



Universidad
Nacional
de Loja

[Desarrollo de Software en Ambientes Cloud]

Resolución de problemas prácticos:
Elaboración de solución de software /
Gestión de bodega

Tutor:

ROBERTH GUSTAVO FIGUEROA DIAZ
roberth.figueroa@unl.edu.ec

Elaborado por:

FREDDY ALEXANDER CHAMORRO ENCALADA
fachamorro@unl.edu.ec

11 de noviembre de 2022



Tabla de contenido

1. OBJETIVOS	3
2. DESARROLLO	3
2.1. Solución de software	3
2.1.1. Problemática	3
2.1.2. Clasificación de actores	4
2.1.3. Diagrama de procesos (actual)	4
2.1.4. Especificación de requerimientos	5
2.1.5. Modelo de Dominio	5
2.1.6. Modelado de Casos de Uso	6
2.1.6.1. Identificación de actores	6
2.1.6.2. Identificación de Casos de Uso	6
2.1.6.3. Modelado de Casos de Uso	6
2.1.6.4. Descripción de Casos de Uso	7
2.1.6.4.1. CU: Iniciar Sesión	7
2.1.6.4.2. CU: Administrar Producto	7
2.1.6.4.3. CU: Administrar Órdenes de Compra	8
2.1.7. Modelo de implementación (Monolito distribuido)	9
2.1.8. Diagrama de Clases	10
2.1.9. Diagrama de despliegue	10
2.2. Tecnologías utilizadas para la implementación de la solución de software	11
2.3. Implementación de la solución de software para la gestión de bodega	12
2.3.1. Backend - API REST	12
2.3.2. Frontend - App Web	14
2.3.3. Ejecución del proyecto	15
2.3.4. Código fuente	17
3. CONCLUSIONES	18
4. BIBLIOGRAFIA	18
5. FIRMAS	18



1. OBJETIVOS

- Identificar los fundamentos y elementos necesarios para el desarrollo de software en ambientes cloud.

2. DESARROLLO

2.1. Solución de software

Sistema para la gestión de ingresos en las bodegas del Hospital Básico El Corazón.

2.1.1. Problemática

En el Hospital Básico El Corazón no se cuenta con una solución tecnológica para la gestión de productos que son adquiridos a diversos proveedores en diferentes fechas durante el ejercicio fiscal vigente.

En vista que la documentación de los ingresos se encuentra de manera física se dificulta el cálculo del stock de los productos disponibles, y su seguimiento que permita planificar una adquisición de nuevos productos de manera oportuna sin dejar desabastecidos a los diferentes departamentos y servicios.

Por tales motivos el personal responsable de las bodegas del Hospital Básico El Corazón ha visto la necesidad de mejorar específicamente en:

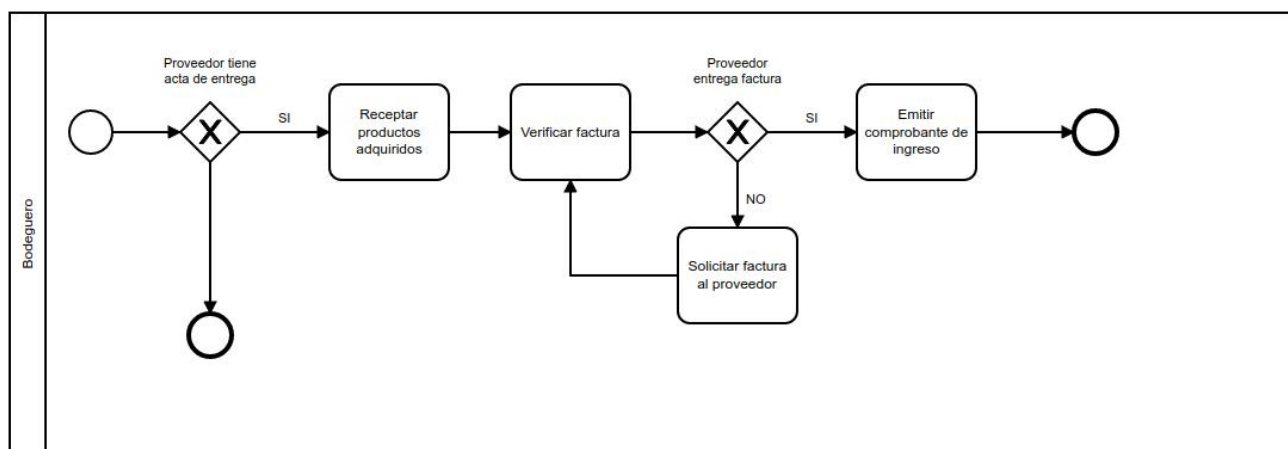
- Mejorar el manejo de órdenes de ingresos/egresos de productos de las bodegas.
- Agilizar el cálculo del stock de los productos disponibles.
- Obtener datos de los proveedores de productos que ingresan a las bodegas del Hospital Básico El Corazón.
- Dar seguimiento a los productos que egresó un cliente(empleo) desde la bodega.
- Categorizar los productos por su tipo.

2.1.2. Clasificación de actores

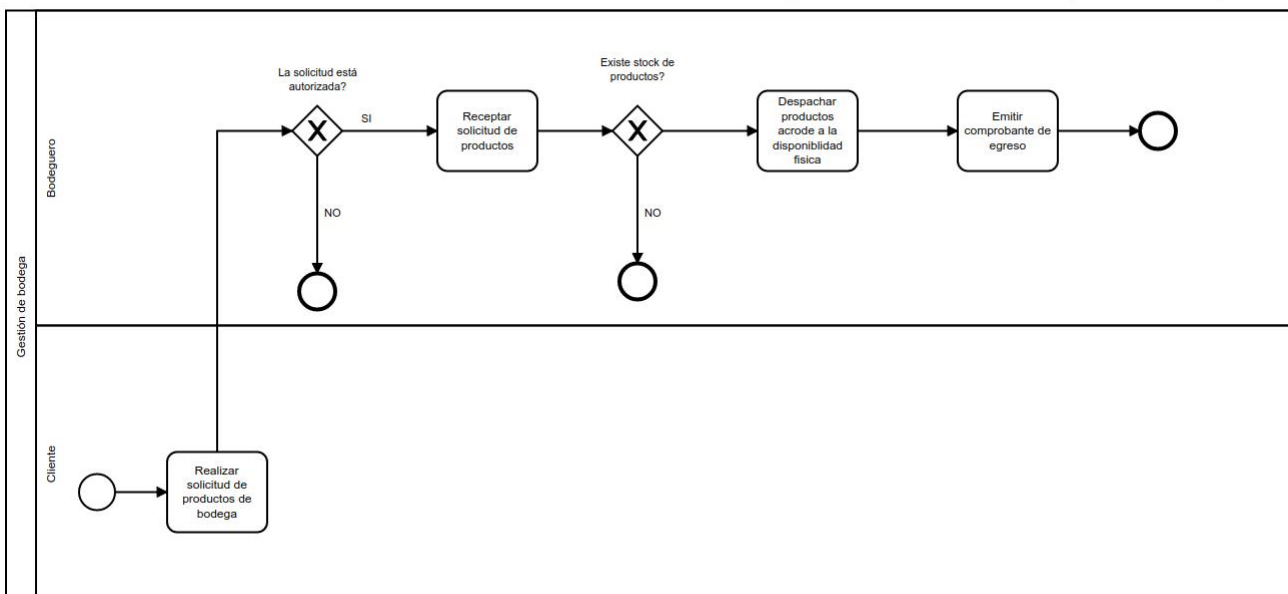
Actor	Necesidad del usuario
Bodeguero	<ul style="list-style-type: none"> ● Registra proveedores ● Registra productos y tipos de productos ● Registra clientes ● Ingresa Órdenes de compras ● Ingresa Órdenes de egresos.
Administrador	<ul style="list-style-type: none"> ● Registra usuarios (Bodeguero)

2.1.3. Diagrama de procesos (actual)

Proceso de ingreso de productos a bodega (actual)



Proceso de egreso de productos de bodega (actual)



2.1.4. Especificación de requerimientos

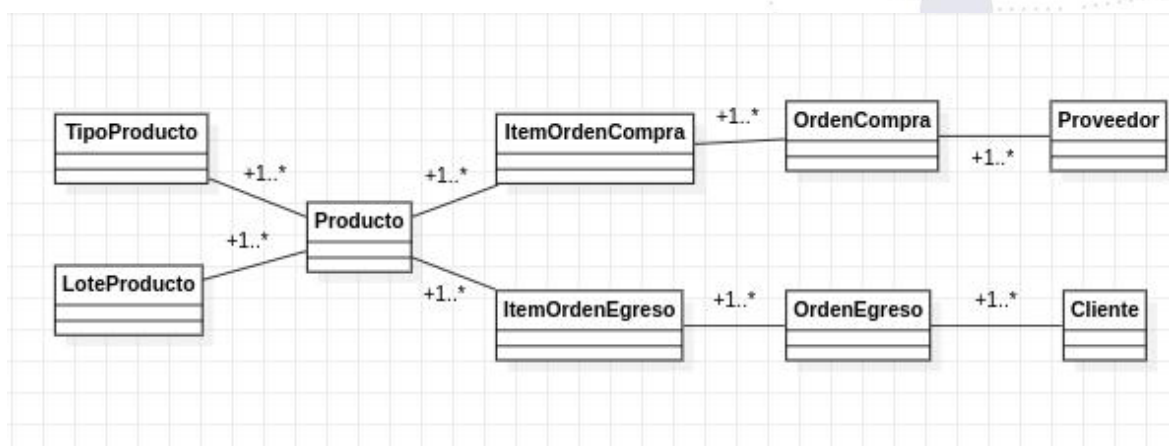
El sistema debe permitir al usuario(bodeguero):

Código	Descripción	Categoría	Prioridad
RF001	Registrar, editar, eliminar y visualizar los productos	Visible	Alta
RF002	Registrar, editar, eliminar y visualizar los tipos de los productos	Visible	Alta
RF003	Registrar, editar, eliminar y visualizar los proveedores	Visible	Alta
RF004	Registrar y visualizar las Órdenes de compra	Visible	Alta
RF005	Registrar y visualizar las Órdenes de egreso	Visible	Alta
RF006	Registrar, editar, eliminar y visualizar los clientes	Visible	Alta

El sistema debe permitir al Administrador:

Código	Descripción	Categoría	Prioridad
RF007	Registrar usuarios del sistema	Visible	Alta
RF008	Iniciar sesión con sus credenciales de cuenta: usuario y contraseña	Visible	Alta

2.1.5. Modelo de Dominio



2.1.6. Modelado de Casos de Uso

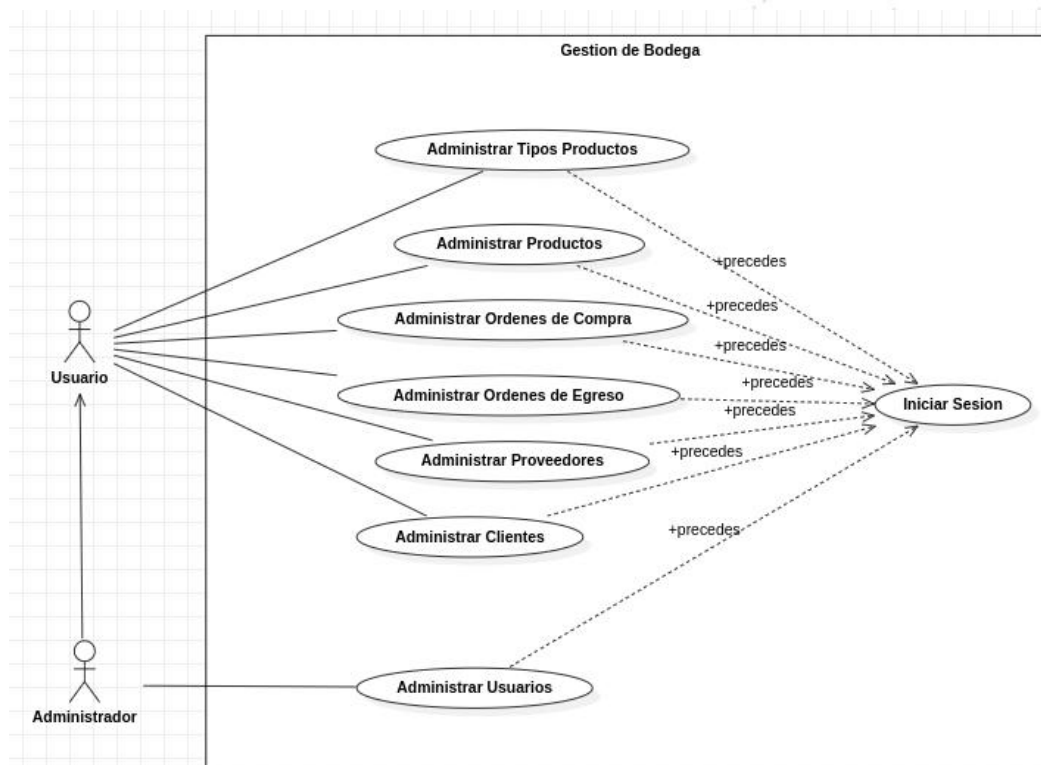
2.1.6.1. Identificación de actores

- Usuario.- Es el encargado de operar el sistema.
- Administrador.- Es el encargado de administrar los usuarios (bodegueros), también puede ejercer todas las acciones del operador en el sistema.

2.1.6.2. Identificación de Casos de Uso

Actores	Funcionalidad	Código	Requerimientos cubiertos
Usuario del sistema (Bodeguero, Administrador)	Iniciar Sesión	CU-001	RF008
	Administrar productos	CU-002	RF001
	Administrar tipos de productos	CU-003	RF002
	Administrar clientes	CU-004	RF003
	Administrar proveedores	CU-005	RF004
	Administrar Órdenes de Compra	CU-006	RF005
	Administrar Órdenes de Egreso	CU-007	RF006
Administrador	Administrar Usuarios	CU-008	RF007

2.1.6.3. Modelado de Casos de Uso



2.1.6.4. Descripción de Casos de Uso

2.1.6.4.1. CU: Iniciar Sesión

Caso de Uso:	Iniciar sesión	Actor(es):	Usuarios del sistema
Código	UC-001	Pantalla(s):	
Ref. Req:	RF008		
Resumen / Descripción:	Los usuarios del sistema (bodeguero, administrador del sistema) a través de sus credenciales de usuario acceden al sistema para su gestión.		
Objetivos:	<ul style="list-style-type: none">• Acceso a la funcionalidad especial del sistema• Autorización a la funcionalidad de gestión del sistema		
Precondiciones:	<ul style="list-style-type: none">• Tener cuenta de usuario registrada.• Encontrarse en el formulario de “inicio de sesión”.		
Poscondiciones:	<ul style="list-style-type: none">• Usuario autorizado y autenticado.• Pantalla de inicio		
CURSO NORMAL DE EVENTOS			
<ol style="list-style-type: none">1. El usuario ingresa sus credenciales de acceso como usuario y contraseña en el formulario de “inicio de sesión”2. El usuario presiona el botón “Ingresar” del formulario de “Iniciar Sesión”.3. El sistema busca al Usuario con las credenciales de acceso ingresadas como usuario y contraseña.4. El sistema redirige a la página principal con las funciones autorizadas disponibles para el usuario.			
CURSO ALTERNO DE EVENTOS			
A. Información requerida			
A.2. El sistema muestra un mensaje indicando que “existe información obligatoria que no esta presente”.			
B. Usuario no existe			
B.3. El sistema muestra un mensaje indicando que “Credenciales No válidas”			

2.1.6.4.2. CU: Administrar Producto

Caso de Uso:	Administrar Producto	Actor(es):	Usuarios del sistema
Código	UC-002	Pantalla(s):	
Ref. Req:	RF001		
Resumen / Descripción:	Los usuarios del sistema (bodeguero, administrador del sistema) una vez autorizados podrán administrar los productos en el sistema.		
Objetivos:	<ul style="list-style-type: none"> Administrar los productos del sistema de bodega (CRUD). 		
Precondiciones:	<ul style="list-style-type: none"> Usuario haya iniciado sesión (Caso de Uso: Iniciar Sesión). 		
Poscondiciones:	<ul style="list-style-type: none"> Pantalla de administración de productos. 		

CURSO NORMAL DE EVENTOS	
1.	El usuario selecciona la opción “Productos”, del menú principal en la página de inicio.
2.	El sistema muestra un menú desplegable con las opciones disponibles para productos.
3.	El usuario selecciona la opción “Lista de Productos” del menú desplegable.
4.	El sistema muestra la pantalla “Lista de Productos” con una tabla de los productos registrados para su consulta.
CURSO ALTERNO DE EVENTOS	
A. Registrar Producto	
A.3.	El usuario selecciona la opción “Registrar Productos” del menú desplegable.
A.4.	El sistema muestra la pantalla “Registro de Productos” para el ingreso de datos del producto.
A.5.	El usuario registra los datos del producto y presiona el botón “Guardar”.
A.6.	El sistema guarda la información y muestra el mensaje “Guardado exitoso”.
B. Editar Producto	
B.5.	El usuario selecciona la opción “Editar” del producto en la tabla de productos de la pantalla “Lista de Productos”.
B.6.	El sistema muestra la pantalla “Editar de Productos” para la modificación de datos del producto.
B.7.	El usuario modifica los datos del producto y presiona el botón “Guardar”.
B.8.	El sistema guarda la información y muestra el mensaje “Actualizado con éxito”.
B. Eliminar Producto	
B.5.	El usuario selecciona la opción “Eliminar” del producto en la tabla de productos de la pantalla “Lista de Productos”.
B.6.	El sistema muestra un mensaje para la solicitud de confirmación “Está seguro de eliminar el producto”.
B.7.	El usuario confirma la eliminación del producto registrado en el sistema.
B.8.	El sistema elimina el producto y muestra el mensaje “Producto eliminado con éxito”.

Nota. La descripción se repite para los Casos de Uso: CU-003, CU-004, y CU-005, por lo cual no se coloca el presente informe.

2.1.6.4.3. CU: Administrar Órdenes de Compra

Caso de Uso:	Administrar Órdenes de Compra	Actor(es):	Usuarios del sistema
Código	UC-006	Pantalla(s):	
Ref. Req:	RF005		
Resumen / Descripción:	Los usuarios del sistema (bodeguero, administrador del sistema) una vez autorizados podrán administrar las órdenes de compra en el sistema.		
Objetivos:	<ul style="list-style-type: none">• Administrar (crear, visualizar) las órdenes de compra en el sistema de bodega.		
Precondiciones:	<ul style="list-style-type: none">• Usuario haya iniciado sesión (Caso de Uso: Iniciar Sesión).• Productos registrados en el sistema.		
Poscondiciones:			
CURSO NORMAL DE EVENTOS			
1. El usuario selecciona la opción “Órdenes de Compra”, del menú principal en la página de			

inicio.

2. El sistema muestra un menú desplegable con las opciones disponibles para órdenes de compra.
3. El usuario selecciona la opción “Lista de Órdenes de Compra” del menú desplegable.
4. El sistema muestra la pantalla “Lista de Órdenes de Compra” con una tabla de las Órdenes registradas en el sistema para su consulta.

CURSO ALTERNO DE EVENTOS

A. Registrar Orden de Compra

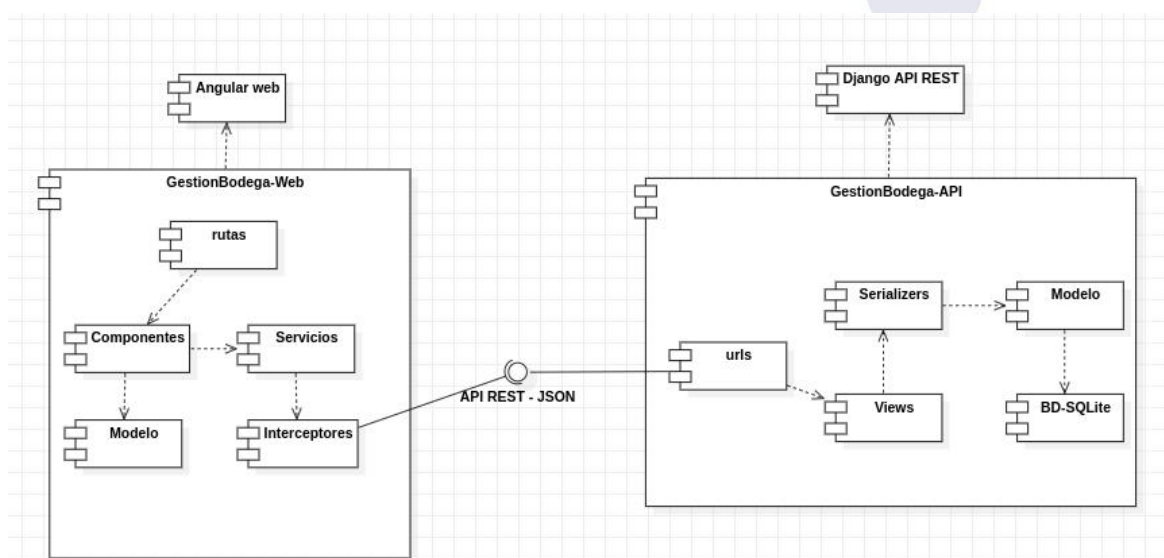
- A.3. El usuario selecciona la opción “Registrar Orden de Compra” del menú desplegable.
- A.4. El sistema muestra la pantalla “Registro Orden de Compra” para el ingreso de datos del orden.
- A.5. El usuario registra los datos de la orden de compra y presiona el botón “Guardar”.
- A.6. El sistema guarda la información y muestra el mensaje “Guardado exitoso”.

B. Visualizar Órdenes de Compra

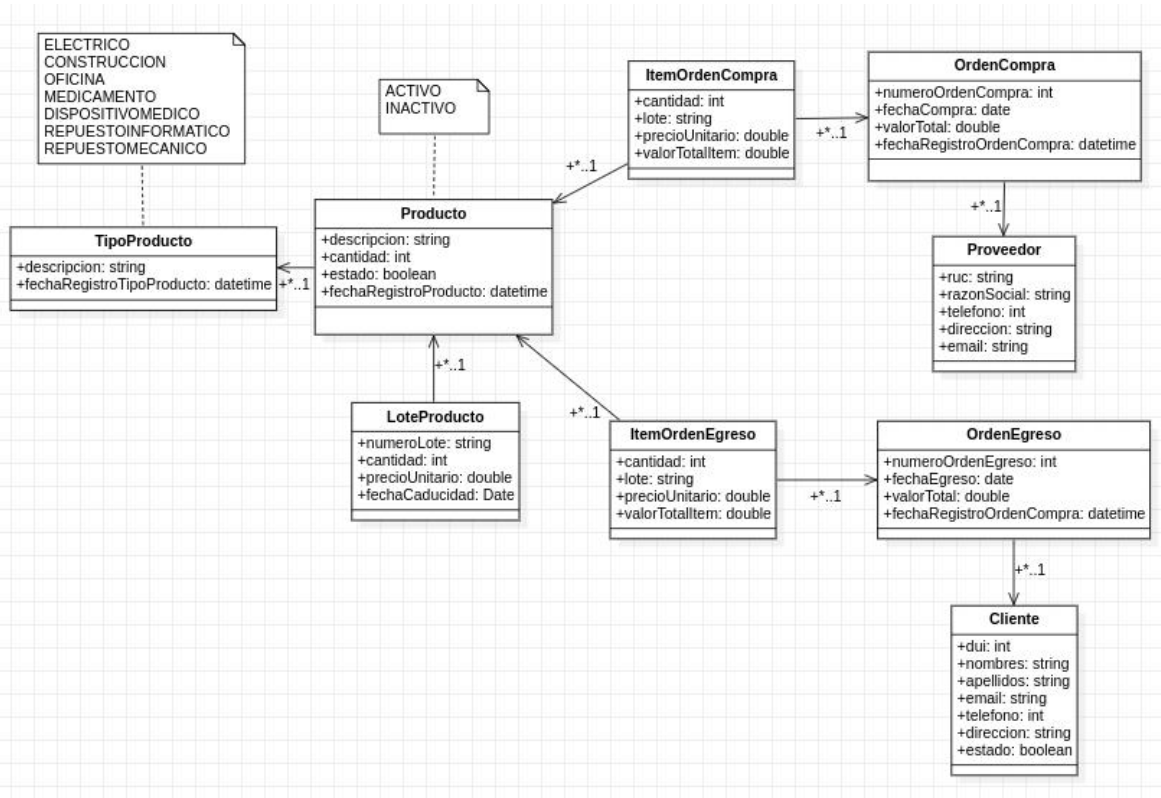
- B.5. El usuario selecciona la opción “Detalles” de la orden de compra en la tabla de la pantalla “Lista de Productos”.
- B.6. El sistema muestra la pantalla “Detalles de la Orden de Compra” para la visualización a detalle de la misma o impresión.

Nota. La descripción se repite para los Casos de Uso: CU-007, por lo cual no se coloca el presente informe.

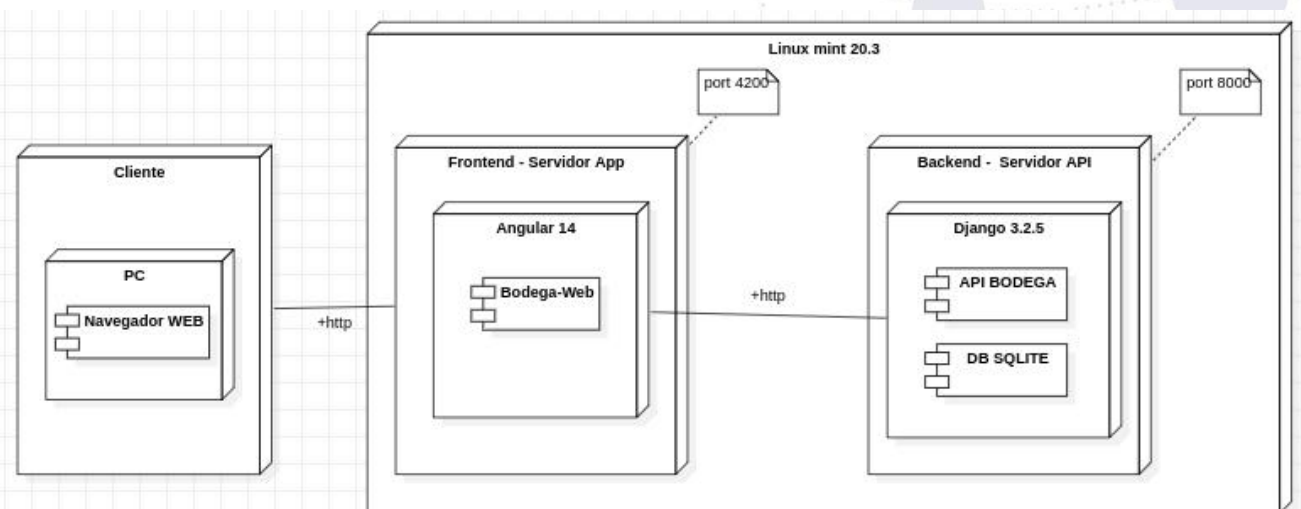
2.1.7. Modelo de implementación (Monolito distribuido)



2.1.8. Diagrama de Clases



2.1.9. Diagrama de despliegue



2.2. Tecnologías utilizadas para la implementación de la solución de software

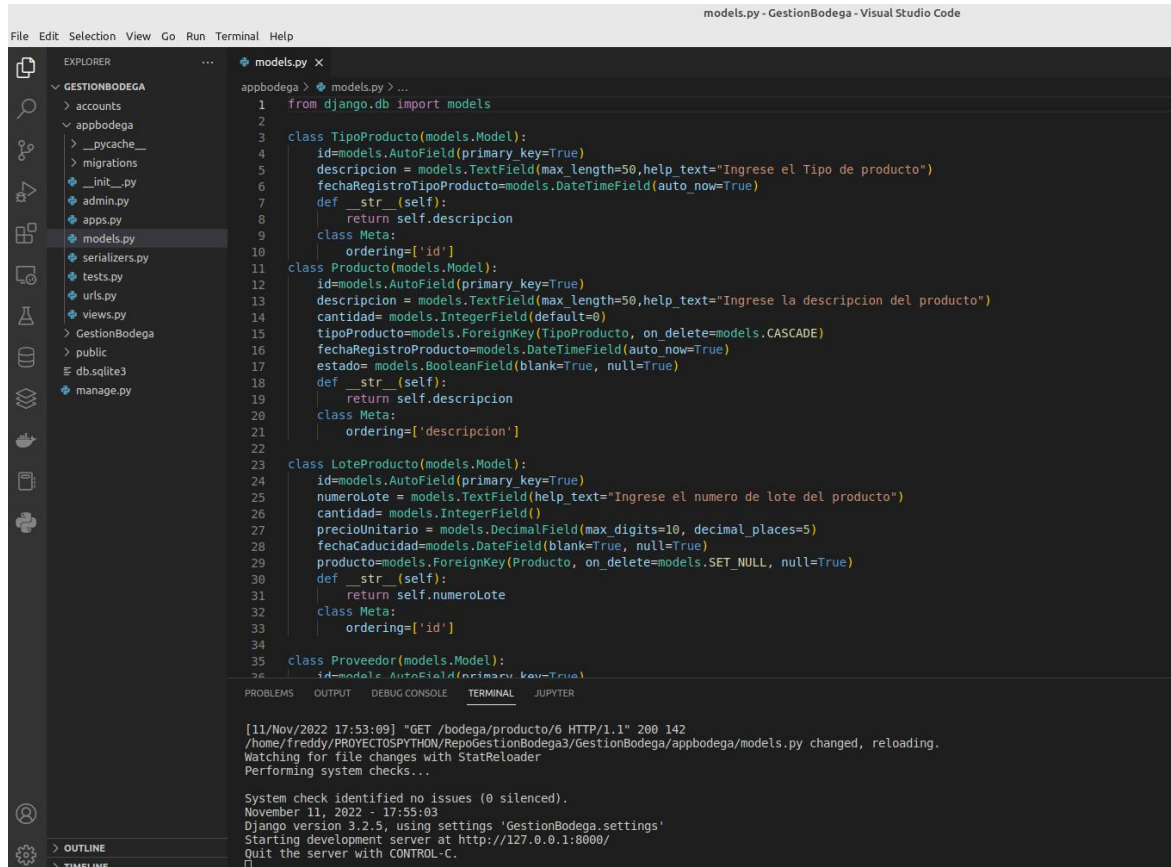
Para el desarrollo de la presente actividad se utilizó las siguientes herramientas:

Herramienta	Versión	Observación
Python	3.9.15	Lenguaje de programación
pip	22.2.2	gestor de paquetes
virtualenv	20.017	herramienta para entorno aislado Python
Django	3.2.5	Framework para el desarrollo de apps
django-cors-headers	3.13.0	Paquete python, permite configurar las solicitudes a la app en Django desde otros orígenes.
django-filter	22.1	Paquete python, permite agregar filtrado dinámico a partir de parámetros URL.
djangorestframework	3.14.0	Paquete python, es un framework que permite el desarrollo de una API REST
Markdown	3.4.1	Paquete python, para utilizar lenguaje de marcado sencillo
sqlparse	0.4.3	Paquete python para gestión de base de datos SQLite
importlib-metadata	5.0.0	Paquete python
Angular (Typescript)	14.2.4	Framework para el desarrollo de apps
Git	2.25.1	Software para el control de versiones

2.3. Implementación de la solución de software para la gestión de bodega

2.3.1. Backend - API REST

La estructura del proyecto en la parte Backend es la siguiente:



```
models.py - GestionBodega - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER
  GESTIONBODEGA
    accounts
    appbodega
    __pycache__
    migrations
    __init__.py
    admin.py
    apps.py
    models.py
    serializers.py
    tests.py
    urls.py
    views.py
  GestionBodega
    public
    db.sqlite3
    manage.py

models.py
1 from django.db import models
2
3 class TipoProducto(models.Model):
4     id=models.AutoField(primary_key=True)
5     descripcion = models.TextField(max_length=50,help_text="Ingrese el Tipo de producto")
6     fechaRegistroTipoProducto=models.DateTimeField(auto_now=True)
7     def __str__(self):
8         return self.descripcion
9     class Meta:
10         ordering=['id']
11
12 class Producto(models.Model):
13     id=models.AutoField(primary_key=True)
14     descripcion = models.TextField(max_length=50,help_text="Ingrese la descripcion del producto")
15     cantidad= models.IntegerField(default=0)
16     tipoProducto=models.ForeignKey(TipoProducto, on_delete=models.CASCADE)
17     fechaRegistroProducto=models.DateTimeField(auto_now=True)
18     estado= models.BooleanField(blank=True, null=True)
19     def __str__(self):
20         return self.descripcion
21     class Meta:
22         ordering=['descripcion']
23
24 class LoteProducto(models.Model):
25     id=models.AutoField(primary_key=True)
26     numeroLote = models.TextField(help_text="Ingrese el numero de lote del producto")
27     cantidad= models.IntegerField()
28     precioUnitario = models.DecimalField(max_digits=10, decimal_places=5)
29     fechaCaducidad=models.DateField(blank=True, null=True)
30     producto=models.ForeignKey(Producto, on_delete=models.SET_NULL, null=True)
31     def __str__(self):
32         return self.numeroLote
33     class Meta:
34         ordering=['id']
35
36 class Proveedor(models.Model):
37     id=models.AutoField(primary key=True)
38
39 PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
[11/Nov/2022 17:53:00] "GET /bodega/producto/6 HTTP/1.1" 200 142
/home/freddy/PROYECTOSPYPYTHON/RepoGestionBodega3/GestionBodega/appbodega/models.py changed, reloading.
Watching for file changes with StatReloader
Performing system checks...
System check identified no issues (0 silenced).
November 11, 2022 - 17:55:03
Django version 3.2.5, using settings 'GestionBodega.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

Explicación:

- El modelado de clases se realizó en el archivo **models.py**. A cada clase se agregó el método especial llamado **__str__** que debe devolver una cadena de caracteres con lo que queremos mostrar. Ese método se invoca cada vez que se llama a la función str.
- En el archivo **admin.py** podemos registrar modelos para emplearlos en una aplicación que nos ofrece Django para realizar la gestión de datos.
- En el archivo **settings.py** del proyecto registramos nuestras aplicaciones en la opción llamada **INSTALLED_APPS**. Este archivo además nos permite realizar ajustes/configuración para este proyecto Django; nos define un

conjunto de configuraciones que podemos aplicar sobre el proyecto de manera global.

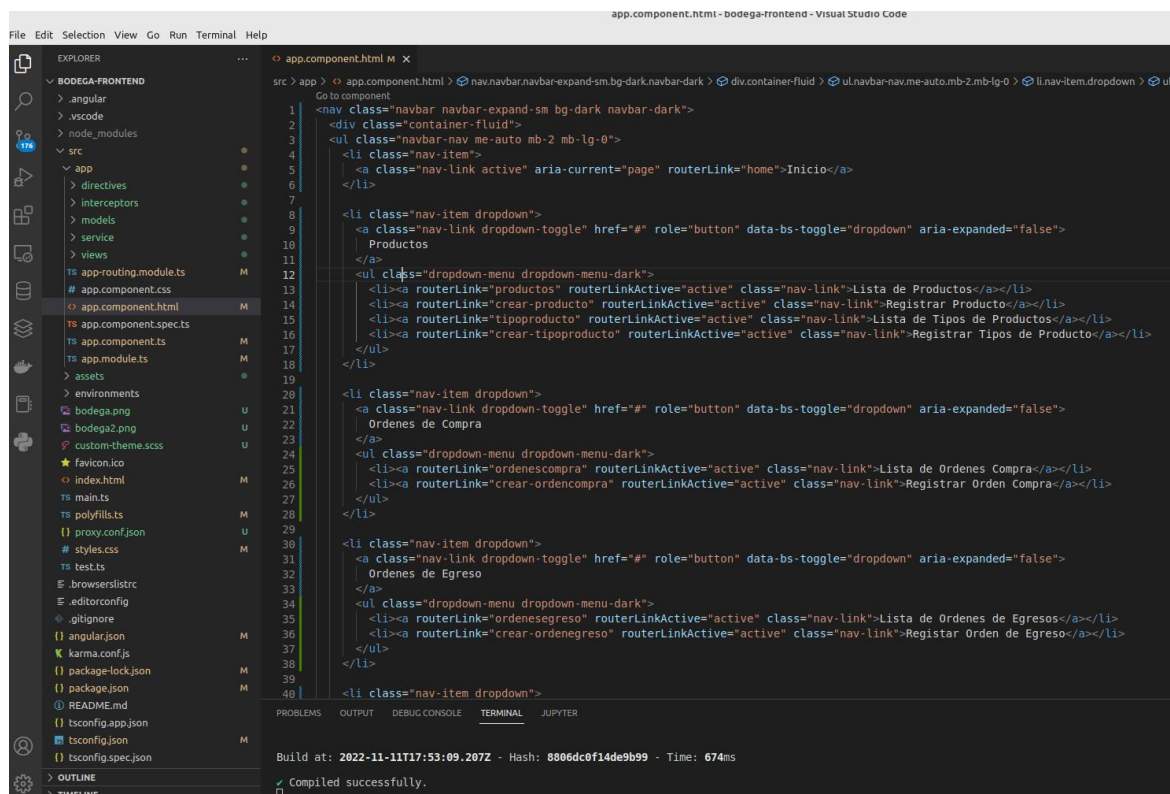
- El archivo **urls.py** nos permite realizar las declaraciones URL para el proyecto Django; desde aquí podemos cargar más archivos de URLs de otras aplicaciones o dependencias.
- El archivo **serializers.py** permitió que los datos de las consultas e instancias de modelos, se conviertan en tipos de datos nativos de Python que luego se pueden representar fácilmente en u otros tipos de contenido JSON o XML. Los serializadores también brindan deserialización, lo que permite que los datos analizados se vuelvan a convertir en tipos complejos, después de validar primero los datos entrantes.
- La base de datos tipo SQLite se encuentra en el archivo **db.sqlite3**, es donde se almacenarán todos los datos ingresados en el sistema de Gestión de Bodega.

Resumen de actividades:

- Se creó un entorno de aislado de python con virtualenv para el desarrollo de la aplicación web.
- Se instaló los paquetes de python necesarios para implementar el API REST para la gestión de bodega (Django, django-cors-headers, django-filter, djangorestframework, Markdown, sqlparse).
- Se procedió con la creación de un proyecto de Django.
- Se creó el super usuario y panel de administración de la aplicación Django.
- Se creó la aplicación GestionBodega para la implementación del API REST.
- Se subió al repositorio Git la API para su revisión.
- Adicionalmente se implementó el API para la autenticación de usuarios registrados en el panel de administrador de Django, para el uso desde aplicaciones externas.

2.3.2. Frontend - App Web

La estructura del proyecto en la parte Frontend es la siguiente:



```
1 <nav class="navbar navbar-expand-sm bg-dark navbar-dark">
2   <div class="container-fluid">
3     <ul class="navbar-nav me-auto mb-2 mb-lg-0">
4       <li class="nav-item">
5         <a class="nav-link active" aria-current="page" routerLink="home">Inicio</a>
6       </li>
7     <li class="nav-item dropdown">
8       <a class="nav-link dropdown-toggle" href="#" role="button" data-bs-toggle="dropdown" aria-expanded="false">
9         Productos
10      </a>
11      <ul class="dropdown-menu dropdown-menu-dark">
12        <li><a routerLink="productos" routerLinkActive="active" class="nav-link">Lista de Productos</a></li>
13        <li><a routerLink="crear-producto" routerLinkActive="active" class="nav-link">Registrar Producto</a></li>
14        <li><a routerLink="tipoproducto" routerLinkActive="active" class="nav-link">Lista de Tipos de Productos</a></li>
15        <li><a routerLink="crear-tipoproducto" routerLinkActive="active" class="nav-link">Registrar Tipos de Producto</a></li>
16      </ul>
17    </li>
18    <li class="nav-item dropdown">
19      <a class="nav-link dropdown-toggle" href="#" role="button" data-bs-toggle="dropdown" aria-expanded="false">
20        Ordenes de Compra
21      </a>
22      <ul class="dropdown-menu dropdown-menu-dark">
23        <li><a routerLink="ordenescompra" routerLinkActive="active" class="nav-link">Lista de Ordenes Compra</a></li>
24        <li><a routerLink="crear-ordencompra" routerLinkActive="active" class="nav-link">Registrar Orden Compra</a></li>
25      </ul>
26    </li>
27    <li class="nav-item dropdown">
28      <a class="nav-link dropdown-toggle" href="#" role="button" data-bs-toggle="dropdown" aria-expanded="false">
29        Ordenes de Egreso
30      </a>
31      <ul class="dropdown-menu dropdown-menu-dark">
32        <li><a routerLink="ordenesegreso" routerLinkActive="active" class="nav-link">Lista de Ordenes de Egresos</a></li>
33        <li><a routerLink="crear-ordenegreso" routerLinkActive="active" class="nav-link">Registrar Orden de Egreso</a></li>
34      </ul>
35    </li>
36  </ul>
37 </div>
38 </nav>
```

Explicación:

- En el directorio **models** se definió las clases con los mismos atributos declarados en el backend.
- El directorio **interceptors** contiene los interceptores (errores, autenticación) que inspecciona las peticiones y respuestas de la aplicación web realizadas desde y hacia el servidor API REST.
- En angular las directivas permiten extender las funcionalidades del HTML, en este caso permitieron restringir el ingreso de datos en los formularios de la aplicación en el caso puntual para el ingreso de números enteros y decimales, el código de las directivas está ubicado en el directorio **directivas** de la aplicación web.
- En el directorio **service** se encuentran los proveedores de datos de la aplicación web denominados **servicios**. Mantienen la lógica de acceso al backend y tienen la responsabilidad de acceder a la información y realizar operaciones con los datos. Los servicios son consumidos por los componentes.
- En el directorio **views** se encuentran los **componentes** que contienen vistas (.html) que permiten al usuario visualizar una interfaz, estilos (.css) que

permiten modificar la interfaz con estilos agradables y la lógica (.ts) que contiene las propiedades y métodos que se usan en las vistas.

Resumen de actividades:

- Se creó un proyecto con angular
- Se creó una aplicación web con Angular.
- Se estableció el modelo de datos mediante la definición de clases.
- Se creó los componentes necesarios y codificó los mismos para ofrecer al usuario las interfaces web en el sistema de Gestión de Bodega.
- Se creó los servicios para la comunicación de la aplicación web con el API REST del backend.
- Se estableció los interceptores para manejar los errores que reciba la aplicación web desde el backend, además se utilizó un interceptor para establecer la autorización de peticiones de usuarios con sesiones activas.
- Las directivas permitieron establecer controles en los formularios de ingreso de datos, específicamente se utilizaron en el presente trabajo para controlar el ingreso de números enteros y decimales.

2.3.3. Ejecución del proyecto

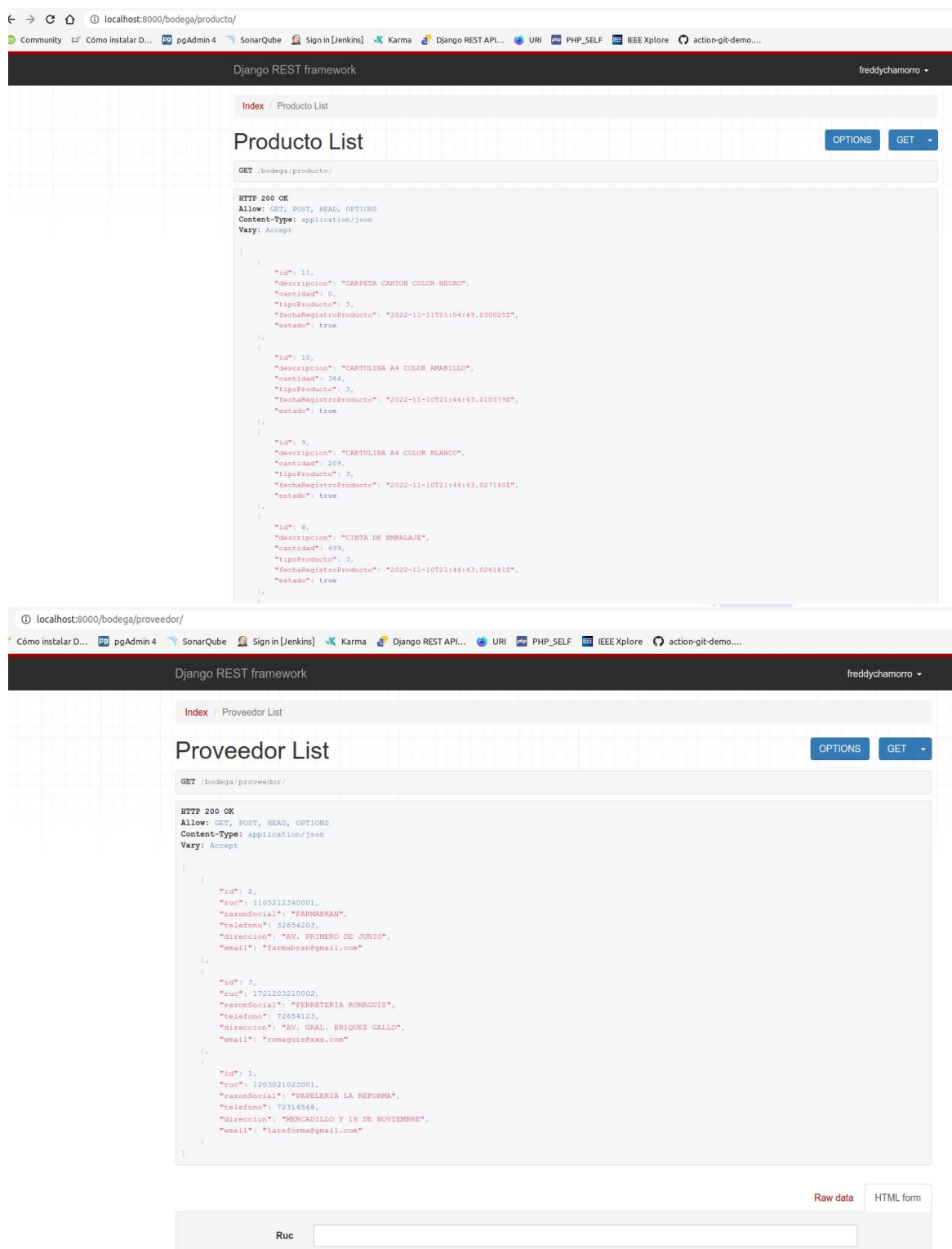
Para ejecutar el proyecto utilizamos los siguientes comandos:

```
#backend
#para la ejecución del proyecto backend utilizamos el comando por defecto se
ejecuta en el puerto 8000
python manage.py runserver

#frontend
#para la ejecución del proyecto frontend utilizamos el comando por defecto se
ejecuta en el puerto 4200
ng serve -o
```

A continuación se muestra la ejecución de la aplicación web para la gestión de bodega.

Backend



The screenshot displays the Django REST framework API browser interface. The top section shows the 'Producto List' endpoint at `GET /bodega/producto/`. The response is a JSON array of product objects, each containing fields like `id`, `descripcion`, `cantidad`, `tipoProducto`, `fechaRegistroProducto`, and `estado`.

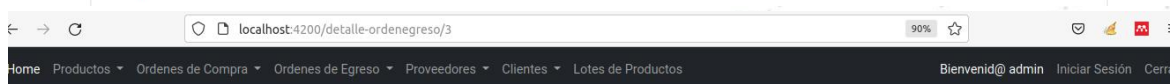
The bottom section shows the 'Proveedor List' endpoint at `GET /bodega/proveedor/`. The response is a JSON array of provider objects, each containing fields like `id`, `ruc`, `razonSocial`, `telefono`, `direccion`, and `email`.

At the bottom of the interface, there is a search bar labeled 'Ruc' and a toggle switch between 'Raw data' and 'HTML form'.



Universidad
Nacional
de Loja

Frontend



Orden de Egreso Nro. 3

Datos Generales

Fecha de Orden de Egreso

2022-11-10T21:44:42.904903Z

Cliente

FREDDY ALEXANDER CHAMORRO ENCALADA

Detalle de la Orden de Egreso

ID	Descripcion	Cantidad	Lote	Precio U.	Valor Total
5	CINTA DE EMBALAJE	3	8	0.12454	0.37362
4	CARTULINA A4 COLOR BLANCO	2	6	0.45555	0.91110
3	CARTULINA A4 COLOR AMARILLO	1	5	0.21455	0.21455
TOTAL					1.49927

Regresar

Imprimir

2.3.4. Código fuente

Repositorio en GitHub

<https://github.com/fachamorro/GestiondeBodega>

3. CONCLUSIONES

- La solución de software desarrollada permite administrar productos y sus tipos, proveedores y clientes. Además, permite administrar Órdenes de compra y egresos, e imprimir comprobantes de los mismos, previo al inicio de sesión del usuario registrado por el administrador.
- El caso de Uso Administrar Usuarios no se desarrollo en vista que Django facilita la administración de usuarios en el panel admin.

4. BIBLIOGRAFIA

- Barranquilla, P. (s/f-b). Una guía completa de Django para principiantes- Parte 1. <https://pybaq.co/>. Recuperado el 30 de octubre de 2022, de <https://pybaq.co/blog/una-guia-completa-de-django-para-principiantes-parte-1/>
- Barranquilla, P. (s/f-a). Una guía completa de Django para principiantes - Parte 2. <https://pybaq.co/>. Recuperado el 30 de octubre de 2022, de <https://pybaq.co/blog/una-guia-completa-de-django-para-principiantes-parte-2/>
- Django tutorial part 9: Working with forms. (s/f). Mozilla.org. Recuperado el 31 de octubre de 2022, de <https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Forms>
- Writing your first Django app, part 1. (s/f). Djangoproject.com. Recuperado el 31 de octubre de 2022, de <https://docs.djangoproject.com/en/4.1/intro/tutorial01/>
- Angular. (s/f). Angular.lat. Recuperado el 8 de noviembre de 2022, de <https://docs.angular.lat/cli>

5. FIRMAS

Nombres y Apellidos	Firma
Freddy Alexander Chamorro Encalada	