

UNIVERSIDAD SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA
ESCUELA DE CIENCIAS Y SISTEMAS
LABORATORIO SISTEMAS ORGANIZACIONALES
Y GERENCIALES 2
SECCIÓN N
AUXILIAR: JOSÉ LUIS REYNOSO TIU

PROYECTO 1

Nombre

Kelly Mischel Herrera Espino
Alberto Gabriel Reyes Ning

Carné:

201900716
201612174

ÍNDICE

Objetivos	1
Objetivo General	1
Objetivos específicos	1
Resumen	2
Manual 1	3
Sistema Odoo	3
Sección 1: Proceso de instalación:	3
Paso 1: Crear VM	3
Paso 2: SSH al VM	4
Paso 3: crear docker-compose yamI	4
Paso 4: Levantar el contenedor	5
Paso 5: Visita el ip externa del VM	6
Sección 2: Funcionamiento de los módulos	6
Módulos instalados	7
Vista general de los módulos:	7
Sección 3	8
Empleados	8
Clients (Clientes)	12
Providers (Proveedores)	12
Purchase (Compras)	12
Sales (Ventas)	15
Invoicing (Facturas)	17
Proceso de generar el información	17
Seccion 4	20
RPA Automatización 2	20
Compras a proveedores de la empresa	23
Manual 2	24
Diagrama de flujo UIPath	24
Diagrama Compras a proveedores de la empresa	26
Diagrama Ventas de productos	27
Diagrama de Flujo de proceso de la empresa desde que entra un producto hasta que este sea vendido	28
Diagrama de compra web	29

Objetivos

Objetivo General

Realizar una implementación del sistema de gestión empresarial Odoo para integrar y automatizar procesos clave de ventas, CRM, sitio web, recursos humanos y comunicación interna entre módulos de la empresa.

Objetivos específicos

- Configurar los módulos para gestionar el proceso de la empresa y poder realizar un control y movimiento de los inventarios.
- Implementación del sistema de gestión a través de modelos de integración como Purchase y Employees, con el fin de organizar la carga y búsqueda de documentos.
- Mejorar la eficiencia operativa mediante el flujo de automatización.

Resumen

Con la implementación de Odoo adaptado a lo solicitado y la solución RPA en MegaCity, se estableció un flujo automatizado donde los archivos Excel generados por el departamento de ventas, que contienen hojas con datos de clientes y proveedores, son identificados por un RPA. Este extrae automáticamente la información relevante de las hojas llamadas “clientes” y “proveedores” y la carga en una base de datos centralizada con tablas específicas. Posteriormente, estos datos se integran con los módulos de Odoo como CRM, Sales, Invoicing y Purchase, permitiendo una gestión más eficiente de contactos, órdenes y facturación. Paralelamente, documentos clave como facturas y contratos son cargados, etiquetados y organizados en el módulo de gestión documental de Odoo, garantizando el cumplimiento normativo y una fácil localización.

Manual 1



Sistema Odoo



Odoo es un sistema de tipo ERP(Enterprise Resource Planning) de código abierto el cual alberga una amplia gama de aplicaciones empresariales para gestionar diferentes áreas de una empresa.

Sección 1: Proceso de instalación:

Para la instalación del sistema se debe de seguir los pasos que se describen a continuación:

Paso 1: Crear VM

Se debe crear una VM.

VM instances							
Filter Enter property name or value							
<input type="checkbox"/> Status	Name ↑	Zone	Recommendations	In use by	Internal IP	External IP	Connect
<input type="checkbox"/> 	odoo	us-east1-c			10.142.0.2 (nic0)	35.211.154.166  (nic0)	SSH ▾ ⋮

Paso 2: SSH al VM

```
C:\Users\Codeprentice\AppData\Local\Google\Cloud SDK>gcloud compute ssh odoo

Codeprentice@odoo: ~
System information as of Wed Apr 30 21:15:29 UTC 2025

System load:  0.07          Processes:            136
Usage of /:   36.4% of 19.20GB Users logged in:        0
Memory usage: 40%          IPv4 address for ens4: 10.142.0.2
Swap usage:   0%

* Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
  just raised the bar for easy, resilient and secure K8s cluster deployment.

  https://ubuntu.com/engage/secure-kubernetes-at-the-edge

Expanded Security Maintenance for Applications is not enabled.

2 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

1 additional security update can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

*** System restart required ***
Last login: Tue Apr 22 16:23:30 2025 from 181.209.201.128
Codeprentice@odoo:~$
```

Paso 3: crear docker-compose yaml

Se deber de crear un docker-compose yaml para levantar odoo y db

```

1  version: '3'
   ▶ Run All Services
2  services:
   ▶ Run Service
3    db:
4      image: postgres:13
5      environment:
6        POSTGRES_DB: postgres
7        POSTGRES_USER: odoo
8        POSTGRES_PASSWORD: odoo
9      volumes:
10     - odoo-db-data:/var/lib/postgresql/data
11
   ▶ Run Service
12    web:
13      image: odoo:16
14      depends_on:
15        - db
16      ports:
17        - "8069:8069"
18      environment:
19        HOST: db
20        USER: odoo
21        PASSWORD: odoo
22      volumes:
23        - odoo-web-data:/var/lib/odoo
24        - ./addons:/mnt/extra-addons
25
26    volumes:
27      odoo-web-data:
28      odoo-db-data:
29

```

Paso 4: Levantar el contenedor

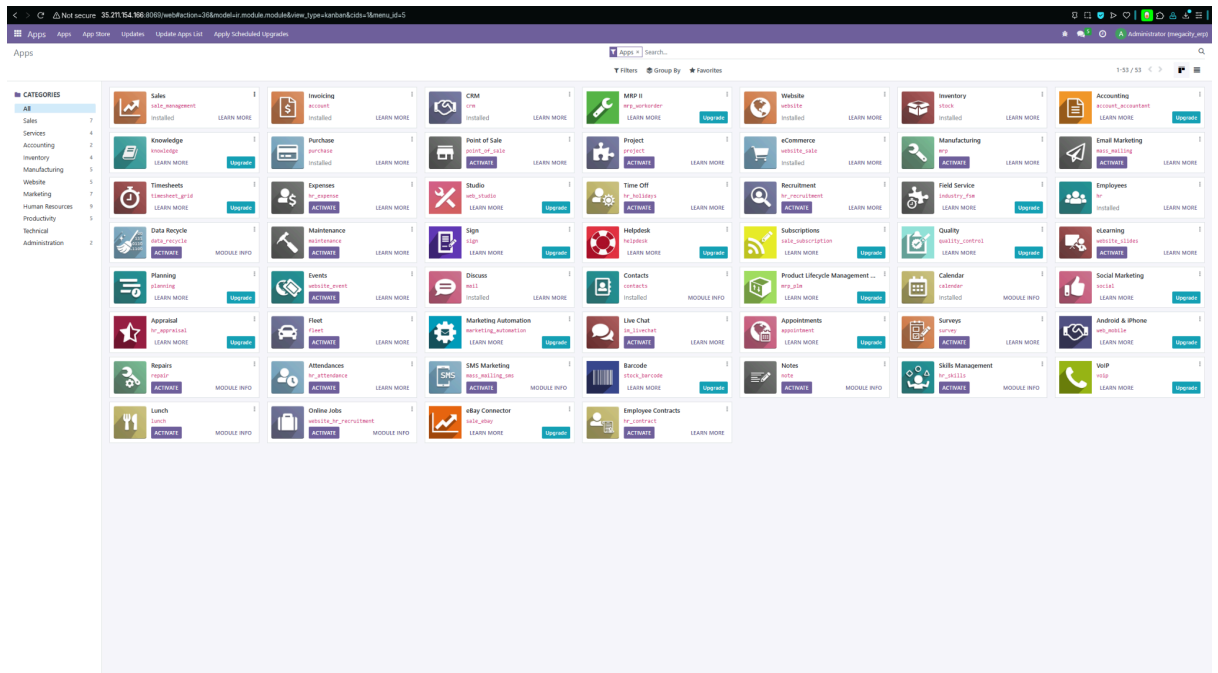
Se debe de levantar el contenedor con odoo y la base de datos.

```

Codeprentice@odoo:~$ sudo docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS
NAMES
#728ddb2e690   odoo:16       "/entrypoint.sh odoo"   8 days ago    Up 8 days    0.0.0.0:8069->8069/tcp, :::8069->8069/tcp, 8071-8072/t
cp   odoo_project_web_1
160e9cdd2b7f   postgres:13   "docker-entrypoint.s..." 8 days ago    Up 8 days    5432/tcp
odoo project_db_1

```

Paso 5: Visita el ip externa del VM



Sección 2: Funcionamiento de los módulos




A continuación se detalla el funcionamiento de los módulos utilizados durante el desarrollo del flujo del sistema.

Módulos instalados

Vista general de los módulos:

The screenshot displays the 'Apps' section of the Megacity ERP system. The top navigation bar includes links for 'Apps', 'App Store', 'Updates', 'Update Apps List', and 'Apply Scheduled Upgrades'. The user is logged in as 'Administrator (megacity_erp)'. The main area shows a grid of installed modules, each with an icon, name, sub-name, status, and a 'LEARN MORE' link. A left sidebar lists categories with counts.

CATEGORIES	Module Name	Sub-name	Status	Action
All	Sales	sale_management	Installed	LEARN MORE
Sales	Invoicing	account	Installed	LEARN MORE
Services	CRM	crm	Installed	LEARN MORE
Accounting	Website	website	Installed	LEARN MORE
Inventory	Inventory	stock	Installed	LEARN MORE
Manufacturing	eCommerce	website_sale	Installed	LEARN MORE
Website	Employees	hr	Installed	LEARN MORE
Marketing	Purchase	purchase	Installed	LEARN MORE
Human Resources	Discuss	mail	Installed	LEARN MORE
Productivity	Contacts	contacts	Installed	MODULE INFO
Technical	Calendar	calendar	Installed	MODULE INFO
Administration				

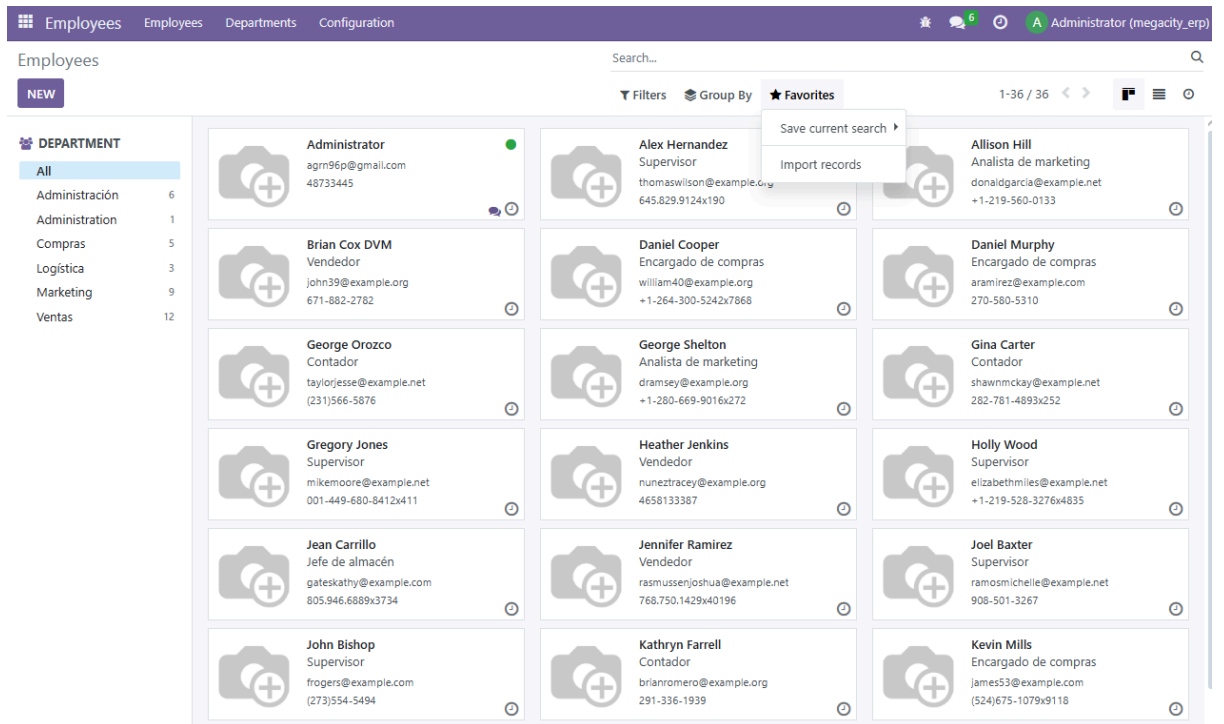
Módulo	Descripción
Sales 	Gestiona cotizaciones, órdenes de venta y el flujo de facturación a clientes. Incluye funciones como firma electrónica y seguimiento automatizado.
Invoicing 	Administra facturas de clientes y proveedores, pagos, asientos contables y cálculo de impuestos. Se integra con ventas y compras.
CRM 	Permite gestionar oportunidades, clientes potenciales y el embudo de ventas para dar seguimiento a interacciones y cerrar negocios.
Website	Permite crear y personalizar el sitio web público con un editor visual. Se integra con eCommerce y herramientas de análisis.

	
Inventory 	Controla niveles de inventario, entradas/salidas de productos, almacenes y trazabilidad. Se conecta directamente con ventas y compras.
Purchase 	Administra pedidos a proveedores, acuerdos de compra, recepción de productos y facturas de proveedores. Permite automatizar reabastecimientos.
eCommerce 	Permite vender productos en línea mediante una tienda virtual conectada al inventario, precios y módulo de ventas. Incluye carrito y pagos.
Employees 	Centraliza la gestión de empleados, puestos, departamentos y contratos. Base para otros módulos de recursos humanos.
Discuss 	Módulo de mensajería interna para comunicación entre empleados. Soporta menciones, canales y conversaciones organizadas.
Contacts 	Almacena todos los registros de contactos (clientes, proveedores, empleados, etc.). Es la base para enlazar entidades entre módulos.
Calendar 	Administra reuniones y eventos. Se integra con CRM, Ventas y Empleados para coordinar horarios y actividades.

Sección 3

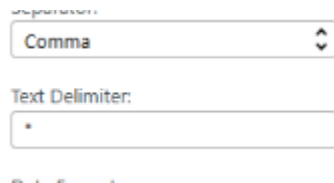
Empleados

Ingresar al módulo de empleados



Seleccionar el archivo **csv** de empleados y colocar el delimitador de la columna.

A continuación se muestra un ejemplo:



Employees

Employees

Departments

Configuration

6

Administrator (megacity_erp)

Employees / Import a File

IMPORT

TEST

LOAD FILE

CANCEL

Imported file

empleados_mega_city.csv

Use first row as header

Formatting

Encoding:

utf-8-sig

Separator:

Comma

Text Delimiter:

,

Date Format:

Datetime Format:

Thousands Separator:

Comma

Decimal Separator:

Dot

Help

Download Template

Go to Import FAQ

Advanced

Track history during import

Allow matching with subfields

File Column	Odoo Field	Comments
Nombre <i>Allison Hill</i>	Ab Employee Name x	
Correo electrónico <i>donaldgarcia@example.net</i>	Ab Work Email x	
Teléfono <i>+1-219-560-0133</i>	Ab Work Phone x	
Puesto de trabajo <i>Analista de marketing</i>	Job Position x	
Departamento <i>Ventas</i>	Department x	

Presionar el botón **IMPORT** para cargar la información

La carga se realiza por medio de una archivo **CSV**.

A continuación se presenta el archivo **csv** y junto con los datos que contiene:

```

Proyecto > datos > csv > empleados_mega_city.csv > data
1  Nombre,Correo electrónico,Teléfono,Puesto de trabajo,Departamento
3  Meredith Barnes,zlawrence@example.org,001-626-254-2351x16155,Vendedor,Administración
4  Kimberly Dudley,smiller@example.net,+1-659-931-0341x316,Supervisor,Compras
5  Holly Wood,elizabethmiles@example.net,+1-219-528-3276x4835,Supervisor,Ventas
6  Renee Morales,clarksherri@example.net,837-767-2423x88496,Analista de marketing,Marketing
7  Zachary Hicks,camposmichelle@example.org,+1-326-691-6697x8480,Vendedor,Marketing
8  Gina Carter,shawnmckay@example.net,282-781-4893x252,Contador,Ventas
9  Brian Cox DVM,john39@example.org,671-882-2782,Vendedor,Ventas
10 Tanya Campos,jenniferross@example.net,+1-557-587-1331x5098,Supervisor,Compras
11 Michael Brown,whiteheadmichele@example.org,334.373.8299x73763,Jefe de almacén,Marketing
12 Heather Jenkins,nuneztracey@example.org,4658133387,Vendedor,Marketing
13 Joel Baxter,ramosmichelle@example.net,908-501-3267,Supervisor,Marketing
14 Stephanie Martin,gallowayjoseph@example.com,268-272-3430x9805,Contador,Compras
15 Kathryn Farrell,brianromero@example.org,291-336-1939,Contador,Marketing
16 Kevin Mills,james53@example.com,(524)675-1079x9118,Encargado de compras,Ventas
17 Gregory Jones,mikemoore@example.net,001-449-680-8412x411,Supervisor,Logística
18 Daniel Cooper,william40@example.org,+1-264-300-5242x7868,Encargado de compras,Administración
19 Stephen Mckee,davenportbrandi@example.org,(420)545-0533x15869,Supervisor,Compras
20 Vanessa Cochran,vjohnson@example.com,(534)221-6073x37543,Encargado de compras,Ventas
21 Jennifer Ramirez,rasmussenjoshua@example.net,768.750.1429x40196,Vendedor,Logística
22 William Wilson,jonathanfletcher@example.org,783.856.1595x14846,Vendedor,Administración
23 Kristen Lee,lbyrd@example.net,001-746-980-4436,Encargado de compras,Marketing
24 Michael Dixon,donnacampbell@example.net,+1-395-913-4332x0037,Encargado de compras,Ventas
25 Mark Lee,psnyder@example.org,+1-401-963-2870x8317,Analista de marketing,Logística
26 Victoria Larson,usalazar@example.net,(387)627-7434x87347,Jefe de almacén,Ventas
27 George Orozco,taylorjesse@example.net,(231)566-5876,Contador,Ventas
28 Jean Carrillo,gateskathy@example.com,805.946.6889x3734,Jefe de almacén,Administración
29 George Shelton,dramsey@example.org,+1-280-669-9016x272,Analista de marketing,Marketing
30 Daniel Murphy,aramirez@example.com,270-580-5310,Encargado de compras,Marketing
31 Alex Hernandez,thomaswilson@example.org,645.829.9124x190,Supervisor,Ventas
32 Miguel Jones,curtisbarton@example.net,+1-819-905-8651x850,Vendedor,Compras
33 Tyler Reid,bzimmerman@example.org,001-684-998-7769x4531,Encargado de compras,Ventas

```

Clients (Clientes)

Providers (Proveedores)

Esta información fue generada al ingresar las Compras.csv. Utilizando la información del proveedor en cada producto

Purchase (Compras)

Por problemas con ingresar por medio del csv, esto fue hecho por medio de XML-RPC.

Se realizamos la conexión del cliente a [odoo](#):

```

1  import xmlrpc.client
2  import csv
3  import os
4  from dotenv import load_dotenv
5
6  # ===== LOAD ENV VARIABLES =====
7  load_dotenv()
8  url = os.getenv('ODOO_URL')
9  db = os.getenv('ODOO_DB')
10 username = os.getenv('ODOO_USER')
11 password = os.getenv('ODOO_PASSWORD')
12
13 # ===== CONNECT TO ODOO =====
14 common = xmlrpc.client.ServerProxy(f'{url}/xmlrpc/2/common')
15 uid = common.authenticate(db, username, password, {})
16 models = xmlrpc.client.ServerProxy(f'{url}/xmlrpc/2/object')

```

Cargamos el archivo csv y creamos las **compras**

```

# ===== LOAD PURCHASES CSV =====
with open('csv/purchase.csv', newline='', encoding='utf-8') as csvfile:
    reader = csv.DictReader(csvfile)
    for row in reader:
        vendor_name = row['Proveedor']
        product_name = row['Producto']
        quantity = float(row['Cantidad'])
        price_unit = float(row['Precio Unitario'])

        vendor_ids = models.execute_kw(db, uid, password, 'res.partner', 'search', [[['name', '=', vendor_name]]])
        if not vendor_ids:
            vendor_id = models.execute_kw(db, uid, password, 'res.partner', 'create', [{
                'name': vendor_name,
                'supplier_rank': 1
            }])
        else:
            vendor_id = vendor_ids[0]

        product_ids = models.execute_kw(db, uid, password, 'product.product', 'search', [[['name', '=', product_name]]])
        if not product_ids:
            print(f'Product not found: {product_name}')
            continue
        product_id = product_ids[0]

        po_id = models.execute_kw(db, uid, password, 'purchase.order', 'create', [{
            'partner_id': vendor_id,
            'date_order': row['Fecha Orden']
        }])

```

Creamos el sub-compra que son los detalles de cada compra

```
po_data = models.execute_kw(db, uid, password, 'purchase.order', 'read', [[po_id]], {'fields': ['name']})
po_name = po_data[0]['name']

models.execute_kw(db, uid, password, 'purchase.order.line', 'create', [{
    'order_id': po_id,
    'product_id': product_id,
    'product_qty': quantity,
    'price_unit': price_unit,
    'name': product_name,
    'product_uom': 1
}])
```

Validamos todo los procesos y creamos las facturas para cada compra

```
models.execute_kw(db, uid, password, 'purchase.order', 'button_confirm', [[po_id]])

pickings = models.execute_kw(db, uid, password, 'stock.picking', 'search', [[
    ['origin', '=', po_name]
]])
for picking_id in pickings:
    move_lines = models.execute_kw(db, uid, password, 'stock.move.line', 'search', [[
        ['picking_id', '=', picking_id]
    ]])
    for move_line_id in move_lines:
        models.execute_kw(db, uid, password, 'stock.move.line', 'write', [[move_line_id], {
            'qty_done': quantity # or move['product_uom_qty'] if doing per move
        }])

    models.execute_kw(db, uid, password, 'stock.picking', 'button_validate', [[picking_id]])

models.execute_kw(db, uid, password, 'purchase.order', 'button_confirm', [[po_id]])

models.execute_kw(db, uid, password, 'purchase.order', 'action_create_invoice', [[po_id]])

invoice_ids = models.execute_kw(db, uid, password, 'account.move', 'search', [[
    ['invoice_origin', '=', po_name]
]])

if invoice_ids:
    models.execute_kw(db, uid, password, 'account.move', 'write', [[invoice_ids[0]], {
        'invoice_date': row['Fecha Orden'],
        'invoice_date_due': row['Fecha Orden']
    }])

    models.execute_kw(db, uid, password, 'account.move', 'action_post', [[invoice_ids[0]]])
    print(f'✅ PO billed: {po_name}')
else:
    print(f'⚠️ No invoice found for PO: {po_name}')
```

Archivo csv


```

Proyecto > datos > csv > purchase.csv > data
1 Proveedor,Producto,Cantidad,Precio Unitario,Precio Total,Fecha Orden
2 Phillips LLC,There TV,99,316.45,31328.55,2025-02-04
3 Little Group,Save Auriculares,41,526.23,21575.43,2025-01-20
4 King Group,Create TV,91,334.59,30447.69,2025-01-23
5 "Avery, Miller and Wright",Church Teclado,17,563.46,9578.82,2025-04-19
6 "Branch, Torres and Oliver",Trade Cámara,18,653.78,11768.04,2025-01-20
7 Ortiz Ltd,Close Cámara,93,707.28,65777.04,2025-04-02
8 French-Weber,East Cable,28,713.51,19978.28,2025-02-08
9 Larson-Holloway,Structure Router,43,234.97,10103.71,2025-02-26
10 "Williams, Reynolds and Morris",Be Mouse,38,723.37,27488.06,2025-02-18
11 "Sampson, Key and Chambers",Parent TV,24,420.14,10083.36,2025-02-22
12 Costa-Harrison,Prove TV,86,169.99,14619.14,2025-03-21
13 "Johnson, Miller and King",Institution Auriculares,77,336.21,25888.17,2025-01-23
14 Dean-Jimenez,National Router,24,695.58,16693.92,2025-03-27
15 "Washington, Johnson and Mccoy",Left Teclado,47,626.45,29443.15,2025-02-16
16 Murray Inc,Thought Monitor,43,390.29,16782.47,2025-01-30
17 Lee-Greene,Check Teclado,90,430.76,38768.4,2025-01-27
18 Hoover-Campbell,Movement Router,57,430.75,24552.75,2025-02-27
19 Kane-Pollard,Worker TV,10,765.12,7651.2,2025-02-16
20 "Pollard, Simpson and Johnson",Record Mouse,56,416.45,23321.2,2025-04-08
21 Harrison LLC,Clear Router,20,93.44,1868.8,2025-03-07
22 Brennan-Johnson,Song Auriculares,78,414.49,32330.22,2025-01-14
23 "Padilla, Garcia and Jackson",Billion TV,31,544.81,16889.11,2025-02-28
24 Thomas-Schmidt,House Router,37,704.32,26059.84,2025-04-20
25 Durham-Thomas,Effect Teclado,61,567.4,34611.4,2025-03-31
26 Jones LLC,Office Laptop,41,724.71,29713.11,2025-02-28
27 "Howard, Phillips and Lyons",I Cámara,39,303.56,11838.84,2025-01-05
28 Parsons-Hall,Model Mouse,17,55.39,941.63,2025-03-10
29 Fuller Ltd,This Cámara,19,729.07,13852.33,2025-01-08
30 "Perry, Gonzalez and Buchanan",General Teclado,79,258.86,20449.94,2025-03-13
31 Brock-Mcdonald,Truth TV,70,751.64,52614.8,2025-01-15

```

Sales (Ventas)

Ingresar al módulo de ventas

Sales Orders / Import a File

IMPORT TEST LOAD FILE CANCEL

Imported file

ventas_mega_city.csv

☒ Use first row as header

Formatting

Encoding: utf-8-sig

Separator: Comma

Text Delimiter: "

Date Format: YYYY-MM-DD

Datetime Format:

Thousands Separator: Comma

Decimal Separator: Dot

Batch Import

Batch limit: 2000 Start at line: 1

Help

[Go to Import FAQ](#)

Advanced

☐ Track history during import

☒ Allow matching with subfields

File Column	Odoo Field	Comments
Cliente Kyle Phillips	Customer	
Producto Song Auriculares	Optional Products Lines / Product	
Cantidad 1	Optional Products Lines / Quantity	
Precio Unitario 117.16	Optional Products Lines / Unit Price	
Precio Total 117.16	To import, select a field...	
Fecha Orden 2025-04-09	Order Date	

Seleccionar la opción de **import**



Invoicing (Facturas)

Las facturas fueron creados por modo manual al confirmar las ventas

Proceso de generar el información

La información fue generado en python con el uso de los librerías pandas, random y faker

```

1 import pandas as pd
2 import random
3 from faker import Faker
4
5 fake = Faker()
6 Faker.seed(42)
7 random.seed(42)
8

```

Primero generamos la información para empleados.csv

```

#Empleados.csv

departments = ['Ventas', 'Compras', 'Administración', 'Logística', 'Marketing']
job_positions = ['Vendedor', 'Supervisor', 'Encargado de compras', 'Contador', 'Jefe de almacén', 'Analista de marketing']

employees = []
for i in range(35):
    full_name = fake.name()
    email = fake.email()
    phone = fake.phone_number()
    job = random.choice(job_positions)
    dept = random.choice(departments)
    employees.append({
        'Nombre': full_name,
        'Correo electrónico': email,
        'Teléfono': phone,
        'Puesto de trabajo': job,
        'Departamento': dept
    })

df_employees = pd.DataFrame(employees)
df_employees.to_csv('csv/empleados_mega_city.csv', index=False, encoding='utf-8-sig')

```

Después generamos la información para compras.csv que se utiliza para popular productos y proveedores

```

# compras.csv
product_categories = ['Telefonía', 'Audio', 'Video', 'Accesorios', 'Redes', 'Computación']
product_catalog = []
purchases = []
used_names = set()

for _ in range(40):
    while True:
        product_name = fake.unique.word().capitalize() + ' ' + random.choice([
            'Smartphone', 'Auriculares', 'TV', 'Cámara', 'Router', 'Laptop', 'Monitor', 'Cable', 'Teclado', 'Mouse'
        ])
        if product_name not in used_names:
            used_names.add(product_name)
            break

        category = random.choice(product_categories)
        vendor = fake.company()
        price_unit = round(random.uniform(50, 800), 2) # realistic cost
        sale_price = round(price_unit * random.uniform(1.2, 1.5), 2) # markup between 20-50%
        quantity = random.randint(10, 100)
        date = fake.date_this_year().isoformat()

        purchases.append({
            'Proveedor': vendor,
            'Producto': product_name,
            'Cantidad': quantity,
            'Precio Unitario': price_unit,
            'Precio Total': round(quantity * price_unit, 2),
            'Fecha Orden': date
        })

# productos.csv
product_catalog.append({
    'Nombre': product_name,
    'Categoría': category,
    'Precio': sale_price,
    'Disponible': '1',
    'Descripción': fake.sentence(nb_words=6),
    'Tipo': 'Storable Product'
})

for _ in range(10):
    purchase = random.choice(purchases)
    purchases.append({
        **purchase,
        'Fecha Orden': fake.date_this_year().isoformat()
    })

df_products = pd.DataFrame(product_catalog)
df_purchases = pd.DataFrame(purchases)

```

Después utilizando esa información, generamos las ventas y guardamos todo en su propio csv

```
# ventas.csv
sales_orders = []
product_names = df_products['Nombre'].tolist()

for i in range(100):
    product = random.choice(product_names)
    customer = fake.name()
    quantity = random.randint(1, 10)
    unit_price = round(random.uniform(60, 1200), 2)
    total = round(quantity * unit_price, 2)
    order = {
        'Cliente': customer,
        'Producto': product,
        'Cantidad': quantity,
        'Precio Unitario': unit_price,
        'Precio Total': total,
        'Fecha Orden': fake.date_this_year().isoformat()
    }
    sales_orders.append(order)

df_sales = pd.DataFrame(sales_orders)

df_products.to_csv('csv/productos_mega_city.csv', index=False, encoding='utf-8-sig')
df_purchases.to_csv('csv/compras_mega_city.csv', index=False, encoding='utf-8-sig')
df_sales.to_csv('csv/ventas_mega_city.csv', index=False, encoding='utf-8-sig')
```

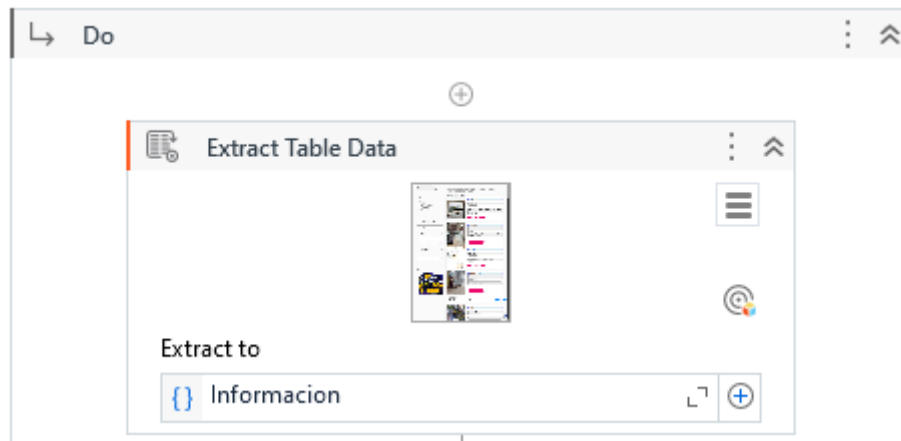
Seccion 4

RPA Automatización 2

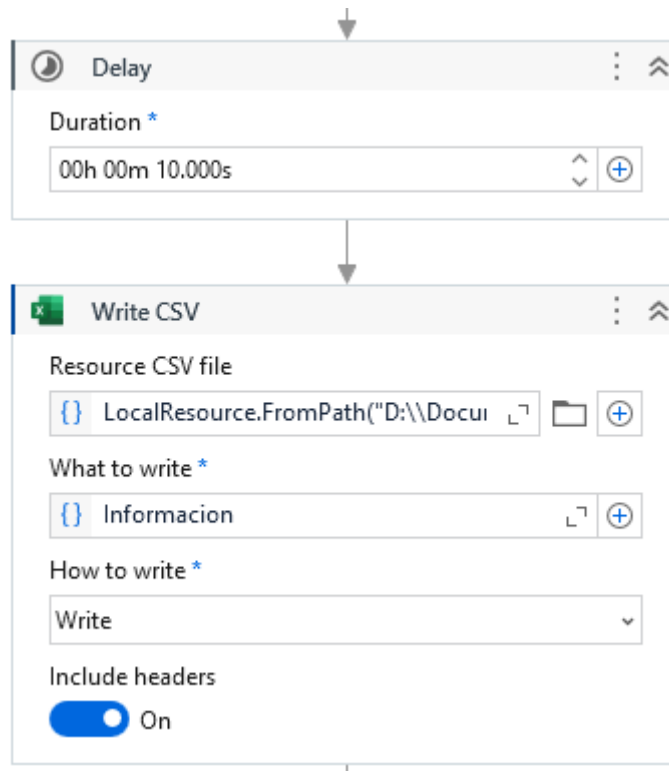
Primero creamos una secuencia para utilizar el browser de Chrome y navegar al pagina de https://www.encuentra24.com/guatemala-es/searchresult/bienes-raices-venta-de-propiedades-comercios?regionslug=guatemala-guatemala-city&q=f_price.-800000




Después extraemos toda el información relevante del pagina



Dejamos un delay de 10s y después escribimos esa información en un archivo csv



ya con ese archivo, generamos un correo y lo enviamos con el archivo generado


Send Email

Gmail

Default (3001554250101@ingenieria.usac.edu....

Save as draft

☐ True ☒ False

To *

agrn96p@gmail.com X

The primary recipients of the email, separated by commas

Subject

{} "info"

Body

B *I* U ~~S~~

Adjunto el csv con el informacion obtenido de bodegas < 800000

Attachment(s)

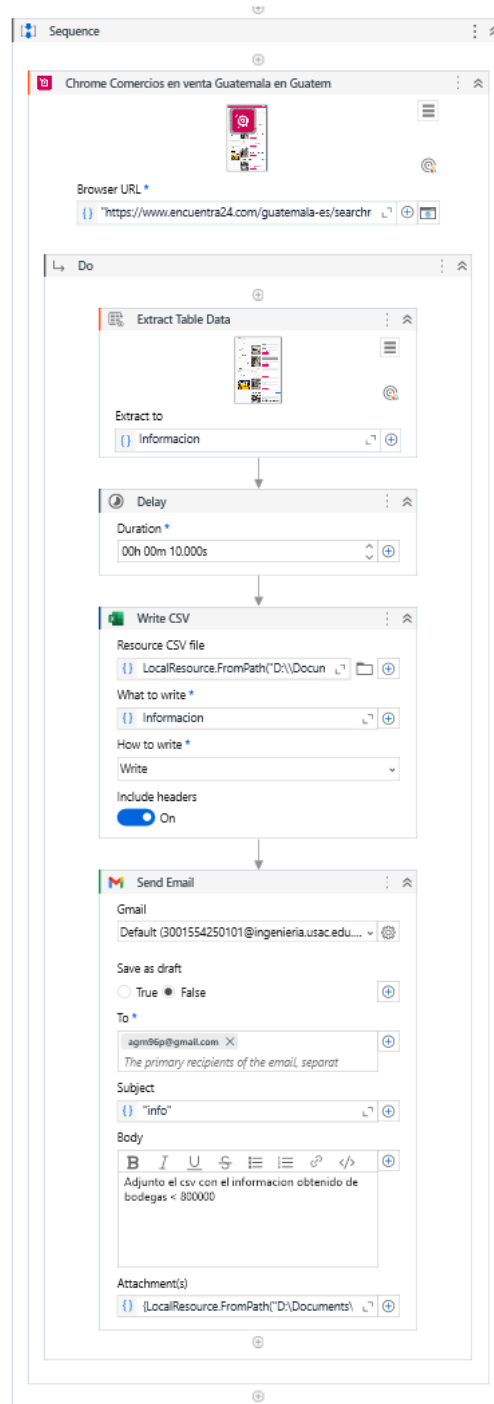
{} {LocalResource.FromPath("D:\Documents

Compras a proveedores de la empresa

Manual 2

Diagrama de flujo UiPath

Búsqueda de lotes menor de 800000



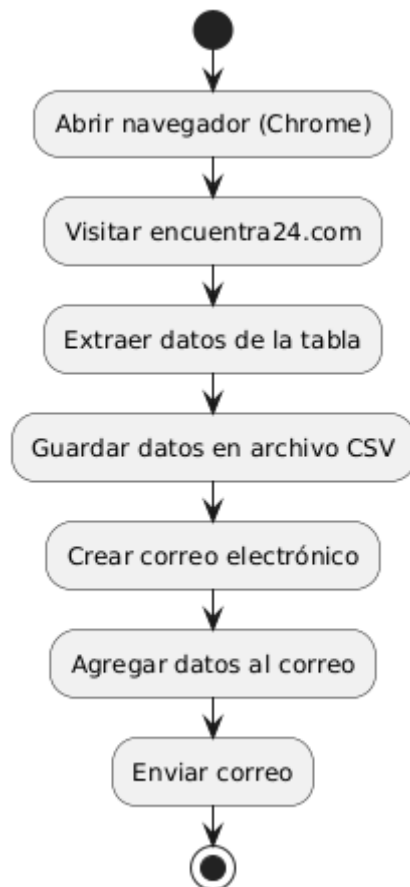


Diagrama Compras a proveedores de la empresa



Diagrama Ventas de productos

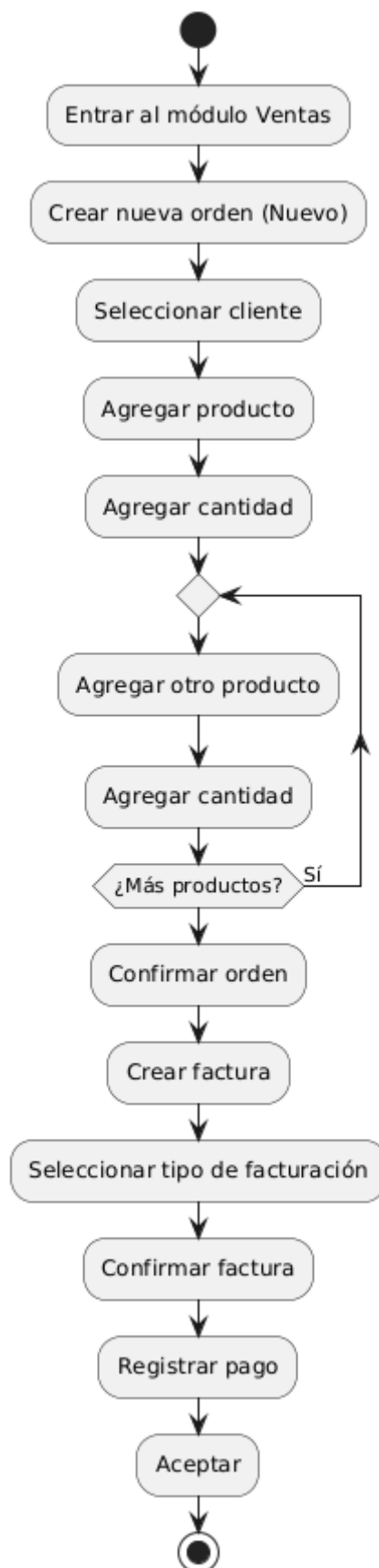


Diagrama de Flujo de proceso de la empresa desde que entra un producto hasta que este sea vendido

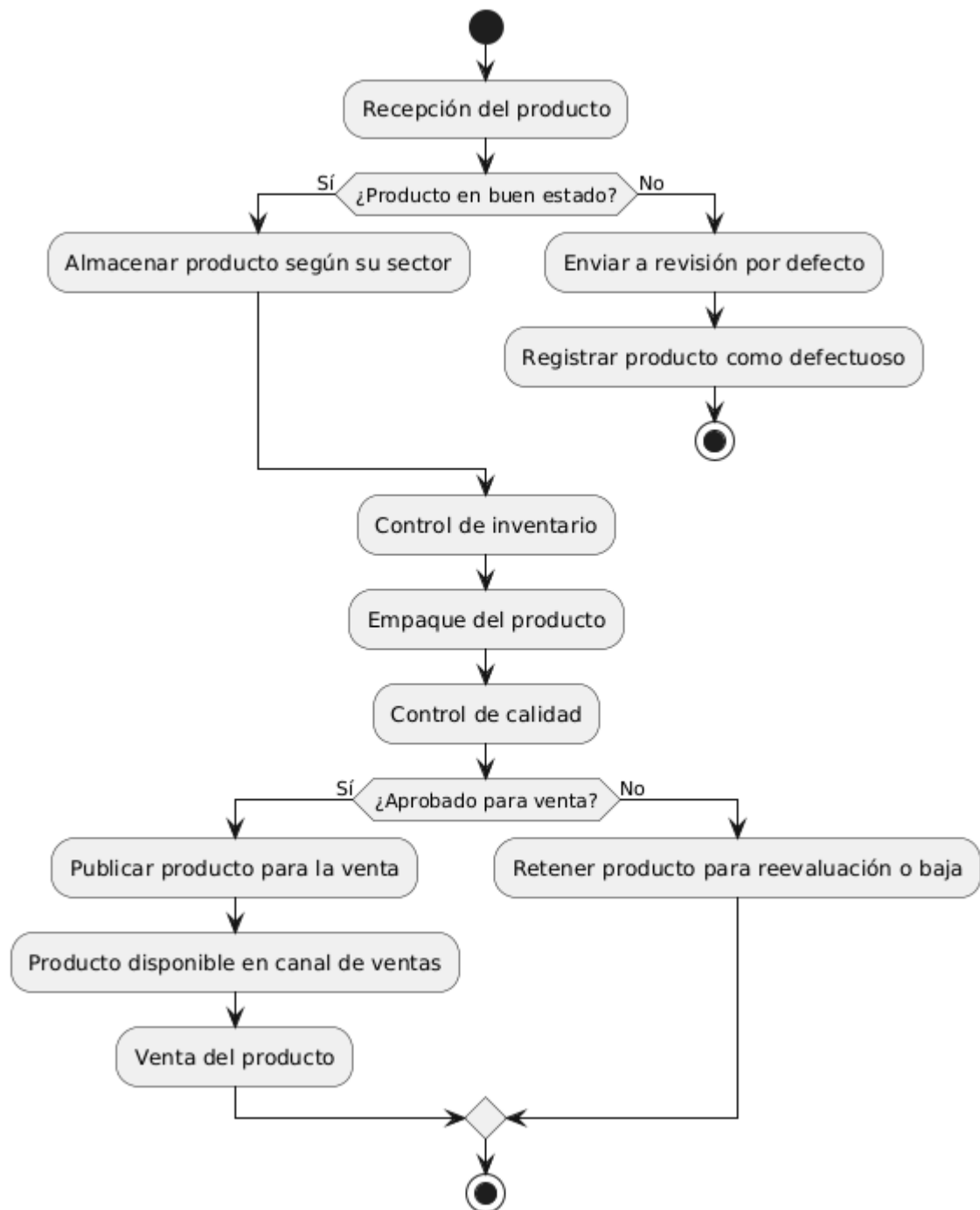


Diagrama de compra web

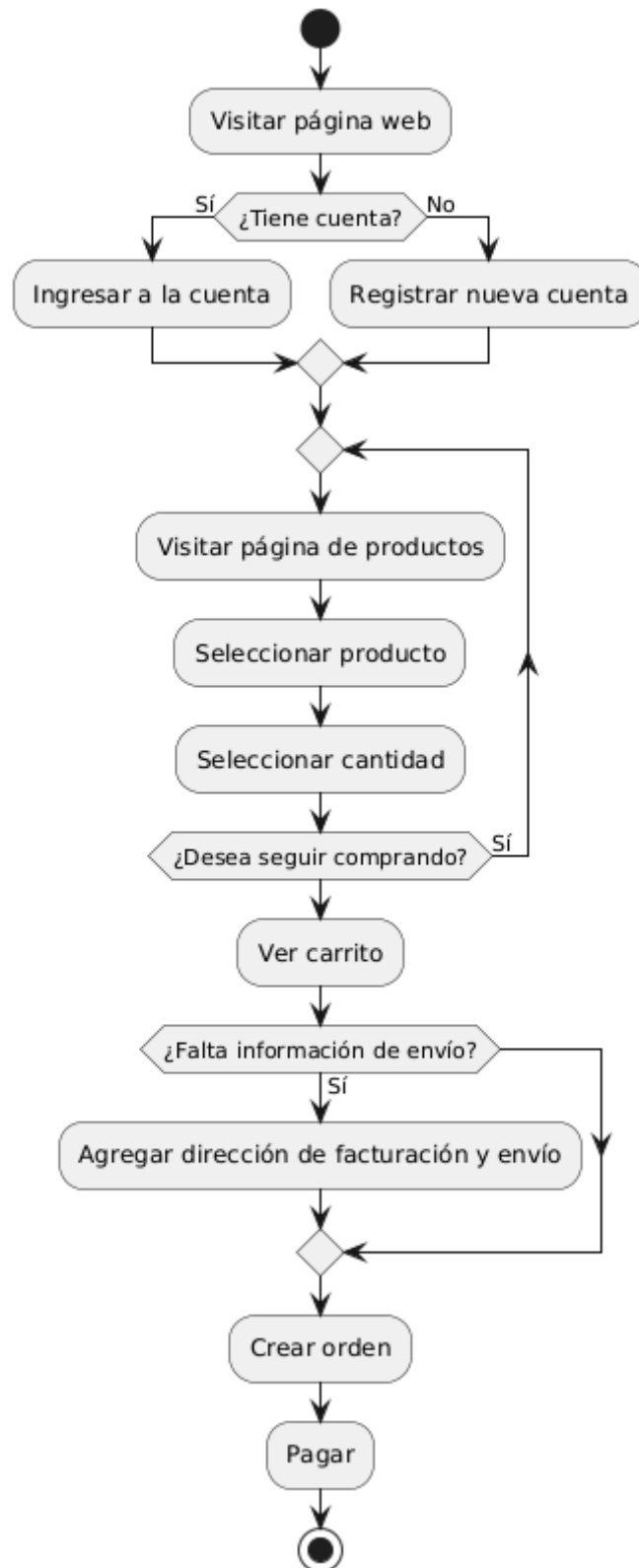


Diagrama de compra tienda

