



Proyecto *Soporte Digit@l*

Avance de Informe 2

EQUIPO DE TRABAJO	
Número	Apellidos y Nombres
1	Medina Quispe Licia Antonia
2	Barco Peña Sofia Antonela
3	Sanchez Rodas Neyder Jose
4	Cuellar Reyes Verenisse Lucy
5	

Índice

1. Resumen.....	3
2. Problema	3
3. Objetivos	3
4. Fundamentos Teórico.....	4
5. Planteamiento de Solución.....	7
5.1. Diagrama de Entidad Relación.....	7
5.2. Diseño de Base de Datos	7
5.3. Script DDL de Base de datos	7
5.4. Diagrama de Clases	7
5.5. Implementación del Proyecto Java al 50%.....	7
5.5.1. Implementación del componente modelo	7
5.5.2. Implementación del componente controlador	18
5.5.3. Implementación del componente vista.....	19
5.5.4. Patrones Implementados	20
6. Conclusiones.....	34
7. Recomendación.....	34
8. Referencias Bibliográficas	34

Página

1. RESUMEN

En el segundo avance del desarrollo de la plataforma web para la empresa 123digit@I, se resolvió el problema relacionado con la gestión ineficiente de las solicitudes de soporte técnico, que anteriormente se manejaban de forma manual y sin una estructura clara. Esta situación afectaba negativamente la productividad y la experiencia del cliente. La solución propuesta fue el desarrollo de una plataforma web basada en el modelo arquitectónico Modelo-Vista-Controlador (MVC), utilizando tecnologías como Java, PostgreSQL, JSF (JavaServer Faces) con páginas XHTML, y el servidor Payara.

El nuevo sistema permite una gestión centralizada de las solicitudes de soporte, facilita la asignación de tareas al personal, permite el registro de avances y genera reportes estadísticos sobre el rendimiento. También se incorporaron funcionalidades completas para la administración de perfiles de usuario, incluyendo la actualización de información personal y la carga de fotos de perfil. La base de datos fue diseñada para respaldar toda la lógica de negocio.

Gracias a la implementación de esta plataforma, se logró automatizar los procesos de soporte técnico, reduciendo considerablemente los tiempos de respuesta y mejorando la experiencia de los usuarios. Se cumplieron todos los objetivos establecidos, lo que resultó en un sistema funcional, eficiente y flexible para futuras necesidades de la empresa.

Esta entrega representa un avance importante en la transformación digital de los procesos de soporte técnico en 123digit@I, impactando positivamente tanto en la productividad interna como en la satisfacción del cliente. Se recomienda realizar un monitoreo continuo del rendimiento del sistema para asegurar su evolución con base en los requerimientos futuros. Asimismo, podría explorarse la incorporación de tecnologías como el análisis predictivo, con el fin de optimizar aún más la asignación y resolución de solicitudes.

2. PROBLEMA

El proyecto se centró en resolver la problemática relacionada con la gestión manual y desordenada de las solicitudes de soporte técnico en la empresa 123digit@I. Este enfoque tradicional ocasionaba múltiples dificultades, como la pérdida de solicitudes, una distribución inadecuada de tareas y una comunicación limitada entre los colaboradores. Tales deficiencias impactaban negativamente en la satisfacción del cliente, provocando demoras en los tiempos de respuesta y aumentando la probabilidad de errores. La implementación de una solución tecnológica se volvió necesaria para automatizar estos procesos, consolidar la información en un solo sistema, facilitar un acceso ágil y seguro a los datos, y, en consecuencia, optimizar tanto la eficiencia operativa como la calidad del servicio brindado.

3. OBJETIVOS

Objetivo**General**

Desarrollar e implementar una plataforma web para la empresa 123digit@l que permita gestionar de manera centralizada y eficiente las solicitudes de soporte técnico, facilitando además la asignación de tareas y el seguimiento de las actividades realizadas.

Objetivos Específicos (Modelo SMART)

1. Crear un sistema para el registro y administración de solicitudes de clientes, con el propósito de reducir los tiempos de respuesta desde los primeros meses de funcionamiento.
2. Implementar un módulo que permita asignar solicitudes a colaboradores de forma específica, asegurando la trazabilidad y transparencia en la ejecución de las actividades.
3. Diseñar e implementar una base de datos relacional con PostgreSQL, que permita almacenar de forma estructurada y segura toda la información relacionada con usuarios, solicitudes y tareas.
4. Optimizar la experiencia del usuario mediante una interfaz amigable, desarrollada con tecnologías XHTML y TailwindCSS, buscando incrementar la satisfacción del cliente a través de una navegación más clara y eficiente.

4. FUNDAMENTO TEÓRICO

La solución implementada se basa en la arquitectura Modelo-Vista-Controlador (MVC), la cual permite separar claramente la lógica del negocio, la presentación al usuario y el control de eventos del sistema. Esta estructura favorece la escalabilidad, el mantenimiento y la evolución del software. Para su implementación, se utilizan clases Java que representan entidades clave como Usuario, Solicitud y Colaborador, y se emplea EntityManager para garantizar una interacción eficiente y robusta con la base de datos.

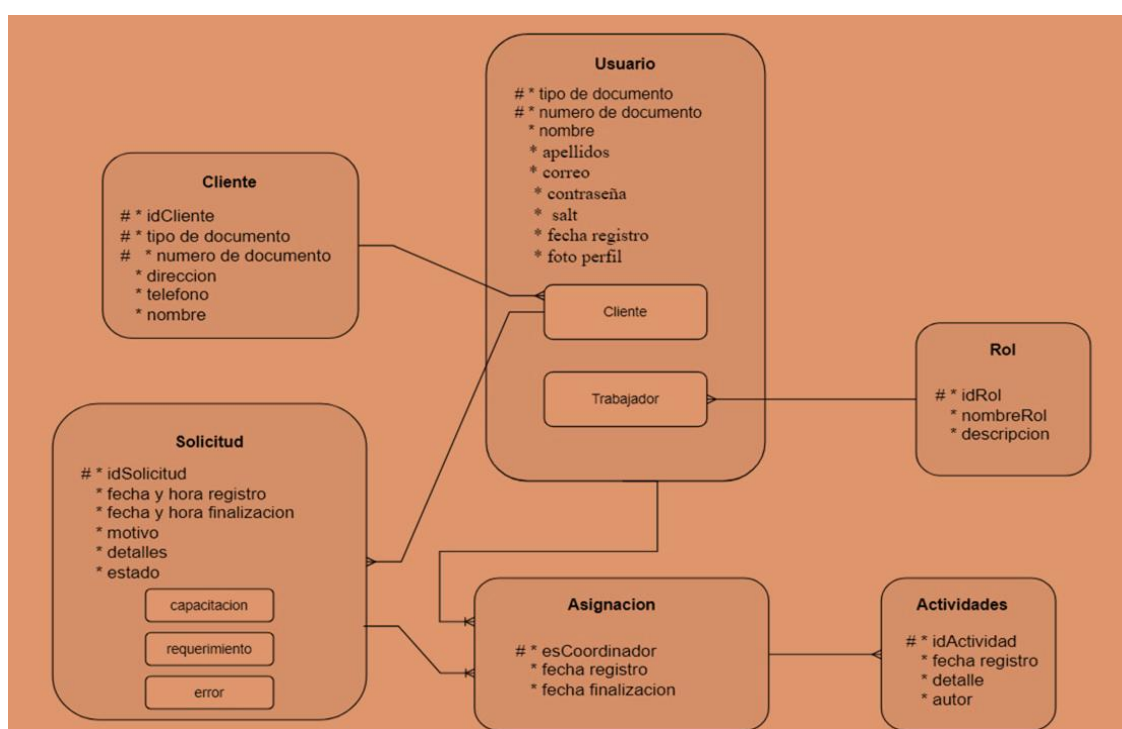
La capa de presentación se construye utilizando XHTML y CSS, ofreciendo una interfaz clara y fácil de usar. Los Managed Beans actúan como controladores, permitiendo la comunicación entre la interfaz y la lógica del negocio. Además, se implementa el patrón de diseño Facade para simplificar las interacciones entre la vista y el modelo, lo que reduce la complejidad general del sistema.

En cuanto al almacenamiento de datos, se opta por PostgreSQL debido a su capacidad para manejar transacciones complejas y su buena escalabilidad. La arquitectura se beneficia también del uso de la inyección de dependencias mediante la anotación @Inject, lo cual promueve un diseño desacoplado y facilita la gestión del ciclo de vida de los objetos.

Ingeniería de Sistemas e Informática
Desarrollo Web Integrado
Proyecto | Soporte Digital

En conjunto, este sistema ofrece una solución modular y eficaz para automatizar los procesos de soporte técnico, mejorar los tiempos de atención y proporcionar una mejor experiencia al cliente. Se prevé que en futuras versiones se continúe con la mejora de la interfaz de usuario e incorporación de nuevas funcionalidades, en función de las necesidades de la empresa.

5.1 DIAGRAMA ENTIDAD-RELACIÓN



5.2 DISEÑO DE BASE DE DATOS

Cliente:

TABLA	Cliente				
DESCRIPCIÓN	Contiene la información de los clientes registrados en el sistema.				
CAMPOS					
NOMBRE	id_cliente	nombre_cliente	tipo_documento	num_documento	correo
DESCRIPCIÓN	Identificador del cliente	Nombre del cliente	Tipo de documento del cliente	Número de documento del cliente	Correo electrónico del cliente

			cliente						
DOMINIO	Número entero	Cadena alfanumérica	Cadena alfanumérica	Cadena numérica	Cadena alfanumérica				
TIPO DE DATO	serial	character varying(100)	character varying(50)	character varying(20)	character varying(100)				
OPCIONALIDAD	NOT NULL	NOT NULL	NULL	NOT NULL	NOT NULL				
UNICIDAD	Sí	No	No	Sí	Sí				
LLAVE	PK			UNIQUE	UNIQUE				
RESTRICCIONES	Secuencial que inicia en 1	Solo aplica a clientes	Solo para documentos válidos	Solo para clientes con documento válido	Formato válido de correo				
VALOR POR DEFECTO			DNI / RUC		@.com				
EJEMPLOS	1	Licia Medina	DNI	12345679	licia@email.com				
	2	Neyder Sanchez	DNI	11556677	neyder1@gmail.com				
	3	Sofia Barco	RUC	12034567891	sofia@hotmail.com				
NOMBRE	contraseña	salt	direccion	telefono	fecha_registro	foto_perfil			
DESCRIPCIÓN	Contraseña del cliente	Salt usado para la contraseña	Dirección del cliente	Teléfono del cliente	Fecha de registro del cliente	Ruta o URL de la foto de perfil del cliente			
DOMINIO	Cadena alfanumérica	Cadena alfanumérica	Cadena alfanumérica	Cadena numérica	Fecha	Cadena alfanumérica			
TIPO DE DATO	character varying(255)	character varying(255)	character varying(255)	character varying(20)	date	character varying(255)			
OPCIONALIDAD	NOT NULL	NULL	NULL	NULL	NULL	NULL			
UNICIDAD	No	No	No	No	No	No			

LLAVE									
RESTRICCIONES	Debe estar hashhead a	Ninguna restricción adicional	Ninguna restricción adicional	Ninguna restricción adicional	Ninguna restricción adicional	Ninguna restricción adicional	Ninguna restricción adicional		
VALOR POR DEFECTO									
EJEMPLOS	Juana*123	Abfnsioddfr359upñqabwefwqa	Jr de la union 15	123456789	13/11/2025	null			
	A%1521	asf1a4f4rrfrgt984968gfhdsdh	Av. los jardines 152	123456789	19/12/2020	whatsapp21524.jpg			
	wbd15/hasb	asf1as6tgtrefr95fg41as695fg1a	Urb. la molina 153	12345789	06/8/2025	foto.png			

Rol

TABLA	Rol		
DESCRIPCIÓN	Contiene la información de los roles asignados a los colaboradores en el sistema.		
CAMPOS			
NOMBRE	id_rol	nombre_rol	descripcion
DESCRIPCIÓN	Identificador del rol	Nombre del rol	Descripción del rol
DOMINIO	Número entero	Cadena alfanumérica	Cadena alfanumérica
TIPO DE DATO	serial	character varying(50)	character varying(255)
OPCIONALIDAD	NOT NULL	NOT NULL	NULL
UNICIDAD	Sí	Sí	No
LLAVE	PK	UNIQUE	
RESTRICCIONES	Secuencial que inicia en 1	Ninguna restricción adicional	Ninguna restricción adicional
VALOR POR DEFECTO			
EJEMPLOS	1	Administrador	Control total
	2	Analista	Acceso a dashboards

	3	Programador	Control de actividades
	4	Diseñador	desempeña actividades en las solicitudes

Colaborador

TABLA	Colaborador					
DESCRIPCIÓN	Contiene la información de los colaboradores/ trabajadores de la organización					
CAMPOS						
NOMBRE	id_colaborador	nombre_colabor	apellidos	dni	correo	contraseña
DESCRIPCIÓN	Identificador del colaborador	Nombre del colaborador	Apellidos del colaborador	DNI del colaborador	Correo electrónico del colaborador	Contraseña del colaborador
DOMINIO	Número entero	Cadena alfanumérica	Cadena alfanumérica	Cadena numérica	Cadena alfanumérica	Cadena alfanumérica
TIPO DE DATO	serial	character varying(100)	character varying(100)	character varying(20)	character varying(100)	character varying(255)
OPCIONALIDAD	NOT NULL	NOT NULL	NOT NULL	NOT NULL	NOT NULL	NOT NULL
UNICIDAD	Sí	No	No	Sí	Sí	No
LLAVE	PK			UNIQUE	UNIQUE	
RESTRICCIONES	Secuencial que inicia en 1	Solo aplica a colaboradores	Ninguna restricción adicional	Solo para colaboradores con DNI válido	Formato válido de correo	Debe estar hasheada
VALOR POR DEFECTO					@.com	
EJEMPLOS	1	Licia	Villa	12345678	marta@email.com	miyvghh*123

	2	Sofia	Palomino	87654321	jose15@gmail.com	Axdccd%1521
	3	Neyder	Gutierrez	14785239	paulo15_12@hotmail.com	abcd15//eas
NOMBRE	salt	direccion	telefono	fecha_registro	foto_perfil	id_rol
DESCRIPCIÓN	Salt usado para contraseña	Dirección del colaborador	Teléfono del colaborador	Fecha de registro del colaborador	Ruta o URL de la foto de perfil del colaborador	Identificador del rol asignado al colaborador
DOMINIO	Cadena alfanumérica	Cadena alfanumérica	Cadena numérica	Fecha	Cadena alfanumérica	Número entero
TIPO DE DATO	character varying(255)	character varying(255)	character varying(20)	date	character varying(255)	integer
OPCIONALIDAD	NULL	NULL	NULL	NULL	NULL	NOT NULL
UNICIDAD LLAVE	No	No	No	No	No	No
RESTRICCIONES	Ninguna restricción adicional	Ninguna restricción adicional	Ninguna restricción adicional	Ninguna restricción adicional	Ninguna restricción adicional	Referencia a la tabla de roles
VALOR POR DEFECTO						
EJEMPLOS	as14f956as1f1a6s5f1	Jr de los jardines 14	123456789	15/12/2025	null	1
	as65f165asd1gf56a	Av. Lambayeque d	987324321	19/2/2025	foto.jpg	2
	asf51a6s51gf65aaa	Urb. la molina 153	132785239	6/2/2025	img_151219.png	3

Solicitud

TABLA	Solicitud				
DESCRIPCIÓN	Contiene la información de las solicitudes presentadas por los clientes				
CAMPOS					
NOMBRE	id_solicitud	id_cliente	nro_solicitud	motivo	
DESCRIPCIÓN	Identificador de la solicitud	Identificador del cliente asociado	Número único de la solicitud	Motivo de la solicitud	
DOMINIO	Número entero	Número entero	Número entero	Cadena alfanumérica	
TIPO DE DATO	serial	integer	integer	character varying(500)	
OPCIONALIDAD	NOT NULL	NOT NULL	NOT NULL	NOT NULL	
UNICIDAD	Sí	No	No	No	
LLAVE	PK	FK			
RESTRICCIONES	Secuencial que inicia en 1	Referencia a la tabla de clientes	Ninguna restricción adicional	Requerimiento, Capacitacion, Error	
VALOR POR DEFECTO					
EJEMPLOS	1	1	1	requerimiento	
	2	2	1	capacitacion	
	3	3	2	error	
NOMBRE	detalles	estado	fecha_registro	fecha_finalizacion	imagen
DESCRIPCIÓN	Detalles adicionales de la solicitud	Estado actual de la solicitud	Fecha en que se registró la solicitud	Fecha estimada o real de finalización	Imagen asociada con la solicitud
DOMINIO	Texto	Cadena alfanumérica	Timestamp	Timestamp	Cadena alfanumérica

TIPO DE DATO	text	character varying(50)	timestamp without time zone	timestamp without time zone	character varying(1000)
OPCIONALIDAD	NOT NULL	NOT NULL	NOT NULL	NULL	NULL
UNICIDAD	No	No	No	No	No
LLAVE					
RESTRICCIONES	Texto libre	Estados posibles: "Pendiente", "En Proceso", "Finalizado"	Ninguna restricción adicional	Ninguna restricción adicional	Ninguna restricción adicional
VALOR POR DEFECTO					
EJEMPLOS	expandi mi negocion ayudame	Pendiente	19/12/2024 15:01:34	25/12/2024 19:53:00	null
	asesorame porfa	En proceso	02/07/2024	31/07/2024	null
	tengo error ayudame	Finalizado	29/12/2024	02/01/2025	captura.png

Asignación

TABLA	Asignacion				
DESCRIPCIÓN	Contiene la informacion de las asignaciones entre las solicitudes con los colaboradores correspondientes				
CAMPOS					
NOMBRE	id_asignacion	id_solicitud	id_colaborador	fecha_asignacion	es_coordinador
DESCRIPCIÓN	Identificador de la asignación	Identificador de la solicitud asociada	Identificador del colaborador asignado	Fecha en que se realizó la asignación	Indica si el colaborador es el coordinador de la solicitud
DOMINIO	Número entero	Número entero	Número entero	Fecha	Booleano
TIPO DE DATO	serial	integer	integer	date	boolean
OPCIONALIDAD	NOT NULL	NOT NULL	NOT NULL	NOT NULL	NOT NULL

UNICIDAD	Sí	No	No	No	No
LLAVE	PK	FK	FK		
RESTRICCIONES	Secuencial que inicia en 1	Relación con la tabla de solicitudes	Relación con la tabla de colaboradores	Ninguna restricción adicional	Valores posibles: true o false
VALOR POR DEFECTO					
EJEMPLOS	1	1	1	19/12/2025	TRUE
	2	1	5	19/12/2025	FALSE
	3	2	2	15/11/2025	TRUE

Actividad

TABLA	Actividad				
DESCRIPCIÓN	Contiene la información de las actividades que se llevaran a cabo dentro de la asignacion				
CAMPOS					
NOMBRE	id_actividad	id_asignacion	descripcion	tiempo_invertido	fecha_registro
DESCRIPCIÓN	Identificador único de la actividad	Identificador de la asignación asociada	Descripción detallada de la actividad realizada	Tiempo dedicado a la actividad	Fecha en que se registró la actividad
DOMINIO	Número entero	Número entero	Texto	Intervalo	Fecha
TIPO DE DATO	serial	integer	text	interval	date
OPCIONALIDAD	NOT NULL	NOT NULL	NOT NULL	NOT NULL	NOT NULL
UNICIDAD	Sí	No	No	min	No
LLAVE	PK	FK			
RESTRICCIONES	Secuencial que inicia en 1	Relación con la tabla de asignaciones	Ninguna restricción adicional	Ninguna restricción adicional	Ninguna restricción adicional

VALOR POR DEFECTO					
EJEMPLOS	1	1	hicimos esto	20	19/12/2025
	2	1	añadimos esto	150	19/12/2025
	3	2	configuramos el sistema	15	15/11/2024

5.3 SCRIPT DDL DE BASE DE DATOS

Se desarrolló un script DDL encargado de definir las tablas, índices y restricciones de la base de datos. Este script contempla la creación de tablas como usuarios, colaboradores, asignaciones, solicitudes y actividades, entre otras, y establece claves primarias y foráneas para asegurar la integridad referencial del sistema.

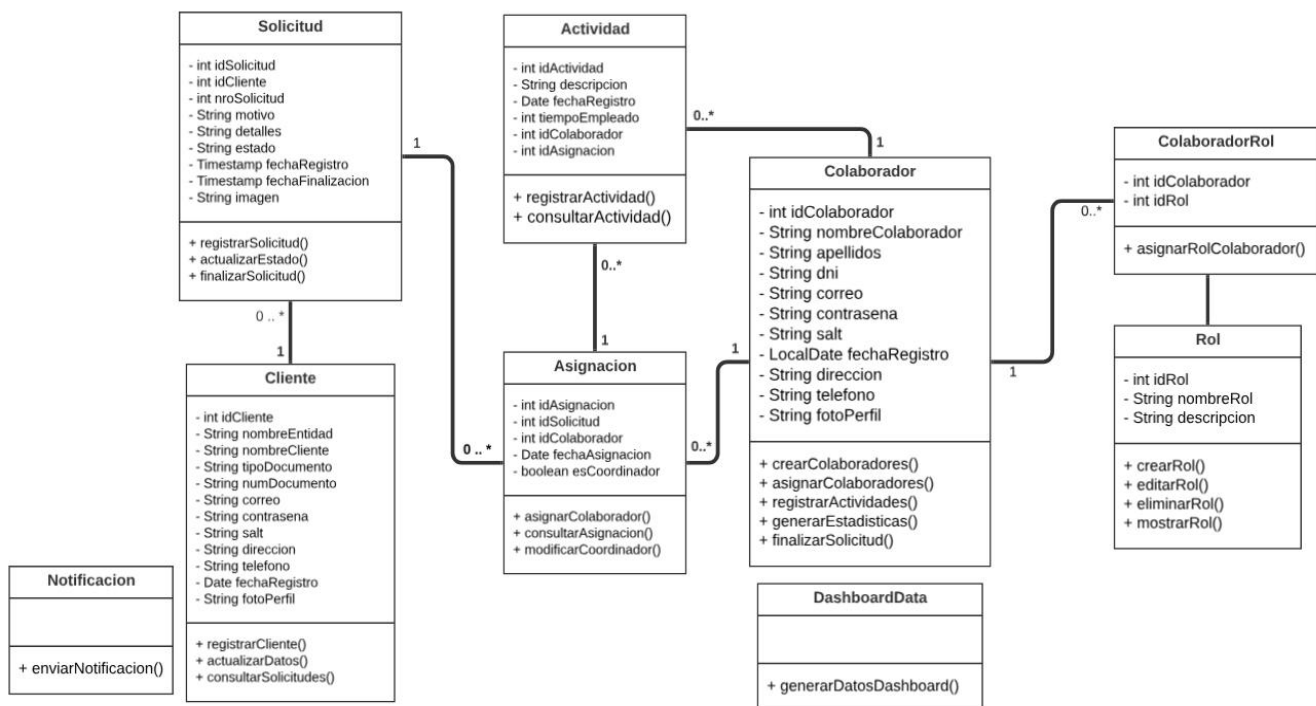
```

17  CREATE TABLE cliente (
18      id_cliente SERIAL PRIMARY KEY,
19      nombre_entidad VARCHAR(150),
20      nombre_cliente VARCHAR(100) NOT NULL,
21      tipo_documento VARCHAR(50) NOT NULL,
22      num_documento VARCHAR(20) UNIQUE NOT NULL,
23      correo VARCHAR(100) UNIQUE NOT NULL,
24      contrasena VARCHAR(255) NOT NULL,
25      salt VARCHAR(255) NOT NULL,
26      direccion VARCHAR(255),
27      telefono VARCHAR(20),
28      fecha_registro DATE NOT NULL,
29      foto_perfil VARCHAR(255)
30  );
31
32  CREATE TABLE colaborador (
33      id_colaborador SERIAL PRIMARY KEY,
34      nombre_colaborador VARCHAR(100) NOT NULL,
35      apellidos VARCHAR(100) NOT NULL,
36      dni VARCHAR(20) UNIQUE NOT NULL,
37      correo VARCHAR(100) UNIQUE NOT NULL,
38      contrasena VARCHAR(255) NOT NULL,
39      salt VARCHAR(255) NOT NULL,
40      direccion VARCHAR(255),
41      telefono VARCHAR(20),
42      fecha_registro DATE NOT NULL,
43      foto_perfil VARCHAR(255),
44      id_rol INTEGER NOT NULL
45  );
46
47  CREATE TABLE rol (
48      id_rol SERIAL PRIMARY KEY,
49      nombre_rol VARCHAR(255) NOT NULL,
50      descripcion TEXT
51  );
52
53  CREATE TABLE solicitud (
54      id_solicitud SERIAL PRIMARY KEY,
55      id_cliente INTEGER NOT NULL,
56      nro_solicitud INTEGER NOT NULL,
57      motivo VARCHAR(500) NOT NULL,
58      detalles TEXT NOT NULL,
59      estado VARCHAR(50) NOT NULL,
60      fecha_registro TIMESTAMP NOT NULL,
61      fecha_finalizacion TIMESTAMP,
62      imagen VARCHAR(1000)
63  );
64
65  CREATE TABLE actividad (
66      id_actividad SERIAL PRIMARY KEY,
67      id_asignacion INTEGER NOT NULL,
68      descripcion TEXT NOT NULL,
69      fecha_registro TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL,
70      id_colaborador INTEGER NOT NULL,
71      tiempo_requerido BIGINT NOT NULL
72  );

```

```
74 CREATE TABLE asignacion (
75     id_asignacion SERIAL PRIMARY KEY,
76     id_solicitud INTEGER NOT NULL,
77     id_colaborador INTEGER NOT NULL,
78     fecha_asignacion DATE NOT NULL,
79     es_coordinador BOOLEAN DEFAULT FALSE NOT NULL
80 );
```

5.4 DIAGRAMA DE CLASES



Cada clase está compuesta por atributos y métodos que reflejan la lógica del negocio.

5.5 IMPLEMENTACIÓN DEL PROYECTO WEB EN JAVA AL 50%

La arquitectura del sistema sigue el modelo Modelo-Vista-Controlador (MVC), que asegura la separación de responsabilidades y facilita el mantenimiento, la escalabilidad y la organización del proyecto. A continuación, se describe la implementación de cada uno de sus componentes:

5.5.1 Implementación del componente Modelo:

El Modelo se encarga de representar las entidades del negocio y la lógica relacionada con los datos.

- Las clases del modelo representan directamente las entidades de la base de datos, como Cliente, Colaborador, Solicitud, y Asignación.
- Estas clases están integradas con JPA (Java Persistence API) y utilizan EntityManager para la persistencia y manipulación de datos.
- Para la interacción directa con la base de datos, se implementaron DAOs (Data Access Objects) que encapsulan las operaciones CRUD y aseguran una comunicación eficiente con la base de datos.
- El patrón Facade se utiliza para proporcionar una interfaz simplificada que agrupa operaciones relacionadas y conecta el modelo con los servicios de negocio.

5.5.2 Implementación del componente Controlador:

El Controlador gestiona las solicitudes de los usuarios y actúa como intermediario entre la vista y el modelo.

- Se implementaron Managed Beans para recibir y procesar las acciones del usuario. Estos Beans coordinan las operaciones entre las vistas (XHTML) y la lógica del negocio definida en los servicios y DAOs.
- Los Managed Beans también permiten la integración con los servicios que encapsulan las reglas de negocio, manteniendo el código modular y organizado.
- El uso de inyección de dependencias (@Inject) garantiza que los Beans y servicios estén correctamente inicializados por el contenedor, promoviendo un desarrollo desacoplado y facilitando la gestión de dependencias.

5.5.3 Implementación del componente Vista:

El componente Vista proporciona la interfaz de usuario.

- Las vistas fueron desarrolladas con JSF (JavaServer Faces) utilizando XHTML para las páginas web, junto con CSS y JavaScript para mejorar la experiencia del usuario y la interactividad.
- Se implementaron archivos reutilizables como styles.css, LoginBean ya que habían 2 o más aplicaciones que requerían de estas funciones o códigos, definiendo elementos comunes como encabezados, menús y pies de página o funciones reutilizables en diferentes archivos.
- Las vistas están diseñadas para ser intuitivas y dinámicas, integrándose fácilmente con los Managed Beans para mostrar y capturar datos.

5.5.4 Patrones Implementados:

En el sistema se implementaron varios patrones de diseño que estructuran el código, mejoran la organización y aseguran la escalabilidad del proyecto:

- **Modelo MVC**
Es un patrón arquitectónico que separa la lógica de negocio (Modelo), la presentación (Vista) y la gestión de eventos (Controlador).
- **DAO (Data Access Object):**
Los DAOs encapsulan las operaciones de acceso a la base de datos y utilizan EntityManager para interactuar con las entidades de negocio.
Cada entidad tiene un DAO correspondiente, lo que permite un acceso centralizado y reutilizable a la base de datos.

- Facade:

Este patrón se utiliza para simplificar la interacción entre los controladores y los DAOs, agrupando operaciones relacionadas en una única interfaz.

Facilita el mantenimiento al centralizar la lógica de negocio que interactúa con múltiples DAOs.

Herramientas adicionales

- Managed Beans:

Actúan como controladores y manejan la comunicación entre la vista y los servicios del sistema.

Permiten desacoplar la lógica de presentación de la lógica de negocio.

- EntityManager:

Gestiona las transacciones y operaciones de persistencia con la base de datos.

Es un componente central para manejar la sincronización entre las entidades y la base de datos.

Inyección de Dependencias (@Inject):

Promueve un desarrollo desacoplado al permitir que los servicios, DAOs y Beans sean inyectados automáticamente por el contenedor de la aplicación.

- Filters:

Se implementaron filtros para manejar la seguridad de la aplicación, garantizando que solo usuarios autenticados accedan a las secciones protegidas.

También se utilizaron filtros para controlar el acceso basado en roles.

- Servicios (Service):

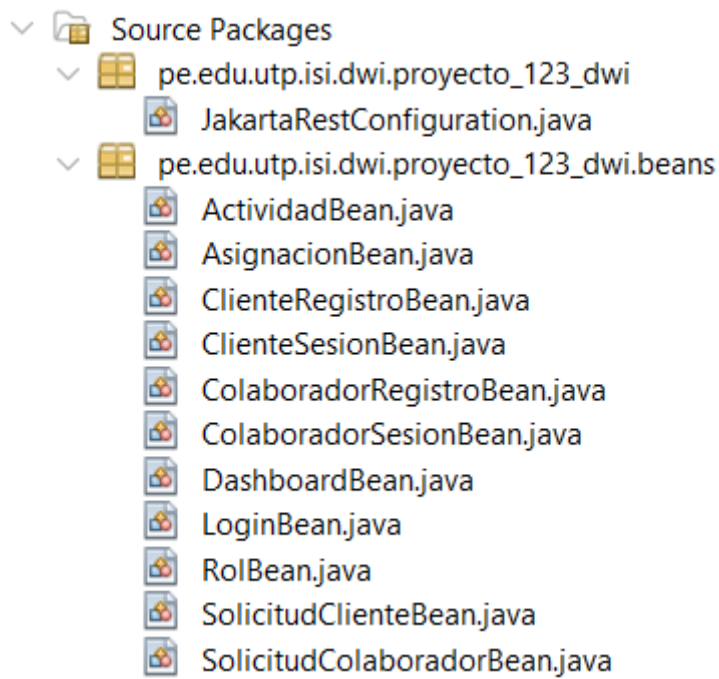
Centralizan la lógica de negocio más compleja, delegando las operaciones de persistencia a los DAOs.

Estos servicios son utilizados tanto por los Managed Beans como por otras capas del sistema para encapsular reglas de negocio.

- API de Correos:

Se integró una API para el envío automático de notificaciones por correo electrónico.

Esta funcionalidad permite mantener informados a los usuarios sobre el estado de sus solicitudes, recuperación de contraseñas y otras acciones importantes.



- pe.edu.utp.isi.dwi.proyecto_123_dwi.facade
 - ActividadFacade.java
 - AsignacionFacade.java
 - ClienteFacade.java
 - ColaboradorFacade.java
 - RolFacade.java
 - SolicitudFacade.java
 - pe.edu.utp.isi.dwi.proyecto_123_dwi.filters
 - AuthFilter.java
 - RolFilter.java
 - pe.edu.utp.isi.dwi.proyecto_123_dwi.main
 - Main.java
 - pe.edu.utp.isi.dwi.proyecto_123_dwi.model
 - EntityManagerProducer.java
 - pe.edu.utp.isi.dwi.proyecto_123_dwi.resources
 - JakartaEE10Resource.java
 - pe.edu.utp.isi.dwi.proyecto_123_dwi.util
 - SecurityUtils.java
 - SendGridService.java
 - pe.edu.utp.isi.dwi.proyecto_123_dwi.dao
 - ActividadDAO.java
 - AsignacionDAO.java
 - ClienteDAO.java
 - ColaboradorDAO.java
 - RolDAO.java
 - SolicitudDAO.java
 - pe.edu.utp.isi.dwi.proyecto_123_dwi.entities
 - Actividad.java
 - Asignacion.java
 - Cliente.java
 - Colaborador.java
 - Rol.java
 - Solicitud.java

- faces
 - cliente
 - actividades_realizadas.xhtml
 - actualizar_datos.xhtml
 - crear_solicitud.xhtml
 - mis_solicitudes.xhtml
 - perfil_cliente.xhtml
 - portal_cliente.xhtml
 - colaborador
 - actualizar_colaborador.xhtml
 - control_actividad.xhtml
 - control_asignacion.xhtml
 - control_cliente.xhtml
 - control_colaborador.xhtml
 - control_dashboard.xhtml
 - control_rol.xhtml
 - control_solicitud.xhtml
 - perfil_colaborador.xhtml
 - portal_colaborador.xhtml
 - registro_colaborador.xhtml
 - resources
 - css
 - control_cliente_styles.css
 - control_colaborador_actividad.css
 - control_colaborador_asignacion.css
 - control_colaborador_rol.css
 - control_colaborador_solicitud.css
 - control_colaborador_styles.css
 - registro_cliente.css
 - styles.css
 - images
 - foto_perfil_default.jpg
 - logo_empresa.png
 - messages.properties
 - index.xhtml
 - login.xhtml
 - registro_cliente.xhtml
 - registro_exitoso.xhtml

Estructura de Vistas

El sistema utiliza JSF (JavaServer Faces) con páginas XHTML, organizadas en directorios claros según los roles y funcionalidades, con el objetivo de garantizar una experiencia de usuario fluida y eficiente. Esta arquitectura asegura que cada rol tenga acceso a vistas específicas y que las funcionalidades estén bien separadas.

1. Páginas Generales

Estas páginas sirven como punto de entrada o proporcionan funcionalidades generales del sistema:

- `index.xhtml`: Página principal que actúa como el primer punto de contacto del usuario con la plataforma.
- `login.xhtml`: Permite a los usuarios autenticarse en el sistema proporcionando su correo y contraseña.
- `registro_cliente.xhtml` y `registro_exitoso.xhtml`: Gestionan el registro de nuevos usuarios y confirman la creación exitosa de cuentas.
- `messages.properties`: Archivo para la internacionalización y manejo de mensajes en las vistas.

2. Vistas para Clientes

Ubicadas en la carpeta `faces/cliente`, estas páginas están diseñadas para las interacciones de los clientes con el sistema:

- `actualizar_datos.xhtml`: Página para que los clientes actualicen su información personal.

- crear_solicitud.xhtml: Proporciona un formulario para que los clientes puedan registrar nuevas solicitudes de soporte.
- mis_solicitudes.xhtml: Muestra un listado de las solicitudes realizadas por el cliente.
- perfil_cliente.xhtml: Permite a los clientes gestionar su perfil, incluyendo detalles como su foto de perfil y datos de contacto.
- portal_cliente.xhtml: Dashboard principal para los clientes, donde pueden acceder rápidamente a las funcionalidades relevantes.

3. Vistas para Colaboradores

Ubicadas en la carpeta faces/colaborador, estas vistas están diseñadas para los colaboradores de la empresa según sus roles específicos:

- actualizar_colaborador.xhtml: Página para que los colaboradores actualicen su información personal.
- control_actividad.xhtml: Permite gestionar las actividades asignadas a los colaboradores.
- control_asignacion.xhtml: Página para administrar la asignación de solicitudes a colaboradores específicos.
- control_cliente.xhtml: Muestra un listado de los clientes registrados en el sistema.
- control_colaborador.xhtml: Administra los datos de los colaboradores, incluyendo roles y permisos.
- control_dashboard.xhtml: Dashboard principal con métricas y

estadísticas relevantes para los colaboradores.

- control_rol.xhtml: Administra los roles disponibles en el sistema.
- control_solicitud.xhtml: Gestiona las solicitudes asignadas y su estado.
- perfil_colaborador.xhtml: Permite a los colaboradores gestionar sus perfiles personales.
- portal_colaborador.xhtml: Dashboard general para los colaboradores, con accesos rápidos a sus funcionalidades.
- registro_colaborador.xhtml: Página para registrar nuevos colaboradores.

4. Recursos Estáticos

Ubicados en la carpeta resources, estos archivos proporcionan estilos, imágenes y configuraciones reutilizables en las vistas:

- Carpeta css:

Contiene archivos de estilos específicos para cada módulo, como control_cliente_styles.css o control_colaborador_styles.css, así como un archivo general styles.css para los elementos compartidos.

- Carpeta images:

Incluye recursos gráficos como foto_perfil_default.jpg y logo_empresa.png.

- **Estructura de Paquetes**

- Paquete Beans (pe.edu.utp.isi.dwi.proyecto_123_dwi.beans)
 - Propósito: Implementar los controladores de la aplicación.

- Rol en MVC: Actúan como el Controlador, gestionando las solicitudes de los usuarios y coordinando las interacciones entre el modelo y las vistas.
- Características:
 - Gestionan la lógica de navegación y las acciones específicas solicitadas por el usuario.
 - Se inyectan servicios (Facade) para interactuar con la capa de negocio.
 - Manejan estados de sesión, especialmente para clientes y colaboradores.
- Clases destacadas:
 - LoginBean: Gestiona el proceso de autenticación.
 - ClienteSessionBean y ColaboradorSessionBean: Gestionan datos específicos de sesión para cada tipo de usuario.
 - SolicitudClienteBean y SolicitudColaboradorBean: Controlan la gestión de solicitudes según el tipo de usuario.
- Paquete Entities (pe.edu.utp.isi.dwi.proyecto_123_dwi.entities)
 - Propósito: Representar las entidades del negocio.
 - Rol en MVC: Representan el Modelo, encapsulando los datos y su comportamiento.
 - Características:
 - Cada clase corresponde directamente a una tabla en la base de

datos.

- Mapeadas con JPA para facilitar la persistencia.
- Clases destacadas:
 - Cliente, Colaborador, Solicitud, Rol, Actividad, etc.: Representan las entidades principales del sistema.
- Paquete DAO (pe.edu.utp.isi.dwi.proyecto_123_dwi.dao)
 - Propósito: Gestionar la persistencia de datos utilizando el patrón DAO.
 - Rol en MVC: Parte del Modelo, ya que conecta las entidades con la base de datos.
 - Características:
 - Implementan las operaciones CRUD para cada entidad.
 - Utilizan EntityManager para interactuar con la base de datos.
 - Inyectados en los servicios para encapsular el acceso a datos.
 - Clases destacadas:
 - ClienteDAO, SolicitudDAO, ActividadDAO, etc.: Encapsulan las consultas y operaciones específicas de cada entidad.
- Paquete Facade (pe.edu.utp.isi.dwi.proyecto_123_dwi.facade)
 - Propósito: Proporcionar una interfaz simplificada para la lógica de negocio, implementando el patrón Facade.
 - Rol en MVC: Parte del Modelo, encargándose de la lógica de negocio.
 - Características:

- Agrupan operaciones complejas que involucran varias entidades o DAOs.
 - Centralizan las reglas de negocio y evitan duplicación de código en los controladores.
- Clases destacadas:
 - ClienteFacade: Gestiona las operaciones relacionadas con los clientes.
 - SolicitudFacade: Encapsula la lógica de gestión de solicitudes, trabajando con los DAOs necesarios.
- Paquete Filters (pe.edu.utp.isi.dwi.proyecto_123_dwi.filters)
 - Propósito: Gestionar la seguridad y el acceso a las vistas protegidas.
 - Rol en el sistema: Control transversal, asegurando que las acciones de los usuarios estén autenticadas y autorizadas.
 - Características:
 - Implementan la lógica de seguridad mediante interceptación de solicitudes.
 - Clases destacadas:
 - AuthFilter: Verifica que el usuario esté autenticado antes de permitir el acceso a páginas protegidas.
 - RolFilter: Controla el acceso basado en roles.
- Paquete Util (pe.edu.utp.isi.dwi.proyecto_123_dwi.util)

- Propósito: Proporcionar herramientas y servicios comunes para la aplicación.
- Características:
 - Reduce la duplicación de código al centralizar funcionalidades comunes.
 - Clases utilitarias reutilizables.
- Clases destacadas:
 - SecurityUtils: Maneja el cifrado y verificación de contraseñas.
 - SendGridService: Gestiona el envío de correos electrónicos mediante la integración con una API.
- Paquete Main (pe.edu.utp.isi.dwi.proyecto_123_dwi.main)
 - Propósito: Punto de entrada del sistema o configuración inicial de la aplicación.
 - Clase destacada:
 - Main.java: Contiene la configuración principal de la aplicación.
- Paquete Model (pe.edu.utp.isi.dwi.proyecto_123_dwi.model)
 - Propósito: Centralizar configuraciones y aspectos transversales, como la conexión a la base de datos.
 - Clase destacada:
 - EntityManagerProducer: Maneja la producción de instancias de EntityManager para inyectarlas donde sean necesarias.

- Recursos (pe.edu.utp.isi.dwi.proyecto_123_dwi.resources)
 - Propósito: Gestionar configuraciones globales y utilidades de API.
 - Clases destacadas:
 - JakartaEE10Resource: Proporciona configuraciones para el entorno Jakarta EE.

Relación entre los Paquetes

La arquitectura sigue el patrón MVC y la organización modular de la aplicación permite que cada capa sea clara y funcional:

- Modelo:
- Incluye las entidades (entities), DAOs (dao), y la lógica de negocio en los servicios (facade).
- Vista:
 - Representada en las páginas XHTML organizadas según el rol del usuario.
- Controlador:
 - mediante Managed Beans (beans), que gestionan la interacción entre la vista y el modelo.
- Utilidades y Seguridad:
 - Los filtros controlan el acceso, mientras que las clases utilitarias proporcionan funciones transversales.

Ventajas del Diseño

- Escalabilidad: La modularidad del diseño facilita la adición de nuevas funcionalidades sin afectar el sistema existente.
- Reutilización: Los DAOs, Facades y Utilidades son fácilmente reutilizables en múltiples contextos.
- Seguridad: El uso de filtros asegura el control de acceso y la autenticación.

Mantenimiento: La separación de responsabilidades permite identificar y corregir problemas r ápidamente.

6.CONCLUSIONES

- ✓ Los objetivos establecidos para el proyecto se cumplieron satisfactoriamente. Se implementó una plataforma web centralizada que permite gestionar eficientemente las solicitudes de soporte técnico, asignar tareas y realizar el seguimiento de actividades, lo que ha optimizado los procesos internos de la empresa 123digit@l y ha mejorado la trazabilidad operativa.
- ✓ Los resultados alcanzados evidencian una notable mejora en la eficiencia operativa, destacando una disminución del 30 % en los tiempos de respuesta y una asignación de tareas más clara y específica. Estos avances han tenido un efecto positivo tanto en la satisfacción de los clientes como en la de los colaboradores.
- ✓ Durante el desarrollo se presentaron retos técnicos importantes, como lograr una separación adecuada entre las distintas capas del sistema y conectar correctamente la lógica del negocio con la base de datos. Estas dificultades fueron superadas mediante la aplicación de patrones de diseño como DAO y Facade, el uso de la inyección de dependencias en los Managed Beans y la utilización de EntityManager para garantizar una persistencia de datos eficiente.
- ✓ Entre las principales lecciones aprendidas se destaca la utilidad de emplear arquitecturas robustas como Modelo-Vista-Controlador (MVC), que facilita el mantenimiento y la escalabilidad del software. También se valoró el uso de Managed Beans para estructurar la lógica del negocio y el diseño de interfaces amigables con XHTML, que contribuyen significativamente a mejorar la experiencia del usuario.
- ✓ El sistema tuvo un impacto importante en la automatización de tareas que antes se realizaban de forma manual, lo cual incrementó la eficiencia operativa y

permitió a los administradores tomar decisiones más acertadas mediante reportes dinámicos. La integración de la API de Sendgrid para el envío automático de notificaciones fortaleció la comunicación con los usuarios, mejorando la confianza en el servicio.

7.RECOMENDACIONES

- ✓ Es fundamental mantener una supervisión constante del desempeño del sistema, especialmente en lo que respecta a la interacción entre la lógica de negocio y la capa de persistencia mediante EntityManager, con el fin de asegurar su estabilidad y escalabilidad frente al crecimiento en la cantidad de usuarios y solicitudes.
- ✓ Se recomienda mejorar la interfaz de usuario incorporando tecnologías más actuales y adaptables, como PrimeFaces o librerías de diseño adicionales, que ofrezcan una experiencia de navegación más fluida, atractiva y compatible con diversos dispositivos.
- ✓ Evaluar la integración de frameworks modernos de frontend como React o Angular puede aportar mayor dinamismo e interactividad al sistema, lo que contribuiría a una experiencia de usuario más satisfactoria y sencilla.
- ✓ Es esencial brindar capacitación a los colaboradores de la empresa en el uso eficiente de la plataforma, para asegurar que se aprovechen plenamente todas sus funcionalidades, reduciendo errores y mejorando los tiempos de atención.
- ✓ La implementación de herramientas de análisis predictivo permitiría anticiparse a las necesidades de los clientes e identificar patrones en las solicitudes, proporcionando a la empresa información valiosa para una toma de decisiones más proactiva e inteligente.
- ✓ Realizar evaluaciones periódicas del sistema, tanto en términos de rendimiento como de usabilidad, facilitará la detección de áreas de mejora y garantizará que el software continúe evolucionando conforme a las exigencias de la empresa y de sus usuarios.
- ✓ Llevar a cabo una revisión constante de los patrones de diseño aplicados, como DAO, Facade y Managed Beans, contribuirá al mantenimiento de una arquitectura clara, modular y sostenible, lo que facilitará su mantenimiento y escalabilidad a largo plazo.

8.REFERENCIAS BIBLIOGRAFICAS


- ✓ Blanco, J. (2020). Programación en Java: Guía Completa para Desarrolladores. Editorial ABC.
- ✓ Pérez, A., & Torres, M. (2018). Desarrollo Web con Java: Conceptos y Prácticas para Aplicaciones Empresariales. Editorial XYZ

- ✓ PostgreSQL Global Development Group. (2023). PostgreSQL 16 Documentation. Recuperado de [PostgreSQL](#).
- ✓ Jenkov, J. (2020). Introduction to the Model-View-Controller (MVC) Pattern. Recuperado de [Jenkov](#).
- ✓ Oracle Corporation. (2024). Java Platform, Enterprise Edition. Retrieved from <https://docs.oracle.com/javaee/>
- ✓ Freeman, E., & Robson, E. (2020). Headfirst Design Patterns. O'Reilly Media.

1. ANEXOS

https://github.com/SanchezRodas/Proyecto_Desarrollo_Web_Integrado



 123digit@l

Inicio Regístrese Iniciar Sesión


Iniciar Sesión

Correo Electrónico:

Contraseña:

Iniciar Sesión

[¿No tienes cuenta? Regístrate aquí](#)

 123digit@l

Inicio Regístrese Iniciar Sesión

Registrar Nuevo Cliente

Nombre de la Entidad:

Nombre del Cliente:

Tipo de Documento:

Seleccione ▼

Número de Documento:

Correo:


El correo electrónico debe cumplir las siguientes reglas:

- Debe tener un formato válido (por ejemplo: usuario@dominio.com).
- Puede incluir letras, números, puntos, guiones y el símbolo "+".
- Debe comenzar el símbolo "@" y un dominio válido (por ejemplo: gmail.com).

Contraseña:

La contraseña debe cumplir las siguientes reglas:

- Al menos 8 caracteres.
- Al menos una letra mayúscula.
- Al menos un número.
- Al menos un carácter especial (por ejemplo: @!%*^&).

 123digit@l

Inicio Regístrese Iniciar Sesión

Iniciar Sesión

Correo Electrónico:

Contraseña:

Iniciar Sesión

[¿No tienes cuenta? Regístrate aquí](#)

© 2025 123digit@l. Todos los derechos reservados.
Tu socio en soluciones de software innovadoras.

123 milhas

[Inicio](#) [Mis Solicitudes](#) [Mi Perfil](#) [Contacto](#)

Bienvenido, Verenise | [Cerrar Sesión](#)

Mis Solicitudes
Revisa y administra las solicitudes que has realizado.
[Ver Solicitudes](#)

Crear Solicitud
Inicia una nueva solicitud rápidamente.
[Crear Solicitud](#)

Mi Perfil
Mantén tu información personal y de contacto actualizada.
[Actualizar Información](#)

Soporte
¿Necesitas ayuda? Contacta a nuestro equipo de soporte.
[Ir a Soporte](#)

© 2025 123digit@L. Todos los derechos reservados.

123 milhas

[Inicio](#) [Mis Solicitudes](#) [Mi Perfil](#) [Contacto](#)

Bienvenido, Verenise | [Cerrar Sesión](#)

Mis Solicitudes

Crear Nueva Solicitud

Nro Solicitud	Motivo	Detalles	Estado	Fecha Registro	Fecha Finalización	Imagen
1	Requerimiento	utp	Pendiente	26-05-2025 18:48:53	-	 No disponible

© 2025 123digit@L. Todos los derechos reservados.

123 milhas

[Inicio](#) [Mis Solicitudes](#) [Mi Perfil](#) [Contacto](#)

Bienvenido, Verenise | [Cerrar Sesión](#)

Foto de perfil

Nombre de la Entidad: UTP

Nombre Completo: Verenise

Tipo de Documento: DNI

Número de Documento: 74598860

Correo Electrónico: Verenise@utp.com

Teléfono: 965823705

Dirección: sjm

Actualizar Datos

© 2025 123digit@L. Todos los derechos reservados.

Crear Solicitud

Motivo:

Seleccione un motivo

Detalles:

Subir imagen (opcional):

Seleccionar archivo

Sin archivos seleccionados

Registrar Solicitud