Universidad San Carlos de Guatemala
Centro Universitario de Occidente
División de Ciencias de la Ingeniería
Manejo e implementación de archivos
Ing. Christian Quiroa

Byron Fernando Torres Ajxup 201731523

Manual Tecnico Game pro Xela

Sistema de Inventarios y sucursales

Quetzaltenango 05 de Septiembre de 2024

Indice

Quetzaltenango 26 de Septiembre de 2024

Introduccion

Este manual técnico está destinado a desarrolladores y administradores del sistema de gestión de sucursales. Proporciona detalles sobre la arquitectura, tecnologías utilizadas y estructura de la base de datos.

Requerimientos Técnicos

Backend: Node.js 16 con Express.
Base de Datos: PostgreSQL.

Frontend react

Arquitectura del Sistema

El sistema está diseñado en una arquitectura de cliente-servidor, donde el frontend se comunica con el backend a través de API REST. La base de datos almacena toda la información necesaria para el funcionamiento del sistema.

Tecnologias

Backend: Node.js y Express para el servidor.

Base de Datos: PostgreSQL para el almacenamiento de datos.

Frontend: React para la interfaz de usuario.

Seguridad: Bcrypt para el manejo de contraseñas y Express-session para la gestión de sesiones.

Estructura de la Base de Datos

```
- Definición de la tabla de productos

CREATE TABLE productos.producto (

   id SERIAL PRIMARY KEY,

   nombre VARCHAR(100) NOT NULL,

   descripcion TEXT,

   precio NUMERIC(10,2) NOT NULL

);


-- Definición de la tabla de sucursales

CREATE TABLE sucursales.sucursal (

   id_sucursal SERIAL PRIMARY KEY,

   nombre VARCHAR(100) NOT NULL,
```

```
    direccion VARCHAR(200) NOT NULL
);


-- Definición de la tabla de empleados
CREATE TABLE sucursales.empleado (
    id_empleado SERIAL PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    apellido VARCHAR(100) NOT NULL,
    username VARCHAR(50) NOT NULL,
    password VARCHAR(255) NOT NULL,
    rol VARCHAR(20),
    numero_caja INTEGER,
    sucursal_id INTEGER NOT NULL REFERENCES
sucursales.sucursal(id_sucursal),
    CONSTRAINT empleado_check CHECK (rol IN ('CAJERO', 'BODEGA',
'INVENTARIO', 'ADMINISTRADOR')),
    CONSTRAINT empleado_rol_check CHECK ((rol = 'CAJERO' AND numero_caja IS
NOT NULL) OR (rol != 'CAJERO'))
);


-- Definición de la tabla de bodegas
CREATE TABLE sucursales.bodega (
    id SERIAL PRIMARY KEY,
    nombre VARCHAR(100),
    sucursal_id INTEGER NOT NULL REFERENCES
sucursales.sucursal(id_sucursal)
);
```

```sql
-- Definición de la tabla de cajas
CREATE TABLE sucursales.caja (
   id SERIAL PRIMARY KEY,
   numero INTEGER NOT NULL,
   sucursal_id INTEGER NOT NULL REFERENCES
sucursales.sucursal(id_sucursal),
   cajero_id INTEGER REFERENCES sucursales.empleado(id_empleado)
);

-- Definición de la tabla de ingreso a bodega
CREATE TABLE sucursales.ingreso_bodega (
   id SERIAL PRIMARY KEY,
   sucursal_id INTEGER NOT NULL REFERENCES
sucursales.sucursal(id_sucursal),
   producto_id INTEGER NOT NULL REFERENCES productos.producto(id),
   empleado_id INTEGER NOT NULL REFERENCES
sucursales.empleado(id_empleado),
   cantidad INTEGER NOT NULL,
   bodega_id INTEGER NOT NULL REFERENCES sucursales.bodega(id),
   fecha TIMESTAMP WITHOUT TIME ZONE DEFAULT now()
);

-- Definición de la tabla de ingreso a pasillo
CREATE TABLE sucursales.ingreso_pasillo (
   id SERIAL PRIMARY KEY,
   sucursal_id INTEGER NOT NULL REFERENCES
sucursales.sucursal(id_sucursal),
   producto_id INTEGER NOT NULL REFERENCES productos.producto(id),
```

```
    cantidad INTEGER NOT NULL,
    pasillo INTEGER NOT NULL,
    empleado INTEGER NOT NULL REFERENCES sucursales.empleado(id_empleado),
    fecha TIMESTAMP WITHOUT TIME ZONE DEFAULT now()
);

-- Definición de la tabla de inventario en bodega
CREATE TABLE sucursales.inventario_bodega (
    id SERIAL PRIMARY KEY,
    producto_id INTEGER NOT NULL REFERENCES productos.producto(id),
    bodega_id INTEGER NOT NULL REFERENCES sucursales.bodega(id),
    cantidad_disponible INTEGER DEFAULT 0 NOT NULL
);

-- Definición de la tabla de inventario en stock
CREATE TABLE sucursales.inventario_stock (
    id SERIAL PRIMARY KEY,
    producto_id INTEGER NOT NULL REFERENCES productos.producto(id),
    pasillo INTEGER NOT NULL,
    cantidad_disponible INTEGER DEFAULT 0 NOT NULL
);

-- Definición de la tabla de clientes
CREATE TABLE ventas.cliente (
    id SERIAL PRIMARY KEY,
    nit VARCHAR(20),
    nombre VARCHAR(100) NOT NULL,
```

```
    apellido VARCHAR(100) NOT NULL,
    total_gastado NUMERIC(10,2) DEFAULT 0,
    puntos INTEGER DEFAULT 0,
    tipo_tarjeta VARCHAR(10),
    direccion VARCHAR(200),
    telefono VARCHAR(15)
);



-- Definición de la tabla de solicitud de tarjetas
CREATE TABLE ventas.solicitud_tarjeta (
    id SERIAL PRIMARY KEY,
    nit VARCHAR(20),
    tipo_tarjeta VARCHAR(10),
    aprobado BOOLEAN DEFAULT false,
    CONSTRAINT solicitud_tarjeta_tipo_tarjeta_check CHECK (tipo_tarjeta IN
('COMUN', 'ORO', 'PLATINO', 'DIAMANTE'))
);



-- Definición de la tabla de ventas
CREATE TABLE ventas.venta (
    id SERIAL PRIMARY KEY,
    numero_factura VARCHAR(50) NOT NULL,
    fecha TIMESTAMP WITHOUT TIME ZONE DEFAULT
random_date('2024-01-01'::DATE, '2024-12-31'::DATE),
```

```
    nit VARCHAR(20),
    total_sin_descuento NUMERIC(10,2),
    total_con_descuento NUMERIC(10,2),
    empleado_id INTEGER REFERENCES sucursales.empleado(id_empleado),
    sucursal_id INTEGER REFERENCES sucursales.sucursal(id_sucursal)
);


-- Definición de la tabla de detalle de venta
CREATE TABLE ventas.detalle_venta (
    id SERIAL PRIMARY KEY,
    venta_id INTEGER NOT NULL REFERENCES ventas.venta(id),
    producto_id INTEGER NOT NULL REFERENCES productos.producto(id),
    cantidad INTEGER NOT NULL,
    precio_unitario NUMERIC(10,2) NOT NULL,
    fecha TIMESTAMP WITHOUT TIME ZONE DEFAULT
random_date('2024-01-01'::DATE, '2024-12-31'::DATE),
    gastado INTEGER NOT NULL
);


CREATE TABLE ventas.historial_descuentos (
    id SERIAL PRIMARY KEY,
    venta_id INTEGER NOT NULL REFERENCES ventas.venta(id),
    descuento INTEGER NOT NULL,
    fecha TIMESTAMP WITHOUT TIME ZONE DEFAULT
random_date('2024-01-01'::DATE, '2024-12-31'::DATE)
);
```

API Endpoints

`POST /api/login`: Iniciar sesión de usuario.

`POST /api/ventas`: Registrar una nueva venta.

`POST /api/clientes`: Crear un nuevo cliente (cajero).

`PUT /api/clientes/:id`: Modificar un cliente (cajero, requiere aprobación).

`POST /api/inventario/mover`: Mover productos (inventario).

`POST /api/reportes`: Generar reportes (administrador).

Rutas Protegidas

Las rutas que requieren autenticación están protegidas mediante middleware. A continuación se muestra un ejemplo de middleware para verificar la autenticación:

javascript

Copiar código

```
const isAuthenticated = (req, res, next) => {
    if (req.session.user) {
        return next();
    }
    res.status(401).send("Acceso no autorizado");
};
```

COnclusiones

El sistema está diseñado para ser escalable y seguro, facilitando la gestión de sucursales y permitiendo la administración de diferentes roles. La implementación de

tecnologías modernas asegura un rendimiento óptimo y una buena experiencia de usuario.