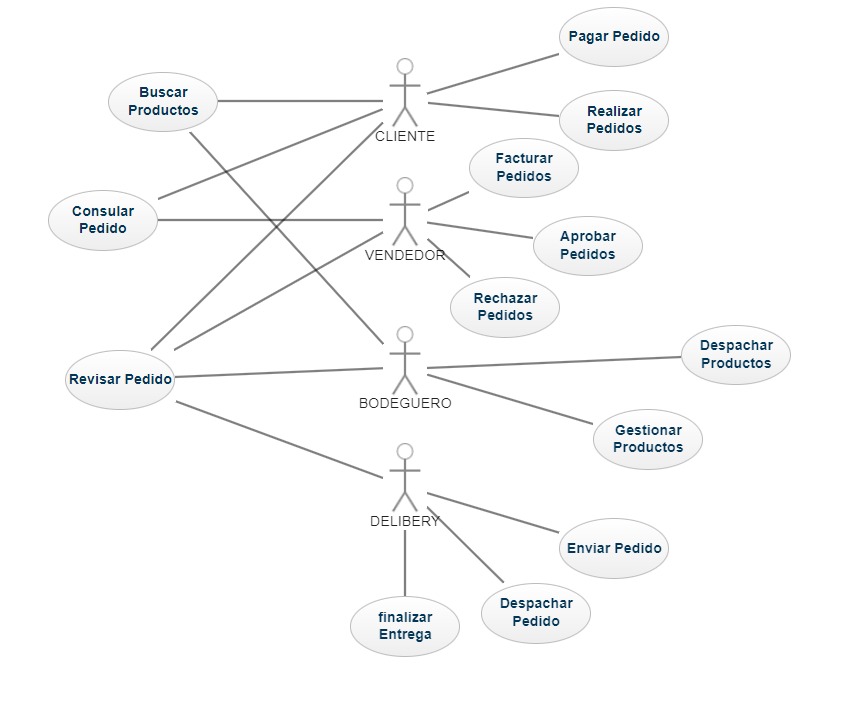
Informe

Tienda online

**Nombre: David Toro**

**Fecha:26/2/2024**

**Caso de usos:**



|  |  |
| --- | --- |
| D | LUX VENDEDOR 1 |
| NOMBRE | FACTURAR PEDIDO |
| DEPENDENCIA | NINGUNA |
| ACTOR | VENDEDOR |
| PRECONDICIÓN | EL VENDEDOR RECIBE UN PEDIDO Y DESEA FACTURARLO |
| DESCRIPCIÓN | EL VENDEDOR GENERA UNA FACTURA PARA UN PEDIDO DE HARDWARE |
| FLUJO NORMAL | 1. EL USUARIO ABRE EL SISTEMA |
| 2. EL USUARIO PRESIONA EL BOTÓN "PEDIDOS PENDIENTES" |
| 3. EL USUARIO BUSCA EL PEDIDO QUE DESEA FACTURAR |
| 4. EL USUARIO PRESIONA EL BOTÓN "GENERAR FACTURA" PARA EL PEDIDO SELECCIONADO |
| 5. EL USUARIO ENVÍA LA FACTURA AL CLIENTE |
| POST CONDICIONES | EL VENDEDOR FACTURA CON ÉXITO EL PEDIDO |
| EXCEPCIÓN | NA |
| COMENTARIOS | LA FACTURA SE GENERA AUTOMÁTICAMENTE EN FORMATO PDF |

|  |  |
| --- | --- |
| ID | LUX VENDEDOR 2 |
| NOMBRE | RECHAZAR PEDIDO |
| DEPENDENCIA | NINGUNA |
| ACTOR | VENDEDOR |
| PRECONDICIÓN | EL VENDEDOR RECIBE UN PEDIDO QUE DESEA RECHAZAR |
| DESCRIPCIÓN | EL VENDEDOR RECHAZA UN PEDIDO DE HARDWARE RECIBIDO |
| FLUJO NORMAL | 1. EL USUARIO PRESIONA EL BOTÓN "PEDIDOS PENDIENTES" |
| 2. EL USUARIO BUSCA EL PEDIDO QUE DESEA RECHAZAR |
| 3. EL USUARIO PRESIONA EL BOTÓN "RECHAZAR PEDIDO" |
| 4. EL USUARIO INGRESA LA RAZÓN DEL RECHAZO |
| 5. EL USUARIO CONFIRMA EL RECHAZO DEL PEDIDO |
| POST CONDICIONES | EL PEDIDO SE RECHAZA EXITOSAMENTE |
| EXCEPCIÓN | NA |
| COMENTARIOS | SE NOTIFICA AL CLIENTE SOBRE EL RECHAZO DEL PEDIDO |

|  |  |
| --- | --- |
| ID | LUX VENDEDOR 3 |
| NOMBRE | APROBAR PEDIDO |
| DEPENDENCIA | NINGUNA |
| ACTOR | VENDEDOR |
| PRECONDICIÓN | EL VENDEDOR RECIBE UN PEDIDO QUE DESEA APROBAR |
| DESCRIPCIÓN | EL VENDEDOR APRUEBA UN PEDIDO DE HARDWARE RECIBIDO |
| FLUJO NORMAL | 1. EL USUARIO PRESIONA EL BOTÓN "PEDIDOS PENDIENTES" |
| 2. EL USUARIO BUSCA EL PEDIDO QUE DESEA APROBAR |
| 3. EL USUARIO PRESIONA EL BOTÓN "APROBAR PEDIDO" |
| 4. EL USUARIO CONFIRMA LA APROBACIÓN DEL PEDIDO |
| EL PEDIDO SE APRUEBA EXITOSAMENTE |
| EXCEPCIÓN | NA |
| COMENTARIOS | SE NOTIFICA AL CLIENTE SOBRE LA APROBACIÓN DEL PEDIDO |

|  |  |
| --- | --- |
| ID | LUX DELIVERY 1 |
| NOMBRE | ENVIAR PRODUCTO |
| DEPENDENCIA | DESPACHAR PRODUCTO |
| ACTOR | DELIVERY |
| PRECONDICIÓN | EL BODEGERO HA DESPACHADO EL PRODUCTO |
| DESCRIPCIÓN | EL DELIVERY RECIBE EL PEDIDO DESPACHADO |
| FLUJO NORMAL | 2. EL DELIVERY PREPARA Y GESTIONA LA ENTREGA DEL PEDIDO |
| POST CONDICIONES | EL ESTADO DEL PEDIDO SE ACTUALIZA A "EN ENVIO" |
| EXCEPCIÓN | NA |
| COMENTARIOS | EL PEDIDO ESTA EN PROCESO DE ENVIO |

|  |  |
| --- | --- |
| ID | LUX DELIVERY 2 |
| NOMBRE | REVISAR PEDIDO |
| DEPENDENCIA | DESPACHAR PRODUCTO |
| ACTOR | DELIVERY |
| PRECONDICIÓN | EL DELIVERY RECIBE EL PEDIDO EMPAQUETADO |
| DESCRIPCIÓN | EL DELIVERY ABRE Y REVISA EL PEDIDO DESPACHADO |
| FLUJO NORMAL | 2. EL DELIVERY INSPECCIONA QUE EL PEDIDO ESTE COMPLETO |
|  | 3. EL DELIVERY BUEVAMENTE ETIQUETA Y EMPAQUETA EL PEDIDO |
| POST CONDICIONES | EL ESTADO DEL PEDIDO SE ACTUALIZA A "EN REVISION" |
| EXCEPCIÓN | NA |
| COMENTARIOS | EL PEDIDO ESTA EN ETAPA DE REVISION Y APROBACIÓN |

|  |  |
| --- | --- |
| ID | LUX DELIVERY 3 |
| NOMBRE | FINALIZAR ENTREGA |
| DEPENDENCIA | REVISAR PEDIDO |
| ACTOR | DELIVERY |
| PRECONDICIÓN | EL DELIVERY REVISA Y EMPAQUETA NUEVAMENTE EL PEDIDO |
| DESCRIPCIÓN | EL DELIVERY REVISA LA ENTREGA DEL PEDIDO |
| FLUJO NORMAL | 2. EL DELIVERY ENVIA EL PEDIDO AL CLIENTE |
|  | 3. EL DELIVERY ENTREGA EL PEDIDO CON ÉXITO |
| POST CONDICIONES | EL ESTADO DEL PEDIDO SE ACTUALIZA A "ENTREGADO" |
| EXCEPCIÓN | NA |
| COMENTARIOS | EL PEDIDO ESTA EN ETAPA DE ENTREGA. |

|  |  |
| --- | --- |
| ID | LUX DELIVERY 3 |
| NOMBRE | DESPACHAR PEDIDO |
| DEPENDENCIA | DESPACHAR PRODUCTO |
| ACTOR | DELIVERY |
| PRECONDICIÓN | EL DELIVERY RECIBE EL PEDIDO EMPAQUETADO |
| DESCRIPCIÓN | EL DELIVERY DESPACHA EL PEDIDO EMPAQUETADO |
| FLUJO NORMAL | 2. EL DELIVERY VERIFICA EL ESTADO DEL PEDIDO |
|  | 3. EL DELIVERY ENVIA A REVISION EL PEDIDO |
| POST CONDICIONES | EL ESTADO DEL PEDIDO SE ACTUALIZA A "EN DESPACHO" |
| EXCEPCIÓN | NA |
| COMENTARIOS | EL PEDIDO ESTA EN ETAPA DE DESPACHO. |

|  |  |
| --- | --- |
| ID | LUX1 |
| NOMBRE | CONSULTAR PEDIDO |
| DEPENDENCIA | NINGUNA |
| ACTOR | CLIENTE |
| PRECONDICIÓN | EL CLIENTE HA REALIZADO UN PEDIDO DE HARDWARE DE COMPUTADORAS |
| DESCRIPCIÓN | EL CLIENTE VERIFICA EL ESTADO DE SU PEDIDO DE HARDWARE EN LA TIENDA ONLINE |
| FLUJO NORMAL | 1. EL CLIENTE INGRESA A SU CUENTA |
| 2.EL SISTEMA MUESTRA LAS OPCIONES DE LA APLICACIÓN |
| 3.EL CLIENTE SELECCIONA EL BOTÓN "PRODUCTOS" |
| 4.EL SISTEMA MUESTRA LOS DETALLES DE LOS PRODUCTOS DE HARDWARE DISPONIBLES |
| POST CONDICIONES | EL CLIENTE OBTIENE INFORMACIÓN SOBRE SU PEDIDO DE HARDWARE |
| EXCEPCIÓN | NA |
| COMENTARIOS | LA INFORMACIÓN DEL PEDIDO SERÁ ACTUALIZADA EN TIEMPO REAL |

|  |  |
| --- | --- |
| ID | LUX 2 |
| NOMBRE | BUSCAR PRODUCTO |
| DEPENDENCIA | NINGUNA |
| ACTOR | CLIENTE |
| PRECONDICIÓN | EL CLIENTE ESTÁ NAVEGANDO EN LA TIENDA ONLINE |
| DESCRIPCIÓN | EL CLIENTE BUSCA UN PRODUCTO DE HARDWARE EN LA TIENDA ONLINE |
| FLUJO NORMAL | 1. EL CLIENTE SELECCIONA EL BOTÓN “BUSCAR” |
| 2.EL CLIENTE INGRESA EL NOMBRE DEL PRODUCTO EN LA BARRA DE BÚSQUEDA |
| 3.EL SISTEMA MUESTRA LOS RESULTADOS DE LA BÚSQUEDA |
| EL CLIENTE OBTIENE INFORMACIÓN SOBRE EL PRODUCTO BUSCADO |
| EXCEPCIÓN | NA |
| COMENTARIOS | LA INFORMACIÓN DEL PRODUCTO SERÁ ACTUALIZADA EN TIEMPO REAL |

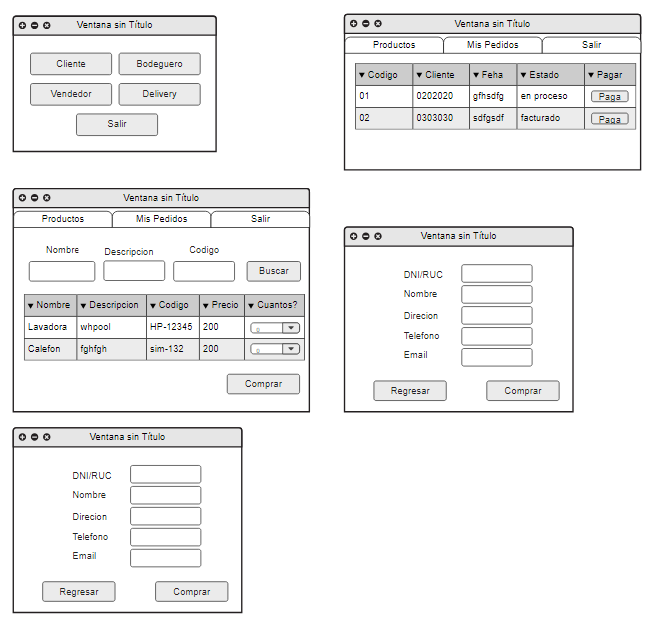
|  |  |
| --- | --- |
| ID | LUX 3 |
| NOMBRE | REALIZAR PEDIDO |
| DEPENDENCIA | NINGUNA |
| ACTOR | CLIENTE |
| PRECONDICIÓN | EL CLIENTE ESTÁ NAVEGANDO EN LA TIENDA ONLINE |
| DESCRIPCIÓN | EL CLIENTE REALIZA UN PEDIDO DE HARDWARE EN LA TIENDA ONLINE |
| FLUJO NORMAL | 1. EL CLIENTE SELECCIONA EL BOTÓN "REALIZAR PEDIDO" |
|  |
| 2. EL CLIENTE AGREGA PRODUCTOS AL CARRITO |
| 3. EL CLIENTE INGRESA LA DIRECCIÓN DE ENVÍO, CÉDULA O RUC, NOMBRES Y APELLIDOS, Y CORREO |
|  | 4. EL CLIENTE SELECCIONA EL MÉTODO DE PAGO |
|  | 5. EL CLIENTE CONFIRMA EL PEDIDO |
| POST CONDICIONES | EL CLIENTE REALIZA CON ÉXITO SU PEDIDO |
| EXCEPCIÓN | NA |
| COMENTARIOS | EL ESTADO DEL PEDIDO SERÁ ACTUALIZADO EN TIEMPO REAL |

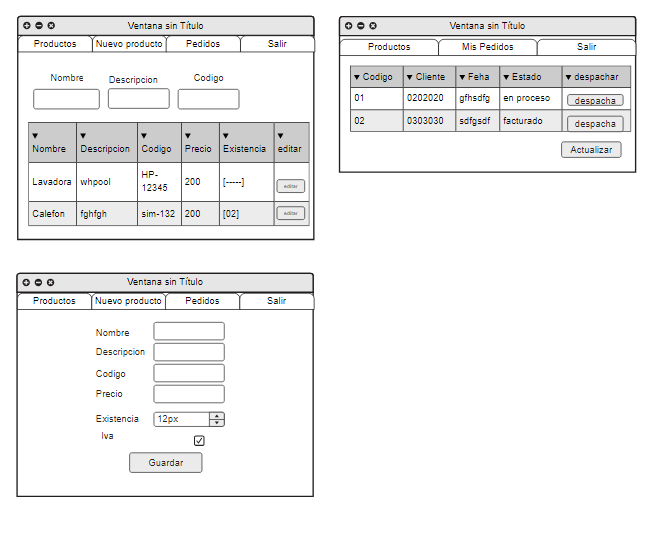
|  |  |
| --- | --- |
| ID | LUX 4 |
| NOMBRE | PAGAR PEDIDO |
| DEPENDENCIA | NINGUNA |
| ACTOR | CLIENTE |
| PRECONDICIÓN | EL CLIENTE HA REALIZADO UN PEDIDO Y DESEA PAGARLO |
| DESCRIPCIÓN | EL CLIENTE REALIZA EL PAGO DE SU PEDIDO DE HARDWARE EN LA TIENDA ONLINE |
| FLUJO NORMAL | 2. EL CLIENTE SELECCIONA "MIS PEDIDOS" |
| 3. EL CLIENTE SELECCIONA EL PEDIDO QUE DESEA PAGAR |
| 4. EL CLIENTE SELECCIONA LA OPCIÓN "PAGAR PEDIDO" |
| 5. EL CLIENTE INGRESA LOS DATOS DE PAGO (TARJETA DE CRÉDITO/DEBITO, ETC.) |
|  | 6. EL CLIENTE CONFIRMA EL PAGO |
| POST CONDICIONES | EL CLIENTE REALIZA CON ÉXITO EL PAGO DE SU PEDIDO |
| EXCEPCIÓN | NA |
| COMENTARIOS | EL ESTADO DEL PEDIDO SERÁ ACTUALIZADO EN TIEMPO REAL |

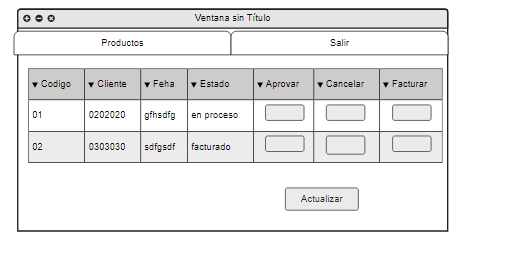
|  |  |
| --- | --- |
| ID | LUX BODEGERO 1 |
| NOMBRE | GESTIONAR PRODUCTOS |
| DEPENDENCIA | FACTURAR PEDIDO |
| ACTOR | BODEGERO |
| PRECONDICIÓN | EL VENDEDOR HA FACTURADO EL PEDIDO |
| DESCRIPCIÓN | EL BODEGERO REALIZA LA GESTION DEL/LOS PRODUCTO/S DEL PEDIDO EN LA BODEGA FISICA DE LA TIENDA |
| FLUJO NORMAL | 2. EL BODEGERO AÑADE LOS PRODUCTOS EN UN EMPAQUE |
| 4. EL BODEGERO ETIQUETA LOS/EL EMPAQUE/S |
| POST CONDICIONES | EL BODEGERO PREPARA Y SELLA LOS EMPAQUES. |
| EXCEPCIÓN | NA |
| COMENTARIOS | EL ESTADO DEL PEDIDO SE ACTUALIZA A "EN PROCESO DE ENVIO" |

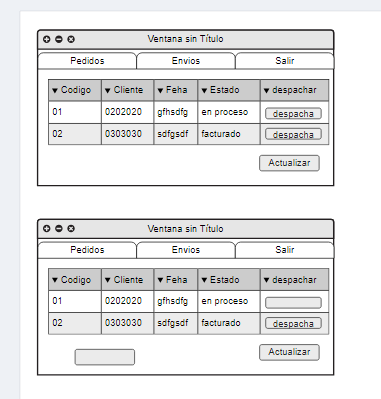
|  |  |
| --- | --- |
| ID | LUX BODEGERO 2 |
| NOMBRE | DESPACHAR PRODUCTOS |
| DEPENDENCIA | GESTIONAR PRODUCTOS |
| ACTOR | BODEGERO |
| PRECONDICIÓN | EL BODEGERO HA EMPAQUETADO EL PEDIDO |
| DESCRIPCIÓN | EL BODEGERO DESPACHA EL PEDIDO |
| FLUJO NORMAL | 2. EL BODEGERO ENTREGA EL PEDIDO AL DELIVERY |
| EL BODEGERO ENTREGA CON ÉXITO EL PEDIDO AL DELIVERY |
| EXCEPCIÓN | NA |
| COMENTARIOS | EL ESTADO DEL PEDIDO SIGUE EN "EN PROCESO DE ENVIO" |

**Mockups:**

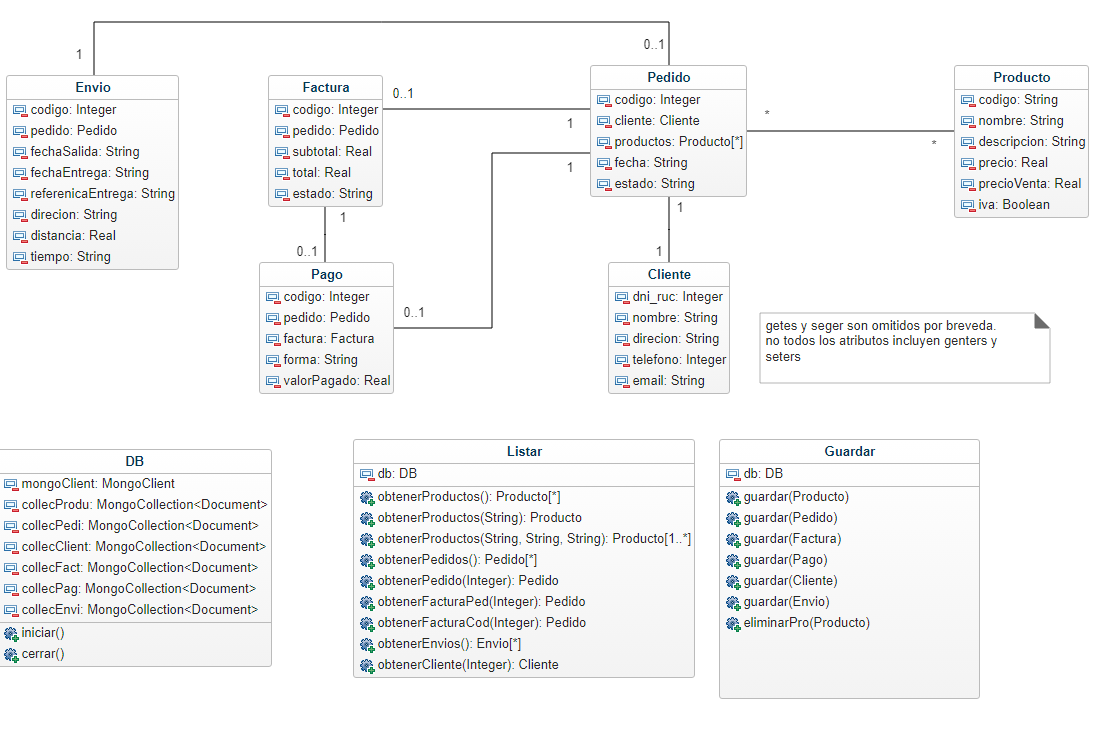
****

****

****

****

**Diagrama de clases:**

****

**Backend**

**Producto**

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template

\*/

package productos;

/\*\*

\*

\* @author david

\*/

public class Producto {

private String codigo;

private String nombre;

private double precio;

private double precioVenta;

private boolean iva;

private String descripcion;

public String getDescripcion() {

return descripcion;

}

public void setDescripcion(String descripcion) {

this.descripcion = descripcion;

}

public String getCodigo() {

return codigo;

}

public void setCodigo(String codigo) {

this.codigo = codigo;

}

public String getNombre() {

return nombre;

}

public void setNombre(String nombre) {

this.nombre = nombre;

}

public double getPrecio() {

return precio;

}

public void setPrecio(double precio) {

this.precio = precio;

this.precioVenta=precio\*1.15;

}

public double getPrecioVenta() {

return precioVenta;

}

public boolean getIva() {

return iva;

}

public void setIva(boolean iva) {

this.iva = iva;

}

}

**Pedido**

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template

\*/

package pedidos;

import productos.Producto;

/\*\*

\*

\* @author david

\*/

public class Pedido {

private int codigo;

public int getCodigo() {

return codigo;

}

public void setCodigo(int codigo) {

this.codigo = codigo;

}

private Cliente cliente;

public Cliente getCliente() {

return cliente;

}

public void setCliente(Cliente cliente) {

this.cliente = cliente;

}

private Producto[] productos;

public Producto[] getProductos() {

return productos;

}

public void setProductos(Producto[] productos) {

this.productos = productos;

}

private String fecha;

public String getFecha() {

return fecha;

}

public void setFecha(String fecha) {

this.fecha = fecha;

}

private String estado;

public String getEstado() {

return estado;

}

public void setEstado(String estado) {

this.estado = estado;

}

public Pedido() {

}

}

**Cliente**

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template

\*/

package pedidos;

/\*\*

\*

\* @author david

\*/

public class Cliente {

private int dni\_ruc;

public int getDni\_ruc() {

return dni\_ruc;

}

public void setDni\_ruc(int dni\_ruc) {

this.dni\_ruc = dni\_ruc;

}

private String nombre;

public String getNombre() {

return nombre;

}

public void setNombre(String nombre) {

this.nombre = nombre;

}

private String direcion;

public String getDirecion() {

return direcion;

}

public void setDirecion(String direcion) {

this.direcion = direcion;

}

private int telefono;

public int getTelefono() {

return telefono;

}

public void setTelefono(int telefono) {

this.telefono = telefono;

}

private String email;

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

}

**Factura**

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template

\*/

package facturacion;

import pedidos.Pedido;

/\*\*

\*

\* @author david

\*/

public class Factura {

private String codigo;

public String getCodigo() {

return codigo;

}

public void setCodigo(String codigo) {

this.codigo = codigo;

}

private Pedido pedido;

public Pedido getPedido() {

return pedido;

}

public void setPedido(Pedido pedido) {

this.pedido = pedido;

}

private float subtotal;

public float getSubtotal() {

return subtotal;

}

public void setSubtotal(float subtotal) {

this.subtotal = subtotal;

}

private float total;

public float getTotal() {

return total;

}

public void setTotal(float total) {

this.total = total;

}

private String estado;

public String getEstado() {

return estado;

}

public void setEstado(String estado) {

this.estado = estado;

}

public Factura(Pedido pedido) {

this.pedido = pedido;

}

}

**Pago**

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template

\*/

package facturacion;

import pedidos.Pedido;

/\*\*

\*

\* @author david

\*/

public class Pago {

private int codigo;

private Pedido pedido;

private Factura factura;

private String forma;

private String valorPagado;

public int getCodigo() {

return codigo;

}

public void setCodigo(int codigo) {

this.codigo = codigo;

}

public Pedido getPedido() {

return pedido;

}

public void setPedido(Pedido pedido) {

this.pedido = pedido;

}

public Factura getFactura() {

return factura;

}

public void setFactura(Factura factura) {

this.factura = factura;

}

public String getForma() {

return forma;

}

public void setForma(String forma) {

this.forma = forma;

}

public String getValorPagado() {

return valorPagado;

}

public void setValorPagado(String valorPagado) {

this.valorPagado = valorPagado;

}

}

**Envio**

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template

\*/

package envios;

import pedidos.Pedido;

/\*\*

\*

\* @author david

\*/

public class Envio {

private int codigo;

public int getCodigo() {

return codigo;

}

public void setCodigo(int codigo) {

this.codigo = codigo;

}

private Pedido pedido;

public Pedido getPedido() {

return pedido;

}

public void setPedido(Pedido pedido) {

this.pedido = pedido;

}

private String fechaSalida;

public String getFechaSalida() {

return fechaSalida;

}

public void setFechaSalida(String fechaSalida) {

this.fechaSalida = fechaSalida;

}

private String fechaEntrega;

public String getFechaEntrega() {

return fechaEntrega;

}

public void setFechaEntrega(String fechaEntrega) {

this.fechaEntrega = fechaEntrega;

}

private String direcion;

public String getDirecion() {

return direcion;

}

public void setDirecion(String direcion) {

this.direcion = direcion;

}

private double distancia;

public double getDistancia() {

return distancia;

}

public void setDistancia(double distancia) {

this.distancia = distancia;

}

private String tiempo;

public String getTiempo() {

return tiempo;

}

public void setTiempo(String tiempo) {

this.tiempo = tiempo;

}

}

DB

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template

\*/

package baseDatos;

import com.mongodb.ConnectionString;

import com.mongodb.MongoClientSettings;

import com.mongodb.ServerApi;

import com.mongodb.ServerApiVersion;

import com.mongodb.client.MongoClient;

import com.mongodb.client.MongoClients;

import com.mongodb.client.MongoCollection;

import com.mongodb.client.MongoDatabase;

import org.bson.Document;

/\*\*

\*

\* @author david

\*/

public class DB {

private MongoClient mongoClient = null;

private MongoCollection<Document> collecProdu = null;

private MongoCollection<Document> collecPedi = null;

private MongoCollection<Document> collecClient = null;

private MongoCollection<Document> collecFact = null;

private MongoCollection<Document> collecPag = null;

private MongoCollection<Document> collecEnvi = null;

public void iniciar() {

String connectionString = "mongodb+srv://fdtoro:WjykdQ47BQOqZoDm@cluster0.mgdhlmc.mongodb.net/?retryWrites=true&w=majority";

ServerApi serverApi = ServerApi.builder()

.version(ServerApiVersion.V1)

.build();

MongoClientSettings settings = MongoClientSettings.builder()

.applyConnectionString(new ConnectionString(connectionString))

.serverApi(serverApi)

.build();

MongoClient mongoClientTem = MongoClients.create(settings);

this.mongoClient = mongoClientTem;

MongoDatabase database = mongoClient.getDatabase("TiendaOnline");

collecProdu=database.getCollection("Productos");

collecClient=database.getCollection("Clientes");

collecPedi=database.getCollection("Pedidos");

collecFact=database.getCollection("Facturas");

collecPag=database.getCollection("Pagos");

collecEnvi=database.getCollection("Envios");

}

public void cerrar() {

this.mongoClient.close();

}

public MongoCollection<Document> getCollecProdu() {

return collecProdu;

}

public MongoCollection<Document> getCollecPedi() {

return collecPedi;

}

public MongoCollection<Document> getCollecClient() {

return collecClient;

}

public MongoCollection<Document> getCollecFact() {

return collecFact;

}

public MongoCollection<Document> getCollecPag() {

return collecPag;

}

public MongoCollection<Document> getCollecEnvi() {

return collecEnvi;

}

}

Guardar

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template

\*/

package baseDatos;

import com.mongodb.client.MongoCollection;

import envios.Envio;

import facturacion.Factura;

import facturacion.Pago;

import org.bson.Document;

import pedidos.Cliente;

import pedidos.Pedido;

import productos.Producto;

/\*\*

\*

\* @author david

\*/

public class Guardar {

private DB db;

public void guardar(Producto pro) {

MongoCollection<Document> collection = db.getCollecProdu();

Document document = new Document("codigo", pro.getCodigo())

.append("nombre", pro.getNombre())

.append("descripcion", pro.getDescripcion())

.append("precio", pro.getPrecio())

.append("precioVenta", pro.getPrecioVenta())

.append("iva", pro.getIva());

Document query = new Document("codigo", pro.getCodigo());

Document result = collection.find(query).first();

if (result != null) {

collection.insertOne(document);

} else {

collection.updateMany(result, document);

}

}

public void guardar(Cliente cliente) {

MongoCollection<Document> collection = db.getCollecClient();

Document document = new Document("dni\_ruc", cliente.getDni\_ruc())

.append("nombre", cliente.getNombre())

.append("direcion", cliente.getDirecion())

.append("telefono", cliente.getTelefono())

.append("email", cliente.getEmail());

Document query = new Document("dni\_ruc", cliente.getDni\_ruc());

Document result = collection.find(query).first();

if (result == null) {

collection.insertOne(document);

} else {

Document filter = new Document("dni\_ruc", cliente.getDni\_ruc());

collection.updateOne(filter, new Document("$set", document));

}

}

public void guardar(Pedido pedido) {

MongoCollection<Document> collection = db.getCollecPedi();

String[] pros=new String[pedido.getProductos().length];

for (int i = 0; i < pedido.getProductos().length; i++) {

pros[i]=pedido.getProductos()[i].getCodigo();

}

Document document = new Document("codigo", pedido.getCodigo())

.append("cliente", pedido.getCliente().getDni\_ruc())

.append("productos", pros)

.append("fecha", pedido.getFecha())

.append("estado", pedido.getEstado());

Document query = new Document("codigo", pedido.getCodigo());

Document result = collection.find(query).first();

if (result == null) {

collection.insertOne(document);

} else {

Document filter = new Document("codigo", pedido.getCodigo());

collection.updateOne(filter, new Document("$set", document));

}

}

public void guardar(Factura factura) {

MongoCollection<Document> collection = db.getCollecFact();

Document document = new Document("codigo", factura.getCodigo())

.append("pedido", factura.getPedido())

.append("subtotal", factura.getSubtotal())

.append("total", factura.getTotal())

.append("estado", factura.getEstado());

Document query = new Document("codigo", factura.getCodigo());

Document result = collection.find(query).first();

if (result == null) {

collection.insertOne(document);

} else {

Document filter = new Document("codigo", factura.getCodigo());

collection.updateOne(filter, new Document("$set", document));

}

}

public void guardar(Pago pago) {

MongoCollection<Document> collection = db.getCollecPag();

Document document = new Document("codigo", pago.getCodigo())

.append("pedido", pago.getPedido())

.append("factura", pago.getFactura())

.append("forma", pago.getForma())

.append("valorPagado", pago.getValorPagado());

Document query = new Document("codigo", pago.getCodigo());

Document result = collection.find(query).first();

if (result == null) {

collection.insertOne(document);

} else {

Document filter = new Document("codigo", pago.getCodigo());

collection.updateOne(filter, new Document("$set", document));

}

}

public void guardar(Envio envio) {

MongoCollection<Document> collection = db.getCollecEnvi();

Document document = new Document("codigo", envio.getCodigo())

.append("pedido", envio.getPedido())

.append("fechaSalida", envio.getFechaSalida())

.append("fechaEntrega", envio.getFechaEntrega())

.append("direcion", envio.getDirecion())

.append("distancia", envio.getDistancia())

.append("tiempo", envio.getTiempo());

Document query = new Document("codigo", envio.getCodigo());

Document result = collection.find(query).first();

if (result == null) {

collection.insertOne(document);

} else {

Document filter = new Document("codigo", envio.getCodigo());

collection.updateOne(filter, new Document("$set", document));

}

}

}

Listar

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template

\*/

package baseDatos;

import com.mongodb.client.FindIterable;

import com.mongodb.client.MongoCollection;

import com.mongodb.client.MongoCursor;

import com.mongodb.client.model.Filters;

import java.util.ArrayList;

import java.util.List;

import org.bson.Document;

import pedidos.Cliente;

import pedidos.Pedido;

import productos.Producto;

/\*\*

\*

\* @author david

\*/

public class Listar {

private DB db;

public Producto[] obtenerProductos() {

MongoCollection<Document> collection = db.getCollecProdu();

List<Producto> lista = new ArrayList<Producto>();

Producto[] pro = new Producto[1];

try (MongoCursor<Document> cursor = collection.find().iterator()) {

while (cursor.hasNext()) {

Document document = cursor.next();

Producto indPro = new Producto();

indPro.setCodigo(document.getString("codigo"));

indPro.setNombre(document.getString("nombre"));

indPro.setDescripcion(document.getString("descripcion"));

indPro.setPrecio(document.getInteger("precio"));

indPro.setIva(document.getBoolean("iva"));

lista.add(indPro);

}

}

pro = lista.toArray(new Producto[0]);

return pro;

}

public Producto obtenerProductos(String a) {

MongoCollection<Document> collection = db.getCollecProdu();

Document query = new Document("codigo", a);

Document result = collection.find(query).first();

Document document = result;

Producto indPro = new Producto();

indPro.setCodigo(document.getString("codigo"));

indPro.setNombre(document.getString("nombre"));

indPro.setDescripcion(document.getString("descripcion"));

indPro.setPrecio(document.getInteger("precio"));

indPro.setIva(document.getBoolean("iva"));

return indPro;

}

public Producto[] obtenerProductos(String nombre, String descripcion, String codigo) {

MongoCollection<Document> collection = db.getCollecProdu();

List<Producto> lista = new ArrayList<Producto>();

Producto[] pro = null;

FindIterable<Document> result;

MongoCursor<Document> cursor = null;

if (nombre.equals("")) {

if (descripcion.equals("")) {

if (codigo.equals("")) {

// Si todos los campos están vacíos, recuperar todos los documentos

result = collection.find();

cursor = result.cursor();

} else {

// Si solo el campo 'codigo' tiene un valor

result = collection.find(Filters.regex("codigo", codigo));

cursor = result.cursor();

}

} else {

if (codigo.equals("")) {

// Si solo el campo 'descripcion' tiene un valor

result = collection.find(Filters.regex("descripcion", descripcion));

cursor = result.cursor();

} else {

// Si 'descripcion' y 'codigo' tienen valores

result = collection.find(Filters.and(

Filters.regex("descripcion", descripcion),

Filters.regex("codigo", codigo)

));

cursor = result.cursor();

}

}

} else {

if (descripcion.equals("")) {

if (codigo.equals("")) {

// Si solo el campo 'nombre' tiene un valor

result = collection.find(Filters.regex("nombre", nombre));

cursor = result.cursor();

} else {

// Si 'nombre' y 'codigo' tienen valores

result = collection.find(Filters.and(

Filters.regex("nombre", nombre),

Filters.regex("codigo", codigo)

));

cursor = result.cursor();

}

} else {

if (codigo.equals("")) {

// Si 'nombre' y 'descripcion' tienen valores

result = collection.find(Filters.and(

Filters.regex("nombre", nombre),

Filters.regex("descripcion", descripcion)

));

cursor = result.cursor();

} else {

// Si todos los campos tienen valores

result = collection.find(Filters.and(

Filters.regex("nombre", nombre),

Filters.regex("descripcion", descripcion),

Filters.regex("codigo", codigo)

));

cursor = result.cursor();

}

}

}

while (cursor.hasNext()) {

Document document = cursor.next();

Producto indPro = new Producto();

indPro.setCodigo(document.getString("codigo"));

indPro.setNombre(document.getString("nombre"));

indPro.setDescripcion(document.getString("descripcion"));

indPro.setPrecio(document.getInteger("precio"));

indPro.setIva(document.getBoolean("iva"));

lista.add(indPro);

}

pro = lista.toArray(new Producto[0]);

return pro;

}

public Pedido[] obtenerPedidos() {

MongoCollection<Document> collection = db.getCollecProdu();

List<Pedido> lista = new ArrayList<Pedido>();

Pedido[] pro = null;

try (MongoCursor<Document> cursor = collection.find().iterator()) {

while (cursor.hasNext()) {

Document document = cursor.next();

Pedido ind = new Pedido();

ind.setCodigo(document.getInteger("codigo"));

ind.setCliente(obtenerCliente(document.getInteger("cliente")));

Producto[] produc = new Producto[((String[]) document.get("productos")).length];

for (int i = 0; i < produc.length; i++) {

produc[i] = obtenerProductos(((String[]) document.get("productos"))[i]);

}

ind.setProductos(produc);

ind.setFecha(document.getString("fecha"));

ind.setEstado(document.getString("estado"));

lista.add(ind);

}

}

pro = lista.toArray(new Pedido[0]);

return pro;

}

public Pedido obtenerPedido(Integer a) {

MongoCollection<Document> collection = db.getCollecClient();

Document query = new Document("codigo", a);

Document result = collection.find(query).first();

Document document = result;

Pedido ind = new Pedido();

ind.setCodigo(document.getInteger("codigo"));

ind.setCliente(obtenerCliente(document.getInteger("cliente")));

Producto[] produc = new Producto[((String[]) document.get("productos")).length];

for (int i = 0; i < produc.length; i++) {

produc[i] = obtenerProductos(((String[]) document.get("productos"))[i]);

}

ind.setProductos(produc);

ind.setFecha(document.getString("fecha"));

ind.setEstado(document.getString("estado"));

return ind;

}

public Cliente obtenerCliente(int ruc) {

MongoCollection<Document> collection = db.getCollecClient();

Document query = new Document("dni\_ruc", ruc);

Document result = collection.find(query).first();

Document document = result;

Cliente ind = new Cliente();

ind.setDni\_ruc(document.getInteger("dni\_ruc"));

ind.setDirecion(document.getString("direcion"));

ind.setNombre(document.getString("nombre"));

ind.setEmail(document.getString("email"));

ind.setTelefono(document.getInteger("telefono"));

return ind;

}

}