

T4 Bigarren Erronka



Data: 2021-11-16

Taldea: T4

Izenak: Asier Blazquez, Julen Mantecon eta Jon Zengotita

GitHub Esteka : <https://github.com/asierblaz/Erronka/Erronka2>

AURKIBIDEA:

AURKIBIDEA:	1
Sarrera	2
Testuingurua	2
Helburuak	2
Helburu Orokorrak	2
Helburu Zehatzak	2
Baliabideak	3
Baliabide Teknikoak	3
Trello Esteka	4
Garapen Teknikoa	5
Multimedia Programazioa Eta Gailu Mugikorrak	5
Connect	5
Hariak	6
Datu Atzipena	10
Spring erabilia eta hibernate orm a ren laguntzarekin postgres datu basetik interesatzen zaizkigun entitateak mapeatu ditugu eta sql express zerbitzari batean entitate berriak sortu ditugu interesatzen zitzaizkigun datuekin.	10
Zerbitzu Eta Prozesuen Programazioa	14
Enpresa Kudeaketako Sistemak	14
Odoo datu basetik, produktuen datuak hartu ditugu csv batean txertatuz	14
Interfazeen Garapena	15
Etorkizunerako ildoak	22
Ondorioak	22

Sarrera

Gomazko ahateak saltzen dituen enpresa batentzat sistema informatiko bat egitea egokitu zaigu. Horretarako Odoo ERP-a erabili behar genuen web orri bat egiteko, honen bitartez salmentak etab egiteko. Horrez gain mugikorretarako aplikazio bat komertzialak erabili dezan.

Testuingurua

Bigarren erronka honetan mahi gaineko aplikazio bat egin behar dugu komertzialak produktu,bezero... en estatistikak ikusteko. Horrez gain androideko aplikazioa odoo zerbitzarira konektatuko da bitartekaririk gabe, honekin salmenta aurrekontuak sortu ahal ditzake. Javako aplikazioak hibernate orm a erabiliz sql server datu base bat mapeatu eta aukera desberdinak xml batera exportatu ahalko ditu.

Helburuak

Helburu Orokorrak

- Android aplikazioa hobetu eta postgresql konexioa egin
- Java aplikazioan hibernate konfiguratu eta xml exportazioa egin
- Visual studio-n mahai gaineko aplikazioa egin, sql serverrera konektatu eta datuak grafiko ezberdinetan irudikatu.

Helburu Zehatzak

- Spring erabilita eta hibernate orm a ren laguntzarekin postgres datu base tik interesatzen zaizkigun entitateak mapeatu ditugu eta sql express zerbitzari batean entitate berriak sortu ditugu interesatzen zitzaizkigun datuekin.
- Xml ak atzitzeko Dom api klasea erabili dugu, log bat eginez. Honek datu baseko artzipenak gordetzen ditu eta horrez gain guk aukeratutako parametro bat gordeko du.
- Android aplikazioan PostgrSQL zerbitzarira konektatu gara datuak jaso eta sartzeko.
- Hariak erabili ditugu kontsultak egiteko eta semaforo bat jarri dugu.
- Salmentak erakutsi eta sortu ahal izateko hobetu dugu aplikazioa.
- Visual studion windows form bat sortu dugu grafikoetan datuak erakusteko
- DLL ak erabili ditugu
- .exe bat sortu dugu aplikazioa exekutatzeko

Baliabideak

Baliabide Teknikoak

Android Studio: Aplikazioa egiteko erabili dugu.

Odoo: Gure ERP-a, produktuen kudeaketa, web orria eta salerosketak eramateko erabili dugu.

Eclipse: Odoo-ko produktuak jasotzeko eta json batera exportatzeko erabili dugu.

DBeaver: Datu basea aztertzeko

GitHub: Bertsioen Kontrolerako erabili dugu.

Trello: Zereginen kontrola eraman ahal izateko erabili dugu.

OpenWebinars: Spring tutorialak jarraitzeko erabili dugu

StackOverFlow: Zalantzak ebazteko erabili dugu.

Visual Studio: Windows form aplikazioa egiteko erabili dugu

Microsoft Management SQL Server Studio 18: Sql server kudeatzeko erabili dugu

Giza Baliabideak

Proiektua egiteko hainbat profil erabili ditugu: idazkaritza, koordinazioa, programazioa etab.

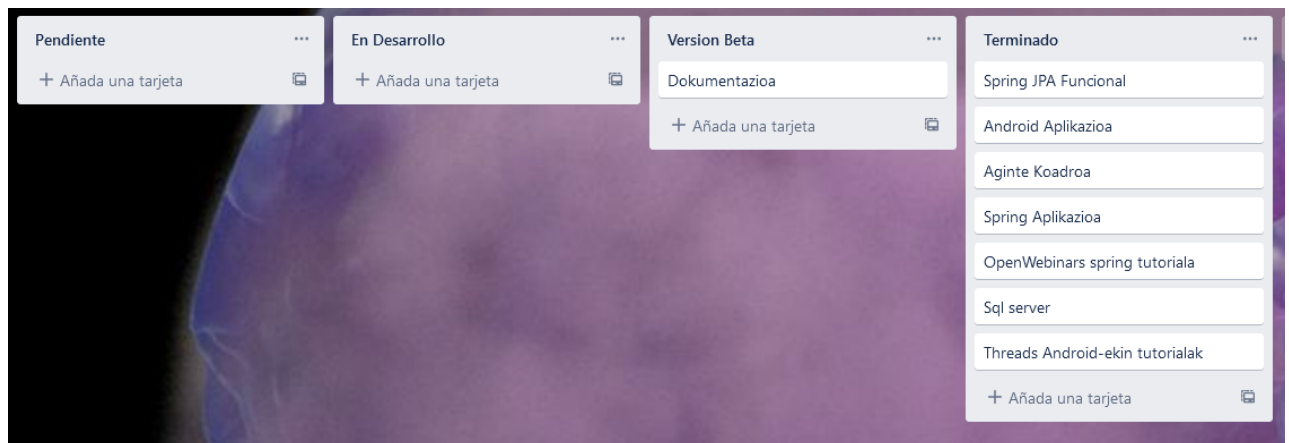
Hasiera batean, rolak banatu genituen. Baina denborak aurrera egin ahala, denok profil guztiak betetzen genituela ohartu ginen eta denok lan mota guztietan hartu dugu parte.

Denbormalizazioa eta Ardurak

Gure erronka taldean hiru rol bereizten dira, koordinatzailea , idazkaria eta bozeramailea.

Kasu honetan egin beharreko atazak ez ditugu banatu, hau da, guztiok egon gara guztia egiten.

Planifikazioa egiteko Trello erabili dugu, honekin ataza bakoitzaren egoera kontrolatzen dugu eta horrela hobeto antolatzen gara.



[Trello Esteka](#)

Garapen Teknikoa

Multimedia Programazioa Eta Gailu Mugikorrak

Android Aplikazio bat garatu dugu odoon dauden produktuak bistaratzeko, horretarako spring erabilia esportatzen dugu csv fitxategi bat produktuen datuekin eta android-en irakurtzen ditugu.

Produktuen irakurketa eta erakusketa egiteko bi gauza erabiltzen ditugu, Connect() funtzioa eta Select-ak dituzten hariak.

Connect

```
//Konexio funtzioa
public static Connection Connect() throws SQLException, ClassNotFoundException {
    //Konexioa egiten saiatzen da
    try {
        //PostgreSQL-ren driver-a
        Class.forName("org.postgresql.Driver");
        /*
        Helbidea, erabiltzailea eta pasahitzaren bitartez konexioa egin eta bueltatzen da
        url => jdbc:postgresql://192.168.65.15:5432/PatitosdeGoma
        user => openpg
        pass => openpgpwd
        */
        Connection conn = DriverManager.getConnection(url, user, pass);
        return conn;
    } catch (SQLException se) {
        Log.d( tag: "SQLException", msg: "No se puede conectar. Error: " + se.toString());
    } catch (ClassNotFoundException e) {
        Log.d( tag: "ClassNotFoundException", msg: "No se encuentra la clase. Error: " + e.getMessage());
    }
    return null;
}
```

Hariak

```
//Produktuen lista hartzen duen haria
static Thread ProduktuakQuery = new Thread(() ->
{
    //Produktuak hartzeko saiakera egiten du
    try {
        //Query-a
        String query = "select product_template.id, product_template.name, " +
            "product_category.complete_name as category, product_template.list_price, " +
            "product_template.default_code from product_template\n" +
            "inner join product_category on product_template.categ_id =product_category.id \n" +
            "order by name asc";
        //Connect() funtzioari deitzen zaio konexioa gordetzeko
        Connection conn = Connect();
        //Query-a gorde eta exekutatzen da
        Statement st = conn.createStatement();
        ResultSet rs = st.executeQuery(query);
        //Hemen bueltatutako produktu guztiak Produktua klasean dagoen produktuak ArrayList-ean gordetzen dira
        while (rs.next()) {
            Produktua p = new Produktua(
                rs.getInt(1), rs.getString(2), rs.getString(3),
                rs.getFloat(4), rs.getString(5).toLowerCase()
            );
            Produktua.produktuak.add(p);
        }
        //Kategoriak Produktua klasean dagoen kategorias ArrayList-ean gordetzen dira
        Produktua.categoriasToArray();
        //Konexioa ixten da
        conn.close();
        //Salbuespena
    } catch (Exception e) {
        Log.d( tag: "Exception", msg: "run: Failed " + e.getMessage());
    }
});
```

```
//Bezeroen lista hartzen duen haria
static Thread BezeroakQuery = new Thread(() ->
{
    //Bezeroak hartzeko saiakera egiten du
    try {
        //Query-a
        String query = "select id, name from res_partner where user_id = 12";
        //Connect() funtzioari deitzen zaio konexioa gordetzeko
        Connection conn = Connect();
        //Query-a gorde eta exekutatzen da
        Statement st = conn.createStatement();
        ResultSet rs = st.executeQuery(query);
        //Hemen bueltatutako bezero guztiak Bezeroa klasean dagoen bezeroak ArrayList-ean gordetzen dira
        while (rs.next()) {
            Bezeroa b = new Bezeroa(
                rs.getInt(1), rs.getString(2)
            );
            Bezeroa.bezeroak.add(b);
        }
        //Konexioa ixten da
        conn.close();
        //Salbuespena
    } catch (Exception e) {
        Log.d( tag: "Exception", msg: "run: Failed " + e.getMessage());
    }
});
```

```

//sale_order taulako azken Id-a hartzen duen haria
static Thread SolIdQuery = new Thread() ->
{
    //Id-a hartzeko saiakera egiten du
    try {
        //Query-a
        String query = "select MAX(id) from sale_order";
        //Connect() funtzioari deitzen zaio konexioa gordetzeko
        Connection conn = Connect();
        //Query-a gorde eta exekutatzen da
        Statement st = conn.createStatement();
        ResultSet rs = st.executeQuery(query);
        /*
        Hemen bueltatutako id-ari 1 gehitzen zaio eta CrearPedido klasean dagoen so_id aldagaian gordetzen da.
        Salmenta egiteko beharrezkoa den salmenta izena era sortzen da eta CrearPedido klasean dagoen so_name aldagaian gordetzen da.
        */
        while (rs.next()) {
            CrearPedido.so_id = rs.getInt(1)+1;
            CrearPedido.so_name = "S000" + CrearPedido.so_id;
        }
        //Konexioa ixten da
        conn.close();
        //Salbuespena
    } catch (Exception e) {
        Log.d( tag: "Exception", msg: "run: Failed " + e.getMessage());
    }
}
});

```

```

//sale_order_line taulako azken Id-a hartzen duen haria
static Thread SolIdQuery = new Thread() ->
{
    //Id-a hartzeko saiakera egiten du
    try {
        //Query-a
        String query = "select MAX(id) from sale_order_line";
        //Connect() funtzioari deitzen zaio konexioa gordetzeko
        Connection conn = Connect();
        //Query-a gorde eta exekutatzen da
        Statement st = conn.createStatement();
        ResultSet rs = st.executeQuery(query);
        //Hemen bueltatutako id-ari 1 gehitzen zaio eta CrearPedido klasean dagoen sol_id aldagaian gordetzen da.
        while (rs.next()) {
            CrearPedido.sol_id = rs.getInt(1)+1;
        }
        //Konexioa ixten da
        conn.close();
        //Salbuespena
    } catch (Exception e) {
        Log.d( tag: "Exception", msg: "run: Failed " + e.getMessage());
    }
}
});

```



```

//Salmenten lista hartzen duen haria
static Thread SalmentakQuery = new Thread() ->
{
    //Salmentak hartzeko saiakera egiten du
    try {
        //Query-a
        String query = "Select id, name, state, date_order, create_date, partner_id, " +
            "invoice_status, amount_untaxed, amount_tax, amount_total from sale_order " +
            "order by name desc";
        //Connect() funtzioari deitzen zaio konexioa gordetzeko
        Connection conn = Connect();
        //Query-a gorde eta exekutatzen da
        Statement st = conn.createStatement();
        ResultSet rs = st.executeQuery(query);
        //Hemen bueltatutako produktu guztiak Salmenta klasean dagoen salmentak ArrayList-ean gordetzen dira
        while (rs.next()) {
            Salmenta s = new Salmenta(
                rs.getInt(1), rs.getString(2), rs.getString(3),
                rs.getString(4), rs.getString(5),
                rs.getInt(6), rs.getString(7),
                rs.getFloat(8), rs.getFloat(9),
                rs.getFloat(10)
            );
            Salmenta.salmentak.add(s);
        }
        //Konexioa ixten da
        conn.close();
        //Salbuespena
    } catch (Exception e) {
        Log.d("tag: \"Exception\", \"msg: \"run: Failed\"+ e.getMessage());
    }
};

```

```

//Salmenta sortzen duen haria
static Thread InsertOrderQuery = new Thread() ->
{
    //Query-a
    String query = "insert into sale_order(id, require_signature, require_payment, " +
        "partner_id, partner_invoice_id, partner_shipping_id, pricelist_id, currency_id," +
        "name, state, date_order, create_date, invoice_status, amount_untaxed, " +
        "amount_tax, amount_total, currency_rate, company_id, team_id, create_uid, " +
        "write_uid, write_date, picking_policy, warehouse_id)" +
        "values (" + CrearPedido.so_id + ", true, false, " + CrearPedido.partner_id + ", " +
        CrearPedido.partner_id + ", " + CrearPedido.partner_id + ", 1, 1, " +
        CrearPedido.so_name + ", 'draft', CURRENT_TIMESTAMP, CURRENT_TIMESTAMP, 'no', " +
        CrearPedido.so_amount + ", 0, " + CrearPedido.so_amount + ", 1, 1, 1, 2, 2, " +
        "CURRENT_TIMESTAMP, 'direct', 1)";
    //Salmenta sortzeko saiakera
    try {
        //Semaforora sartzen da
        CrearPedido.pedido.acquire();
        //Salmenta-ren query-a exekutatzen salatzen da
        try {
            //Connect() funtzioari deitzen zaio konexioa gordetzeko
            Connection conn = Connect();
            //Query-a gorde eta exekutatzen da
            Statement st = conn.createStatement();
            st.executeUpdate(query);
            //Konexioa ixten da
            conn.close();
        } finally {
            //Semaforotik irtetzen da
            CrearPedido.pedido.release();
        }
    }
    //Salbuespena
    catch (Exception e) {
        Log.d("tag: \"InsertOrderQuery\", e.getMessage());
    }
};

```

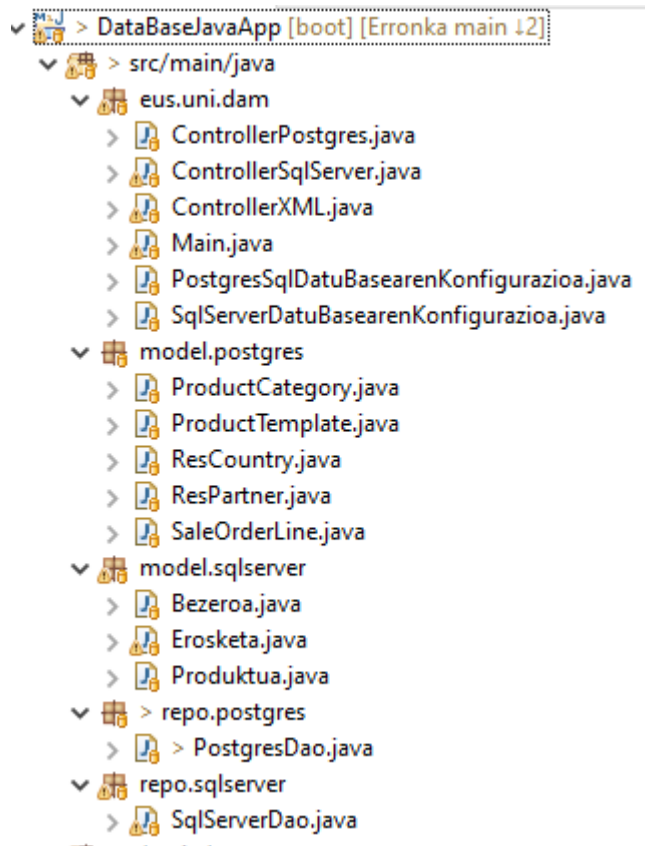
```

//Salmenta lerroa(k) sortzen dituen haria
static Thread InsertLineQuery = new Thread() ->
{
    try {
        //Semaforora sartzen da eta salmenta lerroa(k) sortzeko saiakera
        CrearPedido.pedido.acquire();
        try {
            //Connect() funtzioari deitzen zaio konexioa gordetzeko
            Connection conn = DataConnect.Connect();
            //Lerro bakoitzeko insert bat egiten da
            for (ProductoCarrito p : ProductoCarrito.carrito) {
                //Query-a
                String query = "insert into sale_order_line(id, order_id, name, sequence, invoice_status, " +
                    "price_unit, price_tax, price_subtotal, price_total, price_reduce, price_reduce_taxinc, " +
                    "price_reduce_taxexcl, discount, product_id, product_uom, product_uom_qty, " +
                    "qty_delivered_method, qty_delivered, qty_delivered_manual, qty_to_invoice, " +
                    "qty_invoiced, untaxed_amount_invoiced, untaxed_amount_to_invoice, currency_id, " +
                    "company_id, order_partner_id, state, customer_lead, create_uid, create_date, " +
                    "write_uid, write_date)" +
                    "values( " + CrearPedido.sol_id + ", " + CrearPedido.so_id + ", ' " + p.getProducto().getName() +
                    "' , 10, 'no', " + p.getProducto().getPrezioa() + ", 0, " + p.getPrecio() +
                    ", " + p.getPrecio() + ", " + p.getPrecio() + ", " + p.getPrecio() +
                    ", " + p.getPrecio() + ", 0, " + p.getProducto().getId() + ", 1, " +
                    p.getCantidad() + ", 'stock_move', 0, 0, 0, 0, 0, " + p.getPrecio() +
                    ", 1, 1, " + CrearPedido.partner_id + ", 'draft', 0, 2, CURRENT_TIMESTAMP, 2, " +
                    "CURRENT_TIMESTAMP)";
                //Query-a gorde eta exekutatzen da
                Statement st = conn.createStatement();
                st.executeUpdate(query);
                //Erabilitako id-ari 1 gehitzen zaio eta CrearPedido klasean dagoen sol_id aldagaian gordetzen da.
                CrearPedido.sol_id += 1;
            }
        } finally {
            //Semaforotik irtetzen da eta saskia huzten du
            CrearPedido.pedido.release();
            ProductoCarrito.carrito.clear();
        }
    }
    //Salbuespena
} catch (Exception e) {
    Log.d( tag: "InsertLineQuery", e.getMessage());
    e.printStackTrace();
}
});

```

Datu Atzipena

Spring erabilita eta hibernate orm a ren laguntzarekin postgres datu basetik interesatzen zaizkigun entitateak mapeatu ditugu eta sql express zerbitzari batean entitate berriak sortu ditugu interesatzen zitzaizkigun datuekin.



Aplikazio honetan klase guztiak paketeetan antolatu ditugu, model.postgres-en PostgreSQL datu basetik hartuko ditugun taulak mapeatu ditugu, aldiz model.sqlserver-en aurreko entitateetatik datuak hartzen ditugu eta sortutako entitate berrietan txertatuko ditugu sqlServer-en gordetzeko.

Datu basean egiten diren aldaketak PostgresDao eta SqlServerDao-en daude eta repo paketeetan antolatu ditugu.

Objetuekin operazioak egiteko bi kontroler sortu ditugu, ControllerPostgres-en datuak hartu eta listetan txertatzen ditugu eta ControllerSql-en listetan zehar objektu berriak sortzen joaten gara.

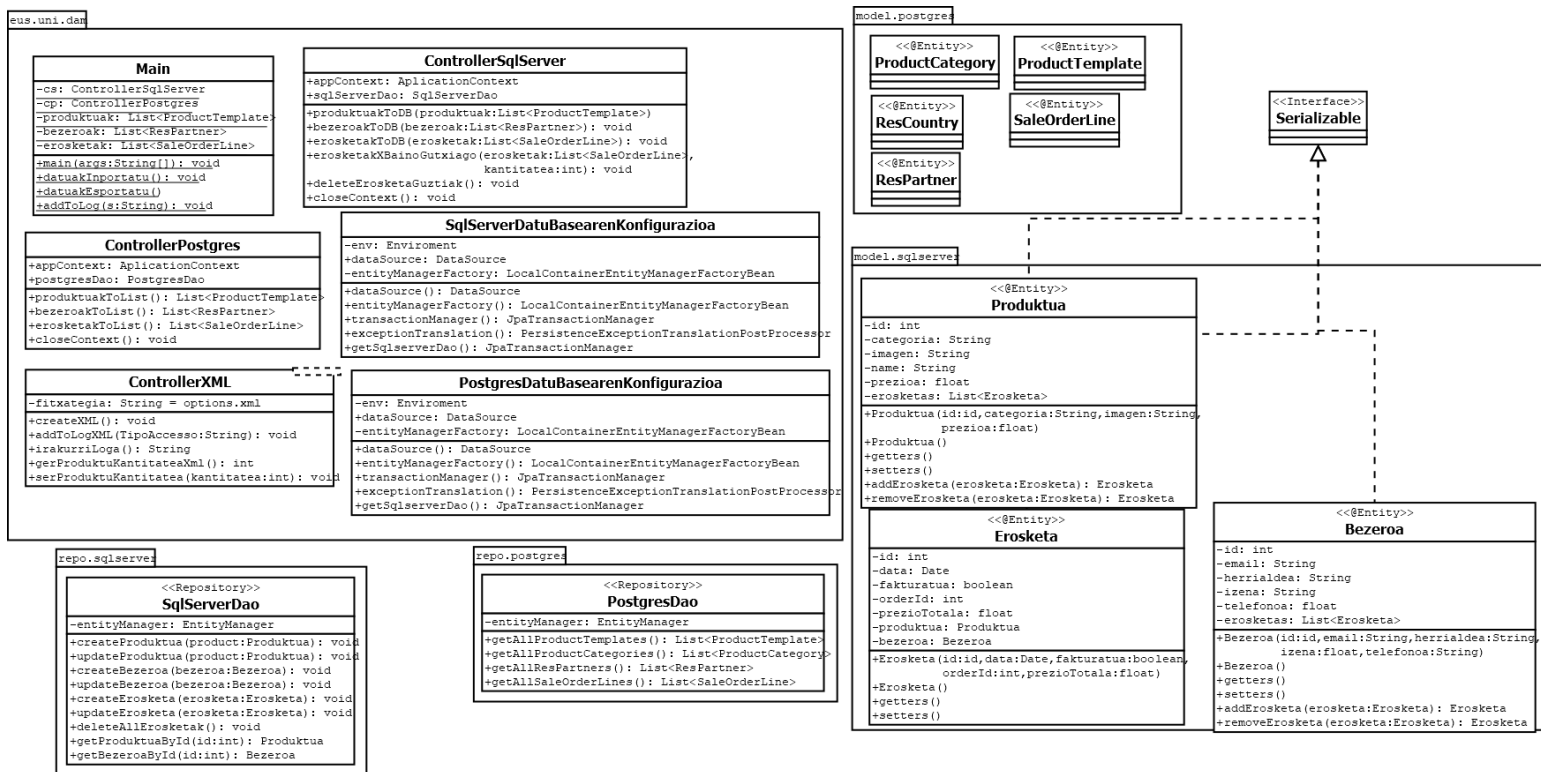
Kasu honetan objektuak datu base batetik bestera eramateko bi datu basearen konfigurazio behar izango ditugu, horretarako application.properties-en definitzen ditugu.

Horretaz gain aplikazioak exportazio bat egiten duen bakoitzean log batean erregistratuko da, horretarako DomApi libreria erabili dugu eta xml batean gordeko da informazioa.

Xml-Aren egitura:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?><?xml-stylesheet type="text/xsl" href="options.xslt"?>
<document>
  <options>
    <ProduktuKopurua>6</ProduktuKopurua>
  </options>
  <log>
    <acceso Date="11-11-2021 12:13:02">Fitxategia Sortu</acceso>
    <acceso Date="11-11-2021 12:13:02">Datu base osoa esportatu da</acceso>
    <acceso Date="11-11-2021 12:28:44">Datu base osoa esportatu da</acceso>
    <acceso Date="18-11-2021 14:29:01">Datu base osoa esportatu da</acceso>
  </log>
</document>
```

Klase Diagrama:



Aplikazioa exekutatzean hurrengo menua ikusiko dugu:

```
-----PATINHOS GOMOSOS-----
      MENUA
=====
1.- Datu Guztiak Esportatu
2.- Esportatu nahi diren datuak aukeratu
3.- Log-a Irakurri
4.- Log-a Ikusi
5-Irten

Aukeratu zenbaki bat
```

1.Datu Guztiak Esportatu:

Datuak esportatuko dira postgresql-tik SqlServer-era eta loguean lerro berri bat gehituko da, fitxategia ez bada existitzen mezu bat aterako da eta xml berri bat sortuko du.

```
Fitxategia ez da existitzen, berri bat sortu da
      MENUA
=====
1.- Datu Guztiak Esportatu
2.- Esportatu nahi diren datuak aukeratu
3.- Log-a Irakurri
4.- Log-a Ikusi
5- Irten

Aukeratu zenbaki bat
```

2.Esportatu nahi diren datuak aukeratu:

```
Aukeratu zenbaki bat 2
      ESPORTATU NAHI DIREN DATUAK AUKERATU
=====
1.- X produktu baino gutxiago dituzten erosketak
2.- Azken Exportazioaren konfigurazioa
3.- Atzera

Aukeratu zenbaki bat
```

1.X produktu baino gehiago dituzten erosketak
aukera hau aukeratzen badugu sartutako zenbakia baino produktu gutxiago dituzten erosketak gordeko dira sqlserveren eta xmlan <ProduktuKopurua> eguneratuko da log-arekin batera.

2. Azken Exportazioaren konfigurazioa;

xml-an ProduktuKopurua irakurriko du, 0 bada esan nahi du ez dagoela aurretik konfiguraziorik, aldiz beste zenbaki bat bada kopuru hori baino produktu gutxiago dituzten produktuak esportatuko ditu.

```
Aurretik dagoen konfigurazioa esportatu da, 5 produktu baino gutxiago dituzten erosketak esportatu dira
EXPORTATU NAHI DIREN DATUAK AUKERATU
=====
1.- X produktu baino gutxiago dituzten erosketak
2.- Azken Exportazioaren konfigurazioa
3.- Atzera
```

3. Atzera

Aurreko menura itzuliko gara

```
Aukeratu zenbaki bat 3
|      MENUA
|
=====
1.- Datu Guztiak Esportatu
2.- Esportatu nahi diren datuak aukeratu
3.- Log-a Irakurri
4.- Log-a Ikusi
5- Irten
```

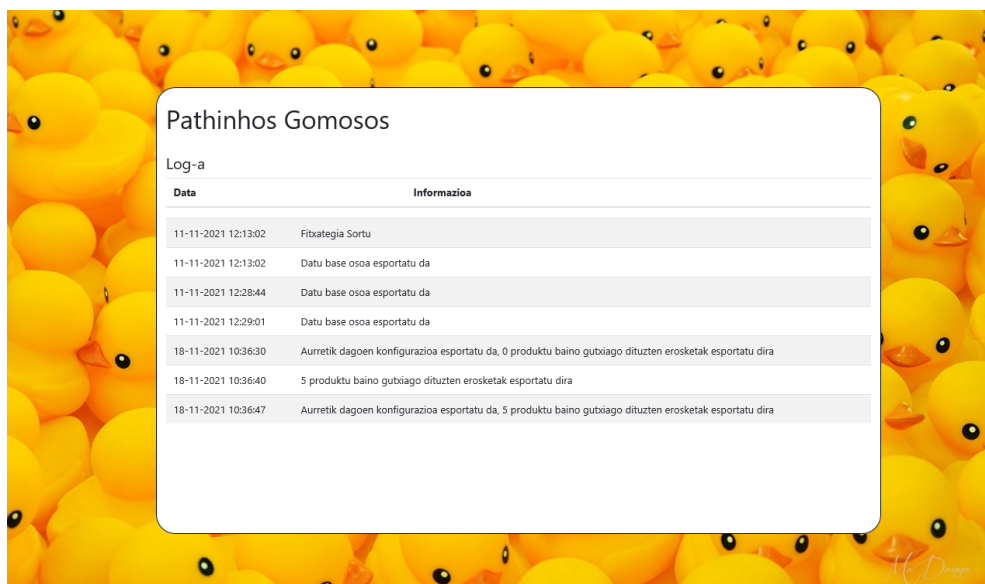
Aukeratu zenbaki bat

3-Log-a Irakurri

```
Log-ean dauden Erregistroak:
1. Konexio Data: 11-11-2021 12:13:02
Informazioa: Fitxategia Sortu
-----
2. Konexio Data: 11-11-2021 12:13:02
Informazioa: Datu base osoa esportatu da
-----
3. Konexio Data: 11-11-2021 12:28:44
Informazioa: Datu base osoa esportatu da
-----
4. Konexio Data: 11-11-2021 12:29:01
Informazioa: Datu base osoa esportatu da
-----
5. Konexio Data: 18-11-2021 10:36:30
Informazioa: Aurretik dagoen konfigurazioa esportatu da, 0 produktu baino gutxiago dituzten erosketak esportatu dira
-----
6. Konexio Data: 18-11-2021 10:36:40
Informazioa: 5 produktu baino gutxiago dituzten erosketak esportatu dira
-----
7. Konexio Data: 18-11-2021 10:36:47
Informazioa: Aurretik dagoen konfigurazioa esportatu da, 5 produktu baino gutxiago dituzten erosketak esportatu dira
-----
```

4- Log-a Ikusi

aukera hau prozesu bat sortuko du eta internet Explorer nabigatzailea options.xml-a irekiko da, xml-ari xslt diseinua jarri diogu.



Data	Informazioa
11-11-2021 12:13:02	Fitxategia Sortu
11-11-2021 12:13:02	Datu base osoa esportatu da
11-11-2021 12:28:44	Datu base osoa esportatu da
11-11-2021 12:29:01	Datu base osoa esportatu da
18-11-2021 10:36:30	Aurretik dagoen konfigurazioa esportatu da, 0 produktu baino gutxiago dituzten erosketak esportatu dira
18-11-2021 10:36:40	5 produktu baino gutxiago dituzten erosketak esportatu dira
18-11-2021 10:36:47	Aurretik dagoen konfigurazioa esportatu da, 5 produktu baino gutxiago dituzten erosketak esportatu dira

Zerbitzu Eta Prosesuen Programazioa

Prozesu bat sortu dugu gure options.xml-a bateratzeko

```
public static void logaIrekiProcess() {  
    try {  
        ProcessBuilder pb = new ProcessBuilder("C:\\Program Files\\Internet Explorer\\iexplore.exe",  
            "file:///C:/Users/blazquez.asier/Documents/DAM2/Erronka/Erronka2/DataBaseJavaApp/options.xml");  
        Process p1 = pb.start();  
    } catch (Exception e) {  
        e.printStackTrace();  
    }  
}
```

Enpresa Kudeaketako Sistemak

Odoo datu basetik, produktuen datuak hartu ditugu csv batean txertatuz

Xehetasun gehiagorako ikusi gure erabiltzaile gida

<https://docs.google.com/document/d/1y7zMkilGcF583jAEZrErFMf-cUQqNvqMEFf91F4dt7Q/edit#heading=h.jg9wy3dlkrhe>

Interfazeen Garapena

Sql Server Management erabiliz datu base bat sortu dugu gero Visual studioarekin datuak jaso ahal izateko. Formulario desberdinak egin ditugu:

Menu Nagusia:



Hiru botoi ditugu, bakoitzak sakatutakoan formulario batera eramaten gaitu eta xagua gainean jartzerakoan irudia desagertu eta eramaten duen formularioaren izena agertuko da



Hauek dira funtzio horiek egiten dituzten metodoak:

```
10 referencias | Asier, Hace 6 días | 3 autores, 4 cambios
public partial class MenuNagusia : Form
{
    Image batJoanImg;
    Image biJoanImg;
    Image hiruJoanImg;

    String batJoanText = "Gehien erosten duten bezeroak";
    String biJoanText = "Produktu Salduenak";
    String hiruJoanText = "Gehien erosten duten herrialdeak";

    4 referencias | Asier, Hace 6 días | 2 autores, 2 cambios
    public MenuNagusia()
    {
        InitializeComponent();
        this.FormBorderStyle = FormBorderStyle.FixedSingle;
        this.MaximizeBox = false;

        batJoan.MouseEnter += OnMouseEnterBat;
        batJoan.MouseLeave += OnMouseLeaveBat;

        biJoan.MouseEnter += OnMouseEnterBi;
        biJoan.MouseLeave += OnMouseLeaveBi;

        hiruJoan.MouseEnter += OnMouseEnterHiru;
        hiruJoan.MouseLeave += OnMouseLeaveHiru;

        batJoanImg = batJoan.Image;
        biJoanImg = biJoan.Image;
        hiruJoanImg = hiruJoan.Image;
    }
}
```

```
1 referencia | Asier, Hace 6 días | 1 autor, 1 cambio
private void OnMouseEnterBat(object sender, EventArgs e)
{
    batJoan.Image = null;
    batJoan.Text = batJoanText;
}

1 referencia | Asier, Hace 6 días | 1 autor, 1 cambio
private void OnMouseLeaveBat(object sender, EventArgs e)
{
    batJoan.Image = batJoanImg;
    batJoan.Text = null;
}

1 referencia | Asier, Hace 6 días | 1 autor, 1 cambio
private void OnMouseEnterBi(object sender, EventArgs e)
{
    biJoan.Image = null;
    biJoan.Text = biJoanText;
}

1 referencia | Asier, Hace 6 días | 1 autor, 1 cambio
private void OnMouseLeaveBi(object sender, EventArgs e)
{
    biJoan.Image = biJoanImg;
    biJoan.Text = null;
}
```

```

1 referencia | zengomax, Hace 8 días | 1 autor, 1 cambio
private void batJoan_Click(object sender, EventArgs e)
{
    BezeroErosketak bat = new BezeroErosketak();
    bat.Show();
    this.Hide();
}

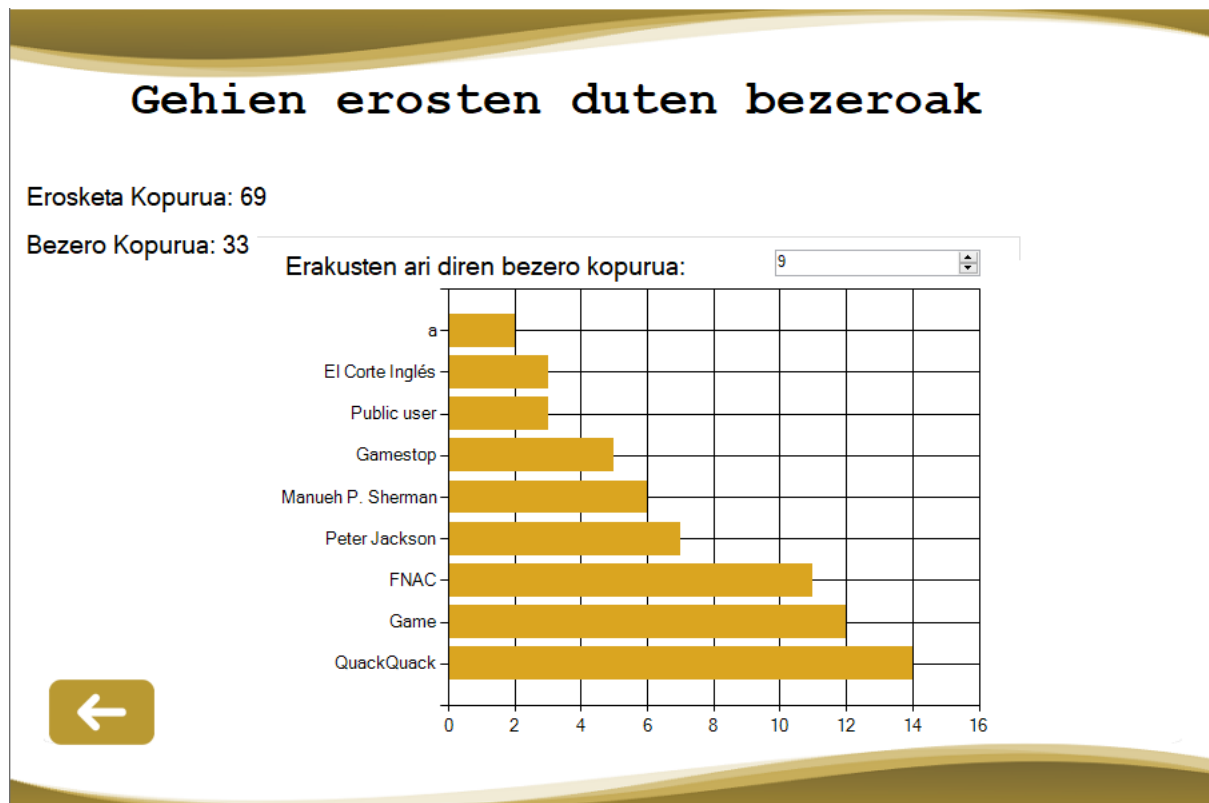
1 referencia | zengomax, Hace 8 días | 1 autor, 1 cambio
private void biJoan_Click(object sender, EventArgs e)
{
    ProduktuSalduenak bi = new ProduktuSalduenak();
    bi.Show();
    this.Hide();
}

1 referencia | zengomax, Hace 8 días | 1 autor, 1 cambio
private void hiruJoan_Click(object sender, EventArgs e)
{
    Herrialdeak hiru = new Herrialdeak();
    hiru.Show();
    this.Hide();
}

1 referencia | julen-mantecon, Hace 6 días | 1 autor, 1 cambio
private void Form_Close(object sender, System.EventArgs e)
{
    System.Windows.Forms.Application.Exit();
}
}

```

Bezero Eroketak:



Bezera eta Erosketa tauletatik datuak jasotzen dituen bista batetik datuak jasotzen dituen grafiko bat daukagu. Goiko balio numerikoaren arabera grafikoak fila kopurua erakutsiko du(kasu honetan 9 bezera erakusten ditu) alboan dauden datuak ateratzeko dli bat erabili dugu, dli horrek dituen funtzioak string batzuk itzultzen dituzte eta deia egiteko gure formularioetan klase horren objektua sortzen dugu eta funtzioari deitzen diogu.

```
0 referencias | Asier, Hace 7 días | 1 autor, 1 cambio
public string getproduktuKantitatea() {
    produktuaTableAdapter.Fill(this.patosDeGomaDataSet.Produktua);
    return this.produktuaTableAdapter.ProduktuaQuery() + "";
}
```

```
Funtzioak f.. = new Funtzioak();

1 referencia | Asier, Hace 7 días | 2 autores, 2 cambios
public ProduktuSalduenak()
{
    InitializeComponent();

    this.FormBorderStyle = FormBorderStyle.FixedSingle;
    this.MaximizeBox = false;

    ProduktuKopurua.Text = "Produktu Kopurua: " + f.getproduktuKantitatea();
}
```

Gezidun botiak formulario nagusira itzuliko gaitu eta formulario hau ixten badugu ere formulario nagusira itzultzen gaitu.

```
1 referencia | julen-mantecon, Hace 6 días | 1 autor, 1 cambio
private void Form_Close(object sender, System.EventArgs e)
{
    MenuNagusia menuNagusia = new MenuNagusia();
    menuNagusia.Show();
}

1 referencia | zengomax, Hace 8 días | 1 autor, 1 cambio
private void ZematErakutsi_ValueChanged(object sender, EventArgs e)
{
    int erakutsiBalio = (int)ZematErakutsi.Value;

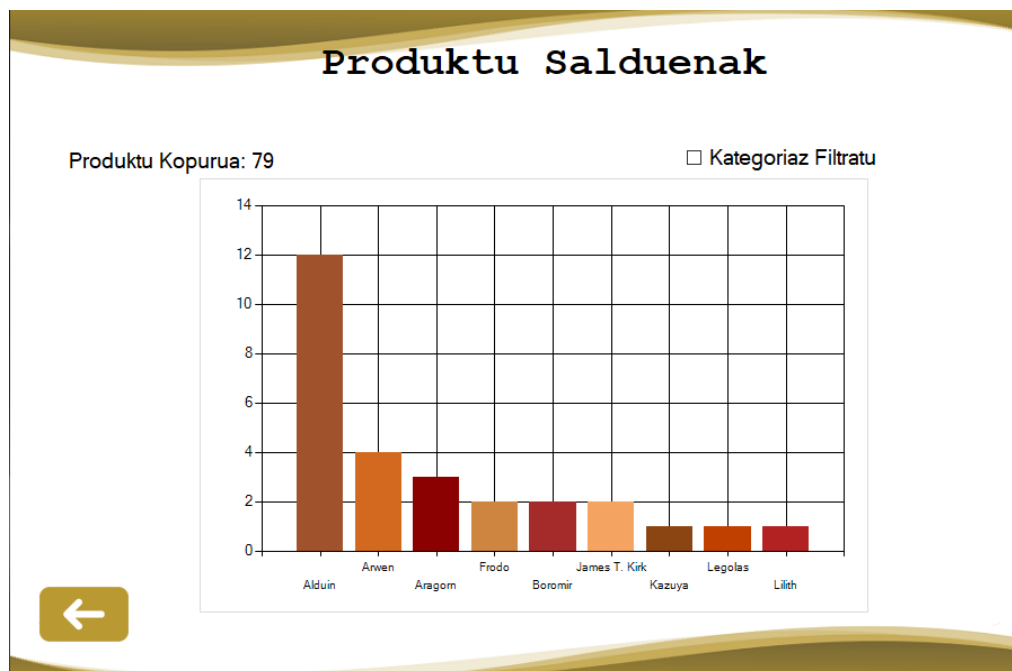
    this.bezeroBistaTableAdapter.Fill(this.bezeroErosketakDataSet.BezeroBista);

    BezeroChart.DataSource = this.bezeroBistaTableAdapter.GetDataBy().Select().Take(erakutsiBalio);

    BezeroChart.Series[0].YValueMembers = "zenbat";
    BezeroChart.Series[0].XValueMember = "izena";
    BezeroChart.DataBind();
}

1 referencia | Asier, Hace 6 días | 1 autor, 1 cambio
private void button1_Click_1(object sender, EventArgs e)
{
    this.Close();
}
```

Produktu Salduenak

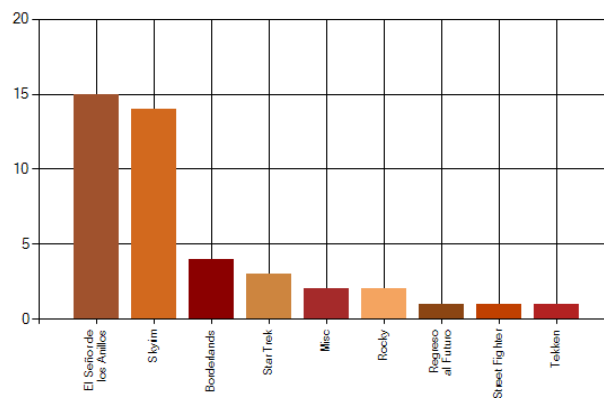


Produktu eta Erosketa tauletatik datuak jasotzen dituen bista batetik datuak jasotzen dituen grafiko bat daukagu. Grafiko hau kategoriaz filtratu daiteke:

Produktu Salduenak

Produktu Kopurua: 79

☒ Kategoriaz Filtratu



1 referencia | zengomax, Hace 9 días | 1 autor, 1 cambio

```
private void KategoriazFiltratu(object sender, EventArgs e)
{
    if (checkBox1.Checked == true)
    {
        this.produktuBistaTableAdapter.Fill(this.produktuSalduenakDataSet1.ProduktuBista);

        ProduktuChart.DataSource = this.produktuBistaTableAdapter.GetDataByCat().Select().Take(9);

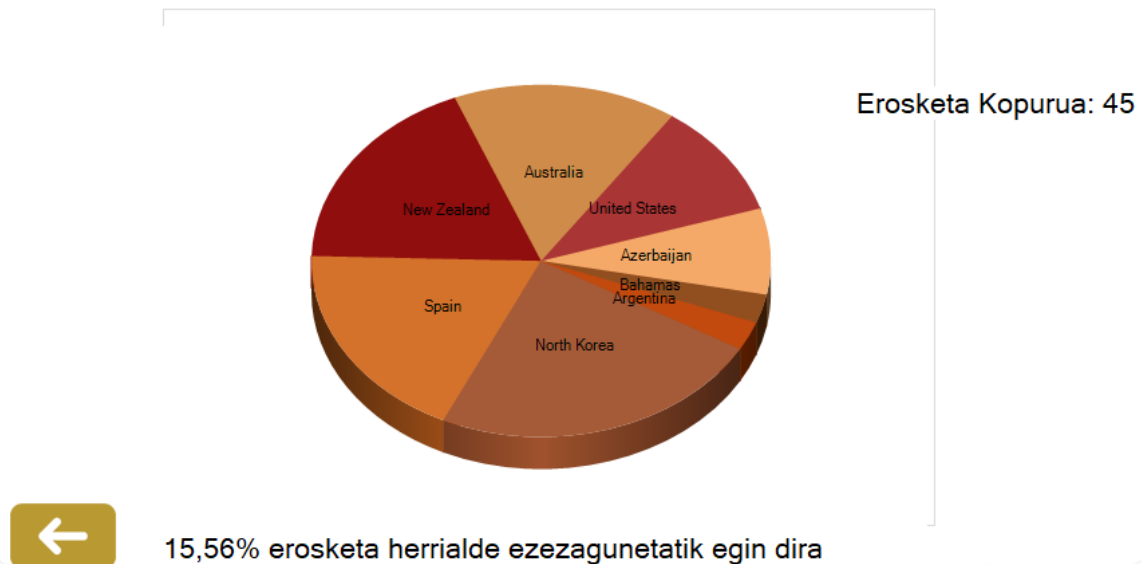
        ProduktuChart.Series[0].YValueMembers = "zenbat";
        ProduktuChart.Series[0].XValueMember = "categoria";
        ProduktuChart.DataBind();
    }
    else
    {
        this.produktuBistaTableAdapter.Fill(this.produktuSalduenakDataSet1.ProduktuBista);

        ProduktuChart.DataSource = this.produktuBistaTableAdapter.GetDataBy().Select().Take(9);

        ProduktuChart.Series[0].YValueMembers = "zenbat";
        ProduktuChart.Series[0].XValueMember = "name";
        ProduktuChart.DataBind();
    }
}
```

Herrialdeak

Gehien erosten duten herrialdeak



Produktu eta Erosketa tauletatik datuak jasotzen dituen bista batetik datuak jasotzen dituen grafiko bat daukagu. Beheko portzentaiak datu basean dauden herrialde guztietatik ezezagunak ehuneko zemat diren erakusten du

```
1 referencia | zengomax, Hace 2 horas | 3 autores, 4 cambios
private void Herrialdeak_Load(object sender, EventArgs e)
{
    // TODO: esta línea de código carga datos en la tabla 'herrialdeDataSet2.HerrialdeBista' Puede moverla o quitarla según sea necesario.
    this.herrialdeBistaTableAdapter1.Fill(this.herrialdeDataSet2.HerrialdeBista);
    this.herrialdeBistaTableAdapter.Fill(this.herrialdeDataSet.HerrialdeBista);

    HerrialdeChart.DataSource = this.herrialdeBistaTableAdapter.GetDataBy().Select();

    HerrialdeChart.Series[0].ValueMembers = "zenbat";
    HerrialdeChart.Series[0].XValueMember = "herrialdea";
    HerrialdeChart.DataBind();

    float ezezagunPortzentaia = (float.Parse(this.herrialdeBistaTableAdapter.FillByEzezaguna().ToString()) * 100) / float.Parse(funtzio.getErosketaKopurua());

    EzezagunakKant.Text = Math.Round(ezezagunPortzentaia, 2) + "% erosketa herrialde ezezagunetatik egin dira";
    ErosketaKopurua.Text = "Erosketa Kopurua: " + funtzio.getErosketaKopurua();
}
```

Etorkizunerako ildoak

Etorkizunean, zenbait gauza hobetzea gustatuko litzaiguke, antolaketari dagokionez, taldearen organizazioa eta komunikazioa hobetu, alde teknikoaren aldetik, multimedien sosliterekin atalen bat egitea eta visual-en beste hobekuntza bisualak egingo genituzke.

Ondorioak

Erronka honen bidez, ikasgelan landutako kontzeptuak barneratu ahal izan ditugu, eta gure kabuz android, figma, odoo, visual, Spring ikasi ahal izan dugu. Gure ustez, erronkak ikasteko modu onak dira, eta gure programazio-ezagutzak hobetzeko balio izan digute baita lan taldean nola egiten den lan joateko ikasten eta gure antolaketa hobetzeko.