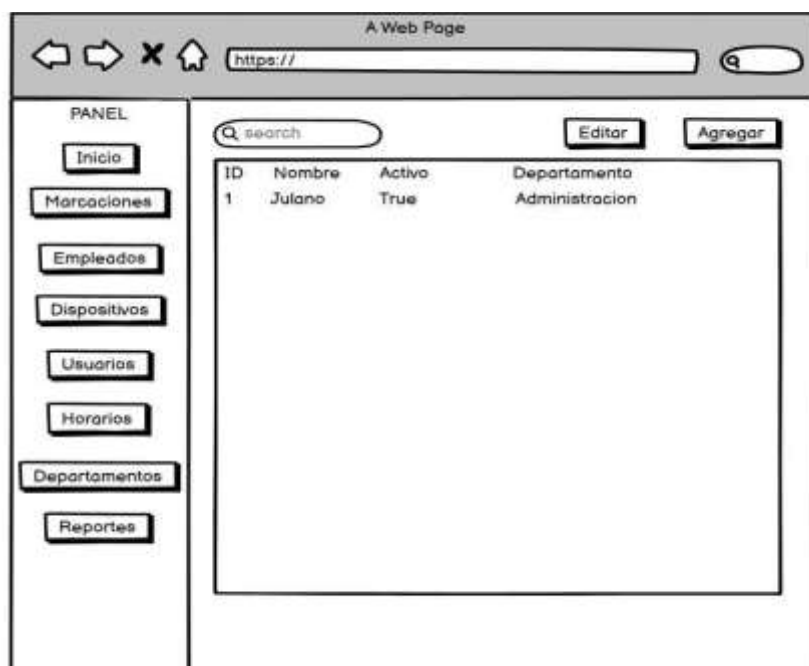


Ilustración 9 Prototipo de pantalla de empleados

Modulo Departamentos:

Esta pantalla mostrara los departamentos recopilados y almacenados en la tabla departamentos de la BD, cuyos campos son: ID, nombre, activo, horarios (a que horario está relacionado el departamento).

Se tendrá acceso a las siguientes pantallas: editar, agregar. Tendrá filtros de búsqueda:

- por nombre, etc.

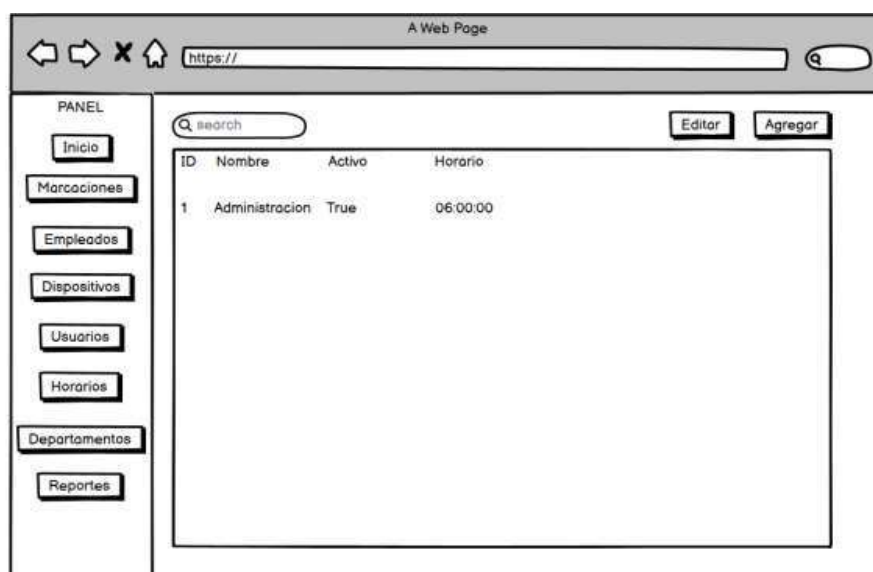


Ilustración 10 Prototipo de pantalla de departamentos

Modulo Horarios:

Esta pantalla mostrara los datos en una tabla de los horarios que están almacenados en la tabla horarios de la BD, cuyos campos son: id, nombre, entrada, salida, tolerancia y activo.

Se tendrá acceso a las siguientes pantallas: editar, agregar.

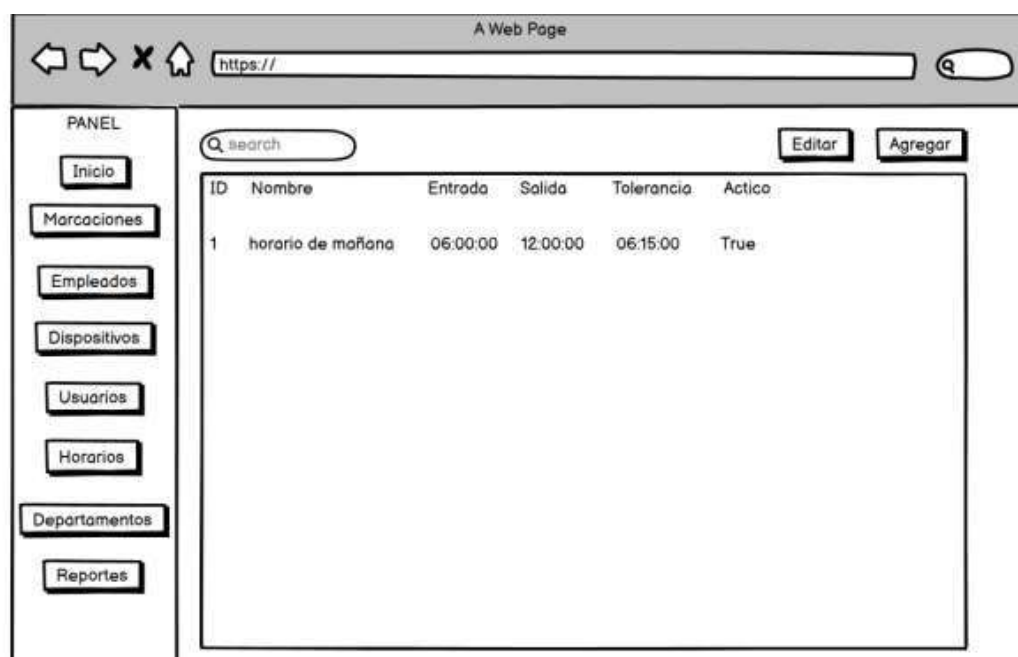


Ilustración 11 Prototipo de pantalla de horarios

Modulo Usuarios:

esta pantalla podrá mostrar usuarios de la tabla usuarios, que solo tendrá acceso el administrador del sistema que tendrá permitido editar y agregar usuarios a la tabla de la BD los usuarios tendrán roles de admin y user

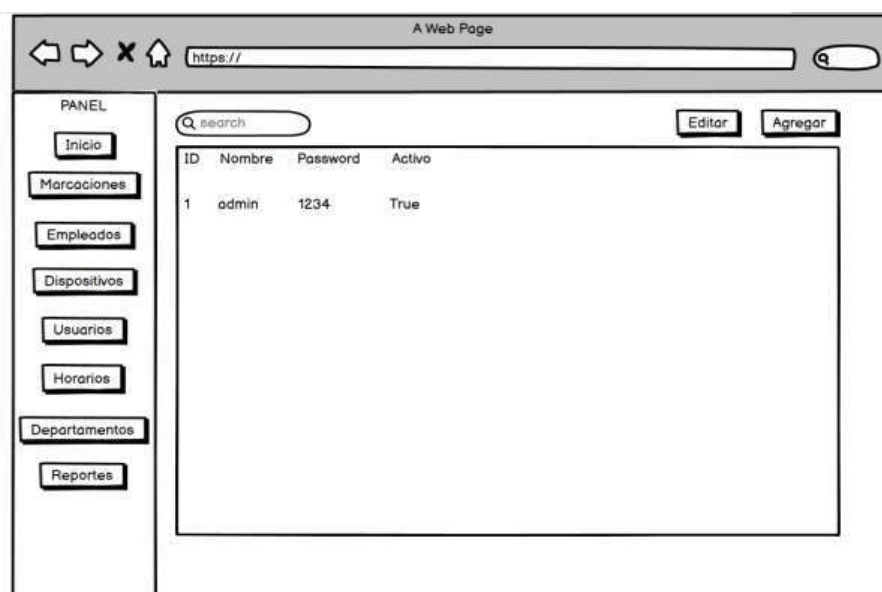


Ilustración 12 Prototipo de pantalla de usuarios

Modulo Reportes:

Esta pantalla mostrar opciones de los diferentes reportes que se podrán generar en el sistema: reporte_empleado, reporte_marcaciones también tendrá un historial de reportes, de los últimos reportes generados por PDF y que hora se generó y el tipo de reporte

Cuya pantalla dará la opción de crear un reporte ir a las diferentes pantallas de reporte:

-reporte_marcaciones

-reporte_empleado

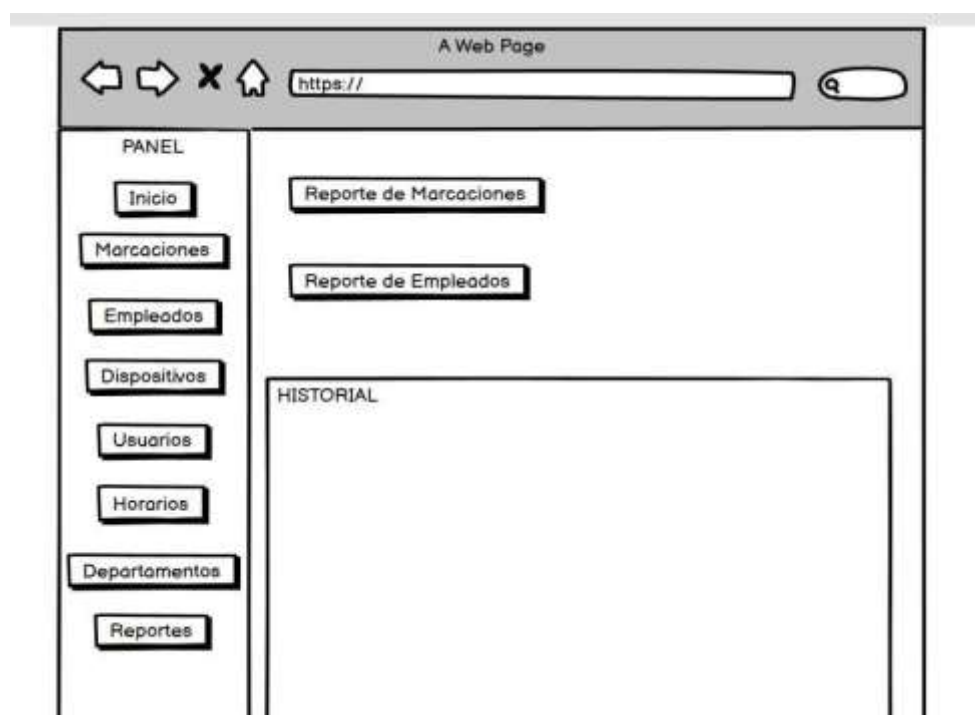


Ilustración 13 Prototipo de pantalla de reportes

Modulo Reporte de Marcaciones:

Esta pantalla mostrar un reporte en formato digital(tabla) cuyo reporte mostrara:

- id
- la marcación
- qué tipo de marcación es (entrada o salida)
- a que empleado está asociada dicha marcación
- a que dispositivo se asoció dicha captura de la marcación

Esto se podrá filtrar por diversas opciones y ordenar

- podrá filtrar por X tiempo (mes, semana, hora, año)
- podrá filtrar por departamento, empleado
- ordenado por paginación
- poder convertir a un formato PDF el reporte

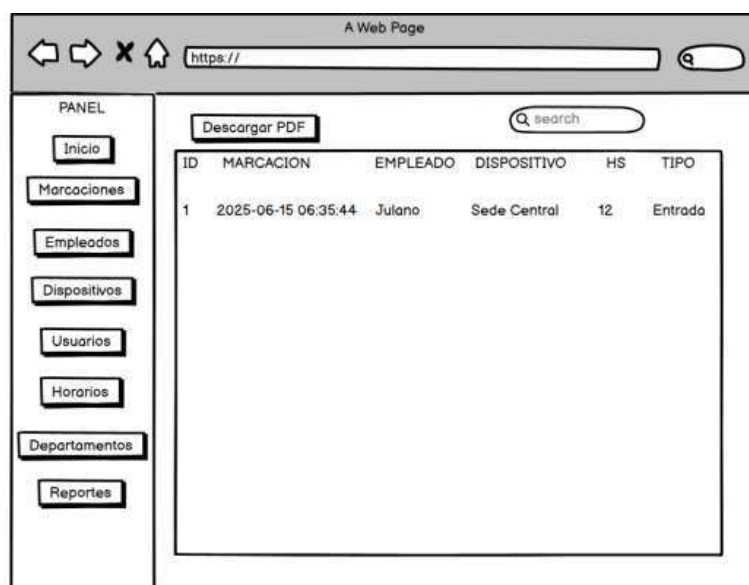


Ilustración 14 Prototipo de pantalla de reportes marcaciones

Modulo reporte empleados:

Esta pantalla tendrá la función de generar un reporte en formato digital(tabla) cuyo reporte mostrará:

- el id del empleado
- el nombre del empleado
- a que departamento está asociado
- mostrara las horas extra que hizo el empleado
- mostrara las horas o días ausentes del empleado
- total de horas trabajadas por X tiempo

Se tendrá un filtrado de búsqueda

- por departamento, id, nombre.

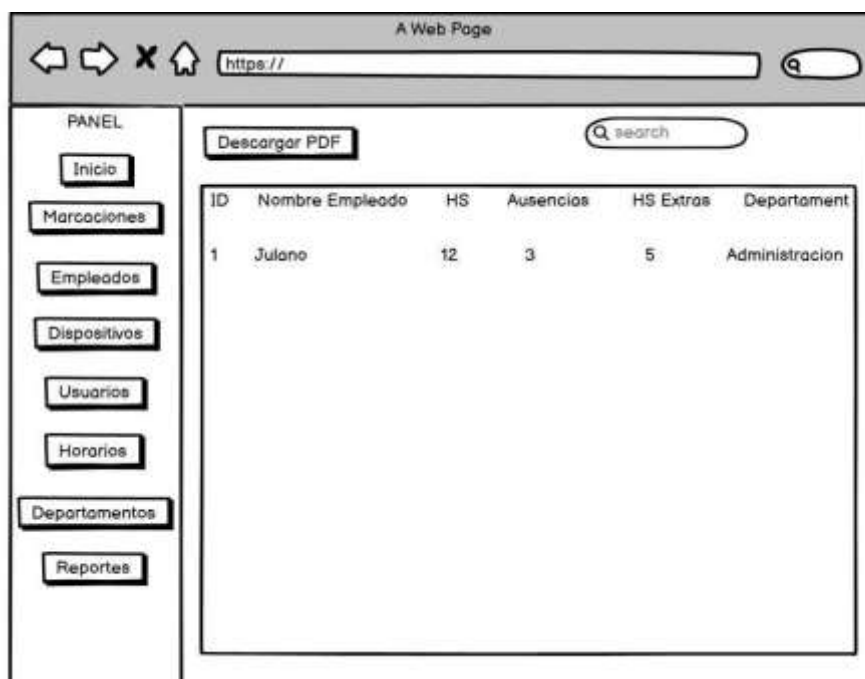


Ilustración 15 Prototipo de pantalla de reportes empleados

6.6.2. Modelos de datos

- Diagrama de Casos de Usos

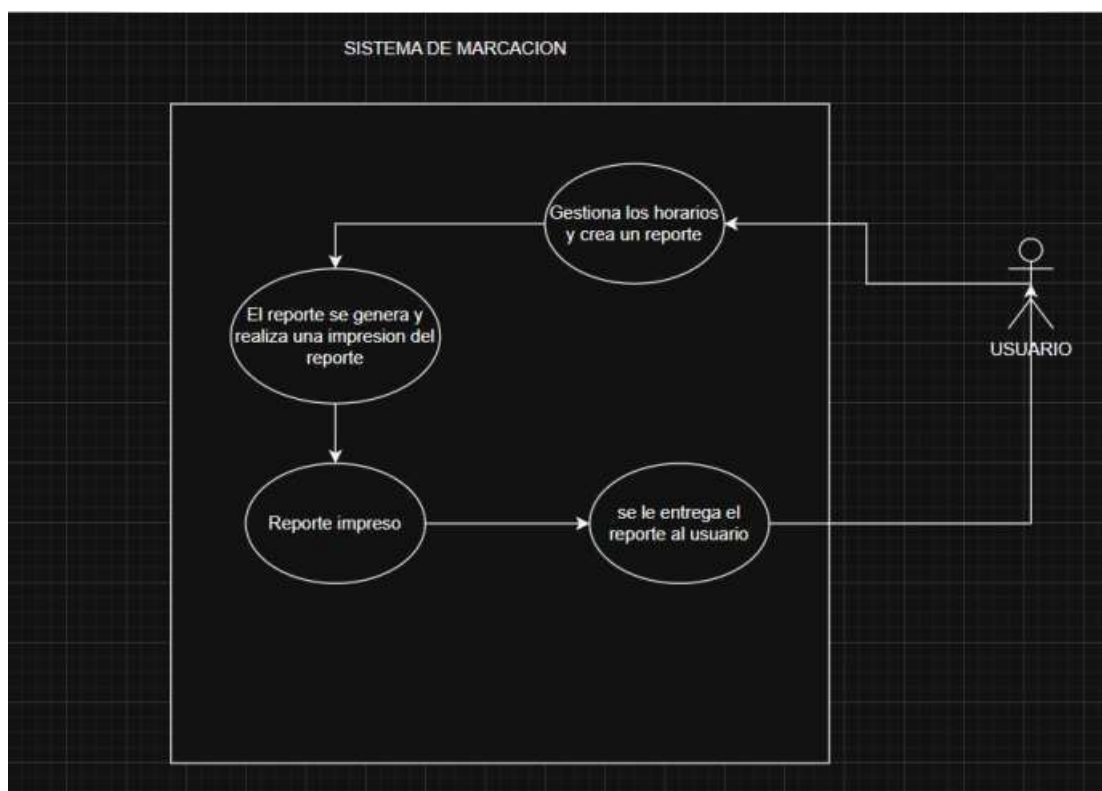


Ilustración 16 Diagrama de casos de uso

- Diagrama de Flujos

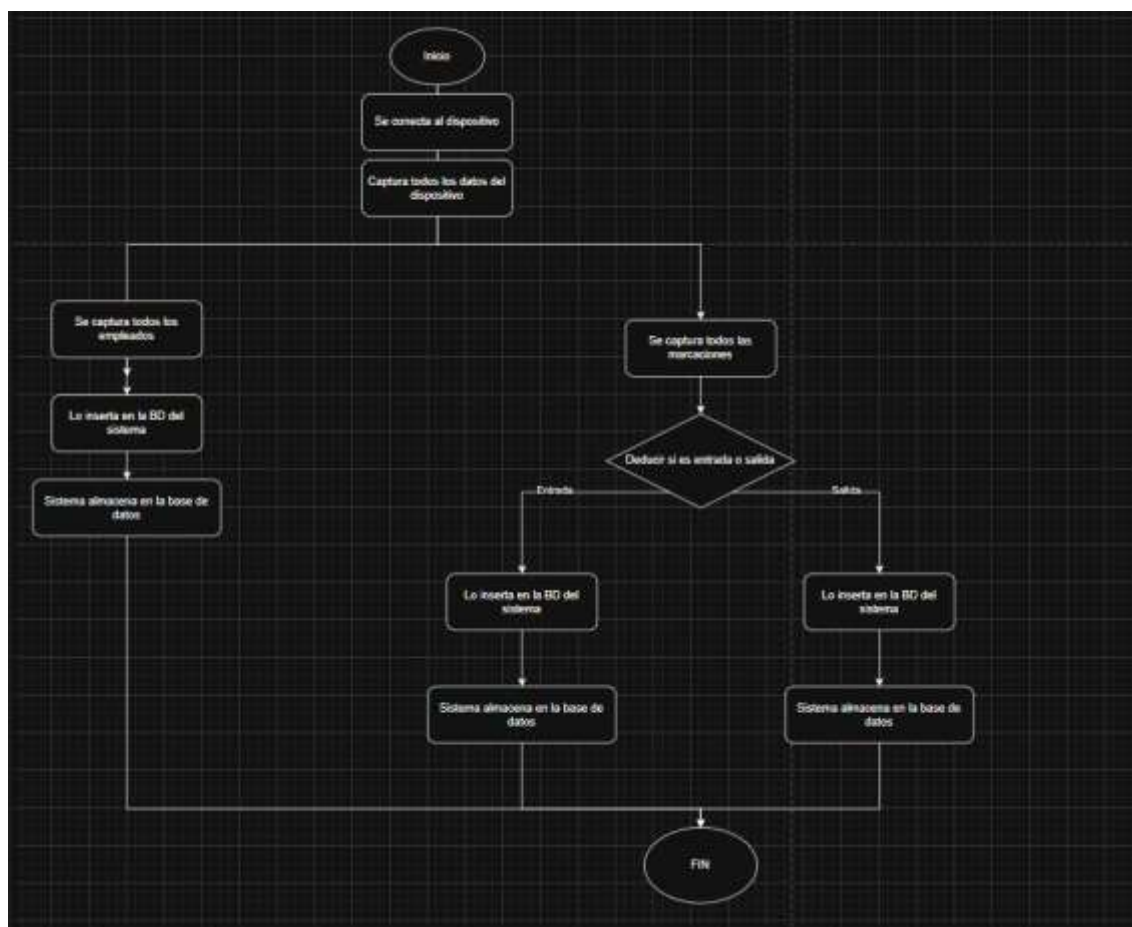


Ilustración 17 Diagrama de flujos de datos

- Diagrama de Entidad-Relación

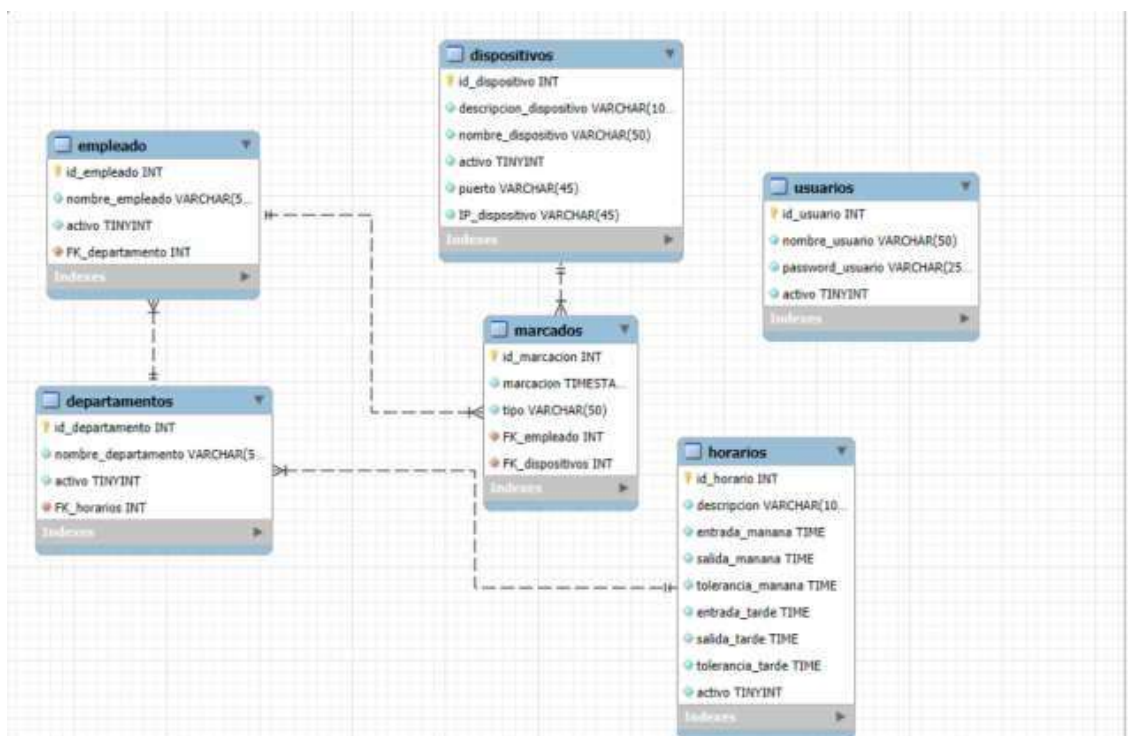


Ilustración 18 Diagrama entidad-relación

- Lógico (MR)



Ilustración 19 Estructura de la Base de datos

Tabla departamentos:

	id_departamento	nombre_departamento	activo	FK_horarios
1	1	administracion	1	1

Ilustración 20 Tabla departamentos de la BD

Tabla Dispositivos:

	id_dispositivo	descripcion_dispositivo	nombre_dispositivo	activo	puerto	IP_dispositivo
▶	1	dispositivo sede central	sede central	1	4370	192.168.150.34
	2	dispositivo prueba	prueba	1	4370	192.168.160.30
*	NULL	NULL	NULL	NULL	NULL	NULL

Ilustración 21 Tabla de dispositivos de la BD

Tabla Empleados:

	id_empleado	nombre_empleado	activo	FK_departamento
▶	1000	pepe	1	1
	1001	jose	1	1
	1002	maria	1	1
	1003	mario	1	1
	1004	javier	1	1
*	NULL	NULL	NULL	NULL

Ilustración 22 Tabla de empleados de la BD

Tabla Horarios:

	id_horario	descripcion	entrada_manana	salida_manana	tolerancia_manana	entrada_tarde	salida_tarde	tolerancia_tarde	activo
▶	1	horario administracion	06:55:00	12:00:00	07:10:00	13:30:00	18:00:00	13:40:00	1
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Ilustración 23 Tabla de horarios de la BD

Tabla de Marcaciones:

	id_marcacion	marcacion	tipo	detalle	horas_trabajadas	FK_empleado	FK_dispositivo
▶	797	2025-05-24 06:55:13	entrada	a tiempo	NULL	1002	2
	798	2025-05-24 06:55:25	entrada	a tiempo	NULL	1001	2
	799	2025-05-24 06:58:09	entrada	a tiempo	NULL	1000	2
	800	2025-05-24 12:01:54	salida	NULL	05:06:29	1001	2
	801	2025-05-24 12:02:19	salida	NULL	05:07:06	1002	2
	802	2025-05-24 12:04:48	salida	NULL	05:06:39	1000	2
	803	2025-05-24 13:27:35	salida	NULL	06:29:26	1000	2
	804	2025-05-24 13:27:56	salida	NULL	06:32:31	1001	2
	805	2025-05-24 13:31:30	entrada	a tiempo	NULL	1002	2
	806	2025-05-24 17:59:08	salida	NULL	04:27:38	1002	2
	807	2025-05-24 18:00:03	salida	NULL	11:01:54	1000	2
	808	2025-05-24 18:00:48	salida	NULL	11:05:23	1001	2
	809	2025-05-25 06:53:50	entrada	a tiempo	NULL	1001	2

Ilustración 24 Tabla marcaciones de la BD

Tabla de Usuario:

	id_usuario	nombre_usuario	password_usuario	activo
▶	1	admin	scrypt:32768:8:1\$KXu603eQVyO4BjSR\$5a3da0...	1
✱	NULL	NULL	NULL	NULL

Ilustración 25 Tabla usuarios de la BD

6.7. Código Fuente

6.7.1. HTML

- Index.html

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Home</title>
  <!-- Bootstrap CSS -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
  <!-- Archivo CSS personalizado para estilos propios -->
  <link rel="stylesheet" href="{{ url_for('static', filename='estilos.css') }}">
</head>
<body>
  <!-- Barra Lateral -->
  <div class="sidebar">
    <h4 class="text-white text-center py-3">AsistOK</h4>
    <a href="{{ url_for('index') }}">Inicio</a>
    <a href="{{ url_for('marcacion') }}">Marcación</a>
    <a href="{{ url_for('dispositivos') }}">Dispositivos</a>
    <a href="{{ url_for('empleados') }}">Empleados</a>
    <a href="{{ url_for('usuarios') }}">Usuarios</a>
    <a href="{{ url_for('departamentos') }}">Departamentos</a>
    <a href="{{ url_for('horarios') }}">Horarios</a>
    <a href="#" onclick="confirmarLogout()">Cerrar sesión</a>
  </div>
  <!-- Contenido Principal -->
  <div class="main-content">
    <!-- Bienvenida -->

    <div class="topbar d-flex justify-content-between align-items-center p-3 bg-light mb-3 rounded">
      <span class="fw-bold">Bienvenido, {{ session['usuario'] }}</span>
    </div>

    <!-- Formularios de accion -->
    <div class="d-flex flex-wrap gap-4">
      <!-- Formulario: Conectar al Dispositivo -->
      <form method="post" onsubmit="mostrarCarga(this)" class="card p-4 shadow-sm" style="min-width: 300px;">
        <h5>Conectar al Dispositivo</h5>
        <div class="mb-3">
          <label for="ip_dispositivo" class="form-label">IP</label>
          <input type="text" class="form-control" name="ip_dispositivo" required>
        </div>
        <div class="mb-3">
          <label for="puerto_dispositivo" class="form-label">Puerto</label>
          <input type="number" class="form-control" name="puerto_dispositivo" required>
        </div>
        <button type="submit" name="accion" value="conectar" class="btn btn-primary w-100">
```




```

        <span class="spinner-border spinner-border-sm me-2 d-none" role="status" aria-hidden="true"></span>
        <span class="texto-boton">Actualizar</span>
    </button>
</form>
<!-- Formulario: Actualizar Todo -->
<form method="post" onsubmit="mostrarCarga(this)" class="card p-4 shadow-sm" style="min-width: 300px; height: fit-
content;">
    <h5>Sincronizar Datos</h5>
    <button type="submit" name="accion" value="actualizar_todo" class="btn btn-success w-100">
        <span class="spinner-border spinner-border-sm me-2 d-none" role="status" aria-hidden="true"></span>
        <span class="texto-boton">Actualizar Todo</span>
    </button>
</form>
</div>
</div>
<!-- Scripts JS -->
<!-- Variable global para URL de logout, utilizada en scripts.js -->
<script>const logoutUrl = "{{ url_for('logout') }}";</script>
<!-- Archivo JS externo con funciones para confirmaciones -->
<script src="{{ url_for('static', filename='scripts.js') }}"></script>

```

```

</body>
</html>

```

```

</html>

```

- Login.html

```

<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Iniciar sesión</title>
    <!-- Bootstrap -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
    <!-- Hoja de estilos personalizada -->
    <link rel="stylesheet" href="{{ url_for('static', filename='estilos.css') }}">
</head>
<body>
    <!-- Contenedor del formulario de login -->
    <div class="login-container">
        <!-- Título principal del sistema -->
        <h2>AsistOK</h2>
        <!-- Subtítulo de la sección de login -->
        <h1>Iniciar sesión</h1>
        <!-- Formulario de inicio de sesión -->
        <form method="post">
            <!-- Campo para el nombre de usuario -->
            <div class="mb-3">
                <label for="usuario" class="form-label">Usuario</label>
                <input type="text" class="form-control" name="usuario" id="usuario" required>
            </div>
            <!-- Campo para la contraseña -->
            <div class="mb-3">
                <label for="password" class="form-label">Contraseña</label>
                <input type="password" class="form-control" name="password" id="password" required>
            </div>
            <!-- Botón para enviar el formulario -->
            <button type="submit" class="btn btn-primary w-100">Entrar</button>
        </form>
        <!-- Bloque para mostrar mensajes flash -->
        {% with messages = get_flashed_messages(with_categories=true) %}

            {% if messages %}
            <div class="mt-3">
                {% for category, msg in messages %}

```





```

    }
  },
  // Personalización del PDF generado
  customize: function (doc) {
    doc.pageMargins = [40, 60, 40, 60];
    doc.defaultStyle.fontSize = 12;
    doc.styles.tableHeader.fontSize = 14;
    doc.styles.title.fontSize = 18;
    doc.styles.title.alignment = 'center';

    // Agregar texto personalizado al principio del PDF
    doc.content.splice(0, 0, {
      text: 'Reporte de: ${usuario} a las ${fechaHora}',
      margin: [0, 0, 0, 10],
      alignment: 'right',
      fontSize: 10,
      italics: true
    });
  }
},
language: {
  url: '//cdn.datatables.net/plug-ins/1.13.4/i18n/es-ES.json' // Traducción al español
}
});
</script>
</head>
<body>
  <!-- Barra lateral de navegación -->
  <div class="sidebar">
    <h4 class="text-white text-center py-3">AsistOK</h4>
    <a href="{{ url_for('index') }}">Inicio</a>
    <a href="{{ url_for('marcacion') }}">Marcacion</a>
    <a href="{{ url_for('dispositivos') }}">Dispositivos</a>
    <a href="{{ url_for('empleados') }}">Empleados</a>
    <a href="{{ url_for('usuarios') }}">Usuarios</a>
    <a href="{{ url_for('departamentos') }}">Departamentos</a>
    <a href="{{ url_for('horarios') }}">Horarios</a>
    <a href="#" onclick="confirmarLogout()">Cerrar sesión</a>
  </div>
  <!-- Contenedor principal -->
  <div class="main-content">
    <!-- Tabla con datos de marcaciones -->
    <table id="example" class="table table-striped table-bordered">
      <thead>
        <tr>
          <th>ID</th>
          <th>Marcacion</th>
          <th>Tipo</th>
          <th>Detalle</th>
          <th>Horas Trabajadas</th>
          <th>Empleado</th>
          <th>Dispositivo</th>
        </tr>
      </thead>
      <tbody>
        {% for marca in marcacion %}
        <tr>
          <td>{{ marca.id_marcacion }}</td>
          <td>{{ marca.marcacion }}</td>
          <td>{{ marca.tipo }}</td>

```

```

        <td>{{ marca.detalle }}</td>
        <td>{{ marca.horas_trabajadas }}</td>
        <td>{{ marca.nombre_empleado }}</td>
        <td>{{ marca.nombre_dispositivo }}</td>
    </tr>
    {% endfor %}
</tbody>
</table>
</div>
<!-- Variable global para usar en scripts.js (logout) -->
<script>const logoutUrl = "{{ url_for('logout') }}";</script>
<!-- Script externo con funciones como confirmarLogout() -->
<script src="{{ url_for('static', filename='scripts.js') }}"></script>
</body>
</html>
- Departamentos.html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Departamentos</title>
    <!-- Bootstrap -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

    <!-- Estilos personalizados -->
    <link rel="stylesheet" href="{{ url_for('static', filename='estilos.css' ) }}">
    <!-- Estilos de Bootstrap para DataTables -->
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.5.2/css/bootstrap.css">
    <!-- Estilos CSS para DataTables con Bootstrap -->
    <link rel="stylesheet" href="https://cdn.datatables.net/2.3.0/css/dataTables.bootstrap4.css">
    <!-- jQuery (necesario para DataTables y Bootstrap JS) -->
    <script src="https://code.jquery.com/jquery-3.7.1.js"></script>
    <!-- Popper.js requerido por Bootstrap para tooltips y popovers -->
    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"></script>
    <!-- Bootstrap JS para funcionalidades JS -->
    <script src="https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.5.2/js/bootstrap.min.js"></script>
    <!-- Libreria DataTables principal -->
    <script src="https://cdn.datatables.net/2.3.0/js/dataTables.js"></script>
    <!-- Integracion de DataTables con Bootstrap 4 -->
    <script src="https://cdn.datatables.net/2.3.0/js/dataTables.bootstrap4.js"></script>
    <!-- Script para inicializar DataTables cuando el DOM este listo -->
    <script>
        $(document).ready(function() {
            $('#example').DataTable();
        });
    </script>
</head>
<body>
    <!-- Barra Lateral -->
    <div class="sidebar">
        <!-- Nombre del sistema en barra lateral -->
        <h4 class="text-white text-center py-3">AsistOK</h4>
        <!-- Enlaces de navegacion -->
        <a href="{{ url_for('index') }}">Inicio</a>
        <a href="{{ url_for('marcacion') }}">Marcacion</a>
        <a href="{{ url_for('dispositivos') }}">Dispositivos</a>
        <a href="{{ url_for('empleados') }}">Empleados</a>
        <a href="{{ url_for('usuarios') }}">Usuarios</a>
        <a href="{{ url_for('departamentos') }}">Departamentos</a>
        <a href="{{ url_for('horarios') }}">Horarios</a>

        <!-- Enlace para cerrar sesion con funcion confirmacion -->
    </div>

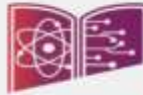
```



```

<a href="#" onclick="confirmarLogout()">Cerrar sesión</a>
</div>
<!-- Contenido Principal -->
<div class="main-content">
  <!-- Boton para ir a la pagina de agregar departamento -->
  <a href="{{ url_for('agregar_departamento') }}">
    <button>Agregar Departamento</button>
  </a>
  <!-- Tabla para mostrar la lista de departamentos con DataTables -->
  <table id="example" class="table table-striped table-bordered">
    <thead>
      <tr>
        <th>ID</th>
        <th>Nombre</th>
        <th>Activo</th>
        <th>Horario</th>
        <th>Accion</th>
      </tr>
    </thead>
    <tbody>
      <!-- Recorre cada departamento enviado desde backend -->
      {% for departamento in departamentos %}
      <tr>
        <td>{{ departamento.id_departamento }}</td>
        <td>{{ departamento.nombre_departamento }}</td>
        <td>{{ departamento.activo }}</td>
        <td>{{ departamento.descripcion }}</td>
        <td>
          <!-- Enlace para editar departamento, con boton -->
          <a href="{{ url_for('editar_departamento', id=departamento.id_departamento) }}">
            <button>Editar</button>
          </a>
        </td>
      </tr>
      {% endfor %}
    </tbody>
  </table>
</div>
<!-- Scripts JS -->
<!-- Variable global para usar dentro del archivo scripts.js -->
<script>const logoutUrl = "{{ url_for('logout') }}";</script>
<!-- Archivo JS externo con funciones de confirmacion -->
<script src="{{ url_for('static', filename='scripts.js') }}"></script>
</body>
</html>
- Agregar departamentos.html
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Agregar Departamento</title>
  <!-- Bootstrap -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
  <!-- Enlace al archivo de estilos personalizado ubicado en la carpeta static -->
  <link rel="stylesheet" href="{{ url_for('static', filename='estilos.css') }}">
</head>
<body>
  <!-- Barra Lateral de Navegacion -->
  <div class="sidebar">
    <!-- Titulo del sistema dentro del sidebar -->
    <h4 class="text-white text-center py-3">AsistOK</h4>
    <!-- Enlaces de navegacion hacia las distintas secciones del sistema -->

```

```
<a href="{{ url_for('index') }}">Inicio</a>
<a href="{{ url_for('marcacion') }}">Marcacion</a>
<a href="{{ url_for('dispositivos') }}">Dispositivos</a>
<a href="{{ url_for('empleados') }}">Empleados</a>
<a href="{{ url_for('usuarios') }}">Usuarios</a>
<a href="{{ url_for('departamentos') }}">Departamentos</a>
<a href="{{ url_for('horarios') }}">Horarios</a>
<!-- Enlace que llama a una funcion JavaScript para confirmar cierre de sesión -->
<a href="#" onclick="confirmarLogout()">Cerrar sesión</a>
</div>
<!-- Contenido Principal -->

<div class="main-content">
  <!-- Boton de regreso a la lista de departamentos -->
  <a href="{{ url_for('departamentos') }}" class="btn btn-secondary btn-lg">Volver</a>
  <!-- Contenedor con el formulario -->
  <div class="container mt-4" style="max-width: 700px;">
    <!-- Titulo centrado del formulario -->
    <h2 class="mb-4 text-center">Agregar Departamento</h2>
    <!-- Formulario de envio POST para agregar un nuevo departamento -->
    <!-- Usa una funcion JS para confirmar antes de enviar -->
    <form method="post" class="card shadow p-5" onsubmit="return confirmarGuardado()">
      <!-- Campo de texto para el nombre del departamento -->
      <div class="mb-4">
        <label for="nombre" class="form-label">Nombre del Departamento</label>
        <input type="text" class="form-control form-control-lg" name="nombre" id="nombre" required>
      </div>
      <!-- Lista desplegable con los horarios disponibles -->
      <div class="mb-4">
        <label for="id_horario" class="form-label">Horario Asignado</label>
        <select name="id_horario" id="id_horario" class="form-select form-select-lg" required>
          <option value="">-- Seleccione un horario --</option>
          {% for horario in horarios %}
            <option value="{{ horario.id_horario }}">{{ horario.descripcion }}</option>
          {% endfor %}
        </select>
      </div>
      <!-- Botones para guardar o cancelar -->
      <div class="d-flex justify-content-between">
        <button type="submit" class="btn btn-primary btn-lg">Guardar</button>
        <a href="{{ url_for('departamentos') }}" class="btn btn-secondary btn-lg">Cancelar</a>
      </div>
    </form>
  </div>
  <!-- Scripts JS -->
  <!-- Variable global para ser usada por scripts.js -->
  <script>const logoutUrl = "{{ url_for('logout') }}";</script>
```

```
<!-- Archivo JS externo con funciones de confirmacion -->
<script src="{{ url_for('static', filename='scripts.js') }}"></script>
</body>
</html>
```

- Editar Departamentos.html

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Editar Departamento</title>
  <!-- Bootstrap -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
  <!-- Estilos personalizados -->
```





```
<link rel="stylesheet" href="{{ url_for('static', filename='estilos.css') }}">
</head>
<body>
<!-- Barra Lateral -->
<div class="sidebar">
  <!-- Nombre del sistema en barra lateral -->
  <h4 class="text-white text-center py-3">AsistOK</h4>
  <!-- Enlaces de navegacion -->
  <a href="{{ url_for('index') }}">Inicio</a>
  <a href="{{ url_for('marcacion') }}">Marcacion</a>
  <a href="{{ url_for('dispositivos') }}">Dispositivos</a>
  <a href="{{ url_for('empleados') }}">Empleados</a>
  <a href="{{ url_for('usuarios') }}">Usuarios</a>
  <a href="{{ url_for('departamentos') }}">Departamentos</a>
  <a href="{{ url_for('horarios') }}">Horarios</a>
  <!-- Enlace para cerrar sesion con funcion confirmación -->
  <a href="#" onclick="confirmarLogout()">Cerrar sesión</a>
</div>
<!-- Contenido Principal -->
<div class="main-content">
  <!-- Boton para volver a la pagina de listado de departamentos -->
  <a href="{{ url_for('departamentos') }}" class="btn btn-secondary btn-lg">Volver</a>
  <div class="container mt-4">
    <!-- Titulo de la pagina -->

    <h2 class="mb-4">Editar Departamento</h2>
    <!-- Formulario para editar el departamento -->
    <form method="post" class="card shadow p-4" onsubmit="return confirmarGuardado()">
      <!-- Campo de texto para nombre del departamento -->
      <div class="mb-3">
        <label for="nombre" class="form-label">Nombre</label>
        <!-- El valor se rellena con el nombre actual del departamento -->
        <input type="text" class="form-control" id="nombre" name="nombre" value="{{
departamentos.nombre_departamento }}" required>
      </div>
      <!-- Checkbox para indicar si el departamento esta activo -->
      <div class="form-check mb-3">
        <!-- Si el departamento esta activo (activo == 1), el checkbox aparecera marcado -->
        <input type="checkbox" class="form-check-input" id="activo" name="activo" {% if departamentos.activo == 1
%}checked{% endif %}>
        <label class="form-check-label" for="activo">Activo</label>
      </div>
      <!-- Select para elegir el horario asignado al departamento -->
      <div class="mb-3">
        <label for="horario" class="form-label">Horario</label>
        <select name="id_horario" id="id_horario" class="form-select" required>
          <option value="">-- Seleccione un horario --</option>
          <!-- Se listan los horarios disponibles desde backend -->
          {% for horario in horarios %}
            <option value="{{ horario.id_horario }}">{{ horario.descripcion }}</option>
          {% endfor %}
        </select>
      </div>
      <!-- Boton para enviar el formulario y guardar cambios -->
      <button type="submit" class="btn btn-primary">Guardar cambios</button>
      <!-- Boton para cancelar y regresar al listado -->
      <a href="{{ url_for('departamentos') }}" class="btn btn-secondary ms-2">Cancelar</a>
    </form>
  </div>
</div>
<!-- Scripts JS -->
<!-- Variable global para usar dentro del archivo scripts.js -->
<script>const logoutUrl = "{{ url_for('logout') }}";</script>
```

<!-- Archivo JS externo con funciones de confirmación -->

<script src="{{ url_for('static', filename='scripts.js') }}"></script>

</body>

</html>

- **Dispositivos.html**

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<title>Dispositivos</title>

<!-- Bootstrap -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

<!-- Estilos personalizados -->

<link rel="stylesheet" href="{{ url_for('static', filename='estilos.css') }}">

<!-- Estilos de Bootstrap para DataTables -->

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.5.2/css/bootstrap.css">

<!-- Estilos CSS para DataTables con Bootstrap -->

<link rel="stylesheet" href="https://cdn.datatables.net/2.3.0/css/dataTables.bootstrap4.css">

<!-- jQuery (necesario para DataTables y Bootstrap JS) -->

<script src="https://code.jquery.com/jquery-3.7.1.js"></script>

<!-- Popper.js requerido por Bootstrap para tooltips y popovers -->

<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"></script>

<!-- Bootstrap JS para funcionalidades JS -->

<script src="https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.5.2/js/bootstrap.min.js"></script>

<!-- Librería DataTables principal -->

<script src="https://cdn.datatables.net/2.3.0/js/dataTables.js"></script>

<!-- Integración de DataTables con Bootstrap 4 -->

<script src="https://cdn.datatables.net/2.3.0/js/dataTables.bootstrap4.js"></script>

<!-- Script para inicializar DataTables cuando el DOM este listo -->

<script>

\$(document).ready(function() {
 \$('#example').DataTable();
});

</script>

</head>

<body>

<!-- Barra Lateral -->

<div class="sidebar">

<!-- Nombre del sistema en barra lateral -->

<h4 class="text-white text-center py-3">AsistOK</h4>

<!-- Enlaces de navegacion -->

Inicio

Marcacion

Dispositivos

Empleados

Usuarios

Departamentos

Horarios

<!-- Enlace para cerrar sesion con funcion confirmacion -->

Cerrar sesión

</div>

<!-- Contenido principal -->

<div class="main-content">

<!-- Boton para ir a la pagina de agregar dispositivo -->

<button>Agregar Dispositivo</button>

<!-- Tabla para mostrar la lista de dispositivos con DataTables -->

<table id="example" class="table table-striped table-bordered">

<thead>

<tr>

<th>ID</th>

```

        <th>Descripcion</th>
        <th>Nombre</th>
        <th>Activo</th>
        <th>IP</th>
        <th>Puerto</th>
        <th>Accion</th>
    </tr>
</thead>
<tbody>
    <!-- Recorre cada dispositivo enviado desde backend -->
    {% for dispositivo in dispositivos %}
    <tr>
        <td>{{ dispositivo.id_dispositivo }}</td>

        <td>{{ dispositivo.descripcion_dispositivo }}</td>
        <td>{{ dispositivo.nombre_dispositivo }}</td>
        <td>{{ dispositivo.activo }}</td>
        <td>{{ dispositivo.IP_dispositivo }}</td>
        <td>{{ dispositivo.puerto }}</td>
        <td>
            <!-- Enlace para editar dispositivo, con boton -->
            <a href="{{ url_for('editar_dispositivo', id=dispositivo.id_dispositivo) }}">
                <button>Editar</button>
            </a>
        </td>
    </tr>
    {% endfor %}
</tbody>
</table>
</div>
<!-- Scripts JS -->
<!-- Variable global para usar dentro del archivo scripts.js -->
<script>const logoutUrl = "{{ url_for('logout') }}";</script>
<!-- Archivo JS externo con funciones de confirmacion -->
<script src="{{ url_for('static', filename='scripts.js') }}"></script>
</body>
</html>
- Agregar dispositivos.html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Agregar dispositivo</title>
    <!-- Bootstrap -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
    <!-- Archivo de estilos -->
    <link rel="stylesheet" href="{{ url_for('static', filename='estilos.css') }}">
</head>
<body>
    <!-- Barra lateral de navegacion -->
    <div class="sidebar">

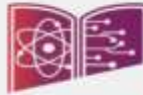
        <!-- Titulo del sistema -->
        <h4 class="text-white text-center py-3">AsistOK</h4>
        <!-- Enlaces de navegacion -->
        <a href="{{ url_for('index') }}">Inicio</a>
        <a href="{{ url_for('marcacion') }}">Marcacion</a>
        <a href="{{ url_for('dispositivos') }}">Dispositivos</a>
        <a href="{{ url_for('empleados') }}">Empleados</a>
        <a href="{{ url_for('usuarios') }}">Usuarios</a>
        <a href="{{ url_for('departamentos') }}">Departamentos</a>
        <a href="{{ url_for('horarios') }}">Horarios</a>
        <!-- Enlace con confirmacion para cerrar sesion -->
    
```



```

<a href="#" onclick="confirmarLogout()">Cerrar sesión</a>
</div>
<!-- Contenido principal -->
<div class="main-content">
  <!-- Boton para volver a la lista de dispositivos -->
  <a href="{{ url_for('dispositivos') }}" class="btn btn-secondary btn-lg">Volver</a>
  <!-- Contenedor del formulario -->
  <div class="container mt-4" style="max-width: 700px;">
    <!-- Titulo del formulario -->
    <h2 class="mb-4 text-center">Agregar Dispositivo</h2>
    <!-- Formulario para agregar un nuevo dispositivo -->
    <form method="post" class="card shadow p-5" onsubmit="return confirmarGuardado()">
      <!-- Campo: Nombre del dispositivo -->
      <div class="mb-4">
        <label for="nombre" class="form-label">Nombre del Dispositivo</label>
        <input type="text" class="form-control form-control-lg" name="nombre" id="nombre" required>
      </div>
      <!-- Campo: Descripcion del dispositivo -->
      <div class="mb-4">
        <label for="descripcion" class="form-label">Descripción del Dispositivo</label>
        <input type="text" class="form-control form-control-lg" name="descripcion" id="descripcion" required>
      </div>
      <!-- Campo: IP del dispositivo -->
      <div class="mb-4">
        <label for="ip" class="form-label">IP del Dispositivo</label>
        <input type="text" class="form-control form-control-lg" name="ip" id="ip" required>
      </div>
      <!-- Campo: Puerto del dispositivo -->
      <div class="mb-4">
        <label for="puerto" class="form-label">Puerto del Dispositivo</label>
        <input type="text" class="form-control form-control-lg" name="puerto" id="puerto" required>
      </div>
      <!-- Botones para guardar o cancelar -->
      <div class="d-flex justify-content-between">
        <button type="submit" class="btn btn-primary btn-lg">Guardar</button>
        <a href="{{ url_for('dispositivos') }}" class="btn btn-secondary btn-lg">Cancelar</a>
      </div>
    </form>
  </div>
</div>
<!-- Scripts JS -->
<!-- Variable global para usar dentro del archivo scripts.js -->
<script>const logoutUrl = "{{ url_for('logout') }}";</script>
<!-- Archivo JS externo con funciones de confirmacion -->
<script src="{{ url_for('static', filename='scripts.js') }}"></script>
</body>
</html>
- Editar dispositivos.html
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Editar Dispositivo</title>
  <!-- Bootstrap CSS -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
  <!-- Estilos personalizados -->
  <link rel="stylesheet" href="{{ url_for('static', filename='estilos.css') }}">
</head>
<body>
  <!-- Barra Lateral -->
  <div class="sidebar">
    <!-- Nombre del sistema en la barra lateral -->

```

<h4 class="text-white text-center py-3">AsistOK</h4>

<!-- Enlaces de navegacion -->

Inicio

Marcacion

Dispositivos

Empleados

Usuarios

Departamentos

Horarios

<!-- Enlace para cerrar sesión con funcion de confirmacion -->

Cerrar sesión

</div>

<!-- Contenido Principal -->

<div class="main-content">

<!-- Boton para volver a la pagina de listado de dispositivos -->

Volver

<div class="container mt-4">

<!-- Titulo de la pagina -->

<h2 class="mb-4">Editar Dispositivo</h2>

<!-- Formulario para editar dispositivo -->

<form method="post" class="card shadow p-4" onsubmit="return confirmarGuardado()">

<!-- Campo para la descripcion del dispositivo, precargada con el valor actual -->

<div class="mb-3">

<label for="descripcion" class="form-label">Descripción</label>

<input type="text" class="form-control" id="descripcion" name="descripcion" value="{{

dispositivos.descripcion_dispositivo }}" required>

</div>

<!-- Campo para el nombre del dispositivo, precargada con el valor actual -->

<div class="mb-3">

<label for="nombre" class="form-label">Nombre</label>

<input type="text" class="form-control" id="nombre" name="nombre" value="{{ dispositivos.nombre_dispositivo }}"

required>

</div>

<!-- Checkbox para marcar si el dispositivo esta activo -->

<div class="form-check mb-3">

<!-- Se marca si el dispositivo esta activo (activo == 1) -->

<input type="checkbox" class="form-check-input" id="activo" name="activo" {% if dispositivos.activo == 1

%}checked{% endif %}>

<label class="form-check-label" for="activo">Activo</label>

</div>

</div>

<!-- Campo para puerto del dispositivo, precargada con el valor actual -->

<div class="mb-3">

<label for="puerto" class="form-label">Puerto</label>

<input type="text" class="form-control" id="puerto" name="puerto" value="{{ dispositivos.puerto }}" required>

</div>

<!-- Campo para direccion IP del dispositivo, precargada con el valor actual -->

<div class="mb-3">

<label for="ip" class="form-label">Dirección IP</label>

<input type="text" class="form-control" id="ip" name="ip" value="{{ dispositivos.IP_dispositivo }}" required>

</div>

<!-- Boton para enviar el formulario y guardar los cambios -->

<button type="submit" class="btn btn-primary">Guardar cambios</button>

<!-- Boton para cancelar y regresar al listado de dispositivos -->

Cancelar

</form>

</div>

</div>

<!-- Scripts JS -->

<!-- Variable global logoutUrl para usar en scripts.js -->

<script>const logoutUrl = "{{ url_for('logout') }}";</script>

<!-- Archivo JS externo con funciones de confirmación -->



```
<script src="{{ url_for('static', filename='scripts.js') }}"></script>
</body>
</html>
```

- Empleados.html

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Empleados</title>
  <!-- Bootstrap -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
  <!-- Estilos personalizados -->
  <link rel="stylesheet" href="{{ url_for('static', filename='estilos.css') }}">

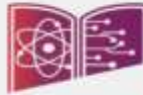
  <!-- Estilos de Bootstrap para DataTables -->
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.5.2/css/bootstrap.css">
  <!-- Estilos CSS para DataTables con Bootstrap -->
  <link rel="stylesheet" href="https://cdn.datatables.net/2.3.0/css/dataTables.bootstrap4.css">
  <!-- jQuery (necesario para DataTables y Bootstrap JS) -->
  <script src="https://code.jquery.com/jquery-3.7.1.js"></script>
  <!-- Popper.js requerido por Bootstrap para tooltips y popovers -->
  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"></script>
  <!-- Bootstrap JS para funcionalidades JS -->
  <script src="https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.5.2/js/bootstrap.min.js"></script>
  <!-- Libreria DataTables principal -->
  <script src="https://cdn.datatables.net/2.3.0/js/dataTables.js"></script>
  <!-- Integración de DataTables con Bootstrap 4 -->
  <script src="https://cdn.datatables.net/2.3.0/js/dataTables.bootstrap4.js"></script>
  <!-- Script para inicializar DataTables cuando el DOM este listo -->
  <script>
    $(document).ready(function() {
      $('#example').DataTable();
    });
  </script>
</head>
<body>
  <!-- Barra Lateral -->
  <div class="sidebar">
    <!-- Nombre del sistema en barra lateral -->
    <h4 class="text-white text-center py-3">AsistOK</h4>
    <!-- Enlaces de navegacion -->
    <a href="{{ url_for('index') }}">Inicio</a>
    <a href="{{ url_for('marcacion') }}">Marcacion</a>
    <a href="{{ url_for('dispositivos') }}">Dispositivos</a>
    <a href="{{ url_for('empleados') }}">Empleados</a>
    <a href="{{ url_for('usuarios') }}">Usuarios</a>
    <a href="{{ url_for('departamentos') }}">Departamentos</a>
    <a href="{{ url_for('horarios') }}">Horarios</a>
    <!-- Enlace para cerrar sesion con funcion confirmacion -->
    <a href="#" onclick="confirmarLogout()">Cerrar sesión</a>
  </div>

  <!-- Contenido principal -->
  <div class="main-content">
    <!-- Boton para ir a la pagina de agregar empleado -->
    <a href="{{ url_for('agregar_empleado') }}"><button>Agregar Empleado</button></a>
    <!-- Tabla para mostrar la lista de empleados con DataTables -->
    <table id="example" class="table table-striped table-bordered">
      <thead>
        <tr>
          <th>ID</th>
          <th>Nombre</th>
        </tr>
      </thead>
    </table>
  </div>
```

```
<th>Activo</th>
<th>Departamento</th>
<th>Accion</th>
</tr>
</thead>
<tbody>
<!-- Recorre cada empleado enviado desde backend -->
{% for empleado in empleados %}
<tr>
<td>{{empleado.id_empleado}}</td>
<td>{{empleado.nombre_empleado}}</td>
<td>{{empleado.activo}}</td>
<td>{{empleado.nombre_departamento}}</td>
<!-- Enlace para editar empleado, con boton -->
<td><a href="{{ url_for('editar_empleado', id=empleado.id_empleado) }}"><button>Editar</button></a></td>
</tr>
{% endfor %}
</tbody>
</table>
</div>
<!-- Scripts JS -->
<!-- Variable global para URL de logout, utilizada en scripts.js -->
<script>const logoutUrl = "{{ url_for('logout') }}"</script>
<!-- Archivo JS externo con funciones para confirmaciones -->
<script src="{{ url_for('static', filename='scripts.js') }}"></script>
</body>
</html>
```

- Agregar Empleados.html

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="UTF-8">
<title>Agregar Empleado</title>
<!-- Bootstrap -->
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
<!-- Archivo CSS personalizado ubicado en la carpeta /static -->
<link rel="stylesheet" href="{{ url_for('static', filename='estilos.css') }}">
</head>
<body>
<!-- Barra lateral de navegación -->
<div class="sidebar">
<!-- Titulo del sistema -->
<h4 class="text-white text-center py-3">AsistOK</h4>
<!-- Enlaces de navegacion hacia las secciones del sistema -->
<a href="{{ url_for('index') }}">Inicio</a>
<a href="{{ url_for('marcacion') }}">Marcacion</a>
<a href="{{ url_for('dispositivos') }}">Dispositivos</a>
<a href="{{ url_for('empleados') }}">Empleados</a>
<a href="{{ url_for('usuarios') }}">Usuarios</a>
<a href="{{ url_for('departamentos') }}">Departamentos</a>
<a href="{{ url_for('horarios') }}">Horarios</a>
<!-- Enlace que invoca funcion para confirmar cierre de sesion -->
<a href="#" onclick="confirmarLogout()">Cerrar sesión</a>
</div>
<!-- Contenido principal -->
<div class="main-content">
<!-- Boton para volver a la lista de empleados -->
<a href="{{ url_for('empleados') }}" class="btn btn-secondary btn-lg">Volver</a>
<!-- Contenedor principal del formulario -->
<div class="container mt-4" style="max-width: 700px;">
<!-- Titulo del formulario -->
<h2 class="mb-4 text-center">Agregar Empleado</h2>
```



<!-- Formulario para agregar un nuevo empleado -->

<!-- Llama a una funcion JS para confirmar antes de enviar -->

```
<form method="post" class="card shadow p-5" onsubmit="return confirmarGuardado()">
  <!-- Campo: Nombre del empleado -->
  <div class="mb-4">
    <label for="nombre" class="form-label">Nombre del empleado</label>
    <input type="text" class="form-control form-control-lg" name="nombre" id="nombre" required>
  </div>
  <!-- Campo: Seleccion del departamento asignado -->
  <div class="mb-4">
    <label for="departamento" class="form-label">Departamento Asignado</label>
    <select name="departamento" id="departamento" class="form-select form-select-lg" required>
      <option value="">-- Seleccione el departamento --</option>
      <!-- Ciclo para mostrar los departamentos disponibles -->
      <% for departamento in departamentos %>
        <option value="{{ departamento.id_departamento }}">{{ departamento.nombre_departamento }}</option>
      <% endfor %>
    </select>
  </div>
  <!-- Botones para guardar o cancelar la acción -->
  <div class="d-flex justify-content-between">
    <button type="submit" class="btn btn-primary btn-lg">Guardar</button>
    <a href="{{ url_for('empleados') }}" class="btn btn-secondary btn-lg">Cancelar</a>
  </div>
</form>
</div>
<!-- Scripts JS -->
<!-- Variable global para usar dentro del archivo scripts.js -->
<script>const logoutUrl = "{{ url_for('logout') }}";</script>
<!-- Archivo JS externo con funciones de confirmacion -->
<script src="{{ url_for('static', filename='scripts.js') }}"></script>
</body>
</html>
```

- Editar Empleados.html

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">

  <title>Editar Empleado</title>
  <!-- Bootstrap CSS -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
  <!-- Archivo CSS personalizado -->
  <link rel="stylesheet" href="{{ url_for('static', filename='estilos.css') }}">
</head>
<body>
  <!-- Barra Lateral -->
  <div class="sidebar">
    <!-- Titulo del sistema en la barra lateral -->
    <h4 class="text-white text-center py-3">AsistOK</h4>
    <!-- Navegacion con enlaces a diferentes secciones del sistema -->
    <a href="{{ url_for('index') }}">Inicio</a>
    <a href="{{ url_for('marcacion') }}">Marcacion</a>
    <a href="{{ url_for('dispositivos') }}">Dispositivos</a>
    <a href="{{ url_for('empleados') }}">Empleados</a>
    <a href="{{ url_for('usuarios') }}">Usuarios</a>
    <a href="{{ url_for('departamentos') }}">Departamentos</a>
    <a href="{{ url_for('horarios') }}">Horarios</a>
    <!-- Enlace para cerrar sesion con funcion de confirmacion -->
    <a href="#" onclick="confirmarLogout()">Cerrar sesión</a>
  </div>
```



```

<!-- Contenido Principal -->
<div class="main-content">
  <!-- Boton para regresar al listado de empleados -->
  <a href="{{ url_for('empleados') }}" class="btn btn-secondary btn-lg">Volver</a>
  <div class="container mt-4">
    <!-- Titulo principal del formulario -->
    <h2 class="mb-4">Editar Empleado</h2>
    <!-- Formulario para editar un empleado -->
    <form method="post" class="card shadow p-4" onsubmit="return confirmarGuardado()">
      <!-- Campo de texto para el nombre del empleado, cargado con el valor actual -->
      <div class="mb-3">
        <label for="nombre" class="form-label">Nombre del Empleado</label>
        <input type="text" class="form-control" id="nombre" name="nombre" value="{{ empleado.nombre_empleado }}"
required>
      </div>

      <!-- Select para elegir el departamento asignado -->
      <div class="mb-3">
        <label for="horario" class="form-label">Departamento Asignado</label>
        <select name="departamento" id="departamento" class="form-select" required>
          <!-- Opcion por defecto para indicarle al usuario que seleccione -->
          <option value="">-- Seleccione un Departamento --</option>
          <!-- Bucle para listar todos los departamentos disponibles -->
          {% for departamento in departamentos %}
            <option value="{{ departamento.id_departamento }}">{{ departamento.nombre_departamento }}</option>
          {% endfor %}
        </select>
      </div>

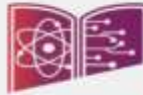
      <!-- Checkbox para marcar si el empleado esta activo -->
      <div class="form-check mb-3">
        <!-- Si empleado.activo es 1, checkbox se marca -->
        <input type="checkbox" class="form-check-input" id="activo" name="activo" {% if empleado.activo == 1 %}checked{%
endif %}>
        <label class="form-check-label" for="activo">Activo</label>
      </div>

      <!-- Boton para guardar los cambios -->
      <button type="submit" class="btn btn-primary">Guardar cambios</button>
      <!-- Boton para cancelar y volver a la lista sin guardar -->
      <a href="{{ url_for('empleados') }}" class="btn btn-secondary ms-2">Cancelar</a>
    </form>
  </div>
</div>

<!-- Scripts JS -->
<!-- Variable global logoutUrl para usar en scripts.js -->
<script>const logoutUrl = "{{ url_for('logout') }}";</script>
<!-- Archivo JS externo con funciones de confirmación -->
<script src="{{ url_for('static', filename='scripts.js') }}"></script>
</body>
</html>
- Horarios.html
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">

  <title>Horarios</title>
  <!-- Bootstrap -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
  <!-- Estilos personalizados -->
  <link rel="stylesheet" href="{{ url_for('static', filename='estilos.css') }}">
  <!-- Estilos de Bootstrap para DataTables -->
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.5.2/css/bootstrap.css">

```

```
<!-- Estilos CSS para DataTables con Bootstrap -->
<link rel="stylesheet" href="https://cdn.datatables.net/2.3.0/css/dataTables.bootstrap4.css">
<!-- jQuery (necesario para DataTables y Bootstrap JS) -->
<script src="https://code.jquery.com/jquery-3.7.1.js"></script>
<!-- Popper.js requerido por Bootstrap para tooltips y popovers -->
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"></script>
<!-- Bootstrap JS para funcionalidades JS -->
<script src="https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.5.2/js/bootstrap.min.js"></script>
<!-- Libreria DataTables principal -->
<script src="https://cdn.datatables.net/2.3.0/js/dataTables.js"></script>
<!-- Integracion de DataTables con Bootstrap 4 -->
<script src="https://cdn.datatables.net/2.3.0/js/dataTables.bootstrap4.js"></script>
<!-- Script para inicializar DataTables cuando el DOM este listo -->
<script>
    $(document).ready(function() {
        $('#example').DataTable();
    });
</script>
</head>
<body>
<!-- Barra Lateral -->
<div class="sidebar">
    <!-- Nombre del sistema en barra lateral -->
    <h4 class="text-white text-center py-3">AsistOK</h4>
    <!-- Enlaces de navegacion -->
    <a href="{{ url_for('index') }}">Inicio</a>
    <a href="{{ url_for('marcacion') }}">Marcacion</a>
    <a href="{{ url_for('dispositivos') }}">Dispositivos</a>
    <a href="{{ url_for('empleados') }}">Empleados</a>
    <a href="{{ url_for('usuarios') }}">Usuarios</a>

    <a href="{{ url_for('departamentos') }}">Departamentos</a>
    <a href="{{ url_for('horarios') }}">Horarios</a>
    <!-- Enlace para cerrar sesion con funcion confirmacion -->
    <a href="#" onclick="confirmarLogout()">Cerrar sesión</a>
</div>
<!-- Contenido principal -->
<div class="main-content">
    <!-- Boton para ir a la pagina de agregar horario -->
    <a href="{{ url_for('agregar_horario') }}"><button>Agregar Horario</button></a>
    <!-- Tabla para mostrar la lista de horarios con DataTables -->
    <table id="example" class="table table-striped table-bordered">
        <thead>
            <tr>
                <th>ID</th>
                <th>Descripcion</th>
                <th>Entrada de la Mañana</th>
                <th>Salida de la Mañana</th>
                <th>Tolerancia de la Mañana</th>
                <th>Entrada de la Tarde</th>
                <th>Salida de la Tarde</th>
                <th>Tolerancia de la Tarde</th>
                <th>Activo</th>
                <th>Accion</th>
            </tr>
        </thead>
        <tbody>
            <!-- Recorre cada horario enviado desde backend -->
            {% for horario in horarios %}
            <tr>
                <td>{{horario.id_horario}}</td>
                <td>{{horario.descripcion}}</td>
                <td>{{horario.entrada_manana}}</td>
```

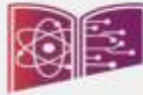
```

<td>{{horario.salida_manana}}</td>
<td>{{horario.tolerancia_manana}}</td>
<td>{{horario.entrada_tarde}}</td>
<td>{{horario.salida_tarde}}</td>
<td>{{horario.tolerancia_tarde}}</td>

<td>{{horario.activo}}</td>
<td>
<!-- Enlace para editar horario, con boton -->
<a href="{{ url_for('editar_horario', id=horario.id_horario) }}">
    <button>Editar</button>
</a>
</td>
</tr>
{% endfor %}
</tbody>
</table>
</div>
<!-- Scripts JS -->
<!-- Variable global para URL de logout, utilizada en scripts.js -->
<script>const logoutUrl = "{{ url_for('logout') }}"</script>
<!-- Archivo JS externo con funciones para confirmaciones -->
<script src="{{ url_for('static', filename='scripts.js') }}"></script>
</body>
</html>
- Agregar Horarios.html
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="UTF-8">
<title>Agregar Horario</title>
<!-- Bootstrap -->
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
<!-- Archivo CSS personalizado -->
<link rel="stylesheet" href="{{ url_for('static', filename='estilos.css') }}">
</head>
<body>
<!-- Barra lateral de navegacion -->
<div class="sidebar">
<!-- Titulo del sistema -->
<h4 class="text-white text-center py-3">AsistOK</h4>
<!-- Menu de navegacion -->
<a href="{{ url_for('index') }}">Inicio</a>

<a href="{{ url_for('marcacion') }}">Marcacion</a>
<a href="{{ url_for('dispositivos') }}">Dispositivos</a>
<a href="{{ url_for('empleados') }}">Empleados</a>
<a href="{{ url_for('usuarios') }}">Usuarios</a>
<a href="{{ url_for('departamentos') }}">Departamentos</a>
<a href="{{ url_for('horarios') }}">Horarios</a>
<!-- Enlace para cerrar sesion con confirmacion -->
<a href="#" onclick="confirmarLogout()">Cerrar sesión</a>
</div>
<!-- Contenido principal -->
<div class="main-content">
<!-- Boton para volver a la lista de horarios -->
<a href="{{ url_for('horarios') }}" class="btn btn-secondary btn-lg">Volver</a>
<!-- Contenedor del formulario -->
<div class="container mt-4" style="max-width: 700px;">
<!-- Titulo del formulario -->
<h2 class="mb-4 text-center">Agregar Horario</h2>
<!-- Formulario para agregar un horario nuevo -->
<!-- Confirmacion al enviar -->

```



```
<form method="post" class="card shadow p-5" onsubmit="return confirmarGuardado()">
  <!-- Descripción del horario -->
  <div class="mb-4">
    <label for="descripcion" class="form-label">Descripción del Horario</label>
    <input type="text" class="form-control form-control-lg" name="descripcion" id="descripcion" required>
  </div>
  <!-- Entrada de la mañana -->
  <div class="mb-4">
    <label for="entrada_manana" class="form-label">Asignar Entrada a la Mañana</label>
    <input type="text" class="form-control form-control-lg" name="entrada_manana" id="entrada_manana" required>
  </div>
  <!-- Salida de la mañana -->
  <div class="mb-4">
    <label for="salida_manana" class="form-label">Asignar Salida a la Mañana</label>
    <input type="time" class="form-control form-control-lg" name="salida_manana" id="salida_manana" required>
  </div>
  <!-- Tolerancia entrada mañana -->
  <div class="mb-4">
    <label for="tolerancia_manana" class="form-label">Asignar Tolerancia de la entrada a la Mañana</label>
    <input type="time" class="form-control form-control-lg" name="tolerancia_manana" id="tolerancia_manana"
required>
  </div>
  <!-- Entrada de la tarde -->
  <div class="mb-4">
    <label for="entrada_tarde" class="form-label">Asignar Entrada a la Tarde</label>
    <input type="time" class="form-control form-control-lg" name="entrada_tarde" id="entrada_tarde" required>
  </div>
  <!-- Salida de la tarde -->
  <div class="mb-4">
    <label for="salida_tarde" class="form-label">Asignar Salida a la Tarde</label>
    <input type="time" class="form-control form-control-lg" name="salida_tarde" id="salida_tarde" required>
  </div>
  <!-- Tolerancia entrada tarde -->
  <div class="mb-4">
    <label for="tolerancia_tarde" class="form-label">Asignar Tolerancia de la entrada a la Tarde</label>
    <input type="time" class="form-control form-control-lg" name="tolerancia_tarde" id="tolerancia_tarde" required>
  </div>
  <!-- Botones guardar o cancelar -->
  <div class="d-flex justify-content-between">
    <button type="submit" class="btn btn-primary btn-lg">Guardar</button>
    <a href="{{ url_for('horarios') }}" class="btn btn-secondary btn-lg">Cancelar</a>
  </div>
</form>
</div>
<!-- Scripts JS -->
<!-- Variable global para usar dentro del archivo scripts.js -->
<script>const logoutUrl = "{{ url_for('logout') }}";</script>
<!-- Archivo JS externo con funciones de confirmacion -->
<script src="{{ url_for('static', filename='scripts.js') }}"></script>
</body>
</html>
- Editar Horarios.html
<!DOCTYPE html>
<html lang="es">
<head>

  <meta charset="UTF-8">
  <title>Editar Horario</title>
  <!-- Bootstrap CSS -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
  <!-- Archivo CSS personalizado para estilos propios -->
```



```
<link rel="stylesheet" href="{{ url_for('static', filename='estilos.css') }}">
</head>
<body>
<!-- Barra Lateral -->
<div class="sidebar">
    <!-- Titulo o nombre del sistema -->
    <h4 class="text-white text-center py-3">AsistOK</h4>
    <!-- Enlaces de navegacion a diferentes secciones -->
    <a href="{{ url_for('index') }}">Inicio</a>
    <a href="{{ url_for('marcacion') }}">Marcacion</a>
    <a href="{{ url_for('dispositivos') }}">Dispositivos</a>
    <a href="{{ url_for('empleados') }}">Empleados</a>
    <a href="{{ url_for('usuarios') }}">Usuarios</a>
    <a href="{{ url_for('departamentos') }}">Departamentos</a>
    <a href="{{ url_for('horarios') }}">Horarios</a>
    <!-- Enlace para cerrar sesion con función confirmacion -->
    <a href="#" onclick="confirmarLogout()">Cerrar sesión</a>
</div>
<!-- Contenido Principal -->
<div class="main-content">
    <!-- Boton para regresar a la lista de horarios -->
    <a href="{{ url_for('horarios') }}" class="btn btn-secondary btn-lg">Volver</a>
    <div class="container mt-4">
        <!-- Titulo de la seccion -->
        <h2 class="mb-4">Editar Horario</h2>
        <!-- Formulario para editar horario, envia datos por POST -->
        <form method="post" class="card shadow p-4" onsubmit="return confirmarGuardado()">
            <!-- Campo texto para descripcion del horario con valor actual -->
            <div class="mb-3">
                <label for="descripcion" class="form-label">Descripcion</label>
                <input type="text" class="form-control" id="descripcion" name="descripcion" value="{{ horarios.descripcion }}"
required>
            </div>

            <!-- Campo para hora de entrada a la mañana -->
            <div class="mb-3">
                <label for="entrada_manana" class="form-label">Entrada a la Mañana</label>
                <input type="time" class="form-control" id="entrada_manana" name="entrada_manana" value="{{
horarios.entrada_manana }}" required>
            </div>
            <!-- Campo para hora de salida a la mañana -->
            <div class="mb-3">
                <label for="salida_manana" class="form-label">Salida a la Mañana</label>
                <input type="time" class="form-control" id="salida_manana" name="salida_manana" value="{{
horarios.salida_manana }}" required>
            </div>
            <!-- Campo para tolerancia en la mañana -->
            <div class="mb-3">
                <label for="tolerancia_manana" class="form-label">Tolerancia a la Mañana</label>
                <input type="time" class="form-control" id="tolerancia_manana" name="tolerancia_manana" value="{{
horarios.tolerancia_manana }}" required>
            </div>
            <!-- Campo para hora de entrada a la tarde -->
            <div class="mb-3">
                <label for="entrada_tarde" class="form-label">Entrada a la Tarde</label>
                <input type="time" class="form-control" id="entrada_tarde" name="entrada_tarde" value="{{ horarios.entrada_tarde
}}" required>
            </div>
            <!-- Campo para hora de salida a la tarde -->
            <div class="mb-3">
                <label for="salida_tarde" class="form-label">Salida a la Tarde</label>
                <input type="time" class="form-control" id="salida_tarde" name="salida_tarde" value="{{ horarios.salida_tarde }}"
required>
            </div>
        </form>
    </div>
</div>
```





```

</div>
<!-- Campo para tolerancia en la tarde -->
<div class="mb-3">
  <label for="tolerancia_tarde" class="form-label">Tolerancia a la Tarde</label>
  <input type="time" class="form-control" id="tolerancia_tarde" name="tolerancia_tarde" value="{{
horarios.tolerancia_tarde }}" required>
</div>
<!-- Checkbox para marcar si el horario esta activo -->
<div class="form-check mb-3">
  <!-- Se marca si horarios.activo es igual a 1 -->

  <input type="checkbox" class="form-check-input" id="activo" name="activo" {% if horarios.activo == 1 %}checked{%
endif %}>
  <label class="form-check-label" for="activo">Activo</label>
</div>
<!-- Boton para enviar el formulario y guardar cambios -->
<button type="submit" class="btn btn-primary">Guardar cambios</button>
<!-- Botón para cancelar y volver sin guardar -->
<a href="{{ url_for('horarios') }}" class="btn btn-secondary ms-2">Cancelar</a>
</form>
</div>
</div>
<!-- Scripts JS -->
<!-- Variable global para URL de logout, utilizada en scripts.js -->
<script>const logoutUrl = "{{ url_for('logout') }}";</script>
<!-- Archivo JS externo con funciones para confirmaciones -->
<script src="{{ url_for('static', filename='scripts.js') }}"></script>
</body>
</html>
- Usuarios.html
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Usuarios</title>
  <!-- Bootstrap -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
  <!-- Estilo personalizado -->
  <link rel="stylesheet" href="{{ url_for('static', filename='estilos.css') }}">
  <!-- Estilos adicionales para DataTables -->
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.5.2/css/bootstrap.css">
  <link rel="stylesheet" href="https://cdn.datatables.net/2.3.0/css/dataTables.bootstrap4.css">
  <!-- Librerías JS necesarias para DataTables -->
  <script src="https://code.jquery.com/jquery-3.7.1.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.5.2/js/bootstrap.min.js"></script>
  <script src="https://cdn.datatables.net/2.3.0/js/dataTables.js"></script>
  <script src="https://cdn.datatables.net/2.3.0/js/dataTables.bootstrap4.js"></script>

  <!-- Inicialización de la tabla DataTable -->
  <script>
    $(document).ready(function() {
      $('#example').DataTable();
    });
  </script>
</head>
<body>
  <!-- Barra lateral de navegacion -->
  <div class="sidebar">
    <h4 class="text-white text-center py-3">AsistOK</h4>
    <!-- Enlaces a las distintas secciones del sistema -->

```

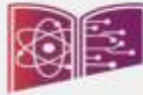



```
<a href="{{ url_for('index') }}">Inicio</a>
<a href="{{ url_for('marcacion') }}">Marcacion</a>
<a href="{{ url_for('dispositivos') }}">Dispositivos</a>
<a href="{{ url_for('empleados') }}">Empleados</a>
<a href="{{ url_for('usuarios') }}">Usuarios</a>
<a href="{{ url_for('departamentos') }}">Departamentos</a>
<a href="{{ url_for('horarios') }}">Horarios</a>
<a href="#" onclick="confirmarLogout()">Cerrar sesión</a>
</div>
<!-- Contenido principal de la pagina -->
<div class="main-content">
  <!-- Botón para agregar usuarios (solo si el usuario tiene permisos de administrador) -->
  <div class="mb-3">
    {% if session['rol'] %}
      <!-- Boton activo si es administrador -->
      <a href="{{ url_for('agregar_usuario') }}">
        <button class="btn btn-success">Agregar Usuario</button>
      </a>
    {% else %}
      <!-- Boton desactivado si no tiene permisos -->
      <button class="btn btn-secondary" onclick="alert('No tienes permisos para agregar usuarios.')" disabled>Agregar
Usuario</button>
    {% endif %}
  </div>
  <!-- Tabla que muestra la lista de usuarios -->
  <table id="example" class="table table-striped table-bordered">

    <thead>
      <tr>
        <th>ID</th>
        <th>Nombre</th>
        <th>Activo</th>
        <th>Editar</th>
      </tr>
    </thead>
    <tbody>
      {% for usuario in usuarios %}
      <tr>
        <td>{{ usuario.id_usuario }}</td>
        <td>{{ usuario.nombre_usuario }}</td>
        <td>{{ 'Si' if usuario.activo else 'No' }}</td>
        <!-- Boton para editar usuario (solo si tiene permisos) -->
        <td>
          {% if session['rol'] %}
            <a href="{{ url_for('editar_usuario', id=usuario.id_usuario) }}">
              <button class="btn btn-primary">Editar</button>
            </a>
          {% else %}
            <button class="btn btn-secondary" onclick="alert('No tienes permisos para editar usuarios.')"
disabled>Editar</button>
          {% endif %}
        </td>
      </tr>
      {% endfor %}
    </tbody>
  </table>
</div>
<!-- Variable global para la función de cierre de sesión (scripts.js) -->
<script>const logoutUrl = "{{ url_for('logout') }}";</script>
<!-- Archivo externo JS con funciones de confirmación como confirmarLogout() -->
<script src="{{ url_for('static', filename='scripts.js') }}"></script>
</body>
</html>
```



```
- Agregar Usuarios.html
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Agregar Usuario</title>
  <!-- Bootstrap -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
  <!-- Estilos personalizados -->
  <link rel="stylesheet" href="{{ url_for('static', filename='estilos.css') }}">
</head>
<body>
  <!-- Barra lateral de navegacion -->
  <div class="sidebar">
    <!-- Nombre del sistema -->
    <h4 class="text-white text-center py-3">AsistOK</h4>
    <!-- Menu de navegacion -->
    <a href="{{ url_for('index') }}">Inicio</a>
    <a href="{{ url_for('marcacion') }}">Marcacion</a>
    <a href="{{ url_for('dispositivos') }}">Dispositivos</a>
    <a href="{{ url_for('empleados') }}">Empleados</a>
    <a href="{{ url_for('usuarios') }}">Usuarios</a>
    <a href="{{ url_for('departamentos') }}">Departamentos</a>
    <a href="{{ url_for('horarios') }}">Horarios</a>
    <!-- Enlace para cerrar sesion con confirmacion -->
    <a href="#" onclick="confirmarLogout()">Cerrar sesión</a>
  </div>
  <!-- Contenido principal -->
  <div class="main-content">
    <!-- Boton para volver a la lista de usuarios -->
    <a href="{{ url_for('usuarios') }}" class="btn btn-secondary btn-lg">Volver</a>
    <!-- Contenedor principal del formulario -->
    <div class="container mt-4" style="max-width: 700px;">
      <!-- Titulo del formulario -->
      <h2 class="mb-4 text-center">Agregar Usuario</h2>
      <!-- Formulario para agregar un usuario nuevo -->
      <!-- Llama a una funcion JS para confirmar antes de enviar -->
      <form method="post" class="card shadow p-5" onsubmit="return confirmarGuardado()">
        <!-- Campo: Nombre del usuario -->
        <div class="mb-4">
          <label for="nombre" class="form-label">Nombre</label>
          <input type="text" class="form-control form-control-lg" name="nombre" id="nombre" required>
        </div>
        <!-- Campo: Contraseña -->
        <div class="mb-4">
          <label for="password" class="form-label">Contraseña</label>
          <input type="password" class="form-control form-control-lg" name="password" id="password" required>
        </div>
        <!-- Campo: Seleccion de rol -->
        <div class="mb-4">
          <label for="rol" class="form-label">Roles</label>
          <select name="rol" id="rol" class="form-select form-select-lg" required>
            <option value="">-- Seleccione un Rol --</option>
            <option value="True">Administrador</option>
            <option value="False">Usuario</option>
          </select>
        </div>
        <!-- Botones para guardar o cancelar -->
        <div class="d-flex justify-content-between">
          <button type="submit" class="btn btn-primary btn-lg">Guardar</button>
          <a href="{{ url_for('usuarios') }}" class="btn btn-secondary btn-lg">Cancelar</a>
        </div>
      </form>
    </div>
  </div>
</body>
</html>
```



```
</div>
</div>
<!-- Scripts JS -->
<!-- Variable global para usar dentro del archivo scripts.js -->
<script>const logoutUrl = "{{ url_for('logout') }}";</script>
<!-- Archivo JS externo con funciones de confirmacion -->
<script src="{{ url_for('static', filename='scripts.js') }}"></script>
</body>
</html>
- Editar Usuarios.html
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Editar Usuario</title>

  <!-- Bootstrap CSS -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
  <!-- Archivo CSS personalizado para estilos propios -->
  <link rel="stylesheet" href="{{ url_for('static', filename='estilos.css') }}">
</head>
<body>
  <!-- Barra Lateral -->
  <div class="sidebar">
    <h4 class="text-white text-center py-3">AsistOK</h4>
    <a href="{{ url_for('index') }}">Inicio</a>
    <a href="{{ url_for('marcacion') }}">Marcacion</a>
    <a href="{{ url_for('dispositivos') }}">Dispositivos</a>
    <a href="{{ url_for('empleados') }}">Empleados</a>
    <a href="{{ url_for('usuarios') }}">Usuarios</a>
    <a href="{{ url_for('departamentos') }}">Departamentos</a>
    <a href="{{ url_for('horarios') }}">Horarios</a>
    <a href="#" onclick="confirmarLogout()">Cerrar sesión</a>
  </div>
  <!-- Contenido Principal -->
  <div class="main-content">
    <a href="{{ url_for('usuarios') }}" class="btn btn-secondary btn-lg">Volver</a>
    <div class="container mt-4">
      <h2 class="mb-4">Editar Usuario</h2>
      <form method="post" class="card shadow p-4" onsubmit="return confirmarGuardado()">
        <!-- Campo: Nombre del Usuario -->
        <div class="mb-3">
          <label for="nombre" class="form-label">Nombre</label>
          <input type="text" class="form-control" id="nombre" name="nombre" value="{{ usuarios.nombre_usuario }}"
required>
        </div>
        <!-- Checkbox: Usuario Activo -->
        <div class="form-check mb-3">
          <input type="checkbox" class="form-check-input" id="activo" name="activo" {% if usuarios.activo == 1 %}checked{%
endif %}>
          <label class="form-check-label" for="activo">Activo</label>
        </div>
        <!-- Campo: Contraseña -->
        <div class="mb-3">
          <label for="password" class="form-label">Contraseña</label>
          <input type="password" class="form-control" id="password" name="password">
        </div>
        <!-- Campo: Rol del Usuario -->
        <div class="mb-3">
          <label for="rol" class="form-label">Roles</label>
          <select name="rol" id="rol" class="form-select form-select-lg" required>
            <option value="">-- Seleccione un Rol --</option>

```

```
<option value="True" {% if usuarios.rol == True %}>selected{% endif %}>Administrador</option>
<option value="False" {% if usuarios.rol == False %}>selected{% endif %}>Usuario</option>
</select>
</div>
<!-- Botones -->
<button type="submit" class="btn btn-primary">Guardar cambios</button>
<a href="{{ url_for('usuarios') }}" class="btn btn-secondary ms-2">Cancelar</a>
</form>
</div>
</div>
<!-- Scripts JS -->
<script>const logoutUrl = "{{ url_for('logout') }}";</script>
<script src="{{ url_for('static', filename='scripts.js') }}"></script>
</body>
</html>
```

6.7.2. Python

- App.py

```
from flask import Flask, render_template, request, redirect, session, url_for, flash
from modulos.conexionDB import conexion as con
from werkzeug.security import check_password_hash
from modulos.funciones import (
    obtener_marcacion, Capturar_DatosZK, dispositivo_ZK, obtener_dispositivos,
    obtener_empleados, obtener_horarios, obtener_departamentos, obtener_usuarios,
    obtener_empleado_Formulario, actualizar_empleado, empleados_formulario, dispositivo_formulario,
    obtener_dispositivo_Formulario, actualizar_dispositivo, horarios_formulario, obtener_horario_Formulario,
    actualizar_horarios, obtener_horarios_Departamentos, departamento_formulario, obtener_departamento_Formulario,
    actualizar_departamentos, PasswordHash, usuario_formulario, actualizar_usuarios, obtener_usuarios_Formulario,
    Departamentos
)
from modulos.generate_zk_testdata import data
app = Flask(__name__)
app.secret_key = 'clave_super_secreta'

# Pantalla login
@app.route('/', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        usuario = request.form['usuario']
        password = request.form['password']
        cursor = con.cursor()
        cursor.execute("SELECT * FROM usuarios WHERE nombre_usuario = %s", (usuario,))
        datos = cursor.fetchone()
        cursor.close()
        if datos:
            if check_password_hash(datos['password_usuario'], password):
                session['usuario'] = datos['nombre_usuario']
                session['rol'] = bool(datos['rol'])
                return redirect(url_for("index"))
            else:
                flash('Contraseña incorrecta', 'danger')
        else:
            flash('Usuario no encontrado', 'danger')
    return render_template("login.html")

# Pantalla index-Home
@app.route('/index', methods=['GET', 'POST'])
def index():
    if 'usuario' not in session:
        return redirect(url_for("login"))
    if request.method == 'POST':
        accion = request.form.get('accion')
        if accion == 'conectar':
            ip = request.form.get('ip_dispositivo')
```

```
        puerto = request.form.get('puerto_dispositivo')
        dispositivo_ZK(ip, puerto)
        flash("Conectado al dispositivo", 'success')
    elif accion == 'actualizar_todo':
        Capturar_DatosZK()
        flash("Todos los dispositivos actualizados correctamente.", 'success')

    return render_template("index.html")

# Pantalla de dispositivos
@app.route('/dispositivos')
def dispositivos():
    if 'usuario' not in session:
        return redirect(url_for('login'))
    dispositivos = obtener_dispositivos()
    return render_template("dispositivos.html", dispositivos=dispositivos)

# Pantalla agregar dispositivos
@app.route('/agregar_dispositivo', methods=['GET', 'POST'])
def agregar_dispositivo():
    if 'usuario' not in session:
        return redirect(url_for('login'))
    if request.method == 'POST':
        descripcion = request.form['descripcion']
        nombre = request.form['nombre']
        puerto = request.form['puerto']
        ip = request.form['ip']
        dispositivo_formulario(descripcion, nombre, puerto, ip)
        flash("Dispositivo agregado correctamente", 'success')
        return redirect(url_for('dispositivos'))
    return render_template('agregar_dispositivos.html')

# Pantalla para editar dispositivos
@app.route('/editar_dispositivo/<int:id>', methods=['GET', 'POST'])
def editar_dispositivo(id):
    if 'usuario' not in session:
        return redirect(url_for('login'))
    dispositivos = obtener_dispositivo_Formulario(id)
    if request.method == 'POST':
        descripcion = request.form['descripcion']
        nombre = request.form['nombre']
        activo = 1 if 'activo' in request.form else 0
        puerto = request.form['puerto']
        ip = request.form['ip']
        actualizar_dispositivo(descripcion, nombre, activo, puerto, ip, )
        flash("Dispositivo actualizado correctamente", 'success')
        return redirect(url_for('empleados'))

    return render_template('editar_dispositivos.html', dispositivos=dispositivos)

# Pantalla de marcacion
@app.route('/marcacion')
def marcacion():
    if 'usuario' not in session:
        return redirect(url_for('login'))
    marcacion = obtener_marcacion()
    return render_template("marcacion.html", marcacion=marcacion)

# Pantalla de empleados
@app.route('/empleados')
def empleados():
    if 'usuario' not in session:
        return redirect(url_for('login'))
    empleados = obtener_empleados()
    return render_template("empleados.html", empleados=empleados)

# Pantalla agregar empleados
@app.route('/agregar_empleado', methods=['GET', 'POST'])
def agregar_empleado():
```



```
if 'usuario' not in session:
    return redirect(url_for('login'))
departamentos = Departamentos()
if request.method == 'POST':
    id_employado = request.form['id']
    nombre = request.form['nombre']
    departamento = int(request.form['departamento'])
    empleados_formulario(id_employado, nombre, departamento)
    flash('Empleado agregado correctamente', 'success')
    return redirect(url_for('empleados'))
return render_template('agregar_empleados.html', departamentos=departamentos)
```

Pantalla para editar empleados

```
@app.route('/editar_employado/<int:id>', methods=['GET', 'POST'])
def editar_employado(id):
    if 'usuario' not in session:
        return redirect(url_for('login'))
    empleado = obtener_employado_Formulario(id)
    departamentos = Departamentos()
```

```
if request.method == 'POST':
    id_employado = request.form['id']
    nombre = request.form['nombre']
    activo = 1 if 'activo' in request.form else 0
    departamento = request.form['departamento']
    actualizar_employado(id_employado, nombre, activo, departamento)
    flash('Empleado actualizado correctamente', 'success')
    return redirect(url_for('empleados'))
return render_template('editar_empleados.html', empleado=empleado, departamentos=departamentos)
```

Pantalla de horarios

```
@app.route('/horarios')
def horarios():
    if 'usuario' not in session:
        return redirect(url_for('login'))
    horarios = obtener_horarios()
    return render_template("horarios.html", horarios=horarios)
```

Pantalla agregar horario

```
@app.route('/agregar_horario', methods=['GET', 'POST'])
def agregar_horario():
    if 'usuario' not in session:
        return redirect(url_for('login'))
    if request.method == 'POST':
        descripcion = request.form['descripcion']
        entrada_manana = request.form['entrada_manana']
        salida_manana = request.form['salida_manana']
        tolerancia_manana = request.form['tolerancia_manana']
        entrada_tarde = request.form['entrada_tarde']
        salida_tarde = request.form['salida_tarde']
        tolerancia_tarde = request.form['tolerancia_tarde']
        horarios_formulario(descripcion, entrada_manana, salida_manana, tolerancia_manana, entrada_tarde, salida_tarde, tolerancia_tarde)
        flash('Horario agregado correctamente', 'success')
        return redirect(url_for('horarios'))
    return render_template('agregar_horarios.html')
```

Pantalla para editar horarios

```
@app.route('/editar_horario/<int:id>', methods=['GET', 'POST'])
def editar_horario(id):
    if 'usuario' not in session:

        return redirect(url_for('login'))
    horarios = obtener_horario_Formulario(id)
    if request.method == 'POST':
        descripcion = request.form['descripcion']
```

```
entrada_manana = request.form['entrada_manana']
salida_manana = request.form['salida_manana']
tolerancia_manana = request.form['tolerancia_manana']
entrada_tarde = request.form['entrada_tarde']
salida_tarde = request.form['salida_tarde']
tolerancia_tarde = request.form['tolerancia_tarde']
activo = 1 if 'activo' in request.form else 0
actualizar_horarios(descripcion, entrada_manana, salida_manana, tolerancia_manana, entrada_tarde, salida_tarde,
tolerancia_tarde, activo)
flash('Horario actualizado correctamente', 'success')
return redirect(url_for('horarios'))
return render_template('editar_horarios.html', horarios=horarios)

# Pantalla de departamentos
@app.route('/departamentos')
def departamentos():
    if 'usuario' not in session:
        return redirect(url_for('login'))
    departamentos = obtener_departamentos()
    return render_template("departamentos.html", departamentos=departamentos)

# Pantalla agregar departamento
@app.route('/agregar_departamento', methods=['GET', 'POST'])
def agregar_departamento():
    if 'usuario' not in session:
        return redirect(url_for('login'))
    horarios = obtener_horarios_Departamentos()
    if request.method == 'POST':
        nombre = request.form['descripcion']
        horario = int(request.form['id_horario'])
        departamento_formulario(nombre, horario)
        flash('Departamento agregado correctamente', 'success')
        return redirect(url_for('departamentos'))
    return render_template('agregar_departamentos.html', horarios=horarios)

# Pantalla para editar departamentos
@app.route('/editar_departamento/<int:id>', methods=['GET', 'POST'])

def editar_departamento(id):
    if 'usuario' not in session:
        return redirect(url_for('login'))
    departamentos = obtener_departamento_Formulario(id)
    horarios = obtener_horarios_Departamentos()
    if request.method == 'POST':
        nombre = request.form['nombre']
        horario = int(request.form['id_horario'])
        activo = 1 if 'activo' in request.form else 0
        actualizar_departamentos(nombre, horario, activo)
        flash('Departamento actualizado correctamente', 'success')
        return redirect(url_for('departamentos'))
    return render_template('editar_departamentos.html', departamentos=departamentos, horarios=horarios)

# Pantalla de usuarios
@app.route('/usuarios')
def usuarios():
    if 'usuario' not in session:
        return redirect(url_for('login'))

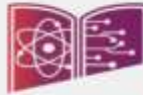
    usuarios = obtener_usuarios()
    return render_template("usuarios.html", usuarios=usuarios)

# Pantalla agregar usuario
@app.route('/agregar_usuario', methods=['GET', 'POST'])
def agregar_usuario():
    if 'usuario' not in session:
        return redirect(url_for('login'))
    if request.method == 'POST':
        nombre = request.form['nombre']
```

```
password = request.form['password']
rol = request.form['rol'] == 'True' # Este es tu campo booleano desde el select
if not password:
    flash('La contraseña no puede estar vacía.', 'danger')
    return redirect(url_for('agregar_usuario'))
password_hash = generate_password_hash(password)
usuario_formulario(nombre, password_hash, rol)
flash('Usuario agregado correctamente', 'success')
return redirect(url_for('usuarios'))

return render_template('agregar_usuarios.html')
# Pantalla para editar usuarios
@app.route('/editar_usuario/<int:id>', methods=['GET', 'POST'])
def editar_usuario(id):
    if 'usuario' not in session:
        return redirect(url_for('login'))
    usuarios = obtener_usuarios_Formulario(id)
    if request.method == 'POST':
        nombre = request.form['nombre']
        password = request.form['password']
        password_hash = password_hash(password)
        rol = request.form['rol'] == 'True'
        activo = 1 if 'activo' in request.form else 0
        actualizar_usuarios(nombre, password_hash, activo, rol)
        flash('Usuario actualizado correctamente', 'success')
        return redirect(url_for('usuarios'))
    return render_template('editar_usuarios.html', usuarios=usuarios)
# Pantalla logout
@app.route('/logout')
def logout():
    session.pop('usuario', None)
    flash('Sesión cerrada correctamente', 'info')
    return redirect(url_for('login'))
if __name__ == '__main__':
    app.run(debug=True)
- ConexionDB.py
#Funcion para conectar con la BD
import pymysql
def obtener_conexion():
    return pymysql.connect(host='localhost', user='root', password="", db='sistemamc',
    cursorclass=pymysql.cursors.DictCursor)
    conexion = obtener_conexion()
- Funciones.py
from modulos.conexionDB import obtener_conexion
from zk import ZK
from datetime import datetime, timedelta, time
from modulos.generate_zk_testdata import data
from werkzeug.security import generate_password_hash

# Funcion para convertir una contraseña a hash
def PasswordHash(password):
    hash = generate_password_hash(password)
    return hash
#Funcion para obtener los id y nombre de los departamentos
def Departamentos():
    conexion = obtener_conexion()
    departamentos = []
    with conexion.cursor() as cursor:
        cursor.execute("SELECT id_departamento, nombre_departamento FROM departamentos")
        departamentos = cursor.fetchall()
    conexion.close()
    return departamentos
```



#Funcion para insertar usuarios a la BD

```
def UsuariIn(users):
    cursor = obtener_conexion()
    for user in users:
        consulta = "INSERT INTO usuarios(id_user, Nombre) VALUES (%s, %s);"
        cursor.execute(consulta, (user.user_id, user.name))
        cursor.commit()
    cursor.close()
```

#Funcion para obtener todas las ID de todos los empleados

```
def obtener_empleados_TODOS():
    conexion = obtener_conexion()
    empleados_TODOS = []
    with conexion.cursor() as cursor:
        cursor.execute("SELECT id_empleado FROM empleado")
        empleados_TODOS= cursor.fetchall()
    conexion.close()
    return empleados_TODOS
```

#Funcion para obtener la ultima marcacion de un empleado cuya marcacion es entrada

```
def ultima_marcacion_empleado(id_empleado, id_dispositivo, fecha_marcacion):
    conexion = obtener_conexion()
    tipo_salida = 'entrada'
    with conexion.cursor() as cursor:

        cursor.execute("SELECT marcacion FROM marcados WHERE FK_empleado = %s AND FK_dispositivos = %s AND tipo = %s AND DATE(marcacion) = %s ORDER BY marcacion DESC LIMIT 1", (id_empleado, id_dispositivo, tipo_salida, fecha_marcacion))
        ultima_marcacion_empleado = cursor.fetchone()
        conexion.close()
    if ultima_marcacion_empleado:
        return ultima_marcacion_empleado['marcacion']
    return None
```

#Funcion para obtener el ultimo empleado de la BD

```
def obtener_ultimo_empleado(id_empleado):
    conexion = obtener_conexion()
    with conexion.cursor() as cursor:
        cursor.execute("SELECT id_empleado FROM empleado WHERE id_empleado = %s", (id_empleado))
        ultimo_empleado = cursor.fetchone()
    conexion.close()
    return ultimo_empleado
```

#Funcion para Insertar empleados a la BD

```
def Empleados(empleado, departamento_unico):
    activo = True
    conexion = obtener_conexion()
    cursor = conexion.cursor()
    consulta = "INSERT INTO empleado(id_empleado, nombre_empleado, activo, FK_departamento) VALUES (%s, %s, %s, %s);"
    cursor.execute(consulta, (empleado.user_id, empleado.name, activo, departamento_unico ))
    conexion.commit()
    cursor.close()
    conexion.close()
```

#Funcion para Insertar dispositivos a la BD

```
def dispositivo_formulario(descripcion, nombre, puerto, ip):
    activo = True
    conexion = obtener_conexion()
    cursor = conexion.cursor()
    consulta = "INSERT INTO dispositivos(descripcion_dispositivo, nombre_dispositivo, activo, puerto, IP_dispositivo) VALUES (%s, %s, %s, %s, %s);"
    cursor.execute(consulta, (descripcion, nombre, activo, puerto, ip ))
    conexion.commit()
    cursor.close()
    conexion.close()
```





#Funcion para Insertar horarios a la BD

```
def horarios_formulario(descripcion, entrada_manana, salida_manana, tolerancia_manana, entrada_tarde, salida_tarde, tolerancia_tarde):
    activo = True
    conexion = obtener_conexion()
    cursor = conexion.cursor()
    consulta = "INSERT INTO horarios(descripcion, entrada_manana, salida_manana, tolerancia_manana, entrada_tarde, salida_tarde, tolerancia_tarde, activo) VALUES (%s, %s, %s, %s, %s, %s, %s, %s);"
    cursor.execute(consulta, (descripcion, entrada_manana, salida_manana, tolerancia_manana, entrada_tarde, salida_tarde, tolerancia_tarde, activo))
    conexion.commit()
    cursor.close()
    conexion.close()
```

#Funcion para Insertar departamentos a la BD

```
def departamento_formulario(nombre, horario):
    activo = True
    conexion = obtener_conexion()
    cursor = conexion.cursor()
    consulta = "INSERT INTO departamentos(nombre_departamento, activo, FK_horario) VALUES (%s, %s, %s);"
    cursor.execute(consulta, (nombre, activo, horario))
    conexion.commit()
    cursor.close()
    conexion.close()
```

#Funcion para Insertar usuario a la BD

```
def usuario_formulario(nombre, password, rol):
    activo = True
    conexion = obtener_conexion()
    cursor = conexion.cursor()
    consulta = "INSERT INTO usuarios(nombre_usuario, password_usuario, activo, rol) VALUES (%s, %s, %s, %s);"
    cursor.execute(consulta, (nombre, password, activo, rol))
    conexion.commit()
    cursor.close()
    conexion.close()
```

#Funcion para Insertar empleados a la BD

```
def empleados_formulario(id, nombre, departamento):
    activo = True
    conexion = obtener_conexion()
    cursor = conexion.cursor()

    consulta = "INSERT INTO empleado(id_empleado, nombre_empleado, activo, FK_departamento) VALUES (%s, %s, %s, %s);"
    cursor.execute(consulta, (id, nombre, activo, departamento))
    conexion.commit()
    cursor.close()
    conexion.close()
```

#Funcion para Insertar marcaciones a la BD

```
def Marcados(marca, nombre_ZK, tipo, detalle, horas_trabajadas):
    conexion = obtener_conexion()
    cursor = conexion.cursor()
    consulta = "INSERT INTO marcados(marcacion, tipo, detalle, horas_trabajadas, FK_empleado, FK_dispositivos) VALUES (%s, %s, %s, %s, %s, %s);"
    cursor.execute(consulta, (marca.timestamp, tipo, detalle, horas_trabajadas, marca.user_id, nombre_ZK))
    conexion.commit()
    cursor.close()
    conexion.close()
```

#Funcion para obtener usuarios de la BD

```
def obtener_usuarios():
    usuarios = []
    conexion = obtener_conexion()
    with conexion.cursor() as cursor:
        cursor.execute("SELECT * FROM usuarios")
        usuarios = cursor.fetchall()
    conexion.close()
    return usuarios
```



#Funcion para obtener horarios de la BD

```
def obtener_horarios_Departamentos():  
    horarios = []  
    conexion = obtener_conexion()  
    with conexion.cursor() as cursor:  
        cursor.execute("SELECT id_horario, descripcion FROM horarios")  
        horarios = cursor.fetchall()  
    conexion.close()  
    return horarios
```

#Funcion para obtener usuario Roles de la BD

```
def obtener_usuario_rol(id):  
    usuarios = []  
    conexion = obtener_conexion()  
    with conexion.cursor() as cursor:  
        cursor.execute("SELECT id_usuario, rol FROM usuarios")  
        usuarios = cursor.fetchall()  
    conexion.close()  
    return usuarios
```

#Funcion para obtener todos los dispositivos de la BD los que estan activos

```
def obtener_dispositivos_activos():  
    dispositivos = []  
    conexion = obtener_conexion()  
    with conexion.cursor() as cursor:  
        cursor.execute("SELECT * FROM dispositivos WHERE activo = 1")  
        dispositivos = cursor.fetchall()  
    conexion.close()  
    return dispositivos
```

#Funcion para obtener todos los dispositivos de la BD

```
def obtener_dispositivos():  
    dispositivos = []  
    conexion = obtener_conexion()  
    with conexion.cursor() as cursor:  
        cursor.execute("SELECT * FROM dispositivos")  
        dispositivos = cursor.fetchall()  
    conexion.close()  
    return dispositivos
```

#Funcion para obtener la ID de un dispositivo en base a los paramentros de IP y Puerto

```
def obtener_dispositivos_ZK(ip, puerto):  
    conexion = obtener_conexion()  
    with conexion.cursor() as cursor:  
        cursor.execute(f"SELECT id_dispositivo FROM dispositivos WHERE IP_dispositivo = {ip} WHERE puerto = {puerto}")  
        dispositivos_ZK = cursor.fetchone()  
    conexion.close()  
    return dispositivos_ZK
```

#Funcion para obtener especificamente un dispositivo que prueba para uso si no se dispone de un dispositivo real

```
def obtener_dispositivos_Prueba():  
    conexion = obtener_conexion()  
    with conexion.cursor() as cursor:  
        cursor.execute("SELECT id_dispositivo FROM dispositivos WHERE id_dispositivo = 2")  
        dispositivosPrueba = cursor.fetchone()  
    conexion.close()  
    return dispositivosPrueba
```

#Funcion para obtener un horario de la BD a traves de la FK de departamento

```
def obtener_horario(id):  
    horarios = []  
    conexion = obtener_conexion()  
    with conexion.cursor() as cursor:  
        cursor.execute(f"SELECT * FROM horarios WHERE id_horario = {id}")  
        horarios = cursor.fetchall()  
    conexion.close()  
    return horarios
```

#Funcion para obtener todos los horarios de la BD

```
def obtener_horarios():
    horarios = []
    conexion = obtener_conexion()
    with conexion.cursor() as cursor:
        cursor.execute(f"SELECT * FROM horarios")
        horarios = cursor.fetchall()
    conexion.close()
    return horarios

#Funcion para obtener un departamento de la BD a traves de la FK del empleado
def obtener_departamento(id):
    conexion = obtener_conexion()
    with conexion.cursor() as cursor:
        cursor.execute("SELECT FK_horarios FROM departamentos WHERE id_departamento = %s", (id))
        departamentos = cursor.fetchone()
    conexion.close()
    return departamentos

#Funcion para obtener departamentos para mostrar en la tabla web del sistema
def obtener_departamentos():
    departamentos = []
    conexion = obtener_conexion()
    with conexion.cursor() as cursor:
        cursor.execute("""SELECT
                        d.id_departamento,
                        d.nombre_departamento,
                        d.activo,
                        h.descripcion
                        FROM departamentos AS d

                        JOIN horarios AS h ON d.FK_horarios = h.id_horario""")
        departamentos = cursor.fetchall()
    conexion.close()
    return departamentos

#Funcion para obtener un empleado de la BD a traves de su id del marcador
def obtener_empleado(id):
    conexion = obtener_conexion()
    with conexion.cursor() as cursor:
        cursor.execute(f"SELECT FK_departamento FROM empleado WHERE id_empleado = {id}")
        empleados = cursor.fetchone()
    conexion.close()
    return empleados

#Funcion para obtener empleados para el formulario para editar empleados
def obtener_empleado_Formulario(id):
    conexion = obtener_conexion()
    with conexion.cursor() as cursor:
        cursor.execute(f"SELECT * FROM empleado WHERE id_empleado = {id}")
        empleados = cursor.fetchone()
    conexion.close()
    return empleados

#Funcion para obtener horarios para el formulario para editar horarios
def obtener_horario_Formulario(id):
    conexion = obtener_conexion()
    with conexion.cursor() as cursor:
        cursor.execute(f"SELECT * FROM horarios WHERE id_horario = {id}")
        horarios = cursor.fetchone()
    conexion.close()
    return horarios

#Funcion para obtener departamentos para el formulario para editar horarios
def obtener_departamento_Formulario(id):
    conexion = obtener_conexion()
    with conexion.cursor() as cursor:
        cursor.execute(f"SELECT * FROM departamentos WHERE id_departamento = {id}")
        departamentos = cursor.fetchone()
    conexion.close()
    return departamentos
```

return departamentos

#Funcion para obtener usuarios para el formulario para editar horarios

```
def obtener_usuarios_Formulario(id):  
    conexion = obtener_conexion()  
    with conexion.cursor() as cursor:  
        cursor.execute(f"SELECT * FROM usuarios WHERE id_usuario = {id}")  
        usuarios = cursor.fetchall()  
    conexion.close()  
    return usuarios
```

#Funcion para obtener dispositivos para el formulario para editar dispositivos

```
def obtener_dispositivo_Formulario(id):  
    conexion = obtener_conexion()  
    with conexion.cursor() as cursor:  
        cursor.execute(f"SELECT * FROM dispositivos WHERE id_dispositivo = {id}")  
        dispositivos = cursor.fetchall()  
    conexion.close()  
    return dispositivos
```

#Funcion para Actualizar empleados de la tabla a de la BD

```
def actualizar_empleado(id, nombre, activo, departamento):  
    conexion = obtener_conexion()  
    with conexion.cursor() as cursor:  
        cursor.execute(f"UPDATE empleado SET nombre_empleado = %s, activo = %s, FK_departamento = %s WHERE  
id_empleado = %s", (nombre, activo, departamento, id))  
    conexion.commit()  
    conexion.close()
```

#Funcion para Actualizar dispositivos de la tabla a de la BD

```
def actualizar_dispositivo(descripcion, nombre, activo, puerto, ip):  
    conexion = obtener_conexion()  
    with conexion.cursor() as cursor:  
        cursor.execute(f"UPDATE dispositivos SET descripcion_dispositivo = %s, nombre_dispositivo = %s, activo = %s, puerto =  
%s", (descripcion, nombre, activo, puerto, ip))  
    conexion.commit()  
    conexion.close()
```

#Funcion para Actualizar horarios de la tabla a de la BD

```
def actualizar_horarios(id, descripcion, entrada_manana, salida_manana, tolerancia_manana, entrada_tarde, salida_tarde,  
tolerancia_tarde, activo):  
    conexion = obtener_conexion()  
    with conexion.cursor() as cursor:
```

```
        cursor.execute(f"UPDATE dispositivos SET descripcion = %s, entrada_manana = %s, salida_manana = %s,  
tolerancia_manana = %s, SET entrada_tarde = %s, SET salida_tarde = %s, SET tolerancia_tarde = %s, SET activo = %s",  
(descripcion, entrada_manana, salida_manana, tolerancia_manana, entrada_tarde, salida_tarde, tolerancia_tarde, activo))  
    conexion.commit()  
    conexion.close()
```

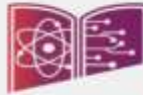
#Funcion para Actualizar departamentos de la tabla a de la BD

```
def actualizar_departamentos(nombre, activo, horario):  
    conexion = obtener_conexion()  
    with conexion.cursor() as cursor:  
        cursor.execute(f"UPDATE departamentos SET nombre = %s, activo = %s, FK_horario = %s", (nombre, activo, horario))  
    conexion.commit()  
    conexion.close()
```

#Funcion para Actualizar usuarios de la tabla a de la BD

```
def actualizar_usuarios(nombre, password, activo, rol):  
    conexion = obtener_conexion()  
    with conexion.cursor() as cursor:  
        cursor.execute(f"UPDATE usuarios SET nombre_usuario = %s, password = %s, activo = %s, rol = %s", (nombre, password,  
activo, rol))  
    conexion.commit()  
    conexion.close()
```

#Funcion para obtener las empleados de la BD



```
def obtener_empleados():
    empleados = []
    conexion = obtener_conexion()
    with conexion.cursor() as cursor:
        cursor.execute("""SELECT
            e.id_empleado,
            e.nombre_empleado,
            e.activo,
            d.nombre_departamento
        FROM empleado AS e
        JOIN departamentos AS d ON e.FK_departamento = d.id_departamento""")
    empleados = cursor.fetchall()
    conexion.close()
    return empleados
```

#Funcion para obtener las marcaciones de la BD

```
def obtener_marcacion():
    cursor = obtener_conexion()

    marcacion = []
    with cursor.cursor() as cursor:
        cursor.execute("""SELECT
            m.id_marcacion,
            m.marcacion,
            m.tipo,
            m.detalle,
            m.horas_trabajadas,
            e.nombre_empleado,
            d.nombre_dispositivo
        FROM marcados AS m
        JOIN empleado AS e ON m.FK_empleado = e.id_empleado
        JOIN dispositivos AS d ON m.FK_dispositivos = d.id_dispositivo""")
    marcacion = cursor.fetchall()
    obtener_conexion().close()
    return marcacion
```

#Funcion para capturar los datos de todos los dispositivos uno a uno

```
def Capturar_DatosZK():
    try:
        dispositivos = obtener_dispositivos()
        for dispositivo in dispositivos:
            zk = ZK(dispositivo["IP_dispositivo"], port=dispositivo["puerto"], timeout=5)
            conn = zk.connect()
            conn.disable_device()
            empleados = conn.get_users()
            marcaciones = conn.get_attendance()
            conn.enable_device()
            conn.disconnect()
            Actualizar_Datos_Empleados(empleados)
            nombre_ZK = dispositivo["id_dispositivo"]
            Actualizar_Datos(nombre_ZK, marcaciones)
    except Exception as e:
        print(f"Error: {e}")
        dispositivoPrueba = obtener_dispositivos_Prueba()
        generador_marcaciones = data
        nombre_ZK = dispositivoPrueba["id_dispositivo"]
        Actualizar_Datos(nombre_ZK, generador_marcaciones)
```

#Funcion que obtiene los datos del formulario de dispositivos para hacer la conexion y tomar los datos

```
def dispositivo_ZK(ip, puerto):
    zk = ZK(ip, port=int(puerto), timeout=5)
    conn = zk.connect()
    conn.disable_device()
    empleados = conn.get_users
    marcaciones = conn.get_attendance()
```





```
conn.enable_device()
conn.disconnect()
Actualizar_Datos_Empleados(empleados)
nombre_ZK = obtener_dispositivos_ZK(ip, puerto)
Actualizar_Datos(nombre_ZK, marcaciones)
```

#Funcion para convertir un valor a time

```
def asegurar_time(valor):
    if isinstance(valor, timedelta):
        return (datetime.min + valor).time()
    return valor
```

#Funcion para saber si la marcacion es de tipo Entrada o Salida y cuantas horas trabajo y que en condiciones llego al trabajo

```
def EntradaoSalida(marcacion, id_dispositivo):
    empleado = obtener_empleado(marcacion.user_id)
    departamento = obtener_departamento(empleado['FK_departamento'])
    horarios_lista = obtener_horario(departamento['FK_horarios'])
    horarios = horarios_lista[0]
    fecha_marcacion = marcacion.timestamp.date()
    hora_marcacion = marcacion.timestamp
    ultima_marcacion = ultima_marcacion_empleado(marcacion.user_id, id_dispositivo, fecha_marcacion)
    entrada_manana = datetime.combine(fecha_marcacion, asegurar_time(horarios['entrada_manana']))
    tolerancia_manana = datetime.combine(fecha_marcacion, asegurar_time(horarios['tolerancia_manana']))
    salida_manana = datetime.combine(fecha_marcacion, asegurar_time(horarios['salida_manana']))
    entrada_tarde = datetime.combine(fecha_marcacion, asegurar_time(horarios['entrada_tarde']))
    tolerancia_tarde = datetime.combine(fecha_marcacion, asegurar_time(horarios['tolerancia_tarde']))
    salida_tarde = datetime.combine(fecha_marcacion, asegurar_time(horarios['salida_tarde']))
    tipo = ""
    detalle = ""
    horas_trabajadas = timedelta(0)
    horas_trabajadas_decimales = None
```

Salida mañana (verificar antes de entrada tarde para evitar conflictos)

```
if salida_manana - timedelta(minutes=10) <= hora_marcacion < entrada_tarde:
    tipo = 'salida'
    detalle = None
    if ultima_marcacion:
        horas_trabajadas = hora_marcacion - ultima_marcacion
        horas_trabajadas_decimales = timedelta(hours=(horas_trabajadas.total_seconds() / 3600))
    else:
        horas_trabajadas = hora_marcacion - entrada_manana
        horas_trabajadas_decimales = timedelta(hours=(horas_trabajadas.total_seconds() / 3600))
```

Entrada mañana

```
elif entrada_manana <= hora_marcacion <= tolerancia_manana:
    tipo = 'entrada'
    detalle = 'a tiempo'
elif tolerancia_manana < hora_marcacion < salida_manana:
    tipo = 'entrada'
    detalle = 'tarde'
```

Salida tarde (verifica antes que cualquier otra entrada por la tarde)

```
elif salida_tarde - timedelta(minutes=10) <= hora_marcacion:
    tipo = 'salida'
    detalle = None
    if ultima_marcacion:
        horas_trabajadas = hora_marcacion - ultima_marcacion
        horas_trabajadas_decimales = timedelta(hours=(horas_trabajadas.total_seconds() / 3600))
    else:
        horas_trabajadas = hora_marcacion - entrada_tarde
        horas_trabajadas_decimales = timedelta(hours=(horas_trabajadas.total_seconds() / 3600))
```

Entrada tarde

```
elif entrada_tarde <= hora_marcacion <= tolerancia_tarde:
    tipo = 'entrada'
    detalle = 'a tiempo'
```




```
elif tolerancia_tarde < hora_marcacion < salida_tarde:  
    tipo = 'entrada'  
    detalle = 'tarde'  
return tipo, detalle, horas_trabajadas_decimales
```

#Funcion para filtrar marcaciones, cuyas marcaciones filtran si estan realacionadas con los empleados de la BD

```
def Filtrados(marcaciones):  
    empleados_id = obtener_empleados_TODOS()  
    ids_validos = {e['id_empleado'] for e in empleados_id}  
    marcas_filtradas = [m for m in marcaciones if int(m.user_id) in ids_validos]  
    return marcas_filtradas
```

#Funcion que toma los datos de la marcacion en bruto, antes de capturar mira si no son iguales y despues devuelve ese valor para que pase por la sgte funcion

```
def Actualizar_Datos(id_dispositivo, marcacion):  
    conexion = obtener_conexion()  
    try:  
        with conexion.cursor() as cursor:  
            cursor.execute(  
                "SELECT marcacion FROM marcados WHERE FK_dispositivos = %s ORDER BY marcacion DESC LIMIT  
1", (id_dispositivo,))  
            ultima_marcacion = cursor.fetchone()  
            conexion.commit()  
        finally:  
            conexion.close()  
        for marca in marcacion:  
            if ultima_marcacion is None or marca.timestamp >= ultima_marcacion['marcacion']:  
                tipo, detalle, horas_trabajadas = EntradaoSalida(marca, id_dispositivo)  
                Marcados(marca, id_dispositivo, tipo, detalle, horas_trabajadas)
```

#Funcion que toma los datos de los empleados en bruto, antes de enviarlo a la BD mira si no es un empleado que ya existe en la tabla en la BD

```
def Actualizar_Datos_Empleados(empleados):  
    for empleado in empleados:  
        ultimo_empleado = obtener_ultimo_empleado(empleado.user_id)  
        if ultimo_empleado is None:  
            Empleados(empleado, 1)
```

6.7.3. JavaScript

- Scripts.js

// Funcion para confirmar el guardado de un dato

```
function confirmarGuardado() {  
    return confirm("¿Estas seguro de que deseas guardar?");  
}
```

// Funcion para confirmar si quiere el usuario cerrar sesion

```
function confirmarLogout() {  
    if (confirm("¿Estas seguro de que deseas cerrar sesion?")) {  
        window.location.href = logoutUrl;  
    }  
}
```

// Funcion para animacion de boton de carga

```
function mostrarCarga(form) {  
    const boton = form.querySelector("button");  
    const spinner = boton.querySelector(".spinner-border");  
    const texto = boton.querySelector(".texto-boton");  
    if (spinner && texto) {  
        spinner.classList.remove("d-none");  
        texto.textContent = "Cargando...";  
    }  
}
```

6. Estructura general del proyecto

Manual del Usuarios

6.7. Inicio de sesión

Cuando se abre la aplicación, primeramente, debe iniciar sesión para poder acceder a la aplicación, con las credenciales validas de la base de datos.

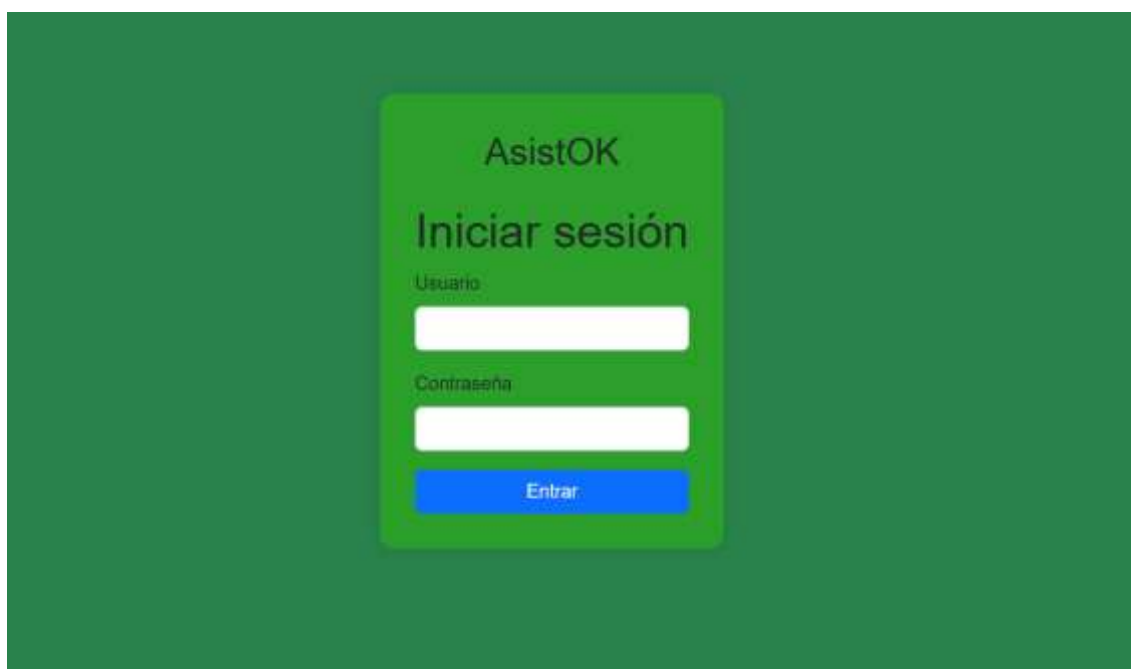


Ilustración 26 Pantalla Iniciar Sesión del sistema AsistOK

6.8. Pantalla Principal

Una vez que se inicia sesión, se podrá acceder a la pantalla principal (Home). Tenemos un panel con diferentes opciones para acceder y conectarse a un dispositivo y capturar los datos del mismo.

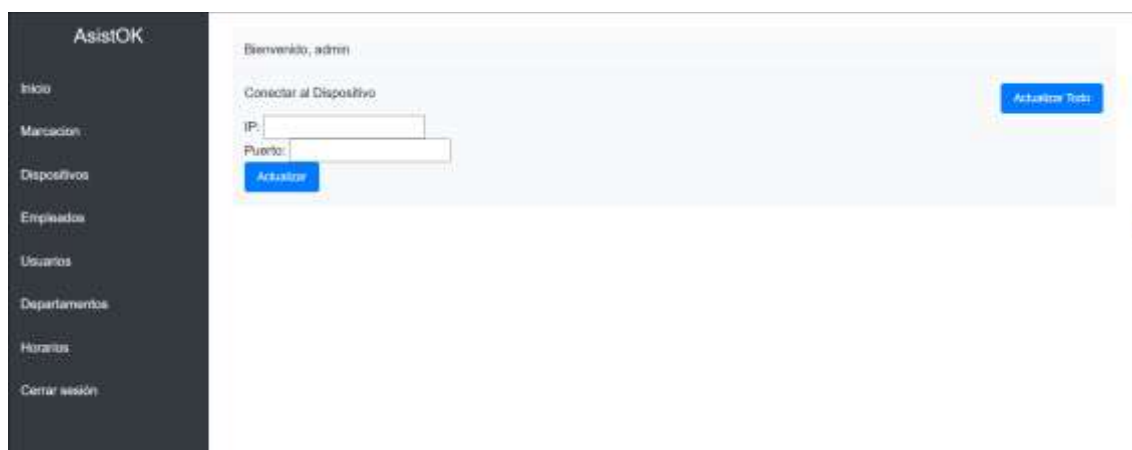
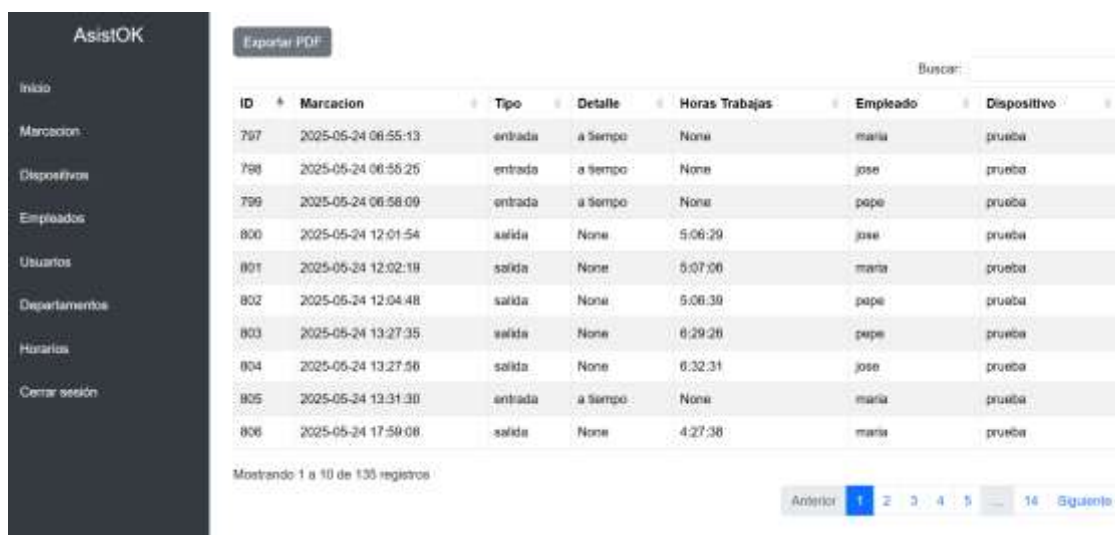


Ilustración 27 Pantalla Principal del sistema AsistOK

6.9. Pantalla de Marcación

Al acceder a la opción marcación, se accederá a una pantalla donde se podrá ver una tabla de todas las marcaciones que se capturaron de los dispositivos de marcación que están almacenados en la Base de datos. Se podrá filtrar a través del buscador y también se podrá exportar la tabla para informes.



ID	Marcación	Tipo	Detalle	Horas Trabajadas	Empleado	Dispositivo
797	2025-05-24 06:55:13	entrada	a tiempo	None	maria	prueba
798	2025-05-24 06:56:25	entrada	a tiempo	None	jose	prueba
799	2025-05-24 06:58:09	entrada	a tiempo	None	pape	prueba
800	2025-05-24 12:01:54	salida	None	5:06:29	jose	prueba
801	2025-05-24 12:02:18	salida	None	5:07:00	maria	prueba
802	2025-05-24 12:04:48	salida	None	5:06:30	pape	prueba
803	2025-05-24 13:27:35	salida	None	6:29:28	pape	prueba
804	2025-05-24 13:27:56	salida	None	6:32:31	jose	prueba
805	2025-05-24 13:31:30	entrada	a tiempo	None	maria	prueba
806	2025-05-24 17:59:08	salida	None	4:27:38	maria	prueba

Ilustración 28 Pantalla de Marcación del sistema AsistOK

6.10. Pantalla de Dispositivos

Al acceder a la opción dispositivos, se accederá a una pantalla donde se podrá ver una tabla de todos los dispositivos que están almacenados en la base de datos. También se podrá acceder a la opción de Agregar dispositivo o Editar dispositivo.

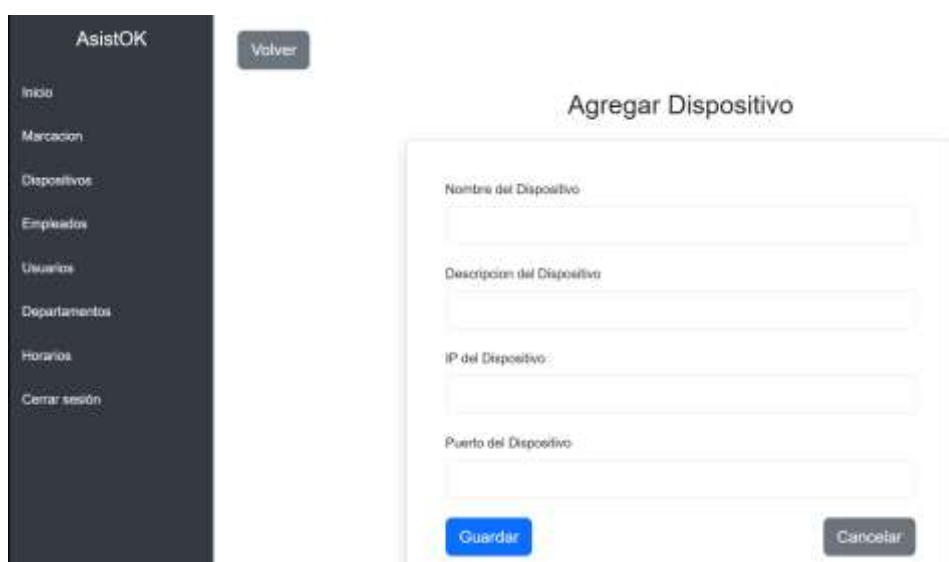


ID	Descripción	Nombre	Activo	IP	Puerto	Acción
1	dispositivo sede central	sede central	1	192.168.150.34	4370	Editar
2	dispositivo prueba	prueba	1	192.168.100.30	4370	Editar

Ilustración 29 Pantalla de Dispositivos del sistema AsistOK

6.11. Pantalla de Agregar Dispositivo

Al presionar el botón agregar dispositivo, te llevará a una pantalla de agregar dispositivo que consiste en un formulario, donde se deberá rellenar los campos para así poder agregar un nuevo dispositivo.



AsistOK

[Volver](#)

Agregar Dispositivo

Nombre del Dispositivo

Descripción del Dispositivo

IP del Dispositivo

Puerto del Dispositivo

[Guardar](#) [Cancelar](#)

Ilustración 30 Pantalla de Agregar Dispositivos del sistema AsistOK

6.12. Pantalla Editar Dispositivo

Al presionar algún botón de editar de cualquier fila de la tabla, se accederá a una pantalla de editar dispositivo, que consiste en un formulario para modificar los campos de dicho dispositivo para después guardar los cambios.



AsistOK

Volver

Editar Dispositivo

Descripción
dispositivo sede central

Nombre
sede central

☒ Activo

Puerto
4370

Dirección IP
192.168.150.34

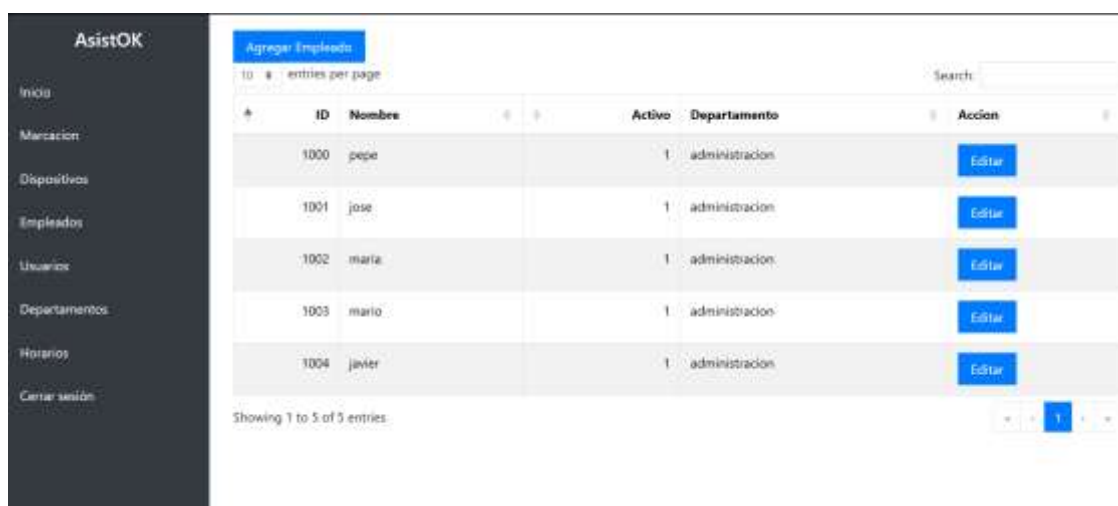
Guardar cambios

Cancelar

Ilustración 31 Pantalla Editar Dispositivo del sistema AsistOK

6.13. Pantalla Empleados

Al acceder a la opción Empleados, se accederá a la pantalla de empleados, que consiste en una tabla el cual mostrara todos los empleados, cuyos empleados se capturo de los dispositivos y que ahora están almacenados en la Base de Datos. Se podrá acceder a las opciones de agregar y editar empleado



AsistOK

Agregar Empleado

10 entries per page

Search

ID	Nombre	Activo	Departamento	Accion
1000	pepe	1	administracion	Editar
1001	jose	1	administracion	Editar
1002	maria	1	administracion	Editar
1003	mario	1	administracion	Editar
1004	javier	1	administracion	Editar

Showing 1 to 5 of 5 entries

1

Ilustración 32 Pantalla Empleados del sistema AsistOK

6.14. Pantalla de Agregar Empleados

Al presionar el botón agregar empleado, se accederá a una pantalla de agregar empleado, que consiste en un formulario, donde se deberá rellenar los campos para así poder agregar un nuevo empleado y guardar.



Ilustración 33 Pantalla de Agregar Empleados del sistema AsistOK

6.15. Pantalla Editar Empleado

Al presionar un botón editar de cualquier fila de la tabla empleados, se podrá editar dicha fila, donde lo llevará a una pantalla de editar empleado el cual que consiste en un formulario, donde se podrá editar un empleado en específico el cual se selecciono en la tabla anterior, para así poder modificarlo y guardarlo.



Ilustración 34 Pantalla Editar Empleado del sistema AsistOK

6.16. Pantalla Usuarios

Al presionar la opción Usuarios, podrá acceder a la pantalla de usuarios, que consiste en una tabla de todos los usuarios que están almacenados en la base de datos. Se podrá filtrar a través del buscador y también tendrá los botones para editar y agregar usuario. Los botones de agregar y editar solo tendrán acceso el administrador, si es un usuario lo bloquea la opción.

Con Admin:



The screenshot shows the 'AsistOK' interface. On the left is a dark sidebar menu with options: Inicio, Marcación, Dispositivos, Empleados, Usuarios, Departamentos, Horarios, and Cerrar sesión. The 'Usuarios' option is selected. The main content area has a green 'Agregar Usuario' button at the top left. Below it is a search bar and a table with columns: ID, Nombre, Activo, and Editar. The table contains two rows: ID 1, Nombre admin, Activo Si; and ID 2, Nombre admin2, Activo Si. Each row has a blue 'Editar' button. At the bottom of the table, it says 'Showing 1 to 2 of 2 entries'.

Ilustración 35 Pantalla Usuarios con Admin del sistema AsistOK

Con Usuario:



The screenshot shows the 'AsistOK' interface. On the left is a dark sidebar menu with options: Inicio, Marcación, Dispositivos, Empleados, Usuarios, Departamentos, Horarios, and Cerrar sesión. The 'Usuarios' option is selected. The main content area has a green 'Agregar Usuario' button at the top left. Below it is a search bar and a table with columns: ID, Nombre, Activo, and Editar. The table contains two rows: ID 1, Nombre admin, Activo Si; and ID 2, Nombre admin2, Activo Si. Each row has a grey 'Editar' button. At the bottom of the table, it says 'Showing 1 to 2 of 2 entries'.

Ilustración 36 Pantalla Usuarios con Usuario del sistema AsistOK

6.17. Pantalla Agregar Usuario

Al presionar el botón agregar usuario, si el usuario en cuestión es administrador, podrá acceder a la pantalla de agregar usuario, el cual consiste en un formulario donde se deberá rellenar los respectivos campos para así poder agregar un nuevo usuario.

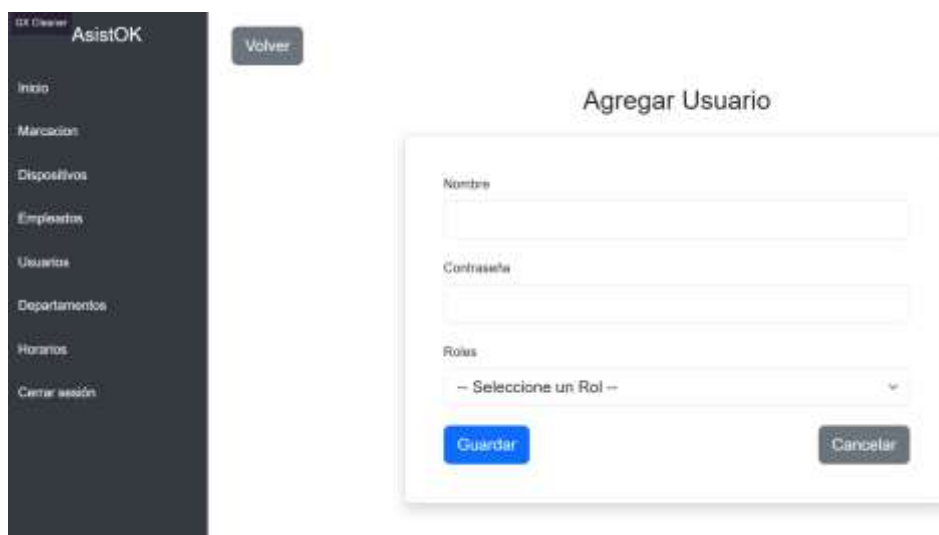


Ilustración 37 Pantalla Agregar Usuario del sistema AsistOK

6.18. Pantalla Editar Usuario

Al presionar el botón editar de alguna fila de la tabla usuarios, si el usuario en cuestión es administrador, podrá acceder a la pantalla editar usuario, el cual consiste en un formulario donde se podrá editar el usuario que se seleccionó en la tabla de usuarios, para así poder modificarlo y guardarlo.

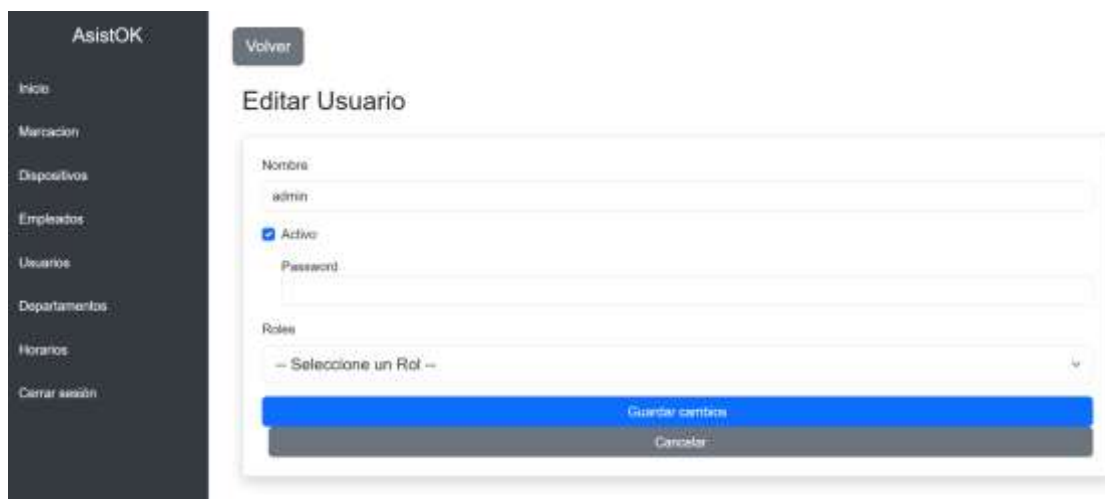


Ilustración 38 Pantalla Editar Usuario del sistema AsistOK

6.19. Pantalla Departamentos

Al presionar la opción departamentos, se podrá acceder a la pantalla de departamentos, el cual consiste en una tabla de todos los departamentos los cuales están almacenados en la base de datos. También se podrá filtrar a través del buscador y tener las opciones de editar y agregar departamentos.

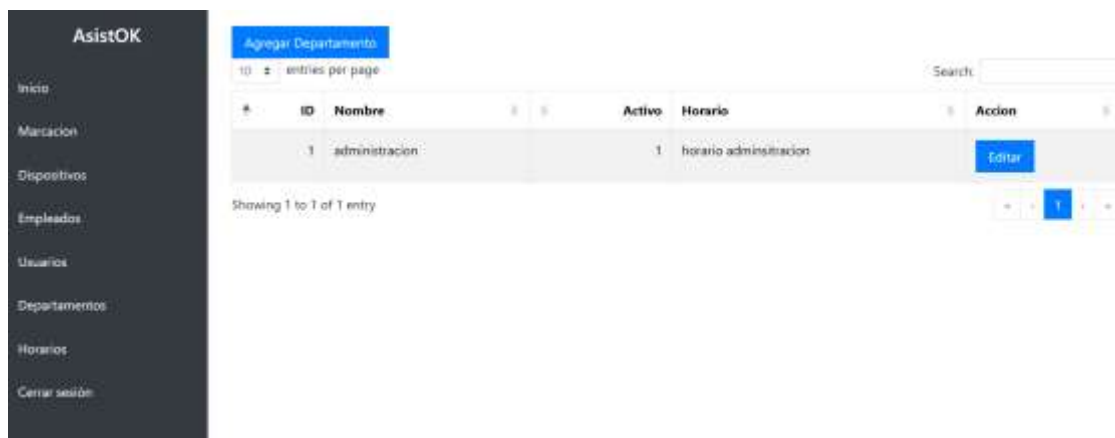


Ilustración 39 Pantalla Departamentos del sistema AsistOK

6.20. Pantalla Agregar Departamento

Al presionar el botón agregar dispositivo, podrá acceder a la pantalla agregar dispositivo, el cual consiste en un formulario donde se deberá rellenar los respectivos campos para así poder agregar un nuevo departamento y guardarlo.

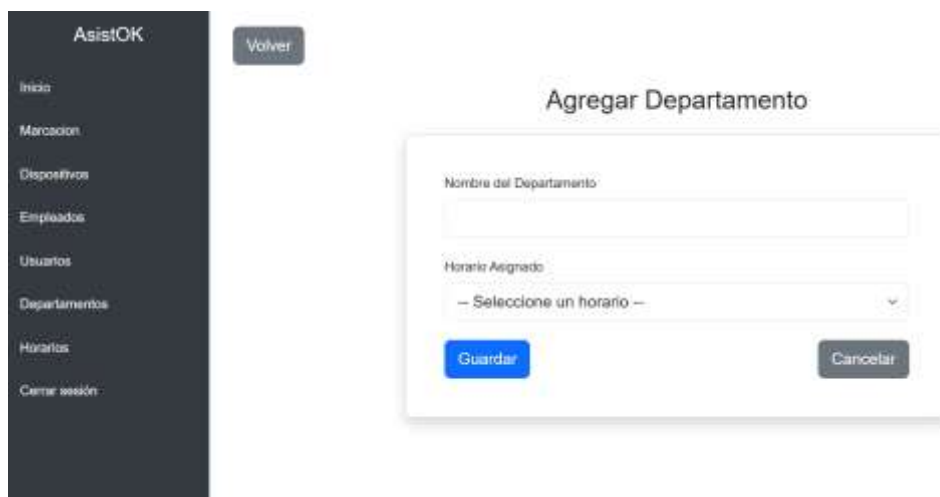


Ilustración 40 Pantalla Agregar Departamento del sistema AsistOK

6.21. Pantalla Editar Departamento

Al presionar el botón editar de alguna fila de la tabla departamentos, se podrá acceder a la pantalla de editar departamento, el cual consiste en un formulario donde se deberá modificar los valores de los campos de dicho departamento sé que selecciono en la tabla para así modificarlo y guardarlo.

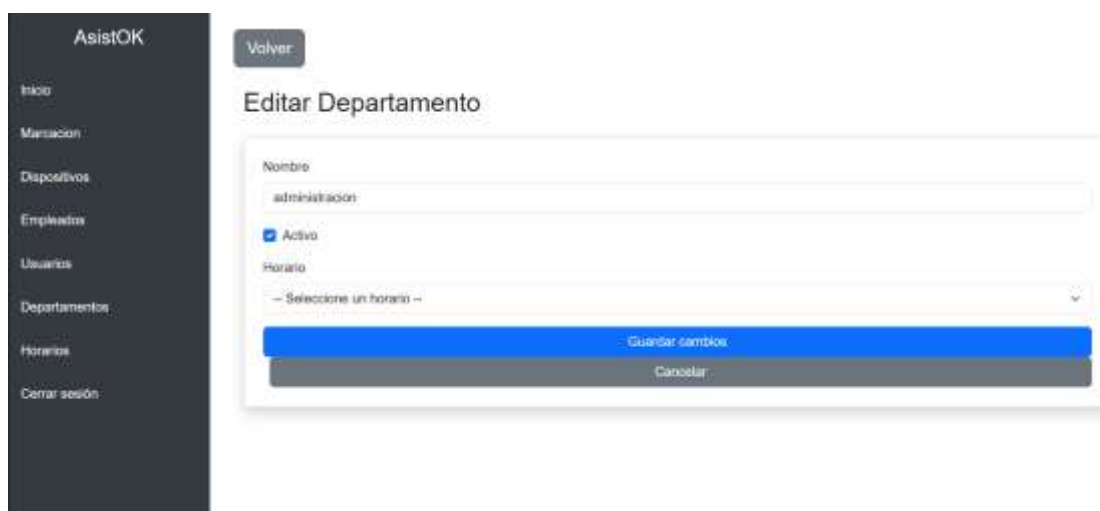


Ilustración 41 Pantalla Editar Departamento del sistema AsistOK

6.22. Pantalla Horarios

Al presionar la opción horarios, se accederá a la pantalla horarios, el cual consiste en una tabla de todos los horarios lo cuales están almacenados en la base de datos. También se podrá filtrar a través del buscador y tener acceso a las opciones de agregar y editar horario.



ID	Descripción	Entrada de la Mañana	Salida de la Mañana	Tolerancia de la Mañana	Entrada de la Tarde	Salida de la Tarde	Tolerancia de la Tarde	Activo	Acción
1	horario administración	6:55:00	12:00:00	7:10:00	13:30:00	18:00:00	13:40:00	1	Editar

Ilustración 42 Pantalla Horarios del sistema AsistOK

6.23. Pantalla Agregar Horario

Al presionar el botón agregar horario, accederá a la pantalla agregar horario, el cual consiste en un formulario donde se deberá rellenar los respectivos campos para si poder agregar un nuevo departamento y guardar.

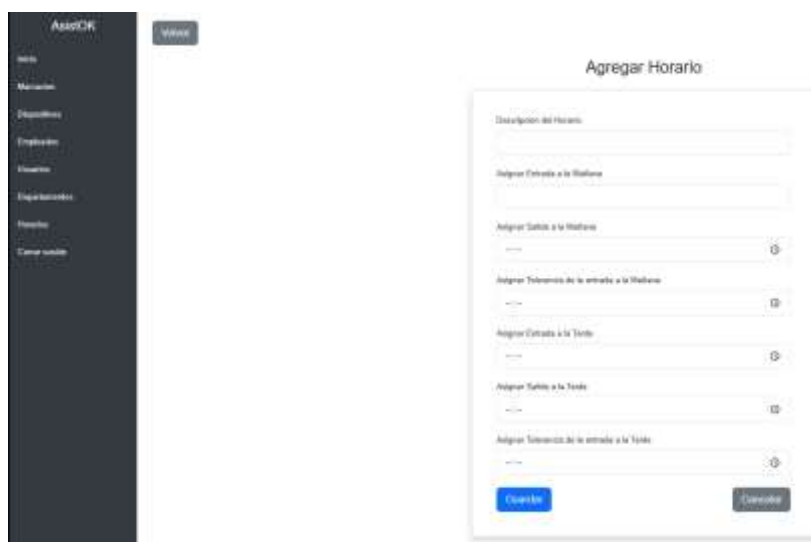


Ilustración 43 Pantalla Agregar Horario del sistema AsistOK

6.24. Pantalla Editar Horario

Al presionar el botón editar, de alguna fila de la tabla horarios, se accederá a la pantalla editar horario, el cual consiste en un formulario donde se deberá modificar los campos de dicho horario que se seleccionó en la tabla horarios para así modificarlo y guardarlo.



Ilustración 44 Pantalla Editar Horario del sistema AsistOK

6.25. Cerrar Sesión

Al presionar la opción cerrar sesión, podrá salir del sistema y lo llevará hacia el login del sistema, antes de salir, te confirma si quieres salir o no del sistema.

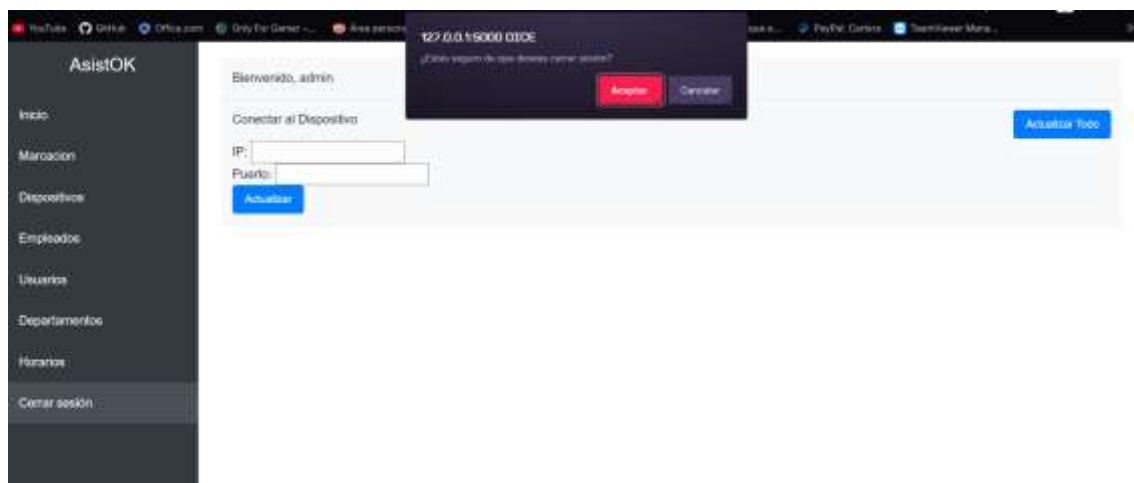


Ilustración 45 Pantalla Cerrar Sesión del sistema AsistOK



7. Conclusión

La pasantía en Eno Bronstrup S.A. ha sido una experiencia enriquecedora y formativa, permitiendo la aplicación práctica de conocimientos en el desarrollo de software y la gestión de proyectos.

La identificación de la problemática en el control de asistencia y el subsiguiente diseño y prototipado de un sistema de marcación automatizado demuestran el valor de la tecnología para optimizar los procesos internos de una organización.

Se logró avanzar significativamente en las fases de análisis, diseño y desarrollo de módulos clave, sentando una sólida base para la implementación futura de una solución integral. Las limitaciones identificadas, lejos de ser impedimentos, establecen un camino claro para futuras iteraciones y mejoras del sistema.

Este proyecto no solo beneficia a la empresa con una propuesta de solución a una necesidad real, sino que también contribuye significativamente al desarrollo profesional del pasante, consolidando habilidades técnicas y de planificación esenciales para la carrera en desarrollo de sistemas. La colaboración con el equipo de TI de la empresa fue fundamental y se espera que el sistema propuesto contribuya a una gestión de asistencia más eficiente y precisa.



8. Anexos

Certificado de aprobación

UNAE UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN SEDE COLONIAS UNIDAS

AÑO INTERNACIONAL DE LA EDUCACIÓN para el desarrollo de la CIENCIA Y LA TECNOLOGÍA UNAE - IREDE 2015

UNAE UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN

(FORMULARIO FUS)

NOTA DE ACEPTACIÓN DEL PASANTE

Hohenau 20 de marzo de 2.025

El representante de ENO BRONSTRUP S.A., la Sra. Fátima Storrer que se desempeña en el cargo de Gerente de Administración manifiesta que es de su interés que el alumno Luis Mario Morinigo Bracho de la carrera Licenciatura en Análisis de Sistemas Informáticos de la **UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN**, desarrolle el trabajo asistente técnico informático y las actividades que el mismo presuponga en esta entidad, enmarcado en la **Práctica Profesional**.

El Sr. Fernando Fernández se designa como **Supervisor de Campo**, nombrado por la Institución y/o Empresa, para realizar el seguimiento de las actividades que realice el alumno en este ámbito.

Eno Brönstrup S.A.
E.B.S.A. RUC: 80013620-9

Firma Alumno

Firma Coordinación Académica

Firma Representante de la Entidad

Aclaración

Aclaración


Aclaración


UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN | Sede Colonias Unidas: Guillermo Closs c/ Aviadores del Chaco, Hohenau, Paraguay | T: 0775 232 608 | E: 0983 796 259 | W: www.unae.edu.py


Ilustración 46 Certificado de aprobación



Formulario información de la empresa

 UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN | SEDE COLONIAS UNIDAS

 AÑO INTERNACIONAL DE LA **EDUCACIÓN** para el desarrollo de la CIENCIA Y LA TECNOLOGÍA UNA E - IREDE POSIS

 UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN

DATOS DE IDENTIFICACIÓN DE LAS PASANTÍAS LABORALES PARA LA CARRERA DE LICENCIATURA EN ANÁLISIS DE SISTEMAS INFORMÁTICOS

Datos de identificación

Nombre de la Dependencia Pública o Privada	ENO BRONSTRUP S.A.
Dirección	Avda. Marcial Samaniego,
Nombre del Titular y cargo	Fátima Storrer - Gerente de Administración.
Correo Electrónico y Teléfono	administracion.ebsa@selecta.com.py
Población o Municipio	Bella Vista

Datos del Estudiante que realiza la Práctica Profesional

Nombre	Luis Mario Morinigo Bracho
Carrera	Licenciatura en Análisis de Sistemas Informáticos
Semestre	5º(Quinto)
Documento de Identidad	5.381.189
Correo Electrónico	luisluismm727@gmail.com
Teléfonos	0972969758

Ilustración 47 Formulario información de la empresa



UNIVERSIDAD
AUTÓNOMA DE
ENCARNACIÓN | SEDE
COLONIAS
UNIDAS



AÑO INTERNACIONAL DE LA
EDUCACIÓN
para el desarrollo de la
CIENCIA Y LA TECNOLOGÍA
UNAE - IREDE POSIS



UNIVERSIDAD
AUTÓNOMA DE
ENCARNACIÓN

Nota de solicitud de pasantía

UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN | SEDE COLONIAS UNIDAS

EDUCACIÓN para el desarrollo de la CIENCIA Y LA TECNOLOGÍA UNAE - IREDE POSIS

UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN

Hohenau, 10 de marzo de 2023

Sra Fátima Storrer

Nos dirigimos a Usted extendiéndole nuestro más cordial saludo. Así mismo deseamos poner a su conocimiento que Luis Mario Morinigo Bracho con C.I. N°5.381.189 es alumno regular de 3° curso de la carrera de Licenciatura en Análisis de Sistemas Informáticos

En este contexto le solicitamos autorización para realizar una pasantía de 200 horas de duración, que se desarrolla dentro de la Malla Curricular de la carrera y que en su Plan de Estudios prevé la realización de un período de pasantía, donde el estudiante pondrá en práctica los conocimientos adquiridos durante los años de preparación de la carrera, en los horarios disponibles de la Empresa.

Los estudiantes poseen las siguientes competencias sobre:

- Aplicar las tecnologías de la información y comunicación de manera efectiva.
- Conocer y saber aplicar técnicas y herramientas actualizadas en sus áreas de competencia.
- Manifestar actitud emprendedora, creativa e innovadora en las actividades inherentes a la profesión.
- Demostrar capacidad de adaptarse a situaciones nuevas y cambiantes.
- Demostrar capacidad de auto aprendizaje y actualización permanente en la formación profesional

En espera de una respuesta favorable en pro de la capacitación y formación de los jóvenes, nos despedimos saludándole atentamente.

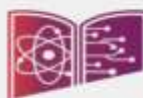
Lic. Pedro Danieli
Docente
Cel: 0982 351555
Correo: pedro.danieli@unae.edu.py

Msc. Carina Ramos
Coordinadora académica
Cel: 0983 597831
Correo: direccioncolonias@unae.edu.py

UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN | Sede Colonias Unidas: Guillermo Closs c/ Aviadores del Chaco, Hohenau, Paraguay | T: 0775 232 608 | E: 0983 796 259 | W: www.unae.edu.py

Ilustración 48 Nota de solicitud de pasantía





Carta compromiso de pasantía

UNAE UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN SEDE COLONIAS UNIDAS

AÑO INTERNACIONAL DE LA EDUCACIÓN para el desarrollo de la CIENCIA Y LA TECNOLOGÍA UNAE - IREDE POSIS

UNAE UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN

(FORMULARIO PAS)

CARTA COMPROMISO DE PASANTÍA


Yo, Luis Mario Morinigo Bracho C.I. No 5.381.189, cursante de la carrera de Licenciatura en Análisis de Sistemas Informáticos de la **Facultad de Ciencia, Arte y Tecnología de la Universidad Autónoma de Encarnación**, por medio del presente documento declaro: "Que acepto y me comprometo a observar íntegramente la normativa que sobre el régimen de pasantía, establece la **Facultad de Ciencia, Arte y Tecnología de la Universidad Autónoma de Encarnación**, en el entendido de que la inobservancia de dicha normativa me acarreará, como consecuencia la no aprobación de la misma, debiendo en este caso inscribir nuevamente la pasantía en el próximo lapso académico. Asimismo convengo expresamente que, para dar por cumplidas las condiciones necesarias para optar al correspondiente acto de grado académico, me comprometo a:

1. Cumplir a cabalidad con el horario de pasantía que se me asigne. En caso de que necesite ausentarme de la institución, deberé justificar mis inasistencias inmediatamente ante la institución y la Coordinación de Pasantía.
2. Desarrollar la pasantía de manera ininterrumpida durante el tiempo académico correspondiente, según cronograma de actividades de pasantía.
3. Notificar de manera inmediata a la Coordinación de Pasantía de la UNAE cualquier cambio de fecha de inicio, o condición de la pasantía por parte de la institución.
4. Entregar en la institución la carta de postulación emitida por la Coordinación de Pasantía de la UNAE en la que deben constar todas las formalidades del proceso de pasantía y las fechas exactas de inicio y de culminación de las mismas. Así como entregar en la Coordinación de Pasantía la carta de aceptación en la institución según cronograma de actividades de pasantía.
5. Notificar inmediatamente a la Coordinación de Pasantía de la UNAE sobre cualquier irregularidad que se presente durante el proceso de pasantía.
6. Presentar todas y cada una de las evaluaciones contempladas en el plan de evaluación de la pasantía de conformidad con los lapsos establecidos en el cronograma de actividades de Pasantía.

El ocultamiento, forjamiento, alteración y/o falsedad de información relativa a cualquier fase del proceso de pasantía o cualquier delito que evidencie la falta de fidelidad de la información sobre la institución, las labores a realizar o el contenido del informe, por parte del pasante, será tomado como falta y será sancionado conforme a la normativa de la UNAE.

Lo no previsto en este convenio y/o el incumplimiento de cualquiera de los numerales anteriores, serán resueltos conforme a las disposiciones del Reglamento de Pasantía o por cualquier otra normativa dictada por el Decanato de la Facultad de Ciencia, Arte y Tecnología, o por el Consejo Superior Universitario de la Universidad Autónoma de Encarnación (UNAE).

Hohenau, a los 20 del mes de marzo de 2.025


Firma y Aclaración

UNAE UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN | Sede Colonias Unidas: Guillermo Closs c/ Aviadores del Chaco, Hohenau, Paraguay | T: 0775 232 608 | E: 0983 796 259 | W: www.unae.edu.py

Ilustración 49 Carta compromiso de pasantía





Plan de trabajo pasantía

UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN SEDE COLONIAS UNIDAS				AÑO INTERNACIONAL DE LA EDUCACIÓN para el desarrollo de la CIENCIA Y LA TECNOLOGÍA UNAE - IREDE POSIS		UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN	
PLAN DE TRABAJO DE PASANTÍA							
DATOS DEL ESTUDIANTE	Apellidos y nombres: Morinigo Bracho, Luis Mario						
	Cédula de identidad: 5.381.189			Teléfonos: 0972969758			
	Carrera: Licenciatura en Análisis de Sistemas Informáticos			Facultad: Ciencia, Arte y Tecnología			
	Inicio de la pasantía: 24/03/25			Final de la pasantía: 23-05-2025			
	Tiempo completo:			Medio tiempo: <input checked="" type="checkbox"/>			
DATOS DE LA EMPRESA	Nombre: ENO BRONSTRUP S.A.			Teléfonos: 0982 982897			
	Dirección: Avda. Marcial Samaniego - Bella Vista						
	Actividad económica: Venta y Producción de Bienes						
	Departamento donde realizará la pasantía: Informática						
DATOS DE LOS TUTORES	Tutor Académico: Ljc. Pedro Danieli			Teléfonos: 0982 351555			
	Tutor Externo: Ulises Martinez			Departamento: Informática			
	Cargo: Analista			Teléfonos: 0985 989746			
TRABAJO DE PASANTÍA							
Título de la pasantía: Pasantía laboral en la empresa ENO BRONSTRUP S.A.							
Descripción del área de trabajo:							
Brindar soporte a los usuarios con respecto a los sistemas de uso, administrar las redes en el sistema de internet. Realizar mantenimiento de sistemas informáticos, administrar gestionar y mantener el buen funcionamiento de los activos informáticos de la empresa.							
Metas:							
Objetivo general:				Objetivos específicos:			
Aprender más sobre manejo de sistemas				-Aprender el uso de las herramientas del área			
Adquirir más conocimiento en el campo laboral				-Observar y aprender los procesos informáticos del área			
(Anexo cronograma de actividades)							
Recursos: Computador, materiales varios							
Observaciones:							
Tutor Académico				Tutor Externo		Pasante	

UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN | Sede Colonias Unidas: Guillermo Closs c/ Aviadores del Chaco, Hohenau, Paraguay | T: 0775 232 608 | F: (0983) 796 259 | E: www.unae.edu.py

Ilustración 50 Plan de trabajo pasantía





Formulario de evaluación (tutor externo)

UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN | SEDE COLONIAS UNIDAS **EDUCACIÓN** para el desarrollo de la CIENCIA Y LA TECNOLOGÍA UNAE - IREDE 2025 UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN

EVALUACIÓN DEL PASANTE (TUTOR EXTERNO)

DATOS PERSONALES Y ACADÉMICOS DEL PASANTE	Apellidos y nombres: Morinigo Bracho, Luis Mario	
	Cédula de identidad: 5.381.189	Teléfonos: 0972969758
	Facultad: Ciencia, Arte y Tecnología	Carrera: Licenciatura en Análisis de Sistemas Informáticos
	Semestre: 5º (Quinto)	Tutor académico: Lic. Pedro Danieli

DATOS DE LA INSTITUCIÓN	Nombre de la empresa: ENO BRONSTRUP S.A.	Teléfonos: 0982 982897
	Dirección: Avda. Marcial Samaniego, Bella Vista	
	Tutor externo: Ulises Martínez	Departamento: Informática

DATOS DE LA PASANTÍA

Departamento	Área o división	Fecha inicio	Fecha de culminación
Administración	Informática	24-03-2025	30-05-2025

INSTRUCCIONES: Califique al pasante en cada casilla, marcando una "X", de acuerdo con la siguiente escala:

DEFICIENTE (1)	REGULAR (2)	BUENO (3)	MUY BUENO (4)	EXCELENTE (5)
-------------------	----------------	--------------	------------------	------------------

ASPECTOS A CONSIDERAR	(1)	(2)	(3)	(4)	(5)
Adquirió conocimiento de la estructura y organización de la empresa, así como de los procedimientos de trabajo y de seguridad.				X	
Calidad del trabajo, habilidad, presentación, precisión, pulcritud					X
Cumplió las normas y procedimientos de la empresa tales como: horario de trabajo, normas de seguridad, tramitación de servicios, etc.					X
Iniciativa y creatividad, capacidad en la toma de decisiones, aporte de ideas.			X		
Puntualidad y asistencia, cumplimiento del horario de trabajo a tiempo.					X
Planeamiento y organización, capacidad para organizar y planificar el trabajo.				X	
Relaciones personales, trato y cortesía con los compañeros de trabajo, cooperación, espontaneidad para colaborar con los compañeros de trabajo y su disposición para mantener buenas relaciones.					X
Presentó varias alternativas de posibles soluciones a cada problema propuesto.				X	
Demostró receptividad a planteamientos diferentes a los presentados por él.					X
Presentó explicaciones escritas de su trabajo en forma clara y precisa.					X
Informe preliminar se ajusta a las necesidades de la empresa.					X
TOTAL PUNTOS (suma de la puntuación asignada a cada factor y dividida entre el número total factores utilizados)	SUMATORIA				DIVISIÓN
	50				4,5

Observaciones del tutor externo:

Eno Brönstrup S.A.
Tutor Externo: 0972969758
(Nombre y teléfono de la institución)
Fecha: 2025/05/25

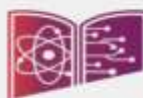
UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN | Sede Colonias Unidas: Guillermo Closs c/ Aviadores del Chaco, Hohenau, Paraguay | T: 0775 232 608 | C: (0983) 796 259 | www.unae.edu.py

Ilustración 51 Formulario de evaluación (tutor externo)





**UNIVERSIDAD
AUTÓNOMA DE
ENCARNACIÓN** SEDE
**COLONIAS
UNIDAS**



AÑO INTERNACIONAL DE LA
EDUCACIÓN
para el desarrollo de la
CIENCIA Y LA TECNOLOGÍA
UNAE - IREDE POSIS



**UNIVERSIDAD
AUTÓNOMA DE
ENCARNACIÓN**

Formulario de evaluación (tutor académico)

(FORMULARIO PL7)

EVALUACIÓN DEL PASANTE (TUTOR ACADÉMICO)

DATOS PERSONALES Y ACADÉMICOS DEL PASANTE	Apellidos y nombres: Morinigo Bracho, Luis Mario		
	Cédula de identidad: 5.381.189	Teléfonos: 0972969758	
	Facultad: Ciencia, Arte y Tecnología	Carrera: Licenciatura en Análisis de Sistemas Informáticos	
	Semestre: 5º(Quinto)	Tutor académico: Lic. Pedro Danieli	
DATOS DE LA INSTITUCIÓN	Nombre de la institución: UNAE Sede Colonias Unidas		Teléfonos: 0983 796259
	Dirección: Guillermo Closs c/ Aviadores del Chaco, Hohenau, Itapúa		
	Tutor institucional: Lic. Pedro Danieli	Departamento y cargo: Docente	
	Área de la pasantía: Informática	Fecha de inicio: 24-03-2025	Fecha de culminación: 30-05-2025

SÓLO PARA USO DEL TUTOR ACADÉMICO

Visita N° 1	Visita N° 2	Visita N° 3
Fecha: 14/04/25	Fecha: 05/05/25	Fecha: 26/05/25
<i>Pdani</i>	<i>Pdani</i>	<i>Pdani</i>

ASPECTOS GENERALES DE LA EVALUACIÓN

INSTRUCCIONES: Califique al pasante en cada casilla, marcando una "X", de acuerdo con la siguiente escala:

DEFICIENTE (1)	REGULAR (2)	BUENO (3)	MUY BUENO (4)	EXCELENTE (5)	
FACTORES					
	(1)	(2)	(3)	(4)	(5)
PUNTUALIDAD Y ASISTENCIA: Cumple con el horario establecido para las reuniones con el tutor académico, justifica razonablemente sus retrasos y responde a las actividades formales e informales que se le indican.					
RESPONSABILIDAD: Tiene una actitud madura y responsable frente a sus funciones, entrega en las fechas previstas sus trabajos.					
INTERÉS: Demuestra interés por las actividades rutinarias e identificación y compromiso con el trabajo.					
CONOCIMIENTOS: Demuestra poseer conocimientos básicos para aplicarlos efectivamente en la práctica. Tiene suficiente base educativa para comprender y resolver problemas relacionados con su área de trabajo.					
CAPACIDAD ANALÍTICA: Demuestra capacidad para razonar planteamientos, de relativa complejidad. Sus criterios y razonamientos son lógicos. Tiene capacidad para aportar soluciones efectivas.					
ASIMILACIÓN: Tiene capacidad para superar situaciones imprevistas en el área de trabajo.					
INICIATIVA: Aporta ideas, tiene capacidad innovadora. Es capaz de recomendar sugerencias y mejoras.					
NORMAS DE LA UNIVERSIDAD: Cumple con el programa de pasantía. Se esfuerza en su cumplimiento de las tareas asignadas.					
IMPRESIÓN SOBRE LA EVOLUCIÓN DEL PASANTE (VISITA N° 1)					X
IMPRESIÓN SOBRE LA EVOLUCIÓN DEL PASANTE (VISITA N° 2)					X
IMPRESIÓN SOBRE LA EVOLUCIÓN DEL PASANTE (VISITA N° 3)					X
TOTAL PUNTOS (suma de la puntuación asignada a cada factor y divida entre el número total factores utilizados)					SUMATORIA DIVISIÓN

Observaciones y Recomendaciones del Tutor Académico:
 Tutor Académico: _____
 (Nombre, firma y sello de la institución) Fecha: _____

UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN | Sede Colonias Unidas: Guillermo Closs c/ Aviadores del Chaco, Hohenau, Paraguay | T: 0775 232 608 | F: (0983) 796 259 | E: www.unae.edu.py

Ilustración 52 Formulario de evaluación (tutor académico)



Dispositivo ZKteco



Ilustración 53 Dispositivo ZKteco

[Github](#)

https://github.com/LuisMM727/Proyecto_Pasantia.git



Bibliografía

- apachefriends. (s.f.). *apachefriends.org*. Obtenido de apachefriends.org:
<https://www.apachefriends.org/es/index.html>
- appvizer. (s.f.). *appvizer.es*. Obtenido de appvizer.es:
<https://www.appvizer.es/it/wireframe/balsamiq-mockups>
- arsys. (s.f.). *arsys.es*. Obtenido de arsys.es: <https://www.arsys.es/blog/que-es-flask-python-y-cuales-son-sus-ventajas#:~:text=Flask%20es%20un%20framework%20para,renderizado%20de%20HTML%20como%20Django>.
- arsys. (s.f.). *arsys.es*. Obtenido de arsys.es: <https://www.arsys.es/blog/que-es-visual-studio-code-y-cuales-son-sus-ventajas#tree-1>
- bootstrap. (s.f.). *bootstrap.com*. Obtenido de bootstrap.com: <https://getbootstrap.com>
- git. (s.f.). *git-scm.com*. Obtenido de git-scm.com: <https://git-scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Fundamentos-de-Git>
- Godaddy. (s.f.). *Godaddy.com*. Obtenido de Godaddy.com:
<https://www.godaddy.com/resources/latam/desarrollo/python-que-es>
- hostinger. (s.f.). *hostinger.com*. Obtenido de hostinger.com:
<https://www.hostinger.com/es/tutoriales/que-es-css>
- IBM. (s.f.). *ibm.com*. Obtenido de ibm.com: <https://www.ibm.com/mx-es/think/topics/structured-query-language>
- mysql. (s.f.). *mysql.com*. Obtenido de mysql.com:
<https://www.mysql.com/products/workbench/>
- Selecta, G. (s.f.). *Selecta*. Obtenido de Selecta: <https://www.selecta.com.py/nosotros/grupo-selecta>
- trello. (s.f.). *trello.com*. Obtenido de trello.com: <https://trello.com/es/tour>
- unir.net. (s.f.). *unir.net*. Obtenido de unir.net: <https://www.unir.net/revista/ingenieria/que-es-javascript/>
- wix. (s.f.). *wix.com*. Obtenido de wix.com: <https://es.wix.com/blog/que-es-html#viewer-fc2o624066>
- zkteco. (s.f.). *zkteco.com*. Obtenido de zkiteco.com: <https://www.zkteco.com/en/>