

TSB



Innovation for a better future

2023

TXOSTENA

AURKIBIDEA

ERRONKAREN TXOSTENA.....	3
Erronkaren hasiera	3
Enpresaren atalak pentsatu:	3
ENPRESA KUDEAKETA SISTEMA	4
Odooren instalazioa eta konfigurazioa:.....	4
KOMERTZIALEN ANDROID APLIKAZIOA	7
Aplikazioaren diseinua pentsatu:.....	7
Aplikazioaren login pantalaila:	7
Aplikazioaren menu:.....	7
Aplikazioak dituen leiho desberdinak:	8
Datu baseko konexioa:	10
Datu base lokaleko sarrerak:.....	11
ENPRESA BARRUKO APLIKAZIOA	13
Aplikazioaren hasiera:	13
Datu baseko konexioak:	13
Datuen gestioa:	14
Aplikazioak dituen aukerak:	16
ENPRESA IDEIA	19
Forma juridikoa:	19
Canvas taula:	19
Balio proposamena:.....	20
Diru sarreren iturria:.....	20
Diru sarreren iturria:.....	20
Kostuen egitura:	21
SGA:	21
Merkatu ikerketa:	21
Lehiakideen deskribapena:.....	24
Hornitzaileak:	25

PYTHON APLIKAZIOA.....	25
-------------------------------	-----------

ERRONKAREN TXOSTENA

Erronkaren hasiera

Enpresaren atalak pentsatu:

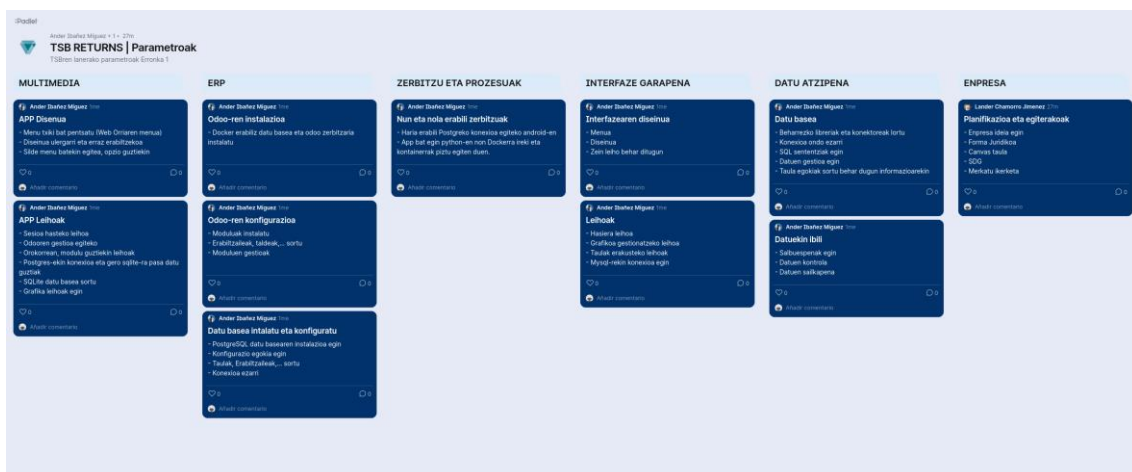
- Egia esan da, enpresa pentsatzea bastante erreza egin zitzagun. Gure enpresa sortuta gengan aurreko Erronka batengatik, orduan, logoa, izena, eslogana, koloreak... Pentzatuta genituen.
- Logoa (izena, eslogana...):
 - TSB Enpresa izena eta eslogana "Innovation for a better future".



- Koloreak: Bost kolore ditugu printzipalki, baina horietako bi txuria eta beltza dira, besteak Urdin berdea (**#00778B**), zeruko urdina (**#5BA4B6**) eta urdin pixkat argiago bat (**#89BCCC**).
- Horren ondoren, taldearen konpromizua zehaztu ditugu.

Erronkaren lehenengo pausoa:

- Erronkaren parametroak zehazten hasi gara, parametroak azkenean, Erronka egin beharrezko pausuak joan gara jartzen: (Clic egin ez gero irudian, handiago ikusi dezakezu)



- Hurrengo pausua, egunerokoari hasiera ematea izango zen, pixket aste bakoitzean zer egin behar dugun planifikatu, eta zeinek egin behar duen parte hori, biok, Lander edo Anderrek.

ENPRESA KUDEAKETA SISTEMA

Odooren instalazioa eta konfigurazioa:







- Odooren instalazioarekin hasteko Docker Desktop instalatu dugu ekipoa. Ondoren, Docker Hub erabilita behar ditugun komanduak berreskuratu eta exekutatu ditugu.
- Lehenengo, datu basea listo jarriko dugu, horretarako hurrengo komandoa exekutatu dugu "CMD" baten, gure kasuan izena eta pasahitza aldatu dugu.

```
C:\Users\ikaltamirapaag2>docker run -d -e POSTGRES_USER=tsb -e POSTGRES_PASSWORD=tsb -e POSTGRES_DB=postgres --name db postgres:15
99788aae1d873d990feee3b6d147efb656b5a663e3d8e00b4db29e736ead42dd
```

- Ondoren, Odoo erp-a jarriko dugu martxa, hurrengo komandoarekin.

```
C:\Users\ikaltamirapaag2>docker run -p 8069:8069 --name tsb --link db:db -t odoo
2023-10-19 06:25:31,324 1 INFO ? odoo: Odoo version 16.0-20230925
2023-10-19 06:25:31,325 1 INFO ? odoo: Using configuration file at /etc/odoo/odoo.conf
2023-10-19 06:25:31,325 1 INFO ? odoo: addons paths: ['/usr/lib/python3/dist-packages/odoo/addons', '/var/lib/odoo/addons/16.0', '/mnt/extra-addons']
2023-10-19 06:25:31,325 1 INFO ? odoo: database: tsb@172.17.0.2:5432
2023-10-19 06:25:31,499 1 INFO ? odoo.addons.base.models.ir_actions_report: Will use the Wkhtmltopdf binary at /usr/local/bin/wkhtmltopdf
2023-10-19 06:25:31,839 1 INFO ? odoo.service.server: HTTP service (werkzeug) running on 2110032a3bd5:8069
```

- Instalazioarekin bukatzeko, Docker-a horrela geratuko da.

<input type="checkbox"/>	Name	Image	Status	CPU (%)	Port(s)	Last started	Actions
<input type="checkbox"/>	db	postgres:15	Running	0.13%		10 minutes ago	  
<input type="checkbox"/>	tsb	odoo	Running	0.03%	8069-8069	7 minutes ago	  

- Instalazioa atzean utzita, orain Odoo-ra sartuko gara, beraren lehenengo konfigurazioa egiteko. Gure kasuan, gure datuak bete eta demo database aukeratu gabe utzi dugu (guk geure datu basea erabiliko dugu).



Warning, your Odoo database manager is not protected. To secure it, we have generated the following master password for it:

p2xa-kqxp-7xvd

You can change it below but be sure to remember it, it will be asked for future operations on databases.

Master Password:

Database Name:

Email:

Password:

Phone number:

Language:

Country:

Demo data: ☐


[Create database](#) or [restore a database](#)

- Behin, barruan egonda, erabiltzailearen izena aldatu dugu, gure enpresaren izena mantentzeko administradore giza.

Nombre
TSB

Dirección de Email [?]
tsbenpresa@gmail.com

Permisos de acceso Preferencias Seguridad de la cuenta



SALES

Ventas [?] Administrador

ACCOUNTING

Facturación [?] Administrador de Facturación

INVENTORY

Inventario [?] Administrador

Compra [?] Administrador

HUMAN RESOURCES

Empleados [?] Administrador

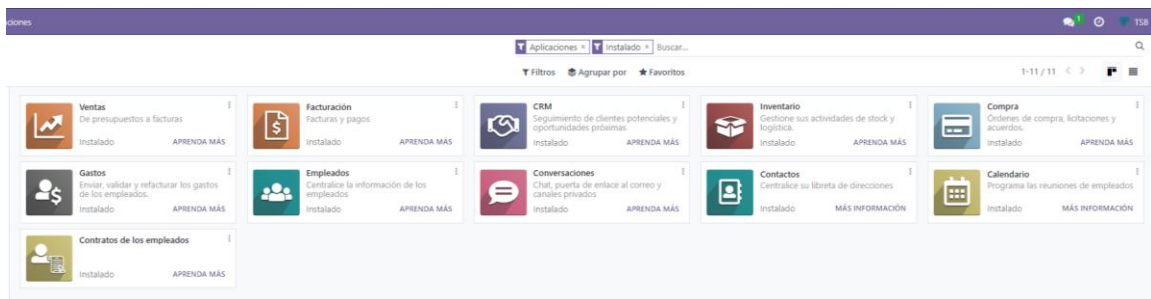
Contratos [?] Administrador

Gastos [?] Administrador

ADMINISTRATION

Administración Ajustes

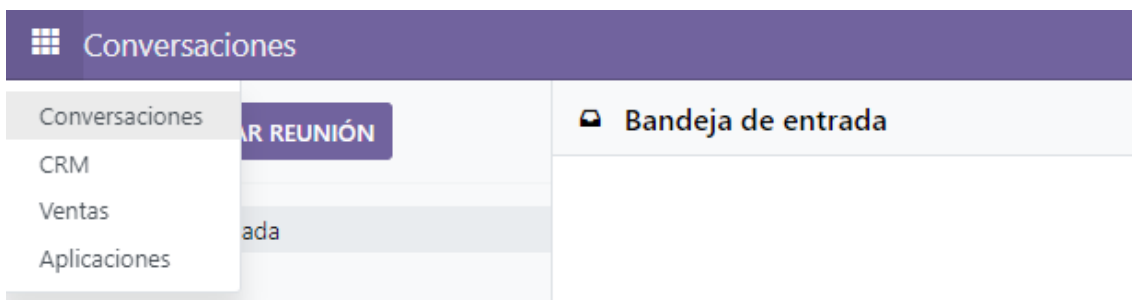
- Ondoren, beharrezko moduluen instalazioa egin dugu, hemen ditugun moduluen kaptura.



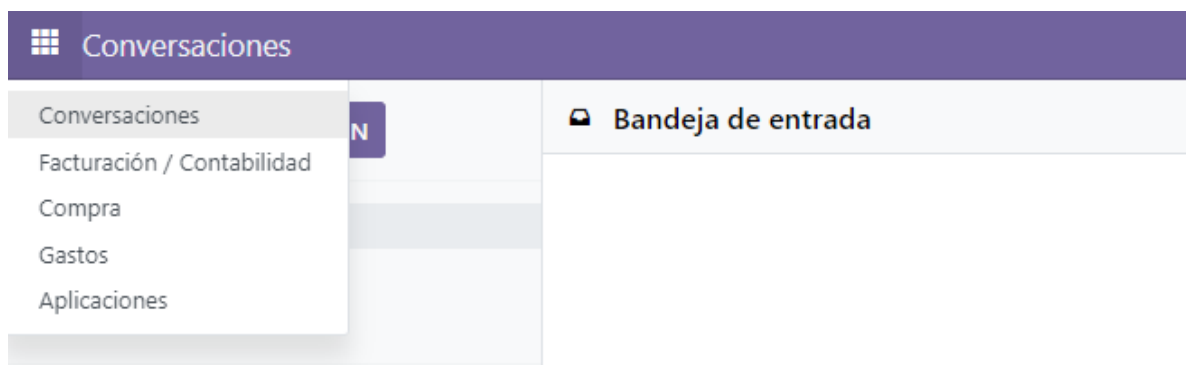
- Bukatzeko, erabiltzaileak sortu eta beraien pantailak/baimenak konfiguratuko ditugu.
- Horretarako, "komertzial" eta "kontabilitatea" erabiltzaileak sortu ditugu. Beraien pantailak konfiguratzeko, modulo batekin eta baimenekin egin dugu.

<input type="checkbox"/>	Nombre	Usuario	Idioma
<input type="checkbox"/>	Komertziala	komertziala@tsb.com	Spanish / Español
<input type="checkbox"/>	Kontabilitatea	kontabilitatea@tsb.com	Spanish / Español
<input type="checkbox"/>	TSB	tsbenpresa@gmail.com	Spanish / Español

- Hemen nola geratu den "KOMERTZIALA" erabiltzailearen pantaila



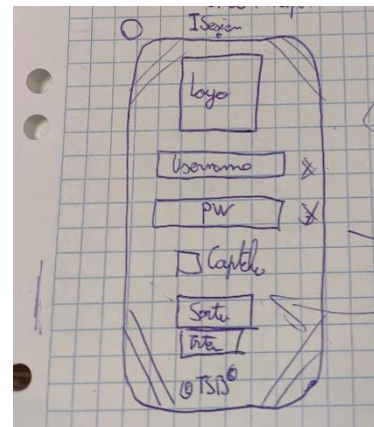
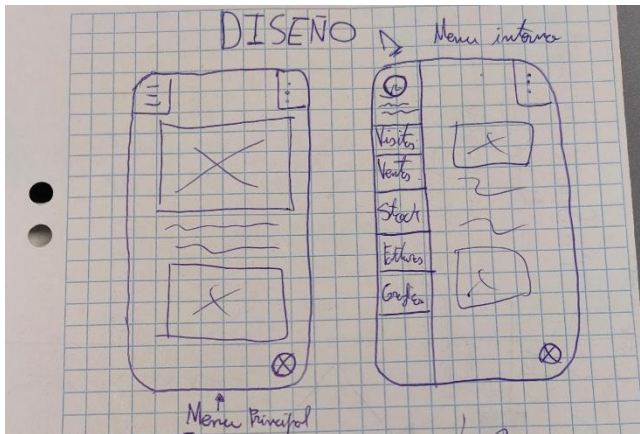
- Amaitzeko, "KONTABILITATEA" erabiltzailearen pantaila.



KOMERTZIALEN ANDROID APLIKAZIOA

Aplikazioaren diseinua pentsatu:

- Lehenengo pausua, aplikazioaren diseinua pentsatzea egin dugu, hau da guk pentsatutako diseinua.



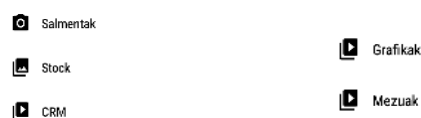
Aplikazioaren login pantalaila:

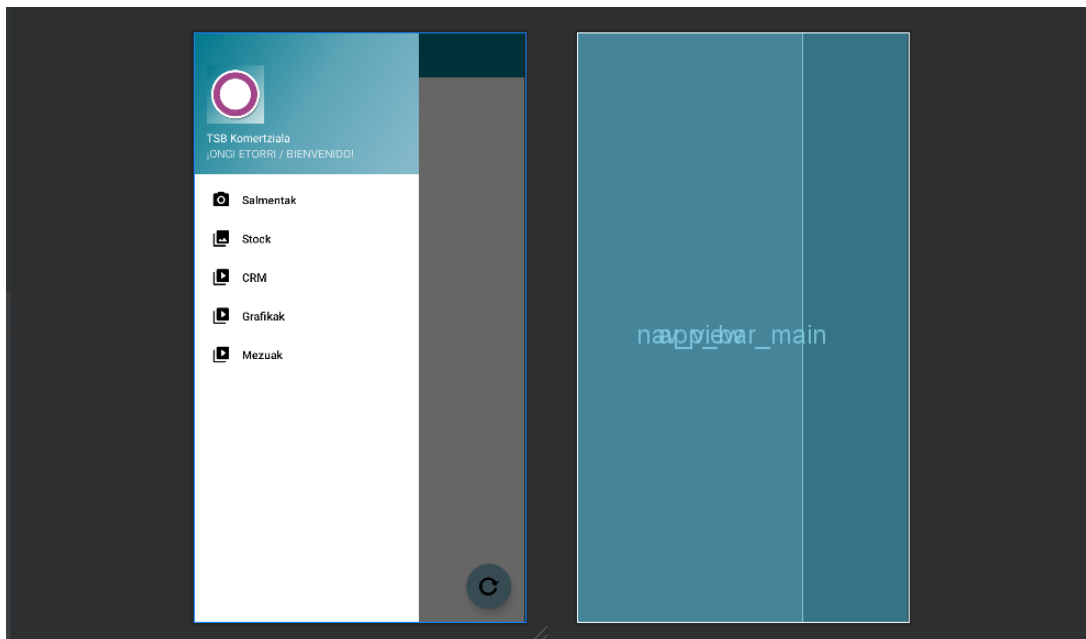
- Lehenengo, pentsatutako diseinua egin dugu. Azkenean gure diseinua horrela geratu da, beraren funtzioekin (hartz marka, user, password...)



Aplikazioaren menu:

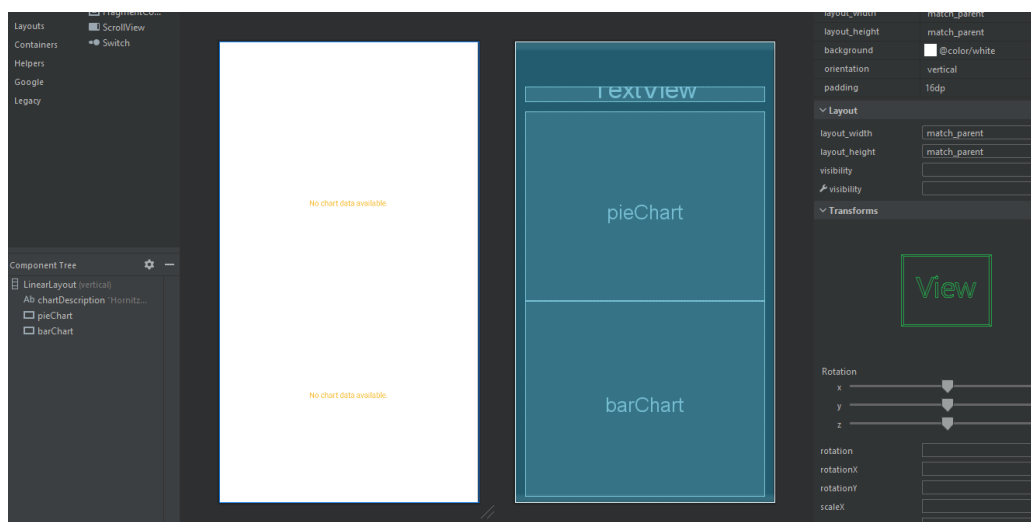
- Menuarren barruan izenak aldatu eta bi aukera berri sartu ditugu. Hau da aplikazioari utzi diogun menua. Salmentak, Stock, CRM, Grafikak eta Hornitzaileak.





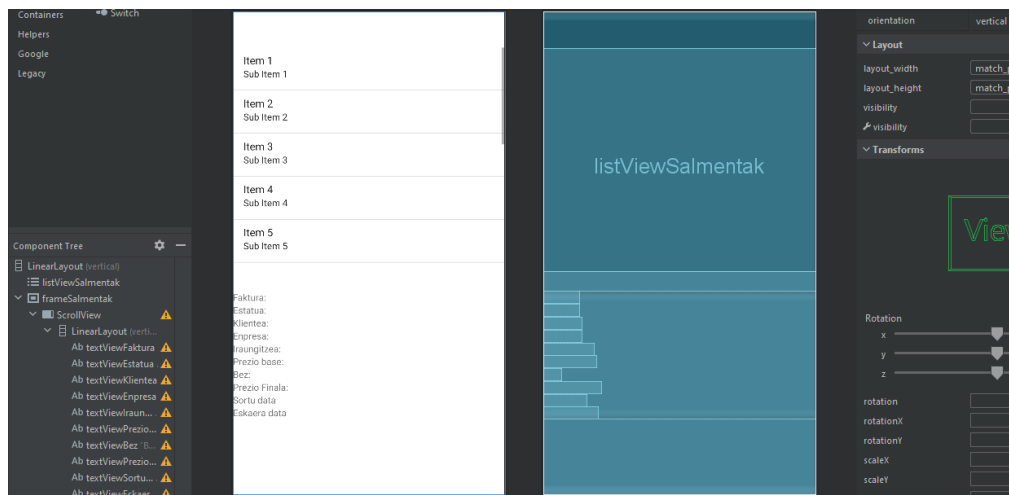
Aplikazioak dituen leiho desberdinak:

- Gure aplikazioak hainbat leihoa desberdin ditu martxan, adibidez, hainbat grafiko bistaratu ahal izateko leiho bat.
- Grafiko leihoa:

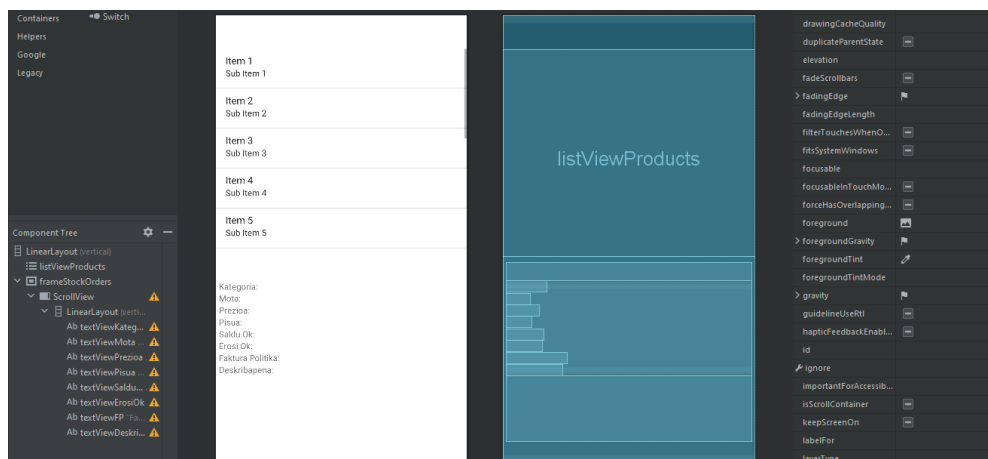


- Hurrengo leihoak, erabiltzaileak/kontableak datuak bistaratu ahal izateko egina dago: Salmentak, Stock, CRM eta Hornitzaileak.

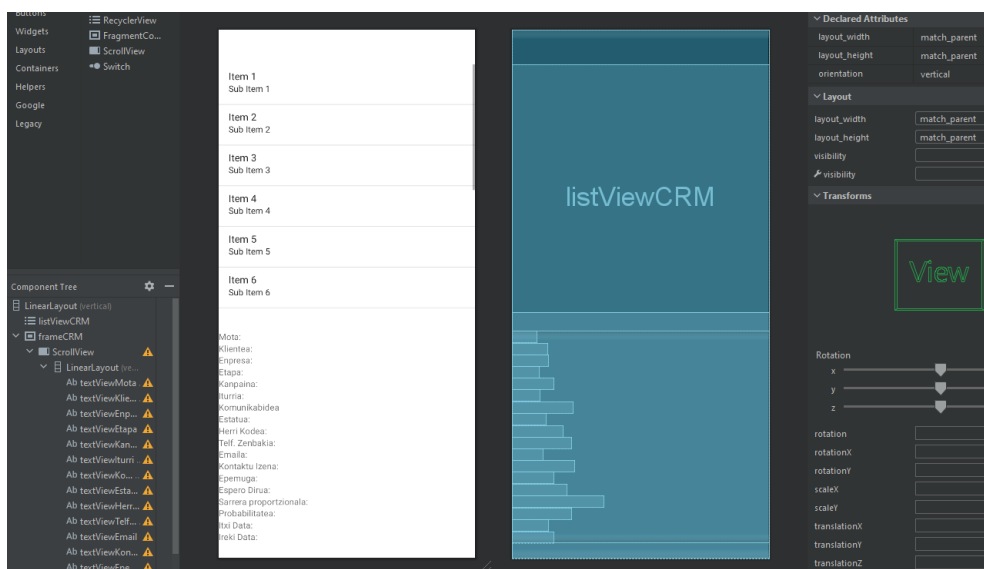
- Salmentak:



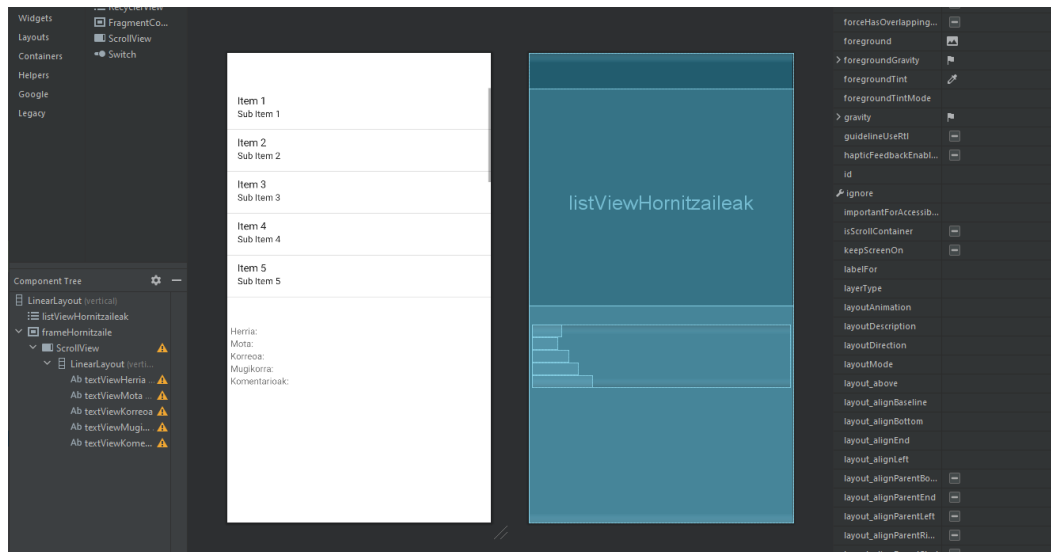
- Stock:



- CRM:

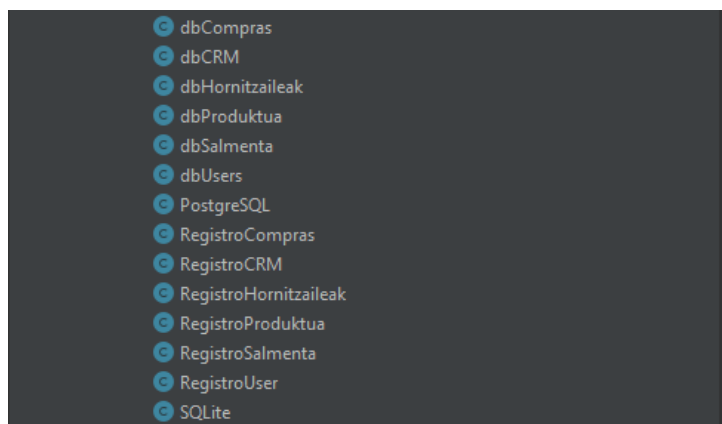


- Hornitzaileak:



Datu baseko konexioa:

- Datu basera konexioa ondo egiteko, hainbat klase sortu ditugu. Horietako bat "PostgreSQLConnection" da. Horrek egiten du konexioaren hasiera eta bukaera.
- Gero, hainbat klase ditugu, bat lokalean ibiltzeko datu basearen taulak sortzen dituen eta bestea datuak sartzen dituen barruan.



- Konexioaren irikiera egiteko orduan, problemak izan ditugu, baina hariaren erabilerarekin konpondu dugu.

```
Thread thread = new Thread(new Runnable() {
    @Override
    public void run() {
        try {
            Class.forName("org.postgresql.Driver");
            connection = DriverManager.getConnection(JDBC_URL, JDBC_USER, JDBC_PASSWORD);
            callback.onConnectionEstablished(connection);
        } catch (Exception e) {
            callback.onConnectionFailed(e);
        }
    }
});
```

- Aurreko danak egiteko, bi “implement” erabili behar izan ditugu, bat PostgreSQL-rako eta bestea, lokaleko datu basearentzako “SQLite”.

```
implementation ("org.postgresql:postgresql:42.2.5")
implementation ("androidx.sqlite:sqlite:2.2.0")
```

Datu base lokaleko sarrerak:

- Datu basera konexioa ondo egiteko, hainbat klase sortu ditugu. Adibidez, “Salmenta” taularen datuekin azalduko dugu.
- Lehenengo pausua datuak hartzea da, gure kasuan PostgreSQL-ko klase barruan hau dugu:

```
List<RegistroSalmenta> registros = new ArrayList<>();

try {
    Statement statement = connection.createStatement();

    String sql = "SELECT so.name, so.invoice_status, so.state, partner.name, company.name, so.validity_date, so.amount_untaxed, so.amount_tax, so.amount_total, so.create_date, so.date_order" +
        "FROM public.sale_order so" +
        "LEFT JOIN public.res_partner partner ON so.partner_id = partner.id" +
        "LEFT JOIN public.res_company company ON so.company_id = company.id;";

    ResultSet resultSet = statement.executeQuery(sql);

    while (resultSet.next()) {
        String izena = resultSet.getString(1);
        String faktura = resultSet.getString(2);
        String estatua = resultSet.getString(3);
        String klientea = resultSet.getString(4);
        String enpresa = resultSet.getString(5);
        String iraungitzea = resultSet.getString(6);
        String prezio_base = resultSet.getString(7);
        String bez = resultSet.getString(8);
        String prezio_finala = resultSet.getString(9);
        String sortu_data = resultSet.getString(10);
        String eskaera_data = resultSet.getString(11);

        RegistroSalmenta registro = new RegistroSalmenta(izena, faktura, estatua, klientea, enpresa, iraungitzea, prezio_base, bez, prezio_finala, sortu_data, eskaera_data);
        registros.add(registro);
    }

    resultSet.close();
    statement.close();
} catch (Exception e) {
    e.printStackTrace();
}

return registros;
```

- Ondoren, Salmenta datuak lortuta ditugula, Salmenta objektu lista betetzen dugu Salmenta datuekin.

```
2 usages
public RegistroSalmenta(String izena, String faktura, String estatua, String klientea, String enpresa, String iraungitzea, String prezio_base, String bez, String prezio_finala,
    String sortu_data, String eskaera_data) {
    this.izena = izena;
    this.faktura = faktura;
    this.estatua = estatua;
    this.klientea = klientea;
    this.enpresa = enpresa;
    this.iraungitzea = iraungitzea;
    this.prezio_base = prezio_base;
    this.bez = bez;
    this.prezio_finala = prezio_finala;
    this.sortu_data = sortu_data;
    this.eskaera_data = eskaera_data;
}
```

- Bukatzeko, SQLite datu base barruan sartzen ditugu datuak:

```
public dbSalmenta(Context context) {
    this.context = context;
    SQLite = new SQLite(context);

    // Datu berriak sartu aurretik, lehenengo datuak ezabatzen ditugu
    SQLiteDatabase db = SQLite.getWritableDatabase();
    db.delete(SQLite.TABLE_SALMENTA, "whereClause: null", "whereArg: null");
}

// Usage
public long salmentaSartu(String izena, String faktura, String estatua, String klientea, String enpresa, String iraugitza, String prezio_base, String bez,
    String prezio_finala, String sortu_data, String eskaera_data) {

    SQLiteDatabase db = SQLite.getWritableDatabase();

    ContentValues values = new ContentValues();

    // izena, faktura, estatua, klientea, enpresa, iraugitza, prezio_base, bez, prezio_finala, sortu_data, eskaera_data

    values.put("izena", izena);
    values.put("faktura", faktura);
    values.put("estatua", estatua);
    values.put("klientea", klientea);
    values.put("enpresa", enpresa);
    values.put("iraugitza", iraugitza);
    values.put("prezio_base", prezio_base);
    values.put("bez", bez);
    values.put("prezio_finala", prezio_finala);
    values.put("sortu_data", sortu_data);
    values.put("eskaera_data", eskaera_data);

    long id = db.insert(SQLite.TABLE_SALMENTA, null, values);

    db.close();

    return id;
}
```

- Hau dana egin aurretik, SQLite barruan "Salmenta" taula sortuta izan behar duzu:

```
// izena, faktura, estatua, klientea, enpresa, iraugitza, prezio_base, bez, prezio_finala, sortu_data, eskaera_data
sqliteDatabase.execSQL("CREATE TABLE " + TABLE_SALMENTA + "(" +
    "id INTEGER PRIMARY KEY AUTOINCREMENT," +
    "izena TEXT," +
    "faktura TEXT," +
    "estatua TEXT," +
    "klientea TEXT," +
    "enpresa TEXT," +
    "iraugitza TEXT," +
    "prezio_base TEXT," +
    "bez TEXT," +
    "prezio_finala TEXT," +
    "sortu_data TEXT," +
    "eskaera_data TEXT" +
    ")");
```

- Esan berra dao, hau dena, aktualizatu botoia sakatu eta PostgreSQLko datu basearekin konexioa duzunean, enpresa barruan egonda.

ENPRESA BARRUKO APLIKAZIOA

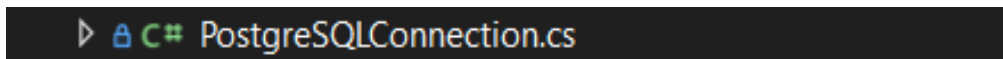
Aplikazioaren hasiera:

- Lehenengo, aplikazio osoaren diseinua pentsatu dugu.
- Esan beharra dugu, hasiera Figma erabiltzea pentsatu genuela, baina azkenean, ikusi genuen guk eskuz egiten ba dugu, gehiago ulertu eta kontrol gehiago izango genuela kodigo eta interfazeaz. Zergaitik, figman dena eginda etortzen da, eta zuk bakarrik "backend" programatu behar duzu, nahi dituzun datuak sartzeko, baina gure ustez, gutxiago kontrolatuko genuen horrela egin da baino.
- Ondoren, gure app-aren hasiera pantaila egin dugu Visual 2022 barruan.



Datu baseko konexioak:

- Hurrengo pausoa, datu baseko konexioa konfiguratzea izan da, bere klaseekin eta dena. Esan beharra dago, PostgreSQL erabili ahal izateko, Visual-ek erreztan digun NuGet bat instalatu behar izan dugu proiektuan.



```
// Parametroak
private string connectionString;
private NpgsqlConnection connection;

// Konexioa zehazteko datuak
private string host = "10.23.28.188:8068";
private string database = "tsb";
private string username = "tsb";
private string password = "tsb";

1 referencia
public PostgreSQLConnection()
{
    // konexioa irekitzeko string-a
    connectionString = $"Host={this.host};Database={this.database};Username={this.username};Password={this.password}";
    connection = new NpgsqlConnection(connectionString);
}

// PostgreSQLko Konexioa ireki ahal izateko
0 referencias
public void KonexioaIreki()
{
    if (connection.State == ConnectionState.Closed)
    {
        connection.Open();
    }
}

// PostgreSQLko Konexioa itxi ahal izateko
0 referencias
public void KonexioaItxi()
{
    if (connection.State == ConnectionState.Open)
    {
        connection.Close();
    }
}

// PostgreSQLko Konexioa lortu ahal izateko
5 referencias
public NpgsqlConnection getKonexioa()
{
    return this.connection;
}
```

- Gure kasuan, datu base printzipala PostgreSQL esan dezakegu dela, baino aplikazioa bera, MySQL gainean ibiltzen da. Zergaitik, PostgreSQL bakarrik datuak jasotzeko erabiltzen dugu, beste aldetik, segurtasuna aldetik igopena lortzen dugu.
- Hau da gure MySQL datu basera konexioa egin ahal izateko klasea:

```
6 {
7     private MySqlConnection connection;
8     private string connectionString;
9
10    2 referencias
11    public MySqlConnection()
12    {
13        // MySQL konexiorako behar ditugun datuak
14        //connectionString = "Server=10.23.28.188;Database=tsb_db;User ID=tsb;Password=tsb";
15        connectionString = "Server=localhost;Database=tsb_db;User ID=root;Password=Ander123";
16        connection = new MySqlConnection(connectionString);
17    }
18
19    // Datu baseko konexioa ireki
20    11 referencias
21    public void KonexioaIreki()
22    {
23        if (connection.State == ConnectionState.Closed)
24        {
25            connection.Open();
26        }
27    }
28
29    // Datu baseko konexioa itxi
30    5 referencias
31    public void KonexioaItxi()
32    {
33        if (connection.State == ConnectionState.Open)
34        {
35            connection.Close();
36        }
37    }
38
39    22 referencias
40    public MySqlConnection getKonexioa()
41    {
42        return this.connection;
43    }
44 }
```

- Hori bai, PostgreSQLrekin pasatzen den moduan, MySQLko libreria instalatu behar izan ditugu proiektuan, Visualek errezten digun "NuGet"-etatik.

Datuen gestioa:

- Orain, behin datu baseko konexioa eginda edukita, beharrezko datuak hartu eta konexioa baldin ba dugu PostgreSQLarekin, datuak eramán MySQL datu basera, gero programan zehar erabili ahal izateko.
- Taula bakoitzaren datuak ondo erabiltzeko, bakoitzari funtzioa bat egin diogu:

```

1 referencia
private void DataBaseDatuaKARGATU()...

// Hornitzaileak taula aktualizatzeko funtzioa
1 referencia
private void hornitzaileak_kargatu(String selectSQL)...

// Erosketa taula aktualizatzeko funtzioa
1 referencia
private void erosketa_kargatu(String selectSQL)...

// Gastua taula aktualizatzeko funtzioa
1 referencia
private void gastuak_kargatu(String selectSQL)...

// Produktuak taula aktualizatzeko funtzioa
1 referencia
private void produktuak_kargatu(String selectSQL)...

// Erabiltzaileak taula aktualizatzeko funtzioa
1 referencia
private void usuarios_kargatu(String selectSQL)...

// Datu baseko datuak lortzerakoan, konprobatu null al diren edo ez, eta beste mota batekoak direnak String moduan bihurtzeko
22 referencias
public static string konprobatuDatua(NpgsqlDataReader reader, int kolumnak)...

```

- Ikusten den moduan, funtzio bakoitzaren barruan, taula bakoitzaren datuak kontrolatzen ditugu, eta baita, null diren datuek errorerik ez emateko, funtzio bat dugu hori konprobatzeko.

- Taula barruko datuak bistaratzeko orduan, funtzioa hau erabili dugu:

```
// Datuak kargatu taularen barruan
try
{
    // Konexioa ireki
    konexioMySQL.KonexioaIreki();

    // Datuak eskuratzeko erabiliko den adaptera sortu
    using (MySqlDataAdapter adapter = new MySqlDataAdapter(selectSQL, konexioMySQL.getKonexioa()))
    {
        // DataTable bat sortu datuak gordetzeko
        DataTable dataTable = new DataTable();

        // DataTable-a bete kontsulta emaitzekin
        adapter.Fill(dataTable);

        // DataTable-a asignatu DataGridView-en DataSource gisa
        DGVtaulak.DataSource = dataTable;
    }
}
catch (Exception ex)
{
    MessageBox.Show("Ezin izan dugu datuak kargatu: " + ex.Message, "ARAZOA");
}
finally
{
    // Konexioa itxi zihoan
    konexioMySQL.KonexioaItxi();

    originalDataTable = ((DataTable)DGVtaulak.DataSource).Copy();
}
}
```

- Funtzio horrekin, datagridview-ko datu guztien kontrola ematen dugu. Behar dugun momentuan, bistaratu nahi ditugun datuak ikusteko.
- Ondoren, grafiko barruko datuak bistaratu ahal izateko, beste era batera egiten dugu, baino azkenean antzekoa da, datuak lortu, MySQL bidez, eta gero era bat edo bestea, grafiko barruan guk nahi bezala jartzea izango zen. Horretarako hainbat funtzio erabiltzen ditugu:

```
// Chart bakoizaren diseinuaren funtzioari hots egiteko funtzioa
1 referencia
private void KonfiguratuCharts()

// Chart-en diseinu konfigurazioa
2 referentziak
private void KonfiguratuChart(Chart chart, string serieIzena, SeriesChartType gpta)

// Chart barruan datuak kargatzeko chart-a
1 referencia
private void datuakKargatuChart()

// Chart barroko hornitzailea datuak kontrolatzeko
1 referencia
private void hornitzaileGrafikoaKargatu()

// Chart barroko salmenta datuak kontrolatzeko
1 referencia
private void salmentakGrafikoaKargatu()

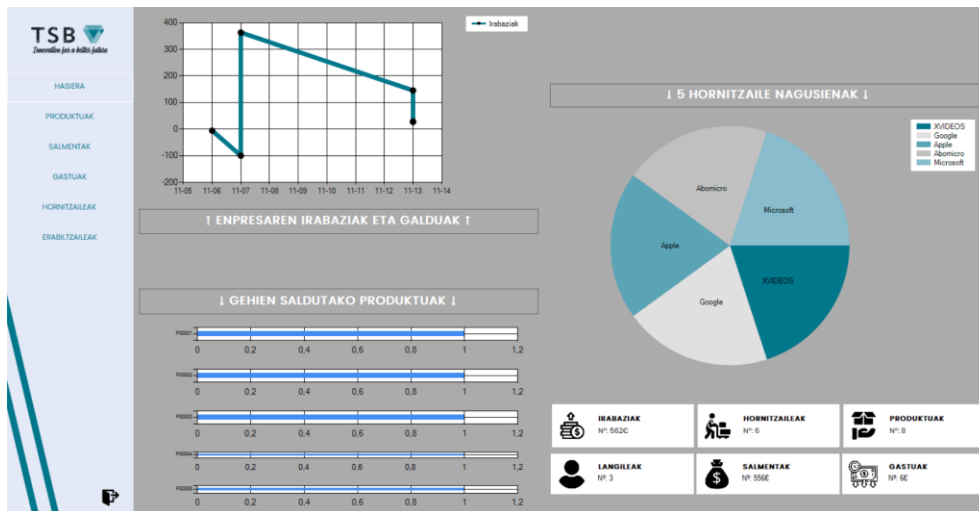
// Chart barroko produktu datuak kontrolatzeko
1 referencia
private void produktuGrafikoa()

// Behoko aldeak azaltzen diren laukien datuak kargatzeko
1 referencia
private void enpresaDatuakKargatu()
1 referencia
private void salmentaTotalakKargatu()
1 referencia
private void hornitzaileakKantitateaKargatu()
1 referencia
private void produktukantitateaKargatu()
1 referencia
private void irabaziakKantitateaKargatu()
1 referencia
private void langileakKantitatea()
1 referencia
private void gastuakKantitatea()
```

Aplikazioak dituen aukerak:

- Gure aplikazioak hainbat leiho ditu, hau da, menuraren barruan ditugu: Hasiera (hemen datu orokorrak bistaratu laike), produktuak (produktu taula bistaratu), Salmentak (Salmentak taula bistaratu), gastuak (gastuak taula bistaratu), Hornitzaileak (hornitzaile taula bistaratu) eta erabiltzaileak (erabiltzaile taula bistaratu).

- Hasiera:



- Produktuak:

TSB
Innovation for a better future

PRODUKTUAK

Bilatu/ak

ID	izena	kategoria	motu	prezioa	prezioa	salbu_ik	erosi_ik	fabrika_pozitua	deskribapena
670	COMM	All	comau	1.00	0.00	False	False	order	["ten_US": "p"]P...
671	TRANS & ACC	All	service	1.00	0.00	False	False	delivery	["ten_US": "p"]P...
672	MIL	All	comau	1.00	0.00	False	False	delivery	["ten_US": "p"]P...
673	FOOD	All	comau	1.00	0.00	False	False	order	["ten_US": "p"]P...
674	GIFT	All	comau	1.00	0.00	False	False	order	["ten_US": "p"]P...
675	Asu	All	service	150.00	0.00	True	True	order	["ten_US": "p"]P...
676	HAMB	Salable	comau	5.00	0.00	True	True	delivery	["ten_US": "p"]P...
677	EXP_GEN	Expenses	service	1.00	0.00	True	True	order	["ten_US": "p"]P...

- Salmentak:

TSB
Innovation for a better future

SALMENTAK

Bilatu/ak

ID	izena	estatus	fabrika	klartea	enpresa	prezio_jasoa	beir	prezio_totala	erakutsi_data	salmentak_data
313	P00001	cancel	no	XVIDEOS	TSB Enpresa	0.00	0.00	0.00	2023-11-07 08:4	2023-11-07 08:4
314	P00002	purchase	invoiced	Google	TSB Enpresa	300.00	63.00	363.00	2023-11-07 08:4	2023-11-07 08:4
315	P00003	purchase	no	Apple	TSB Enpresa	120.00	25.20	145.20	2023-11-13 09:0	2023-11-13 09:0
316	P00004	purchase	no	Abomiro	TSB Enpresa	25.00	3.50	28.50	2023-11-13 12:1	2023-11-13 12:1
317	P00005	purchase	no	Microsoft	TSB Enpresa	25.00	1.50	26.50	2023-11-13 12:1	2023-11-13 12:1

- Gastuak:

GASTUAK										
Bilatuak										
id	izena	statusa	enpresa	enbisa	produkua	ordendu_mota	ordatu_data	ordendu_data	desbidapena	ordendu prezioa
105	Hoody for your W...	done	TSB Enpresa	TSB	GIFT	over_account	2023-11-06 00:0...	2023-11-06 00:0...		6,00
210	Ratones	draft	TSB Enpresa	TSB	EXP_GEN	over_account	2023-11-07 00:0...		Ratones para em...	100,00

- Hornitzaileak:

HORNITZAILEAK						
Bilatuak						
id	izena	hema	maila	kontakoa	telefonoa	komentaria
117	XVIDEOS	DE	contact	xvideos@gmail.c...	+34 623 76 76 53	q>Best pomsite...
118	Abomoro	AO	contact	abomoro@gmail...	+34 623 76 76 53	q>AWG5-q>
119	Google	US	contact	google@gmail.com	+34 623 76 76 53	q>Google Inc. c...
120	Microsoft	AO	contact	microsoft@gmail...	+34 623 76 76 53	q>msd-q>
121	Apple	KH	contact	apple@gmail.com	+34 623 76 76 53	q>Patas de mer...
122	Mario	AF	contact	mario@gmail.com	+34 623 76 76 53	q>Explicar-q>

- Erabiltzaileak:

ERABILTZAILEAK			
Bilatuak			
id	erabiltzailea	email	enpresa
105	KOMERTZIALA	komentzale@hob...	TSB Enpresa
266	KONTABILITATEA	kontabilitate@ts...	TSB Enpresa
267	TSB	tsbempresa@gsn...	TSB Enpresa

ENPRESA IDEIA

- Lehenengo, gure enpresaren ideia azaldu dugu.

ENPRESA IDEIA

Gure enpresaren ideia honakoa da, orain pertsona gehio eta gehio bizitza osasuntzua eduki nahi dute, horretarako, aplikazio bat sortuko dugu. Aplikazioak km0-ko janari guztia agertuko zaizu mapa batean, bertan ikusi dezakezu bere prezioa, nun dagoen edota saltzen duen kantitatea, txat bat edukiko du saltzailearekin hizketan aritzeko, nola geratu edo ordaintzeko metodoa azalduko.

- Ondoren, gure enpresaren forma juridikoa egin behar genuen, Sozietate mugatua aukeratu genuen, enpresarentzako kudeaketa askoz errazagoa delako.
- Enpresaren hainbat datu agertzen dira ere bai.

Forma juridikoa:

- Sozietate mugatua: Honako forma juridikoa baldintza onenak eskaintzen digu gure enpresa motarentzat. Tramiteak errazago kudeatzea, sozio kopurua eta kapital kopurua, adibidez.
- 2 bazkide eta 7 langile.
- Kapital soziala: 100k euro
- Erantzukizuna: Mugitua
- Zergak: Sozietate gaineko zergak
- Gero, "Canvas" taula bete genuen, hainbat sekzio ditu, orduan tauletan banatuta eta bakoitzaren azalpen txiki bat jarri genuen.

Canvas taula:

- Funtsezko bazkideak: Hasieran 2 bazkide baina hainbat langileekin, laguntzailerekin ez da egongo hasiera batean, eta bazkideak 2 bakarrik. Salmenta puntua fisikoki ez dago, dena aplikazio baten bitartez izango da, zuk km0-ko janaria nahi badezu pertsona horrekin hitz egin beharko dezu prezioa adosteko edota lekua zure saltzeko.
- Funtsezko jarduerak:

1. Jatorrizko Produktuak Eskuratzea	Produktuak herriko jatorritik jasotzeko prozesua
2. Saltzaileri aurrekoa	Produktuak saltzeko erabiliko diren kanalak, denda fisikoa, online denda, etab...
3. Marketing eta Osasun-promozioa	Osasun eta kilometro zero produktuen berri emateko estrategiak
4. Bezeroen Arreta eta Aholkularitza Osasungarria	Bezeroei arreta ematea eta nutrizio-aholkularitza eskaintzea
5. Herriko nekazariarekin lankidetzak	Herriko nekazariak sustatzen eta baliatzen dituen lankidetzak

6. Kalitatea eta Segurtasun nahasia	Produktuen kalitatea eta segurtasuna bermatzen den prozesua
7. Ikerketa eta Garapen etenbea	Produktuak eta prozesuak etenbeak mantentzen dira eta garapena da

Balio proposamena:

- Pertsonak bizitza osasuntzua edukitzea eskaintzen dugu, km0 janaria erraz eta ondo erosteko, aplikazioa erraza izango da, ulertzeko eta erabiltzeko, gainera, atencion al cliente egongo da. Ikusi degunez ez dago konpetentzia direkto bat ideia honekin.

Bezeroekin harremana	Kontaktu zenbaki bat egongo da aplikazioan bertan, email bat eta formulario bat zure galdera bidaltzeko.
Bezero segmentua	Edade mugarik ez dago, persona bat mugikor batekin instalatu dezake gure aplikazioa.
Kanalak	Web orria, irratien publizitatea, publizitatea sare sozialetan, eta abar.

Diru sarreren iturria:

Afiliatu programa	Kanpoko bazkideek zure plataforma produktuak sustatzeko zerbitzuak, eta erreferentzia bidez eragindako salmenta bakoitza komisio batekin ordaintzen dute programa afiliatua ezarri.
Ekitaldi eta azokak online	Internet bidezko ekitaldiak edo azokak antolatu, non saltzaileek beren produktuak sustatu eta saltzeko aukera izango dute, eta parte hartzeko ordaindu egin beharko dute.
Produktu propioen salmenta	Zure plataforma produktuak edo merchandisinga salgai jartzea.
Ekitaldi eta azokak online	Ekitaldiak eta azokak internet bidez antolatu, non saltzaileek beren produktuak sustatu eta saltzeko aukera izango dute, eta parte hartzeko ordaindu egin beharko dute.

Diru sarreren iturria:

Garapen taldea	Programatzaileak, diseinatzaileak, eta besteak
Infraestruktura teknologikoa	Hosting, datu baseak, segurtasun sistema, eta baliabide teknologikoak
Merkatuan sartzea	Publizitatea, markinari buruzko estrategia, eta bestelakoak

Ekipo komertziala	Bezeroekin eta hornitzaileekin negoziatzea eta harremana mantentzea
-------------------	---

Kostuen egitura:

- Zerbitzaria, langileak eta publizitatea gehien bat.

SGA:

- SDG 2: Gose Zero: Merkatuak lokalak saltzea eta produkzioa bultzatuz, elikagaiak freskoak eta osasuntsuak sortzea.
- SDG 12: Produkzioa eta Kontsumo Arduratsua: Elikadura lokal eta jasangarriaren ekoizpena eta kontsumoa hobetzeko erantzunkizun eta jasangarritasun praktikak bultzatu.
- SDG 8: Enplegu Dignoa eta Hazkundera Ekonomikoa: Gure merkatuak nekazari lokalak bultzatzeko eta nekazaritzarako eta ekoizpenak enplegu aukerak hazkundera ekonomikoa bultzatu ahal izango dugu.
- SDG: Industria, Berrikuntza eta Azgitura: Elikadura lokalak eta jasangarria sustatzeko, azpiegitura lokalak garatzeko ekarpena egin, hobekuntza eta hobekuntza bultzatuz.
- SDG 11: Hirigintza eta Jasangarritasuneko Okertzea: Elikagai lokalak eta jasangarriak eskuratzeko hiriak eta komunitateak jasangarriagoak sortzeko lagunduko dugu, karbono-huella murriztuz eta nekazari lokalak babestuz.
- SDG 17: Helburuak Lortzeko Elkartasuna: Nekazariekin, gizarte zibileko erakundeekin eta beste eragileekin lankidetzan aritzeko lokalak eta jasangarriak bultzatzen lagunduko dugu.

Merkatu ikerketa:

- Zuzentzen garen merkatuko ezaugarriak: Nola gaur egun geroz eta jende gutxiago janari ez osasuntzua jaten duen, gure aplikazioaren ideia nagusia jendea janaria osasuntzua erostea izango da, era erraz eta azkar batean.
- Geroz eta jende gehiago, gaztiak baten bat, erre, eran eta drogatu egiten dira, eta ez dute janari osasuntzua jaten, obesitatea geroz eta pisu gehiago hartzen hari da urte hauetan. Hona hemen web orri bat hau azaltzen duena:

<https://cuidateplus.marca.com/familia/adolescencia/diccionario/malnutricion-adolescencia.html>

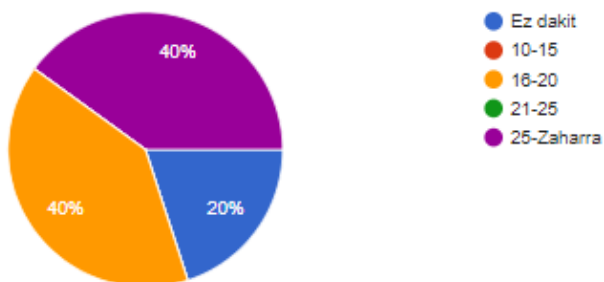
- Orduan, obesitatea geratzeko eta jendea janari osasuntzua jaten hasteko, aplikazio hau sortu dugu.
- Bezeoen analisia: Gure aplikazioa jende gazteari zuzenduta egongo da, baina berez edozein pertsona erabili dezake. Aplikazioa doainekoa izango da. Erosketak egitea oso errexa eta intuitiboa izango da, probedorea selekzionatu, produktua aukeratu eta berekin geratu ordaintzeko eta zure janaria hartzeko. Urteko garai ezberdinetan

tenporadako janaria salmentan jarriko da, besteak beste. Formulario bat egin dugu datuak ateratzeko, hona hemen erantzunak.

Zenbat urte dituzu?

 Copiar

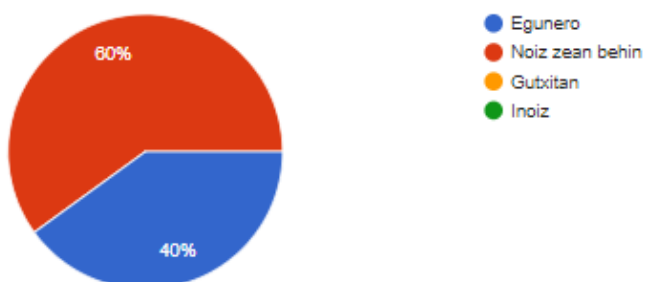
5 respuestas



Janari osasuntzua jaten duzu?

 Copiar

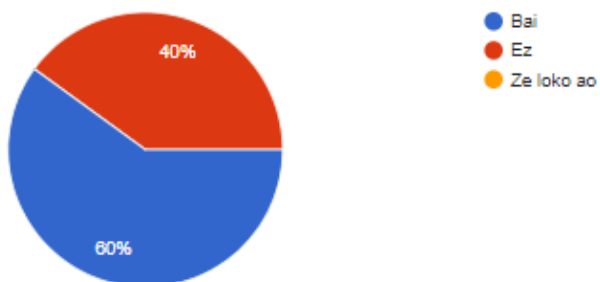
5 respuestas



Janaria online erosiko zenuke?

 Copiar

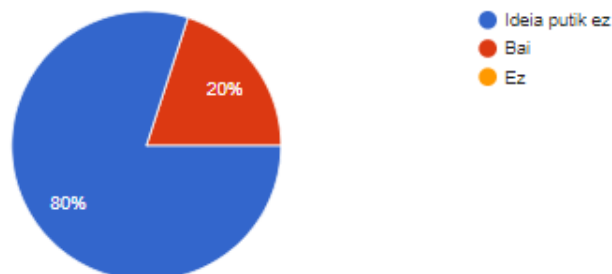
5 respuestas



Zure hirian km0-ko denda asko daude?

 Copiar

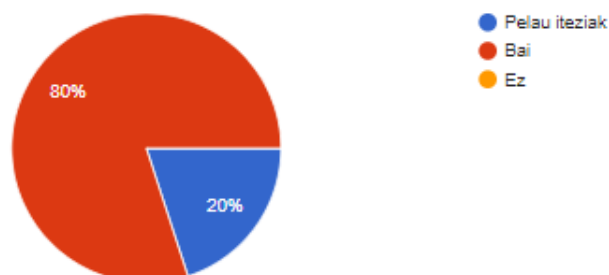
5 respuestas



Zure ustez gende gaztia bizitza osasuntzua eraman beharko luke?

 Copiar

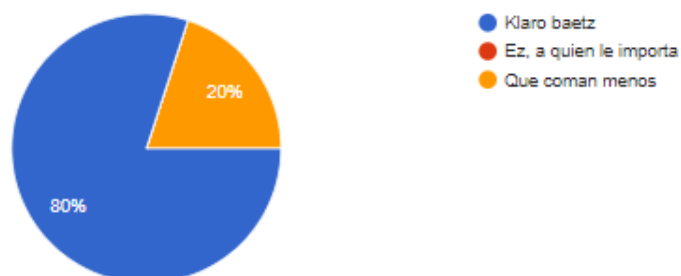
5 respuestas



Obesitatea gaur egun arazo larria da?

 Copiar

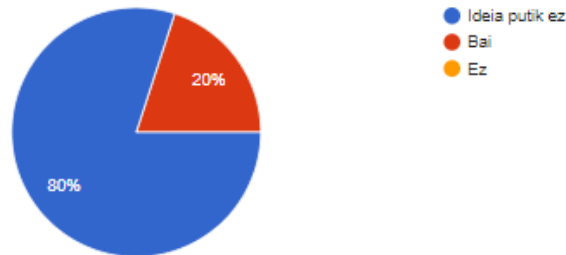
5 respuestas



Zure hirian km0-ko denda asko daude?

 Copiar

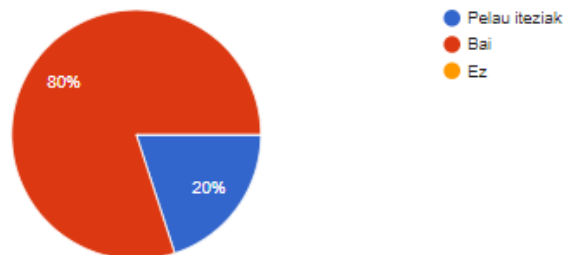
5 respuestas



Zure ustez gende gaztia bizitza osasuntzua eraman beharko luke?

 Copiar

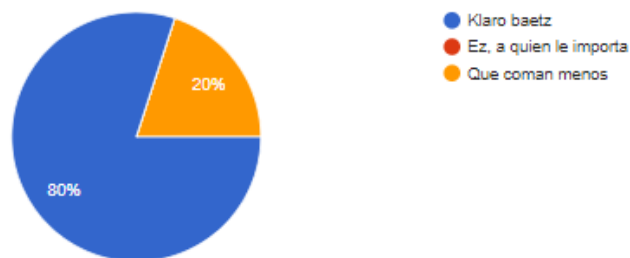
5 respuestas



Obesitatea gaur egun arazo larria da?

 Copiar

5 respuestas



Lehiakideen deskribapena:

Aplikazio edo web gune bat bereziki ez dago ideia honekin, orduan lehiakide asko ez daude, gure lehiakide antzekoena jatetxeetan eta tabernetan uzten den janaria erosteko aplikazioak, sobratzen diren pintxoak edo janaria dira; izan ere, erosketagatik diru asko gastatu nahi ez duen jendeak ez botatzeko, baina km0 janaria saltzeko eta erosteko aplikaziorik ez dago, eta, beraz, egin duten oso gutxi horien artean egongo gara, Espainian behin betiko.

Hornitzaileak:

Guk, berez, ez dugu hornitzailerik, saltzen dutenak baserritarrak direlako edo janaria km-0 duen jendea saldu nahi duelako, app-ean edo web orrian jartzen duelako eta jendeak ikusi eta eros dezakeelako, eta gero hartzera joan. Hornitzaile bat edukitzera irits gaitezke, baldin eta gu, enpresa bezela, gure planetan dagoen janaria km-0 saltzen jarriko bagina, baina oraingoz ez.

PYTHON APLIKAZIOA

- Lehenengo, "dockerIreki()" funtzioa dugu, docker hub irekitzeko funtzioa du, honek prozesuekin exekutatzen du.

```
def dockerIreki():
    # Docker irekitzeko komandoa
    docker_desktop_command = "Start-Process -NoNewWindow -FilePath 'C:\\Program Files\\Docker\\Docker\\Docker Desktop.exe'"

    try:
        # Try-catch bat docker irekitzen danean
        subprocess.run(["powershell", "-Command", docker_desktop_command], shell=True, check=True)
        print("Docker ondo ireki da.")
    except subprocess.CalledProcessError as e:
        print(f"Errore bat gertatu da Docker irekitzerakoan: {e}")
```

- Ondoren, bezero bat sortu dockerrera sartzeko, kontenedore guztiak listatu eta bukle batekin berifikatu egiten du beraien estatua, ikusten du batenbat linkeatuta dagoen host-era, db datu basea badago ikusten du bere estatua eta itzalduta badago lehenengo hori pizten du arazoak ez edukitzeko besteekin, db pizterakoan .start-ekin bestiak pizten dira.

```
def kontenedoreGuztiakPiztu():
    # Docker kliente bat sortu egiten du
    client = docker.from_env()

    # Kontainer lista bat lortzen du
    all_containers = client.containers.list(all=True)

    if not all_containers:
        print("Ez daude kontenedoreak.")
        return

    # Kontenedore bakoitza pizten du
    for container in all_containers:
        # Bere estatua konprobatzen du
        if container.status == 'exited':
            # Kontenedorea binkulatuta badago ikusten du
            links = container.attrs.get('HostConfig', {}).get('Links', [])

            # Berifikatu egiten du
            if links and any('/db' in link for link in links):
                # /db kontenedorea lehenengo pizten du, erroreak ez edukitzeko
                db_container = client.containers.get('db')
                if db_container.status == 'exited':
                    print(f"/db kontenedorea pizten arazoak ez edukitzeko besteekin.")
                    db_container.start()

            # Kontenedore aktuala pizten du
            print(f"Kontenedorea pizten: {container.name}")
            container.start()
        else:
            print(f"{container.name} kontenedorea piztuta dago.")
```

- Bukatzeko, main barrotik hots egiten diogu bi funtzio hauei.

```
if __name__ == "__main__":  
    dockerIreki()  
    kontenedoreGuztiakPiztu()
```