

PRESENTACION DE PÁGINA WEB CONECTADA A UNA API - REST PARA LA EMPRESA INTRIAL



INTRODUCCIÓN

INTRIAL es una empresa especializada en suministrar instrumentos de medición, accesorios, partes y equipos industriales para controlar variables físicas y fluidos. Sus áreas de especialización incluyen automatización industrial, electrónica, mecánica hidráulica y seguridad industrial. INTRIAL ha desarrollado soluciones para diferentes sectores industriales y mineros, destacándose por su experiencia en aplicaciones complejas. Forma parte del grupo internacional INTRIAL y cuenta con el respaldo de inversionistas alemanes para su desarrollo futuro.



PROBLEMATICA

- El proceso actual de consulta de productos es vía telefónica siendo complejo y lento, lo cual dificulta una atención al cliente.
- El proceso de revisión de registros de pagos y facturas es prolongado y manual.
- El proceso de verificación de productos es demasiado lento y manual, lo cual retrasa la entrega de productos a los clientes y dificulta la gestión eficiente del ciclo de ventas.
- Las consultas suelen ser confusas para el cliente ya que no se da la suficiente información.



OBJETIVO GENERAL

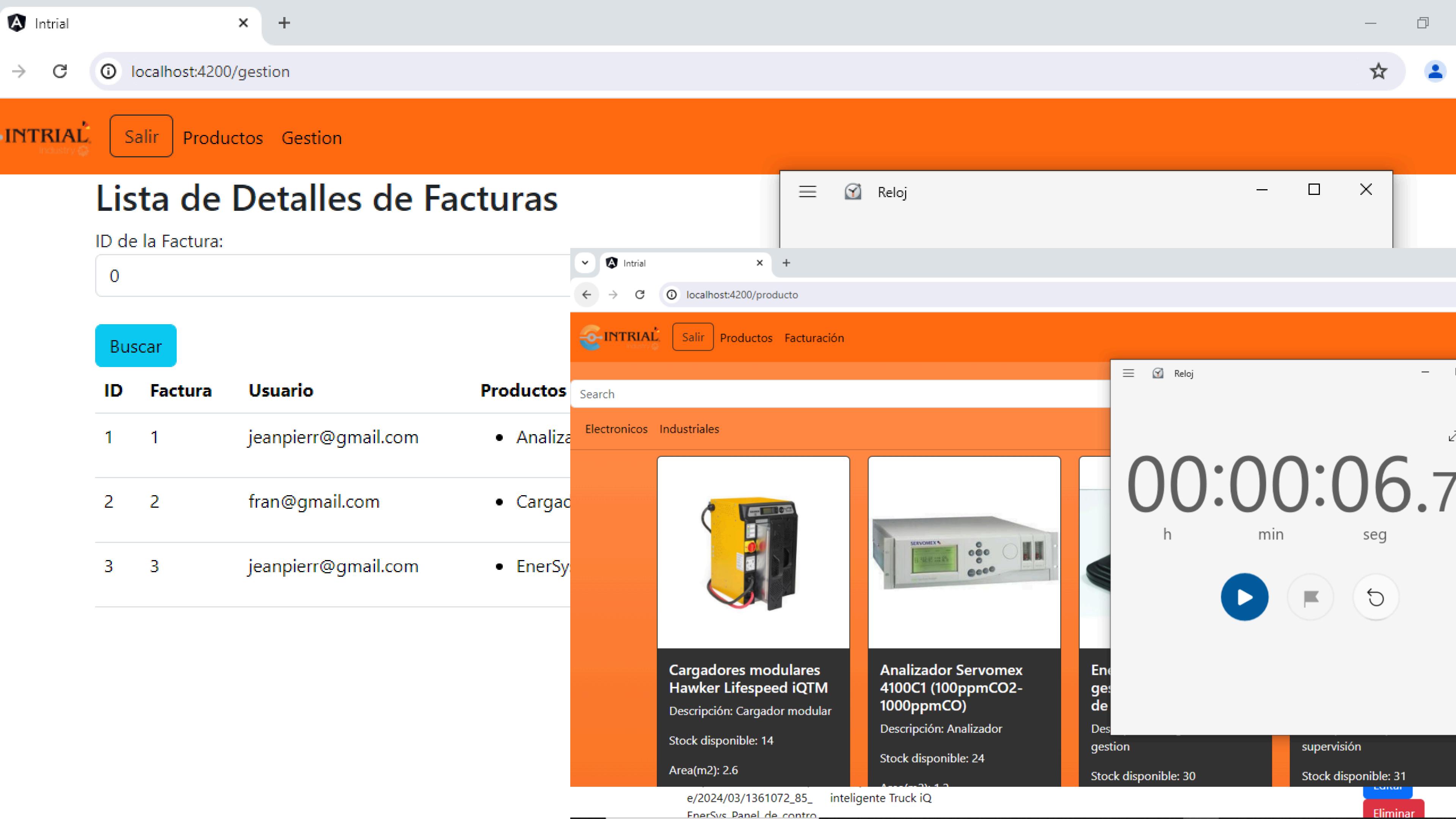
Modernizar el manejo de la información de la empresa usando una página web conectada a una API - REST que brinde información tanto al personal como a los clientes renovando la relación con ellos aumentando la cantidad de clientes mensuales en un 40%.



OBJETIVOS ESPECÍFICOS

- Reducir un 80% el tiempo en la consulta de productos para una mejora atención al cliente.
- Disminuir el tiempo que se tarda en revisar los registros en un 90% generando un registro de los pagos y facturas. (se corroboró mediante uso de cronometro que el tiempo era de 3 min y se busca conseguir un tiempo de 30 seg. aprox).
- Aumentar la velocidad en la verificación de productos en un 87% (se comprobó mediante uso de cronometro que el tiempo era de 5 min y se busca conseguir un tiempo de 65 seg aprox).
- Ofrecer información a través de la página web de los productos que tiene la empresa, cantidad, precio, etc. organizando por categoría para que al cliente se le facilite la búsqueda, mejorar el tiempo de consulta de clientes promedio de 10 - 15 minutos a 1 - 3 minutos cronometrados.





A Intrial

localhost:4200/productoad

INTRIAL

Salir Productos G

Lista de Pro

ID	Imagen
1	https://web.intrial.com.e/2024/03/1361065.Cargadores_modulaHawker_Lifespeed_iC.jpg
2	https://web.intrial.com.e/2017/09/149825_-Servomex.jpg
3	https://web.intrial.com.e/2024/03/1361074.EnerSy.jpg
4	https://web.intrial.com.p.e/2024/03/1361073_82_EnerSys_Dispos.jpg
5	https://web.intrial.com.p.e/2024/03/1361072_85_EnerSys_Panel_de_c

localhost:4200/producto

INTRIAL

Salir Productos Facturación

Search

Electronics Industrials

Cargadores modulares Hawker Lifespeed iQTM
Descripción: Cargador modular
Stock disponible: 14
Area(m2): 2.6

Análizador Servomex 4100C1 (100ppmCO2-1000ppmCO)
Descripción: Analizador
Stock disponible: 24
Area(m2): 1.2

Energía de Gestión de Desarrollo
supervisión
Stock disponible: 30

Stock disponible: 31

Reloj

00:00:06.70

h min seg

Search

9

0.25 2 234 46 1

Editar

Eliminar

Intrial

localhost:4200/producto

STARIAL Industry

Salir Productos Facturación

onicos Industriales



Cargadores modulares Hawker Lifespeed iQTM

Descripción: Cargador modular

Stock disponible: 14

Area(m2): 2.6



Analizador Servomex 4100C1 (100ppmCO2-1000ppmCO)

Descripción: Analizador

Stock disponible: 24

Reloj

00:00:06.70

h min seg

Play button

Stop button

Reset button

Ene
ges
de

Des
gestion

Stock disponible: 30

supervisión

Stock disponible: 31

OBJETIVOS ESPECÍFICOS

- Se comprobó que el tiempo actualmente se realiza más rápido que antes usando
- Disminuir el tiempo que se tarda en revisar los registros en un 90% (de 3 min a 30 seg), generando un registro de los pagos y facturas de la empresa.
- Aumentar la velocidad en la verificación de productos en un 87% (de 5 min a 65 seg).
- Ofrecer información a través de la página web de los productos que tiene la empresa, cantidad, precio, etc. organizando por categoría para que al cliente se le facilite la búsqueda, mejorar el tiempo de consulta de clientes promedio de 10 - 15 minutos a 1 - 3 minutos.



PROPUESTA DE SOLUCIÓN

- El proceso actual de consulta de productos es por la pagina web.
- El proceso de revisión de registros de pagos y facturas se hará mediante el controller y services de la api.
- El proceso de verificación de productos es mas rápida gracias a verificación con la base de datos
- Las consultas del cliente son mas eficiente ya que se tiene una información detallada por producto.



Filter objects

Tables

- ▶ categoria
- ▶ detalle_factura
- ▶ detalle_factura_producto
- ▶ detalle_usuario
- ▶ factura
- ▶ pago
- ▶ producto
- ▶ rol
- ▶ tipo_usuario
- ▶ usuario
- ▶ usuario_rol

Views

Stored Procedures

Functions

DESCRIPCION DE LA BASE DE DATOS

- **Tabla: categoría:** Almacenar las categorías a las que se pueden clasificar los productos.
- **Tabla: detalle_factura:** Registrar los detalles de cada producto incluido en una factura.
- **Tabla: detalle_factura:** Almacena id factura, id producto
- **Tabla: detalle_usuario:** Almacenar informacion del usuario
- **Tabla: factura:** Registrar las ventas realizadas y los datos asociados a cada una.
- **Tabla: pago:** Registrar los pagos realizados por los clientes para las facturas.
- **Tabla: producto:** Almacenar información sobre los productos que se ofrecen a la venta.
- **Tabla: tipo_usuario:** Divide los usuarios en tipos con diferentes características.
- **Tabla: usuario:** Almacenar los datos de los usuarios que tienen acceso al sistema.

SOLUCIÓN API - REST

ApiError.java: utilizado para manejar errores en una API REST.

Atributos: timestamp, status, message, errors.

Categoría.java: Define las categorías de productos en el sistema. Atributos: id, nombre.

Relaciones: Many-to-One con Producto: Un producto pertenece a una única categoría.

Factura.java: Representa una factura del sistema.

Atributos: id, fecha, usuario_id.

Relaciones: Many-to-Many con Producto: Un producto puede estar presente en varias facturas, y una factura puede incluir varios productos (relación gestionada mediante tabla intermedia detalle_factura).

Pago.java: Representa un pago asociado a una factura.

Atributos: id, factura_id.

Relaciones: One-to-One con Factura: Un pago se relaciona con una única factura.

com.application.intrial.web.mode
ApiError.java
Categoría.java
Detalle_factura.java
Detalle_usuario.java
Factura.java
Pago.java
PerfilUsuario.java
Producto.java
Rol.java
Tipo_usuario.java
Usuario.java
UsuarioDto.java

SOLUCIÓN API - REST

PerfilUsuario.java: Almacena la información detallada del perfil de un usuario.

Atributos: id, nombre, dni_ruc, direccion, telefono.

Relaciones: One-to-One con Usuario: Un usuario solo tiene un perfil asociado (relación inversa desde Usuario).

Producto.java: Representa un producto en el sistema.

Atributos: id, nombre, descripcion, precio, categoria_id.

Relaciones:

Many-to-One con Categoría: Un producto pertenece a una única categoría.

Many-to-Many con Factura: Un producto puede estar presente en varias facturas, y una factura puede incluir varios productos (relación gestionada mediante tabla intermedia detalle_factura).

com.application.intrial.web.mode
ApiError.java
Categoria.java
Detalle_factura.java
Detalle_usuario.java
Factura.java
Pago.java
PerfilUsuario.java
Producto.java
Rol.java
Tipo_usuario.java
Usuario.java
UsuarioDto.java

SOLUCIÓN API - REST

Tipo_usuario.java: Tipo_usuario: Define los diferentes tipos de usuarios que existen en el sistema.

Atributos: id, nombre.

Relaciones:

One-to-One con Usuario: Un tipo de usuario solo está asociado a un usuario (relación inversa desde Usuario).

Usuario.java: Representa la entidad principal para gestionar usuarios del sistema.

Atributos: id, nombre, email, password, tipo_usuario_id.

Relaciones:

One-to-One con Tipo_usuario: Un usuario tiene un tipo asociado que define su rol (administrador, cliente, etc.).

Many-to-Many con Rol: Un usuario puede tener varios roles asignados, y un rol puede estar asignado a varios usuarios.

One-to-One con PerfilUsuario: Un usuario solo tiene un perfil asociado que almacena su información detallada.

com.application.intrial.web.mode
ApiError.java
Categoria.java
Detalle_factura.java
Detalle_usuario.java
Factura.java
Pago.java
PerfilUsuario.java
Producto.java
Rol.java
Tipo_usuario.java
Usuario.java
UsuarioDto.java

SERVICES

```
IUsuarioService.java X
1 package com.application.intrial.web.service;
2
3+import java.util.Optional;
11
12 public interface IUsuarioService{
13
14     Optional<Usuario> buscarPorCorreo(String nombre);
15
16     UsuarioDto buscarPorId(Integer id);
17
18     Usuario buscarPorIdUsuario(Integer id);
19
20     Page<Usuario> listarUsuario(String nombre, Pageable pageable);
21
22     PaginationMod<UsuarioDto> listarProductoDtoPaginado(String nombre, Pageable pageable);
23
24     Usuario guardar(Usuario usuario);
25
26     void eliminar(Integer id);
```

UserService.java X

```
1 package com.application.intrial.web.service;
2
3+ import java.util.Arrays;
16
17 @Service
18 public class UserService implements IUsuarioService{
19
20@    @Autowired
21    private ModelMapper mapper;
22
23@    @Autowired
24    private IUsuarioDao usuarioDao;
25
26@    @Override
△27    public Optional<Usuario> buscarPorCorreo(String nombre) {
28        return usuarioDao.findByCorreo(nombre);
29    }
30
31@    @Override
△32    public Page<Usuario> listarUsuario(String nombre, Pageable pageable) {
33        return usuarioDao.findByCorreoContaining(nombre, pageable);
34    }
35
36
37@    @Override
△38    public UsuarioDto buscarPorId(Integer id) {
39        Usuario usu = usuarioDao.findById(id).orElse(null);
40
41        UsuarioDto dto = mapper.map(usu, UsuarioDto.class);
42        return dto;
```

UserService.java ×

```
41     UsuarioDto dto = mapper.map(usu, UsuarioDto.class);
42     return dto;
43 }
44
45@Override
46 public PaginationMod<UsuarioDto> listarProductoDtoPaginado(String nombre, Pageable pageable) {
47     Page<Usuario> paginacion = usuarioDao.findByCorreoContaining(nombre, pageable);
48
49     PaginationMod<UsuarioDto> paginationMod = new PaginationMod<UsuarioDto>();
50     UsuarioDto[] entityDtos = mapper.map(paginacion.getContent(), UsuarioDto[].class);
51     paginationMod.setValue(paginacion, Arrays.asList(entityDtos));
52     return paginationMod;
53 }
54
55@Override
56 public Usuario guardar(Usuario usuario) {
57     return usuarioDao.save(usuario);
58 }
59
60@Override
61 public Usuario buscarPorIdUsuario(Integer id) {
62     return usuarioDao.findById(id).orElse(null);
63 }
64
65@Override
66 public void eliminar(Integer id) {
67     usuarioDao.deleteById(id);
68 }
69 }
70 }
```

CONTROLLER

```
1 package com.application.intrial.web.controller;
2
3+import java.util.Optional;■
22
23 @RestController
24 @RequestMapping("api/intrial")
25 public class AccesoController {
26
27@  @Autowired
28  private AuthenticationManager authManager;
29
30@  @Autowired
31  private JwtTokenUtil jwtUtil;
32
33@  @Autowired
34  private IUsuarioService usuarioService;
35
36@  @PostMapping("/login")
37  public ResponseEntity<?> login(@RequestBody AuthRequest request)
38  {
39      try {
40
41          Authentication authentication = this.authManager.authenticate(new UsernamePasswordAuthenticationToken(request.getCorre
42          System.out.println(authentication);
43
44          Optional<Usuario> user = usuarioService.buscarPorCorreo(request.getCorreo());
45
46          String accessToken = jwtUtil.generarToken(user.get());
47
48          AuthResponse response = new AuthResponse(request.getCorreo(), accessToken);
49          return ResponseEntity.ok(response);
50
51      } catch (BadCredentialsException e) {
52
53          return ResponseEntity.status(HttpStatus.UNAUTHORIZED).build();
54      }
55  }
56 }
```

CONTROLLER

```
ApiUsuarios.java X
1 package com.application.intrial.web.controller;
2
3 import org.springframework.beans.factory.annotation.Autowired;[]
4
5 @RestController
6 @RequestMapping("/api/usuarios")
7 public class ApiUsuarios {
8
9     @Autowired
10    private IUsuarioService usuarioService;
11
12    @Autowired
13    private PasswordEncoder passwordEncoder;
14
15    @GetMapping("/listar")
16    public PaginationMod<UsuarioDto> usuarios(@RequestParam(required = false, defaultValue = "0") Integer page, @RequestParam(required = false, defaultValue = "0") Integer size) {
17
18        return usuarioService.listarProductoDtoPaginado(nombre, PageRequest.of(page, 5, Sort.by("id").descending()));
19    }
20
21    @GetMapping("/listar/{id}")
22    public UsuarioDto usuariosId(@PathVariable Integer id) {
23        return usuarioService.buscarPorId(id);
24    }
25}
```

CONCLUSIONES

Intrial S.A.C. puede mejorar su competitividad y diferenciarse en el mercado mediante una estrategia integral que combine la innovación tecnológica, la mejora del servicio al cliente y el desarrollo de productos y servicios de alta calidad. La implementación de una página web y una API REST, junto con la optimización de procesos internos, permitirá a la empresa brindar una mejor experiencia a sus clientes y aumentar su base de clientes.

REFERENCIAS

- Calderón, X. N. Y., Soledispa, R. A. V., & Poveda, M. L. P. (2021). Crecimiento empresarial: estrategia de desarrollo del mercado en el sector MIPYMES. *Revista Publicando*, 8(31), 82-95.
- Alva Fructuoso, A. M. (2020). Plan de negocios para la empresa B2B Soluciones Industriales SAC.

iGRACIAS!