UNIVERSIDAD TÉCNICA DE AMBATO

Facultad de Ingeniería en Sistemas, Electrónica e Industrial

"Proyecto Académico de Fin de Semestre"

Título: Sistema de Gestión de Combustible de una Gasolinera

Carrera: Software

Ciclo Académico y Paralelo: Tercero "A"

Alumnos participantes: Oviedo Sánchez Bryan Steven

Saquinga Chango Kevin Alexander

I. INFORME DEL PROYECTO

1.1 Título

Sistema de Gestión de Combustible de una Gasolinera

1.2 Objetivos

Objetivo General

- Demostrar las diferentes herramientas, opciones que se ha trabajado, a través del programa starUML utilizando las diferentes plantillas UML que nos ofrece.

Objetivo Específico

- Reforzar los conocimientos de los diferentes modelados UML que se han trabajado para la realización de este proyecto.
- Conocer que herramientas se han utilizado para la transformación de diagramas UML hacia un lenguaje de programación.
- Demostrar la funcionalidad del programa cumpliendo todos los parámetros que se establecieron.

1.3 Resumen

El presente proyecto tiene como fin demostrar un sistema de gestión de combustible a través de una gasolinera, una manera de reflejar a través de diferentes modelados UML (Actividades, Caso de Uso, Clases, Secuencia, Despliegue) aprendidos en clase, siguiendo los pasos de estos diagramas cuidadosamente analizados, fue posible realizarla y a la vez transformarla a código siendo una buena guía para programar, además aplicamos los paradigmas orientados a objetos una manera más eficiente de organizar el codificado.

Se utilizó diferentes clases como son el Cliente, Administrador, Gestión de combustible, factura, etc., para distribuir los diferentes métodos, atributos para el programa. Por mientras a través de un servidor web llamado XAMPP generamos una base de datos en el phpMyAdmin, creamos una base llamada *gestorcombustible* así fue posible guardar cada uno de estos registros de los clientes una vez que hayan entrado al sistema, se espera a través de esto explotar todo el conocimiento reflejado en clases hacia el programa starUML y el codificado.

1.4 Introducción

Para la realización de un sistema de gestión de combustible a través de una gasolinera el lenguaje utilizado fue Python uno de los más curiosos para programar, es muy importante conocer las diferentes herramientas que nos proporciona, que se han utilizado y fueron necesarias, para que este programa funcione correctamente, adquiriendo nuevos conocimientos en la programación. También hay que tener en cuenta el orden de cómo se ha ido avanzando a través de los Diagramas UML, las bases para construir un determinado sistema y llevarlo a cabo mediante el codificado mucho más accesible y eficiente para él programador.

Este proyecto, nos ayuda a levantar muchos conocimientos ya que nos puede servir de base para desarrollar muchos proyectos más aplicando este tipo de metodología. Los materiales que se utilizaron fue Visual Studio Code una herramienta para poder programar en Python junto con sus respectivas librerías, y que este a la vez esté conectada a una base de datos para guardar los diferentes datos que deseamos almacenar.

Cuando se fue desarrollando este sistema, se han generado muchas preguntas y a la vez dudas de cómo puede implementarse este mecanismo, la forma de cómo debe estar bien diseñado y darle un correcto funcionamiento para que no haya algún tipo de error mientras se va desarrollando este sistema.

1.5 Materiales y Metodología

Materiales:

- Software: StarUML (Diagramas: Actividades, Casos de Uso, Clases, Secuencia, Despliegue)
- Lenguaje de Programación: Python
- Software: Visual Studio Code
- Software: XAMPP
- Hardware: Laptop

Metodología:

Para esto hay que primero conocer algunas definiciones básicas para poder comprender el proyecto:

- **Diagramas UML: "**Es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el Object Management Group (OMG). Un lenguaje gráfico que nos permite visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y compuestos reciclados" (Wikipedia, 2021).

- **Python: "**Es un lenguaje de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código. Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos,

programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, dinámico y multiplataforma. También posee una licencia de código abierto" (Wikipedia, 2021).

- **XAMPP: "**Es un paquete de software libre, que consiste principalmente en el sistema de gestión de bases de datos MySQL, el servidor web Apache y los intérpretes para lenguajes de script PHP y Perl. El nombre es en realidad un acrónimo: X (para cualquiera de los diferentes sistemas operativos), Apache, MariaDB/MySQL, PHP, Perl. El programa se distribuye con la licencia GNU y actúa como un servidor web libre, fácil de usar y capaz de interpretar páginas dinámicas" (Wikipedia, 2021).

- **Visual Studio Code: "**Es un editor de código fuente desarrollado por Microsoft para Windows, Linux y macOS. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código. También es gratuito y de código abierto" (Wikipedia, 2020).

Mediante el enunciado es la parte básica para crear diversas ideas de cómo realizar los diagramas UML y con eso la realización de la codificación correspondiente, los puntos están esquemáticamente ordenados como se muestra en la Ilustración 1.

Diagramas UML

Los diferentes Diagramas UML que se han utilizado son los siguientes:

Casos de Uso:

Es la descripción de una acción o actividad. También es una descripción de las actividades que deberá realizar alguien o algo para llevar a cabo algún proceso. Los personajes o entidades que participarán en un diagrama de caso de uso se denominan actores. En nuestro proyecto fue la base esencial para resaltar diferentes ideas a través de una descripción de cada contenido se enfocó cómo se debe elaborarlo, paso tras

paso entonces como se indica en las siguientes tablas correspondientes nuestra respectiva elaboración Tabla 1, Tabla 2, Tabla 3, Tabla 4, Tabla 5, Tabla 6.

Actividades:

Los diagramas de actividades tal y como su nombre lo menciona son diagramas que nos detallan la funcionalidad o como exactamente esta realizado el proyecto, es decir los pasos que tenemos que seguir al momento de codificar es por eso que en los diagramas de actividades debemos detallar completamente todos los pasos que se van a desarrollar. Es por eso que es uno de los diagramas que más detalla cómo sería la codificación. Para lo cual se elaboró en base a los Casos de Uso correspondientes para tenerlo como una guía como se indica en la Ilustración 2, Ilustración 3, Ilustración 4, Ilustración 5, Ilustración 6, Ilustración 7.

Secuencia:

Este describe cómo los objetos intercambian mensajes en un orden determinado como el usuario lo considere. Los objetos son los bloques de construcción básicos de los diagramas UML y esta vez representan características de un elemento del sistema, que varían dependiendo del diagrama. En el proyecto se utilizó como una ayuda de los Casos de Uso y Diagrama de Actividades formando pasos bien organizados, como se indica en la Ilustración 8, Ilustración 9, Ilustración 10, Ilustración 11, Ilustración 12, Ilustración 13.

Clases:

Este nos describe cómo está estructurado el sistema indicando las diferentes clases del sistema, sus atributos, métodos, y las relaciones entre los objetos. Para nosotros a través del enunciado como se indica en la Ilustración 1, las señalamos, y a través de eso las organizamos de forma organizada junto con su multiplicidad correspondiente como se indica en la Ilustración 14.

Despliegue:

Se lo representan como dos tipos de elementos, los nodos y las conexiones, estos a su vez van distribuidos a través de componentes del sistema de información con su respetivo almacenamiento de este sistema. Como el proyecto también se enfoca a través de una Base Datos lo tomamos en cuenta ya que debe tener una conexión con el lenguaje de programación Python con sus respectivas descripciones como se indica

en la Ilustración 15.

Codificación:

En la codificación correspondiente se realizó los diversos controles en el proyecto propuesto además nos ayudamos con la guía de los Diagramas UML explicados anteriormente se recopiló la información respectiva, las observaciones de cada una de ellas para poder ser implementadas. En la Ilustración 16, Ilustración 17, Ilustración 18, Ilustración 19, Ilustración 20, Ilustración 21, Ilustración 22, Ilustración 23, se puede visualizar las diferentes Clases que se implementaron gracias al Diagrama de Clases, con sus respectivos atributos y diferentes métodos. Y en la Ilustración 24 es la ejecución del programa en sí con todo lo aprendido anteriormente.

Funcionalidades

El programa nos permitirá manejar un sistema de gestión de combustible, comunicándose entre sí con la base de datos *gestorcombustible* para así simularlo y ayudándonos a adquirir nuevos conocimientos.

Requisitos mínimos del Computador.

El programa funciona para todos los computadores ya que no abarca de muchas herramientas para que este puede ser ejecutada. En nuestro caso los requisitos mínimos que utilizamos fueron:

Edición de Windows

Windows 10 Home

Sistema

Procesador: AMO Ryzen 5 3550H with Radeon Vega Mobile Gfx
2.1O GHz Memoria instalada (RAM): 8,00 GB (5,88 GB utilizable)

Tipo de sistema: Sistema operativo de 64 bits,procesador x64

Edición de Windows Windows 10 Pro

Sistema

Procesador: lntel(R) Core(TM) iS-7200U CPU @ 2.5OGHz 2.70 GHz

Memoria instalada (RAM): 8,00 GB (1,78 GB utilizable)

Tipo de sistema: Sistema operativo de 64 bits, procesador x64

1.6 Resultados y Discusión

Se logró generar un sistema de gestión de combustible a través de los diagramas UML que se especificaron anteriormente siguiendo todos los pasos correspondientemente, así transformar a código formando un menú de opciones y cada uno que tenga una funcionalidad diferente, el cual está enfocado a la venta de combustibles si es cliente o si el caso de ser administrador, ver los despachos, reportes. Mediante el desarrollo se ha generado ciertos errores sean en los diagramas para lo cual nos enfocamos a las bases para hacer un determinado programa a Python, mientras fuimos avanzando pudimos aclarar las dudas que tuvimos, generando una programación correcta de los puntos que hemos establecido, como paso final hicimos cambios necesarios en los Diagramas UML y eliminando datos innecesarios tanto en el Diagrama UML como en el codificado.

Sin embargo, al momento de realizar este proyecto nos hemos enfocado de utilizar el lenguaje Python entonces surgen las preguntas: ¿Qué pasaría si utilizaríamos otro lenguaje de programación? ¿Se haría la misma manera de programar? ¿Qué cambios se deben hacer para el funcionamiento de ésta? Y no solo el lenguaje también en el servidor web que hemos aplicado. ¿Qué pasaría en vez de instalar un servidor instalamos un software enfocado a la base datos como el ServerManager?

1.7 Conclusiones

Mediante los conocimientos que hemos abarcado durante la materia, hemos aprendido

a desarrollar diferentes modelados UML para la realización de este proyecto, la cual starUML nos ayudó a diseñar y tener una observación diferente de cómo guiarnos y programar.

Existen diferentes tipos de software que nos permiten modelar, el más apropiado fue starUML, ya que es gratuita, sencilla, potente y fiable al momento de realizar los diagramas UML, sin embargo, la mayoría de estos programas son de paga y no es conveniente utilizarlos para este tipo de proyecto.

Demostramos la funcionalidad de este programa, ya que las opciones que se establecieron fueron básicas para la ejecución cumpliendo así todos los parámetros que hemos abarcado, generando un conocimiento más amplio y una manera diferente de pensar de cómo interpretar los diferentes diagramas UML, lo cual no fue sencillo al inicio, pero con esfuerzo y dedicación pudimos llevarlo a codificación, aclarando las diversas dudas y demostrándolo hacia el proyecto.

1.8 Referencias bibliográficas

Lenguaje unificado de modelado. (2021, 15 de enero). *Wikipedia, La enciclopedia libre*. desde https://es.wikipedia.org/w/index.php?title=Lenguaje_unificado_de_modelado&oldid=132428852.

Python. (2021, 26 de enero). *Wikipedia, La enciclopedia libre*. desde https://es.wikipedia.org/w/index.php?title=Python&oldid=132722206.

Visual Studio Code. (2020, 31 de diciembre). *Wikipedia, La enciclopedia libre*. desde https://es.wikipedia.org/w/index.php?title=Visual_Studio_Code&oldid=132071565

XAMPP. (2021, 20 de enero). *Wikipedia, La enciclopedia libre*. desde https://es.wikipedia.org/w/index.php?title=XAMPP&oldid=132552436.

2.10. Anexos

Sistema De gestión De Combustible

Una empresa desea generar un sistema que le ayude con la gestión de combustible el cual ellos necesiten que cumplan con los siguientes requerimientos:

1. El sistema tiene que registrar la compra de los diferentes tipos de combustibles (Diesel, Gasolina)

2. El sistema tiene crear al cliente con sus respectivos datos y almacenando en la base de datos de la gasolinera y su el cliente ya existe el sistema debe buscar sus datos en la base de datos

3. El sistema debe poder registrar la venta de combustibles.

4. El sistema debe estar en las condiciones de aceptar pagos en efectivo y tarjeta de crédito.

5. Si el pago es por tarjeta de crédito el sistema debe verificar si tiene fondos.

6. El sistema tiene que verificar si es pago por tarjeta de crédito se debe aplicar un descuente por ser cliente afiliado caso contrario si el pago

es en efectivo no se va a efectuar el descuento.

7. Una vez ingresa todos los datos y validado si aplica o no descuento el sistema debe mostrar el costo a pagar por la compra de combustible.

8. EL sistema al momento que inicie debe mostrar las opciones de 1. Administrador 2. Cliente.

9. Si el sistema de va por la opción de administrador tiene el siguiente menú 1. Compra de combustible 2. Ver despacho de combustible 3. Ver reporte de ventas 4. Ver el reporte de combustible 5. Salir

10. Si el sistema se va por la opción de cliente se muestra el siguiente menú 1. Ingresar 2. Registrarse, si va por la opción 1 muestra las siguientes opciones Compra gasolina y si va por la segunda opción se debe registrar primero y de ahí ir a la opción número 1.

Ilustración 1: Enunciado Proyecto

NOMBRE	Compra Combustible
Actores	Administrador
Descripción	Registrar los diferentes tipos de combustible que está encargada la empresa para su compra.
Disparador	Notificar medio de un mensaje la compra de un tipo de combustible.
Precondiciones	1. El administrador debe estar logueado al sistema. 2. La empresa a la que se va a realizar la compra debe llevar los registros de sus despachos de combustibles.

Postcondiciones	1. Queda registrado la compra de combustible. 2. Se registra en la bitácora los diferentes tipos de combustibles. 3. Se le notifica al administrador vía correo electrónico la compra de combustible.
Flujo normal	1. El cliente debe estar logueado al sistema 2. El administrador solicita la compra de combustible. 3. El sistema muestra la interfaz de compra con los datos necesarios para entrar. 4. El administrador registra la compra de combustible (S1). 5. El sistema valida la compra de combustible. 6. El sistema envía un notifica los datos de la compra de combustible al administrador. 7. El sistema registra en la bitácora de compra de combustible. 8. Finaliza el caso de uso
Flujos alternativos	S1. El administrador abandona el registro del caso de uso. S1.1 El sistema pregunta al administrador si desea abandonar el proceso.

	S1.2 El administrador si dice que si, el sistema notifica al administrador que no se pudo completar con la compra.
Excepciones	E1. El administrar no registra todos los datos obligatorios para a compra. E1.1 El sistema debe indicar que faltan datos obligatorios. E1.2 Ir al paso 3 del flujo normal. E2 El administrador registra un valor incorrecto al momento de realizar la compra. E2.1 El sistema notifica el error que se cometió. E2.2 Ir al paso 4 del flujo normal.
Reglas de negocio	

Tabla 1: Caso de Uso.- Compra Combustible

NOMBRE	Crear cliente
Actores	Cliente
Descripción	El sistema registrar los datos del cliente para la venta de combustible
Disparador	Notificación al cliente por medio de un mensaje la venta de combustible.
Precondiciones	1. El cliente debe estar logueado al sistema.
Postcondiciones	1. Queda registrado y activo el cliente 2. Se le informa al cliente de los diferentes tipos de combustibles.

Flujo normal	1. El sistema muestra una interfaz con los datos necesarios para crear un cliente. 2. El cliente registra los datos (S1). 2.1 El cliente crea su usuario y su clave para acceso al sistema. 3. El sistema valida los datos. 4. El sistema guarda los datos y los almacena en una base de datos. 5. El sistema notificación con un mensaje que sus datos fueron almacenados. 6. Finaliza el caso de uso
Flujos alternativos	S1. El cliente abandona la ejecución del caso de uso S1.1 El sistema pregunta si desea abandonar registro de datos S1.2 Si el cliente dice que sí, se notificara por medio de un mensaje que abandono el registro de datos.
Excepciones	E1. Faltan datos por llenar. E1.1 El sistema debe indicar que faltan datos. E1.2 Ir al paso 2 del flujo normal. E2. El cliente ya existe en la base de datos. E2.1 El sistema debe notificar que el cliente ya existe. E2.2 Ir al paso 2 de flujo normal E3. El nombre del usuario ya existe. E3.1 Hay que notificar que ya hay un usuario con ese mismo nombre E3.2 Ir al paso 2.1 del flujo normal.
Reglas de negocio	

Tabla 2: Caso de Uso.- Crear Cliente

NOMBRE	Venta Combustible
Actores	Cliente
Descripción	Notificar los diferentes tipos de combustible que existen para su venta.

Disparador	Notificar por mensaje la venta de combustible.
Precondiciones	1. El cliente debe estar logueado al sistema. 2. El sistema debe llevar el registro de ventas de combustible de la gasolinera.
Postcondiciones	1. Queda registrado la venta de combustible. 2. Se registra en las bitácoras las diferentes ventas de combustibles. 3. Se le notifica al cliente por mensaje la venta de combustible.
Flujo normal	1. El sistema solicita la venta de combustible. 2. El sistema muestra la interfaz de venta de combustible. 3. El cliente ingresa sus respectivos datos. 4. El sistema busca los datos en la base de datos de la gasolinera. 5. El sistema despliega los datos del cliente y muestra el menú de opciones de venta de combustible. 5.1 El cliente elige el tipo de combustible deseado (S1). 5.2 El cliente realiza la compra de combustible. 6. El sistema valida la venta de combustible. 7. El sistema notifica por mensaje que su venta se ha realizado.
	8. El sistema registra en las bitácoras de venta de combustible por día y por mes. 9. Finaliza el caso de uso
Flujos alternativos	S1. El cliente abandona al momento de elegir el tipo de combustible. S1.1 El sistema pregunta al cliente si desea abandonar el proceso. S1.2 El cliente si dice que si, el sistema notifica que no se pudo completar con la venta.

Excepciones	E1. El cliente no registra todos los datos. E1.1 El sistema debe indicar que faltan datos. E1.2 Ir al paso 3 del flujo normal. E2 El sistema no encuentra al cliente, por ingresar mal sus datos. E2.1 El sistema notifica al cliente que no se encuentran sus datos. E2.2 Ir al paso 4 del flujo normal. E3 El cliente ingresa un tipo de combustible que no existe E3.1 El sistema notificara que no hay ese tipo de gasolina. E3.2 Ir al paso 5.1 del flujo normal.
Reglas de negocio	

Tabla 3: Caso de Uso.- Venta de Combustible

NOMBRE	Reportes
Actores	Sistema
Descripción	Registrar los reportes de compra y venta de combustible.
Disparador	Notificar sobre los diferentes reportes realizados por día y por mes.
Precondiciones	1. El programa a la que se va a llenar el reporte ya debe estar creado.
Postcondiciones	1. El sistema asigna a los diferentes reportes que existan. 2. Se notifica por medio de un mensaje los diferentes reportes.

Flujo normal	1. El sistema uno vez terminada la venta y compra de combustible registra en los diferentes reportes que existen (S1). 2. El sistema guarda los reportes. 3. El sistema notifica que los reportes han sido guardados 4. Fin de caso de uso
Flujos alternativos	S1. El sistema no termino de realizar la venta y compra de combustible S1.1. El sistema notificara por mensaje que no se puede registrar los reportes porque no se completó el proceso de compra y venta.
Excepciones	E1. El sistema no registró los reportes. E1.1 El sistema notificara por mensaje que no se registró los reportes. E1.2 Ir al paso 1 del flujo normal. E2. El sistema no guardo el registro en los reportes E2.1 El sistema notifica por mensaje que no se guardó los reportes E2.2 Ir al paso 2 del flujo normal. E3. El sistema guarda los registros de los reportes en un reporte diferente. E3.1 Notificar por mensaje que ese registro no pertenece a este reporte E3.1 Ir al paso 1 del flujo normal.
Reglas de negocio	

Tabla 4: Caso de Uso.- Reportes

NOMBRE	Facturación
Actores	Sistema
Descripción	Realizar la de las diferentes ventas.
Disparador	Notificar al cliente sobre sus diferentes compras realizadas.
Precondiciones	1. El cliente debe estar logueado en el sistema. 2. La cuenta del cliente ya debe estar creada. 3. La base de datos de los usuarios ya debe estar creada.

Postcondiciones	1. El cliente es registrado y activo. 2. El sistema indica los diferentes tipos de combustibles al cliente.
Flujo normal	1. El cliente ingresa sus datos 2. El sistema busca los datos en la base de datos 3. El sistema despliega los datos del cliente. 4. El sistema muestra las opciones de pago (S1). 5. El sistema efectúa y valida el pago. 6. El sistema imprime la facturación. 7. Finaliza el caso de uso
Flujos alternativos	S1. Si el pago es por tarjeta el sistema efectúa un descuento por ser cliente afiliado caso contrario si el pago es en efectivo no aplica descuento. S1.1. El sistema verifica que tipo de pago se realiza.
Excepciones	E1. No se ingresaron todos los datos. E1.1 El sistema debe indicar que faltan datos. E1.2 Ir al paso 1 del flujo normal E2. El cliente no aparece en la base de datos E2.1 El sistema debe notificar que el cliente no existe. E2.2 Ir al paso 2 de flujo normal E3. El número de tarjeta es el incorrecto. E3.1 Notificar al cliente que el número de la tarjeta es invalida. E3.2 Ir al paso 4 del flujo normal
Reglas de negocio	

Tabla 5: Caso de Uso.- Facturación

NOMBRE	Crear Base Datos
Actores	Programador
Descripción	Crear la base de datos para ingresar los datos de los clientes
Disparador	Notificación por mensaje que su base de datos ha sido creada
Precondiciones	1. El programador debe saber qué tipo de lenguaje va a utilizar 2. El programador el conocimiento adecuado para base de datos

Postcondiciones	1. El programador ya sabe que tipo de lenguaje utilizara 2. El programador analiza los campos que va a necesitar
Flujo normal	1. El programador elige el lenguaje de programación para empezar a crear 2. El programador crea la tabla (S1)
	3. El programador define y crea su clave primaria 4. El programador crea el campo nombre, apellido y contraseña 5. El programador crea el método ingresar 6. El programador crea el método guardar 7. El programador crea el método visualizar 8. Fin del caso de Uso
Flujos alternativos	S1. El programador crea una tabla que ya existe S1.1 El sistema notifica que esa tabla ya está creada S1.2 El programador decide abandonar la creación de la tabla porque ya existe.
Excepciones	E1. Se ha definido mal la clave primaria E1.1 El sistema debe indicar que clave primaria está mal definida. E1.2 Ir al paso 3 del flujo normal E2. El cliente no cumple con el formato especifico E2.1 El sistema debe notificar que el formato no es correcto E2.2 Ir paso 5 del flujo normal.
Reglas de negocio	

Tabla 6: Caso de Uso.- Crear una Base de Datos

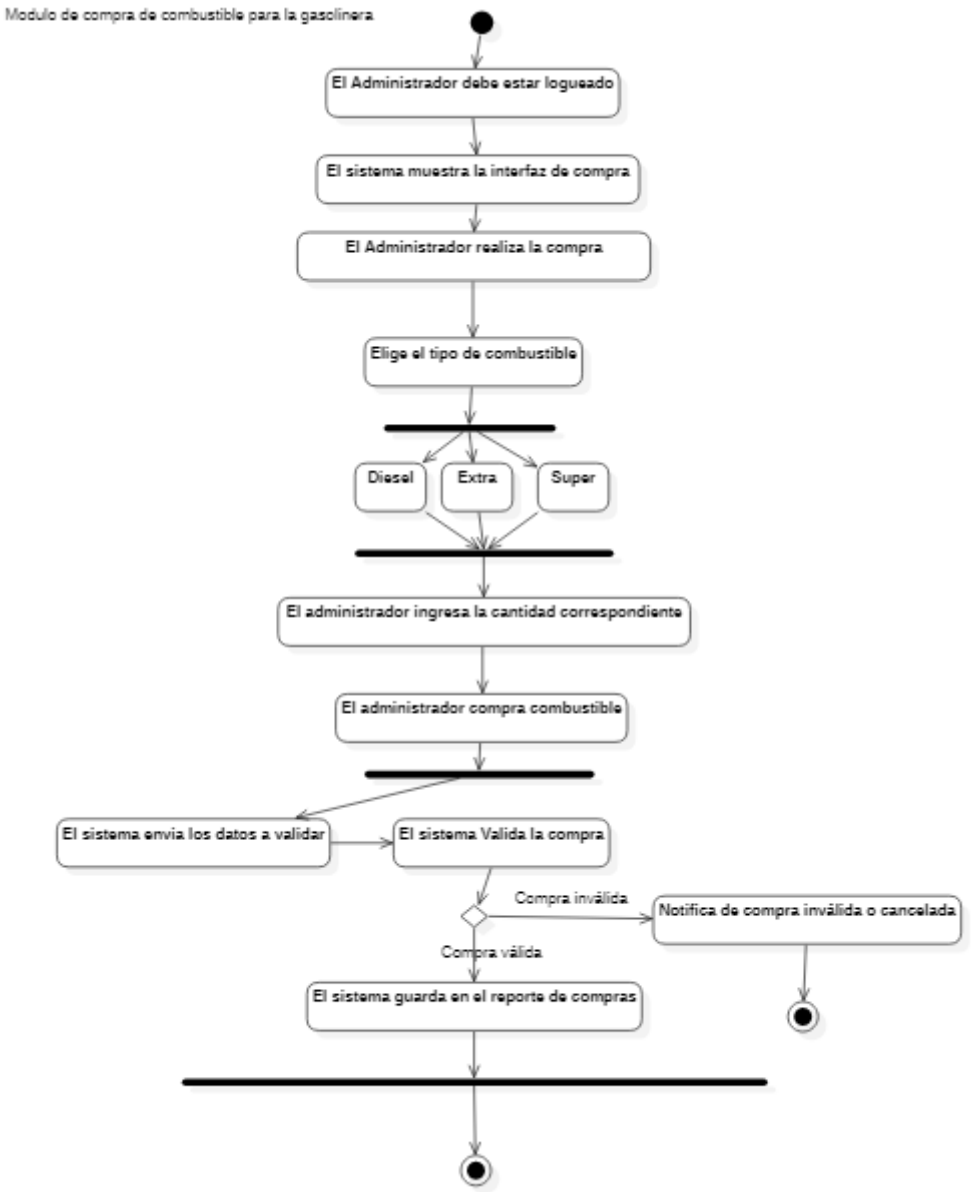

Ilustración 2: Diagrama Actividad.- Compra de Combustible

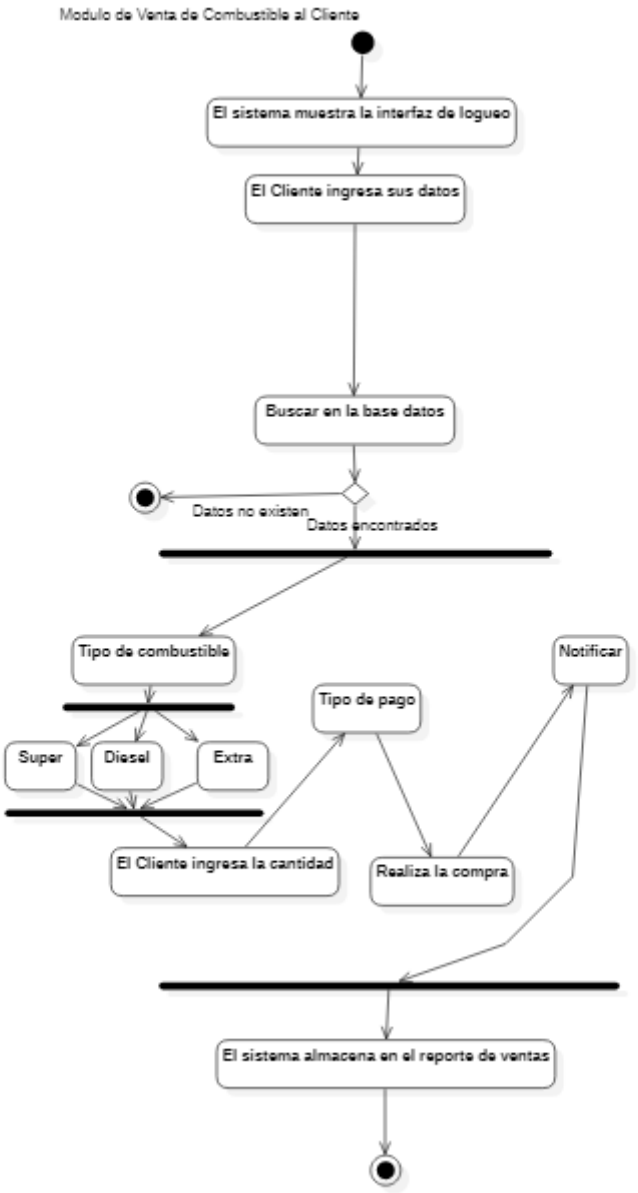

Ilustración 3: Diagrama Actividad. - Venta de Combustible

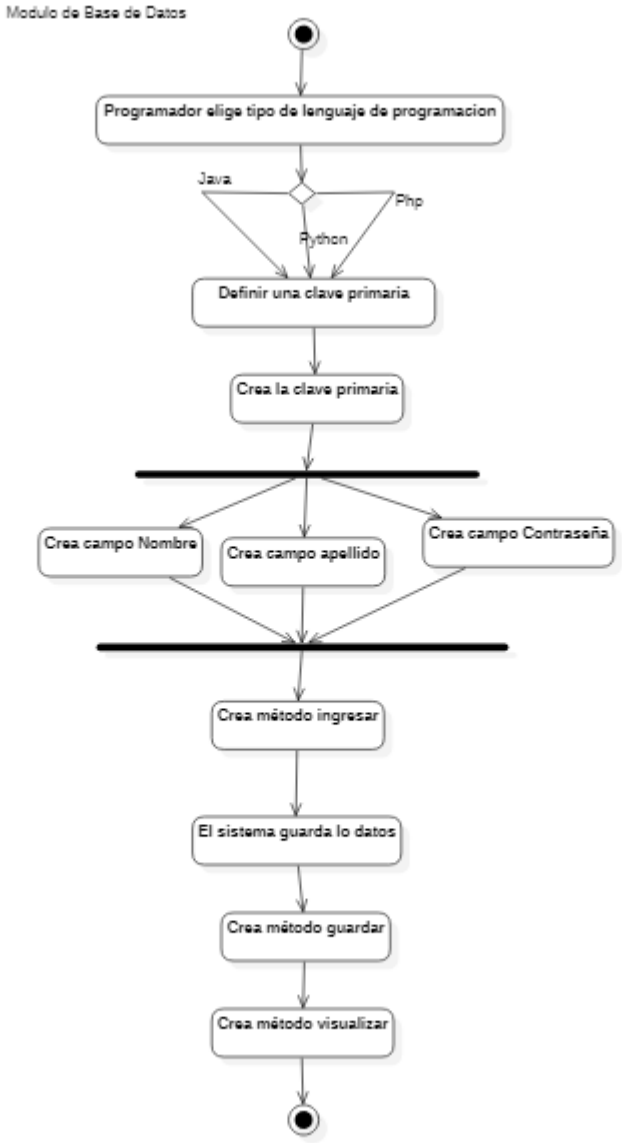

Ilustración 4: Diagrama Actividad. – Base de Datos

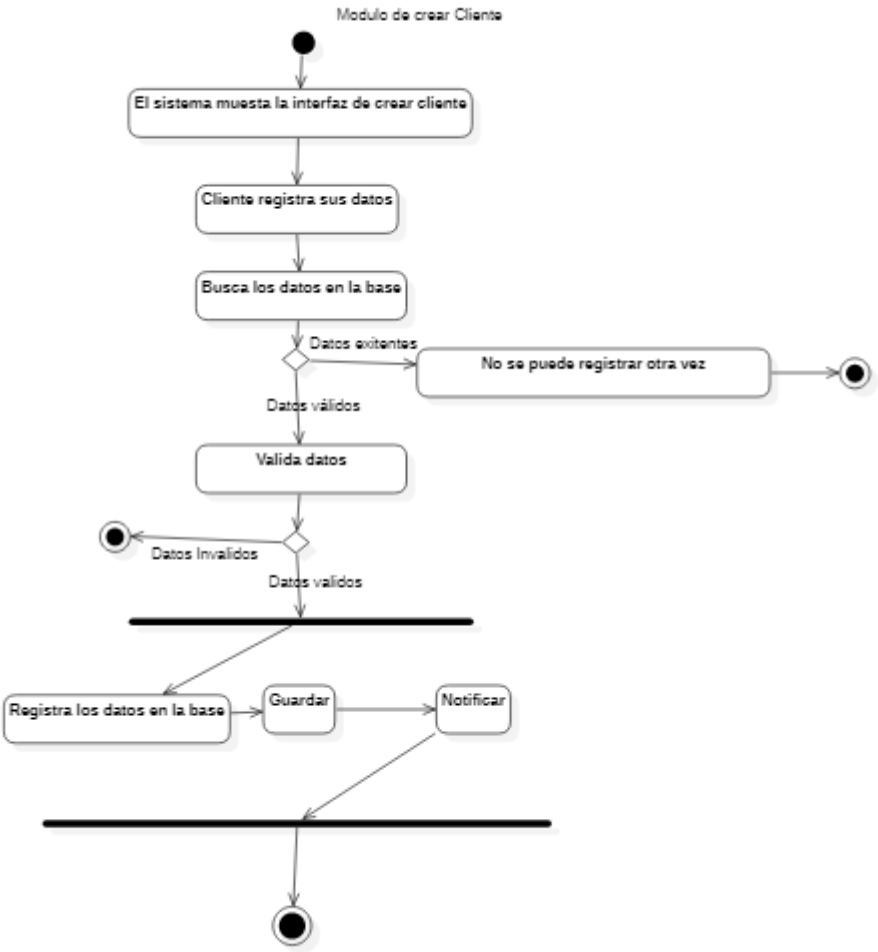

Ilustración 5: Diagrama Actividad. – Crear Cliente

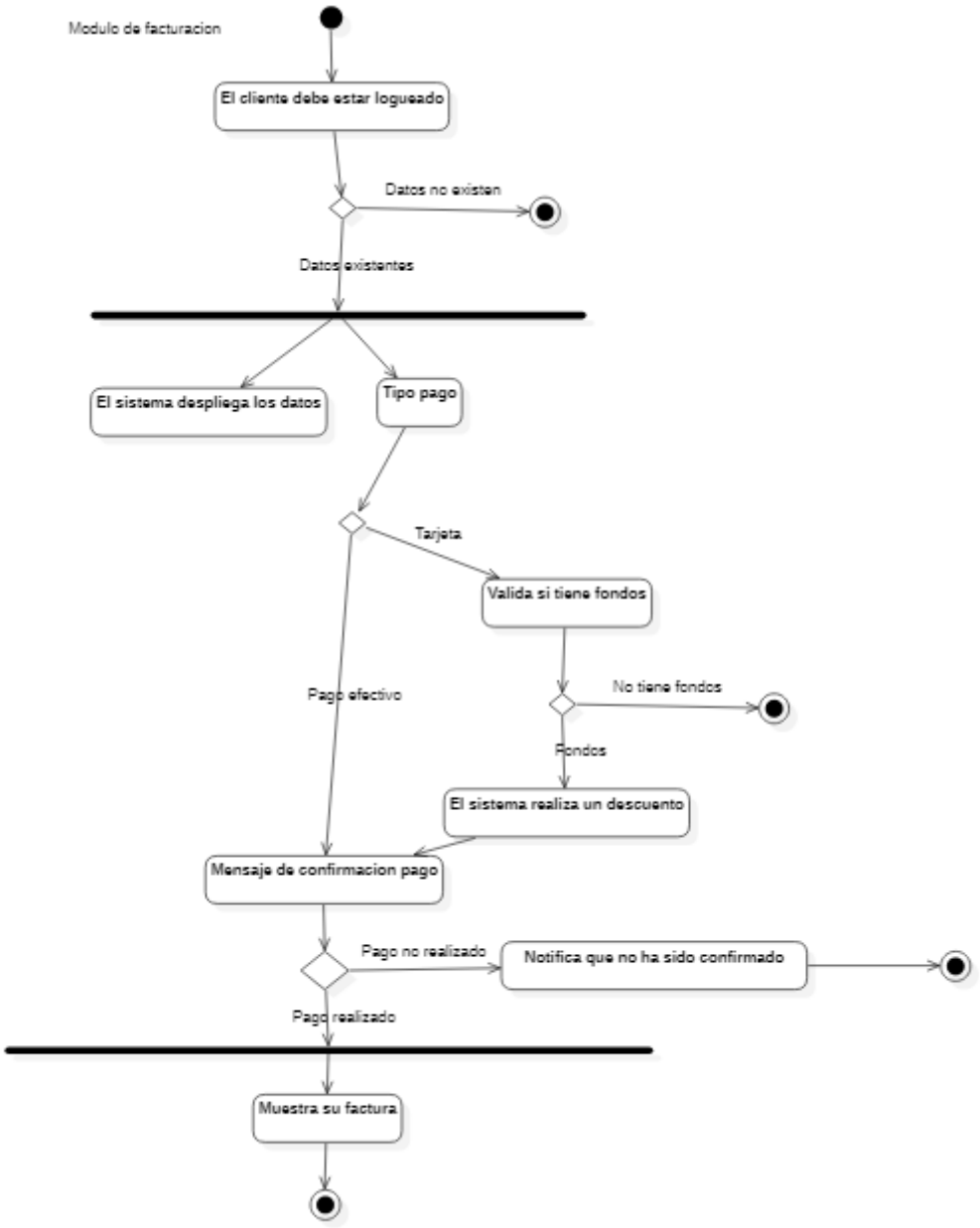

Ilustración 6: Diagrama Actividad. – Facturación

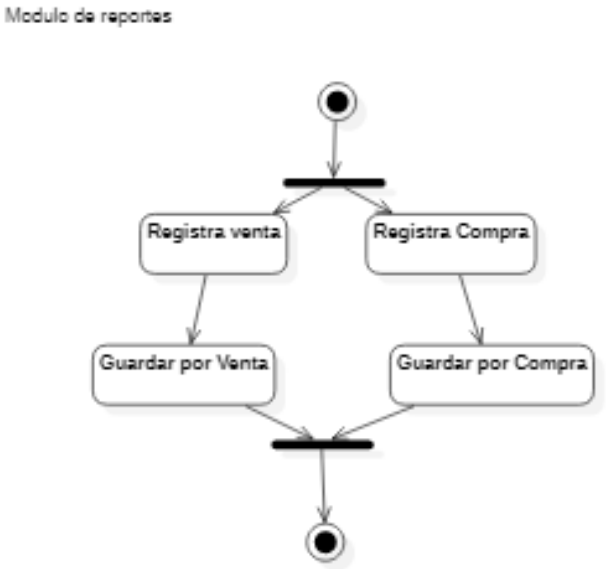

Ilustración 7: Diagrama Actividad. – Reportes

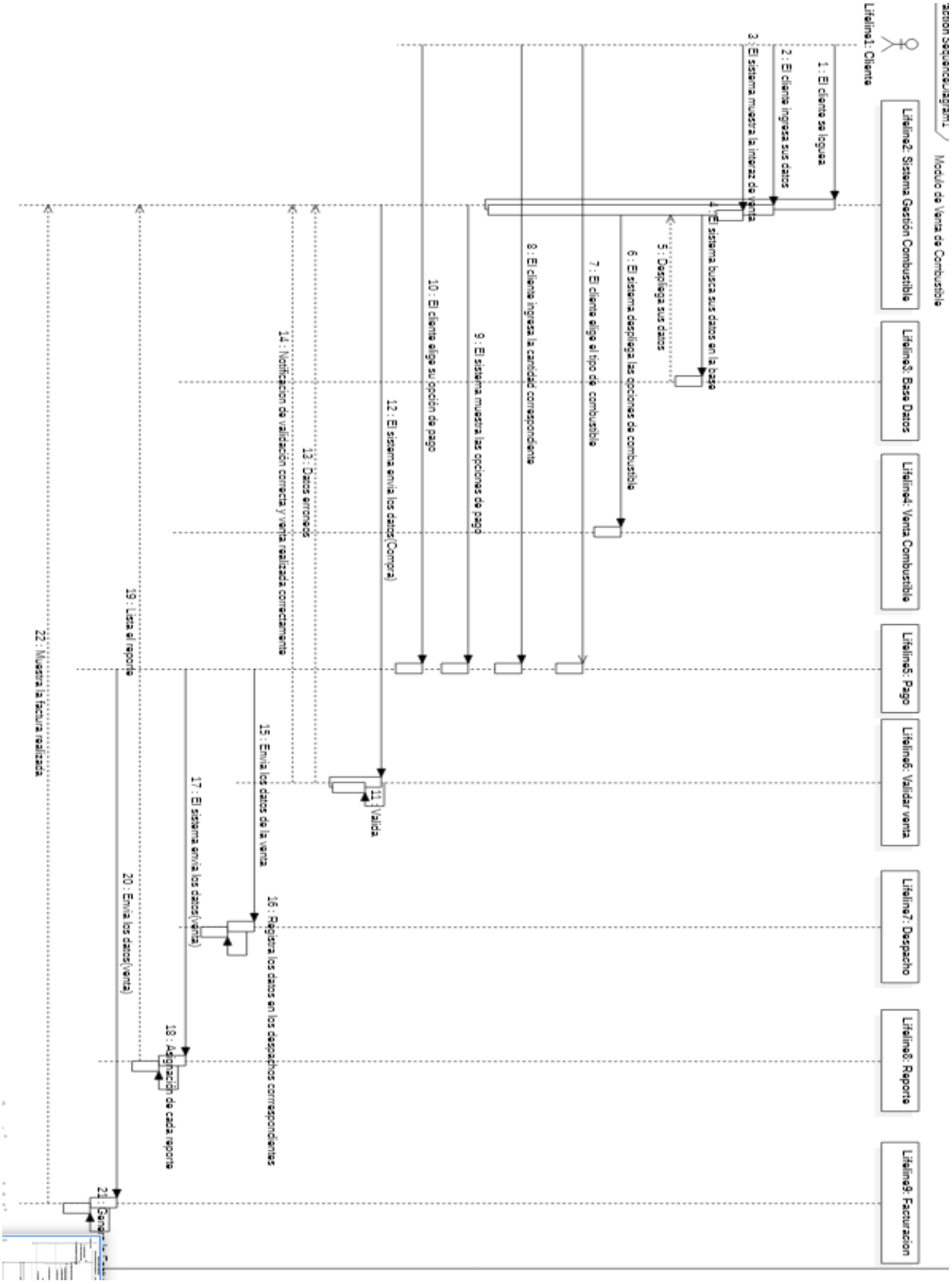
Lifeline1: Cliente
Lifeline2: Sistema Gestión Combustible
Lifeline3: Base Datos
Lifeline4: Venta Combustible
Lifeline5: Pago
Lifeline6: Validar venta
Lifeline7: Despacho
Lifeline8: Reporte
Lifeline9: Facturacion
1 : El cliente se loguea
2 : El cliente ingresa sus datos
3 : El sistema muestra la interaz de venta
4 : El sistema busca sus datos en la base
5 : Despliega sus datos
6 : El sistema despliega las opciones de combustible
7 : El cliente elige el tipo de combustible
8 : El cliente ingresa la cantidad correspondiente
9 : El sistema muestra las opciones de pago
10 : El cliente elige su opción de pago
11 : Valida
12 : El sistema envia los datos(Compra)
13 : Datos erroneos
14 : Notificación de validación correcta y venta realizada correctamente
15 : Envia los datos de la venta
16 : Registra los datos en los despachos correspondientes
17 : El sistema envia los datos(venta)
18 : Agregación de cada reporte
19 : Lista el reporte
20 : Envia los datos(venta)
21 : Gene
22 : Muestra la factura realizada

Ilustración 8: Diagrama Secuencia. - Venta de Combustible

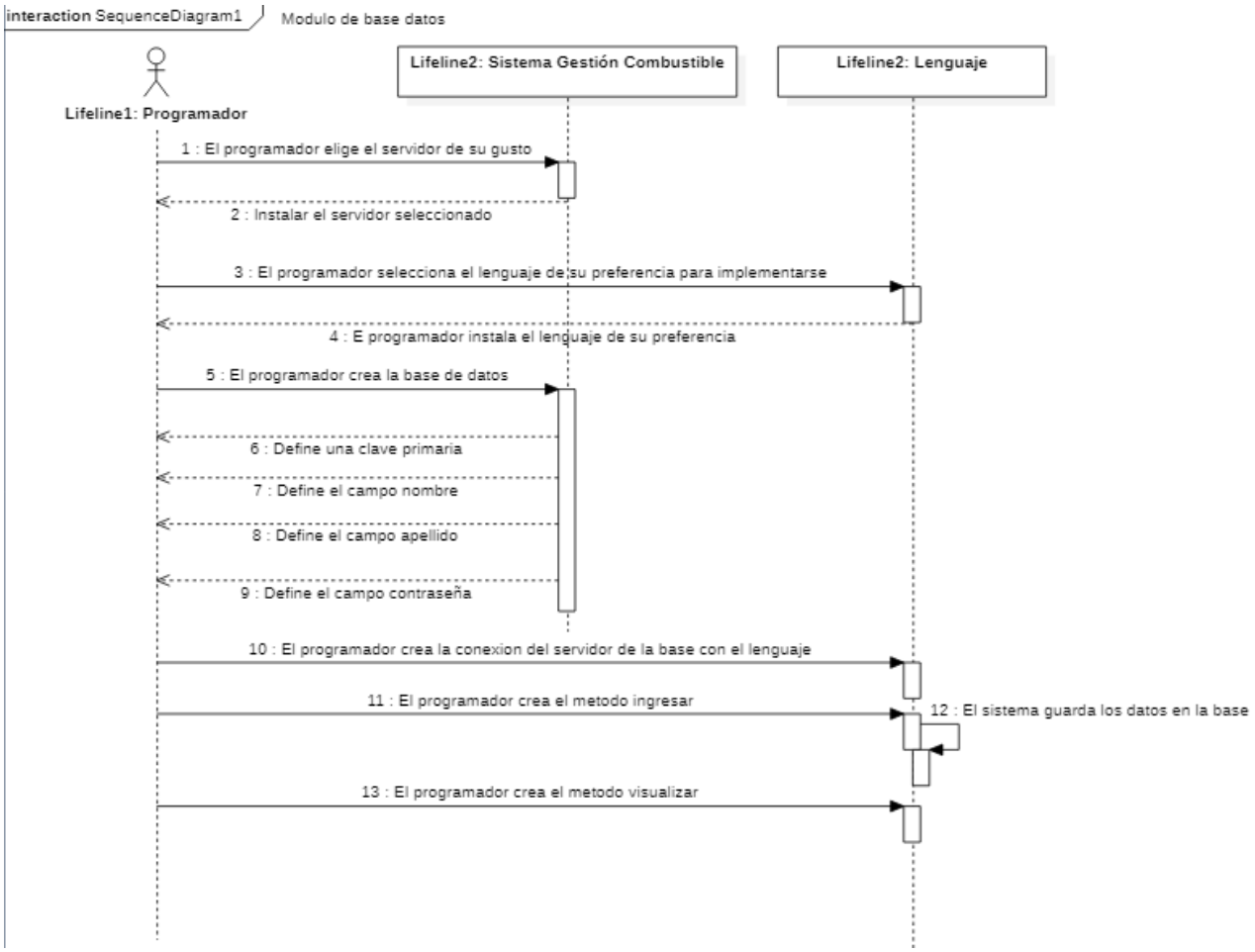

Ilustración 9: Diagrama Secuencia. – Base de Datos

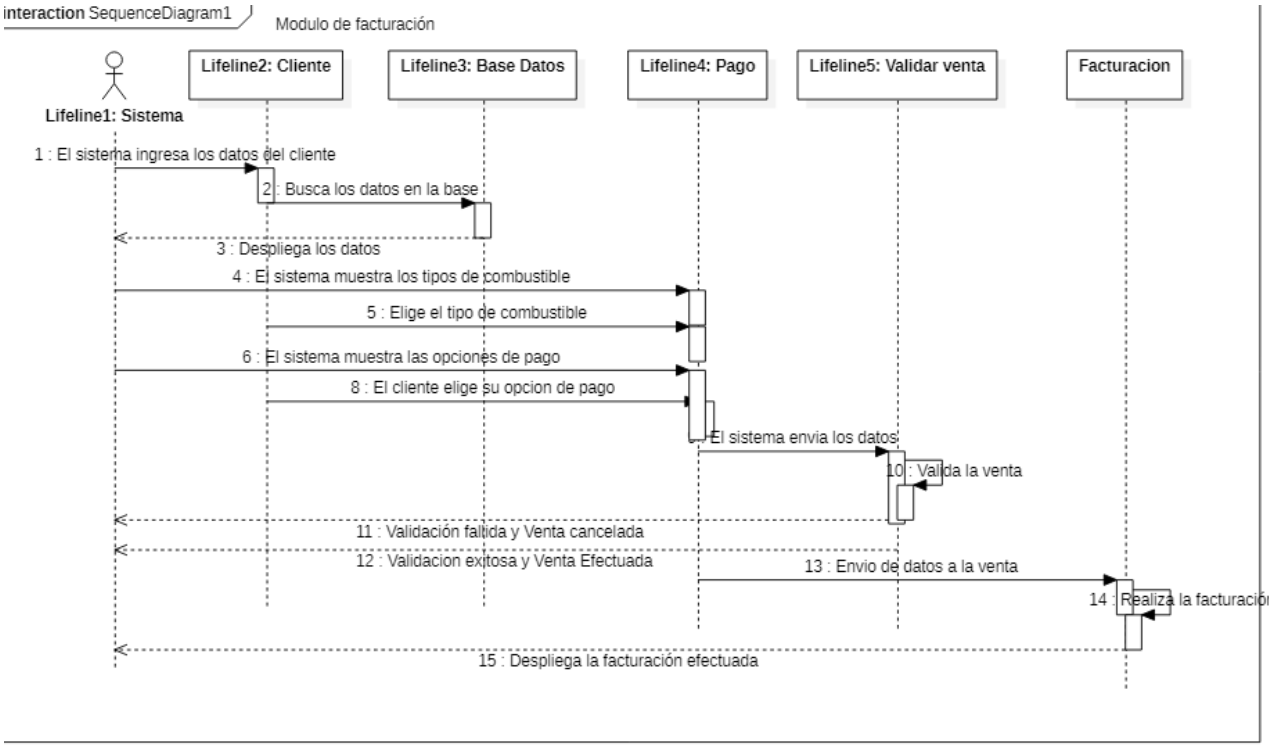

Ilustración 10: Diagrama Secuencia. – Facturación

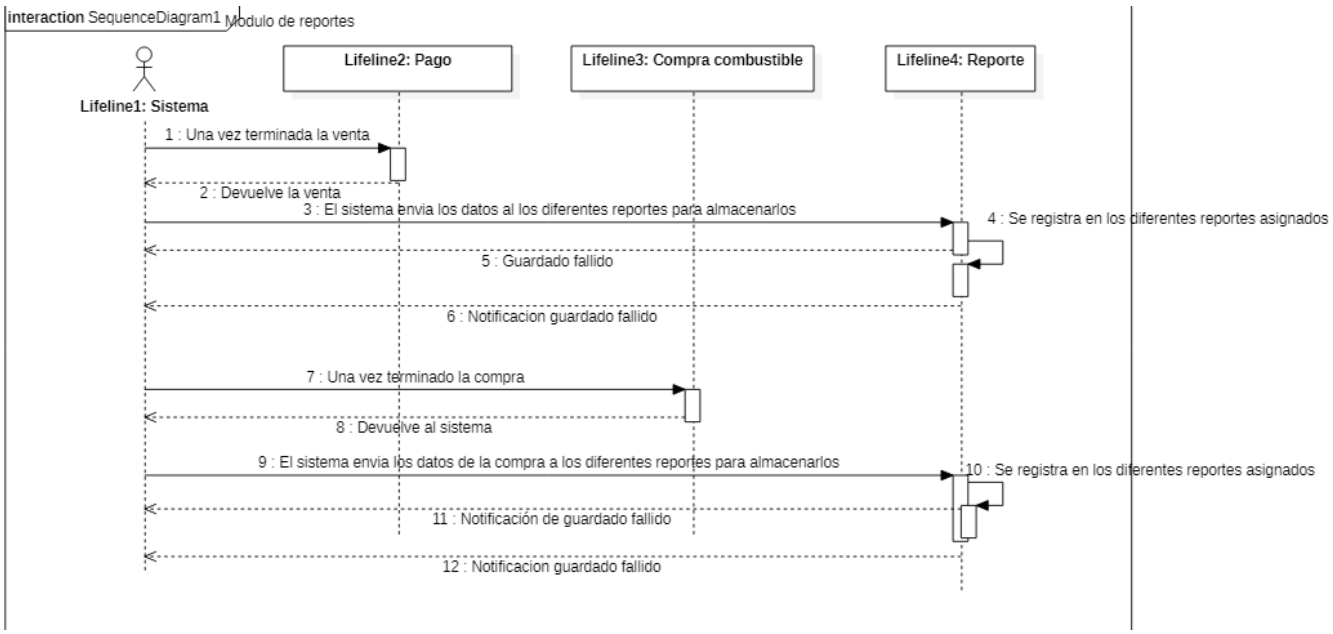

Ilustración 11: Diagrama Secuencia. – Reportes

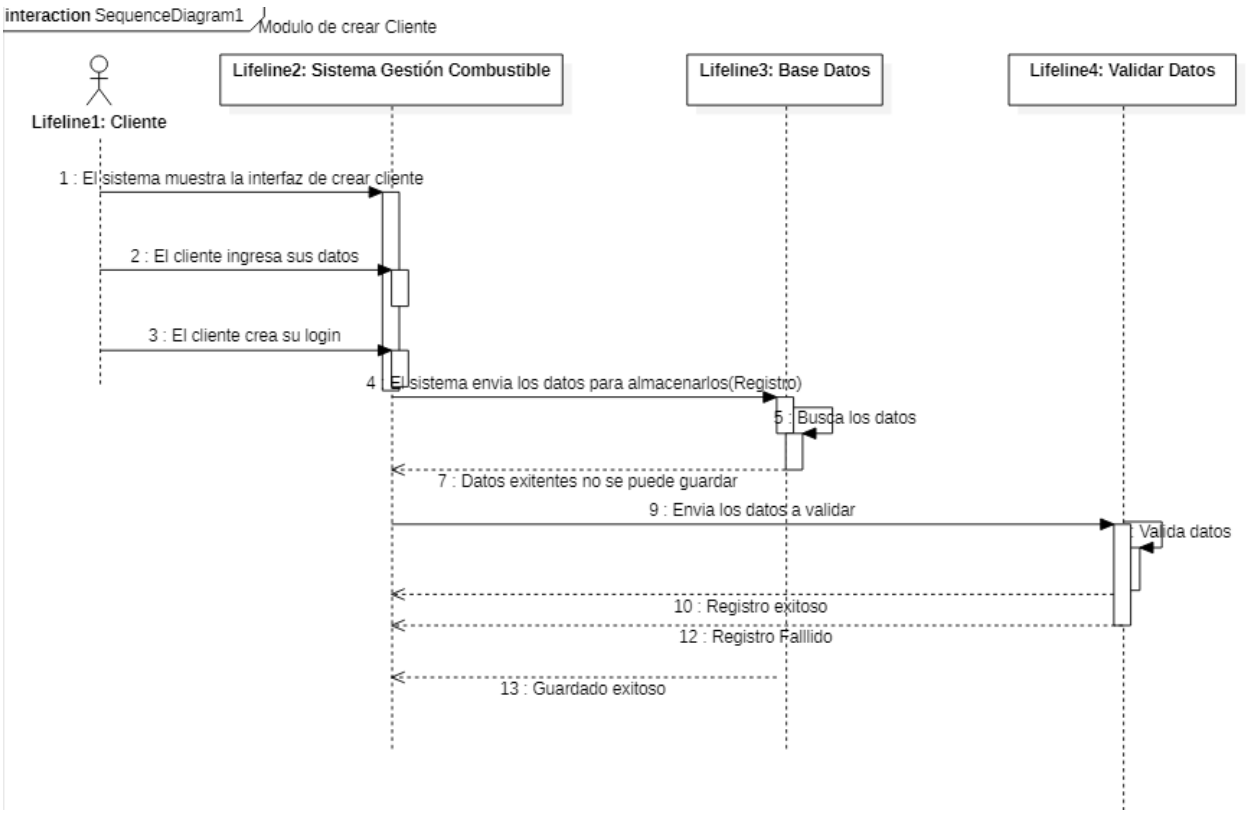

Ilustración 12: Diagrama Secuencia. – Crear Cliente

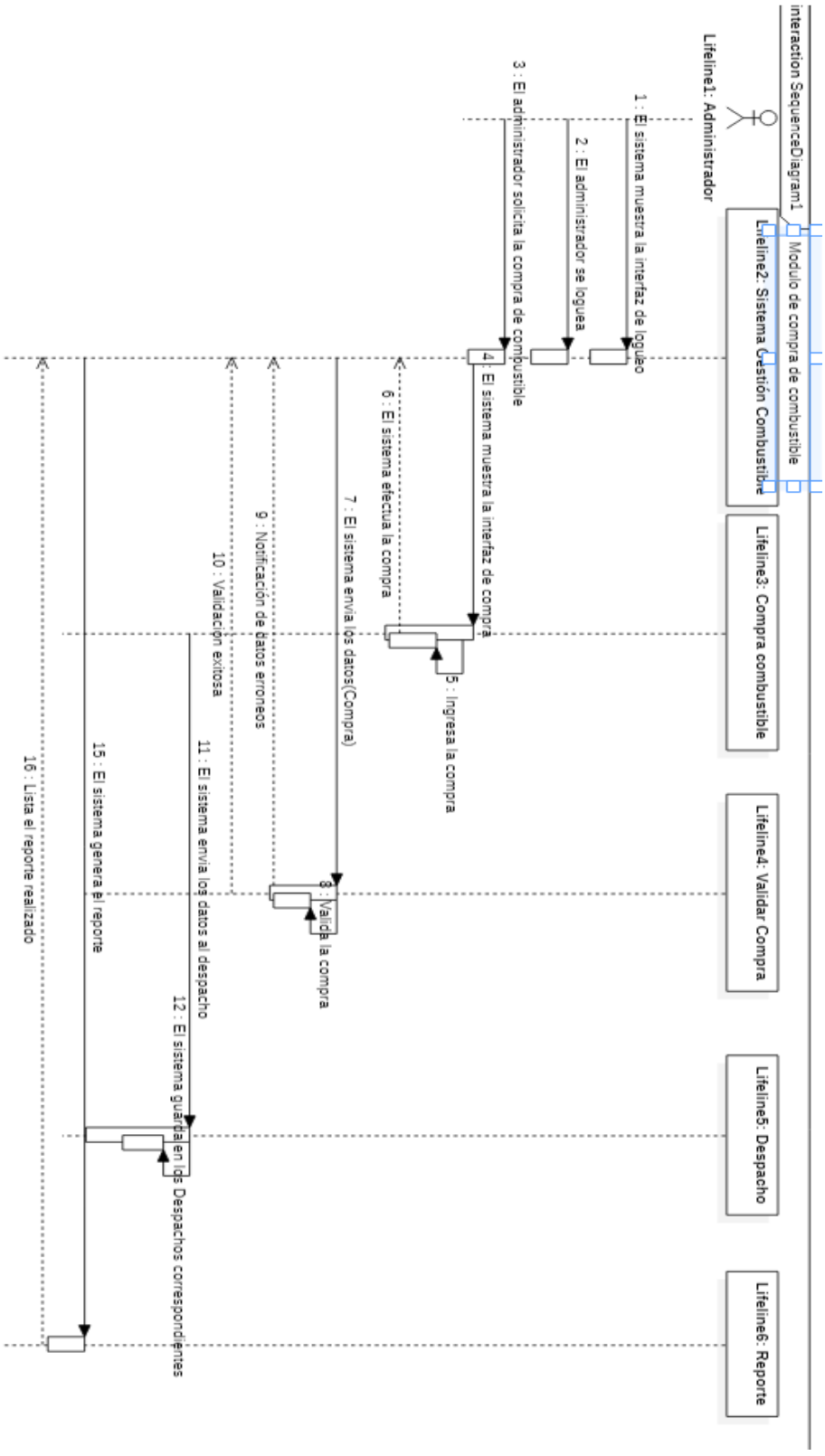

Ilustración 13: Diagrama Secuencia. - Compra de Combustible

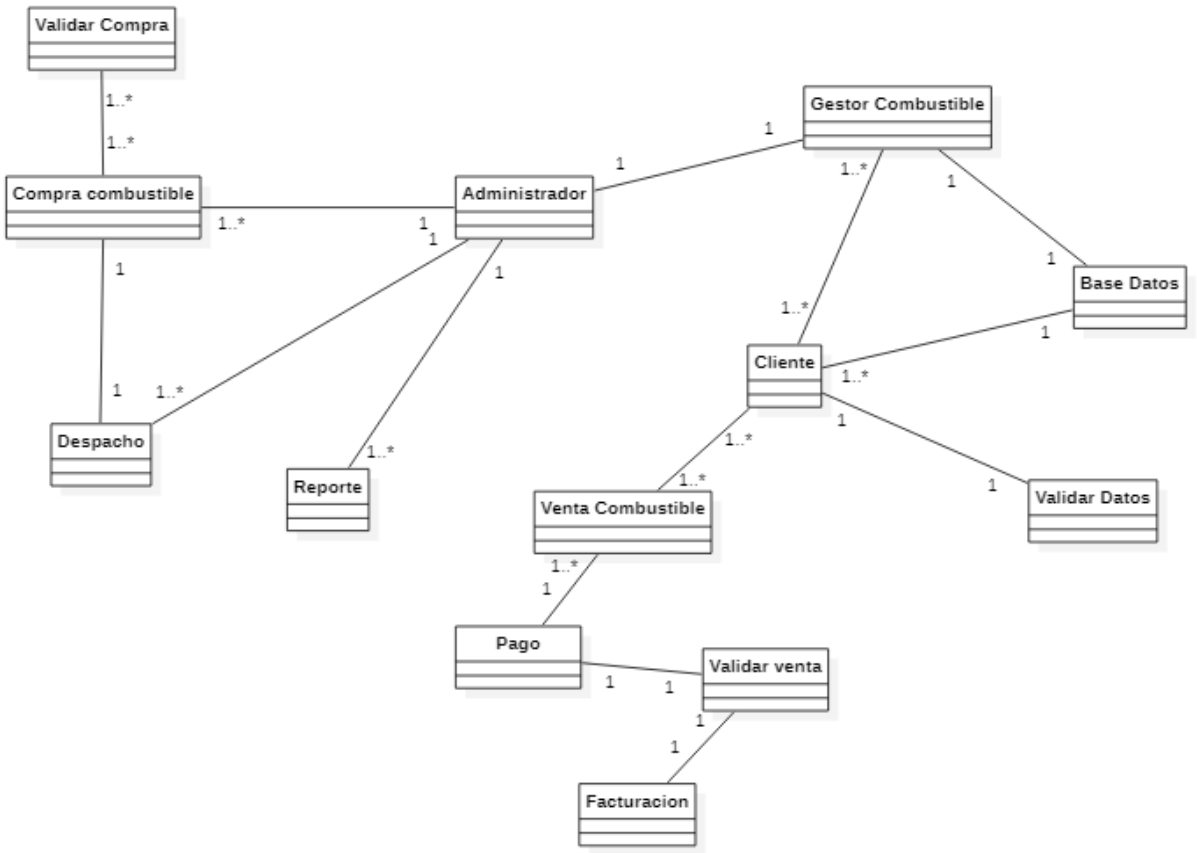

Ilustración 14: Diagrama Clases. – Sistema Combustible

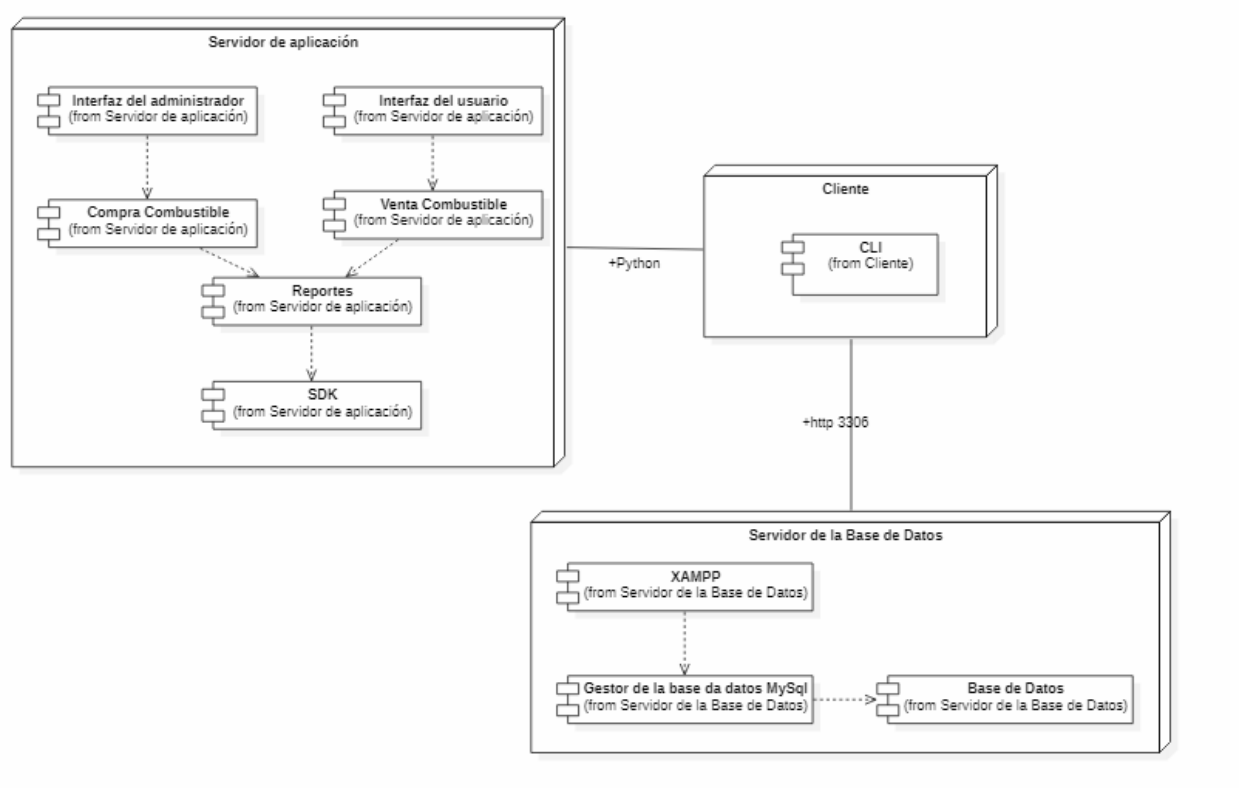

Ilustración 15: Diagrama de Despliegue

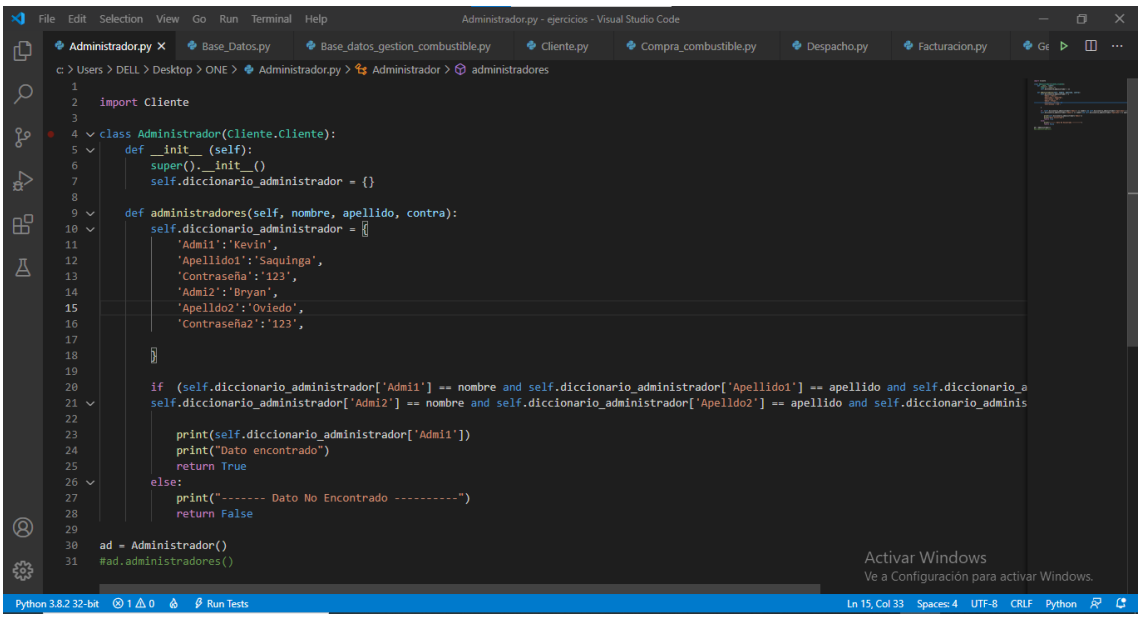

Ilustración 16: Codificación.- Clase Administración

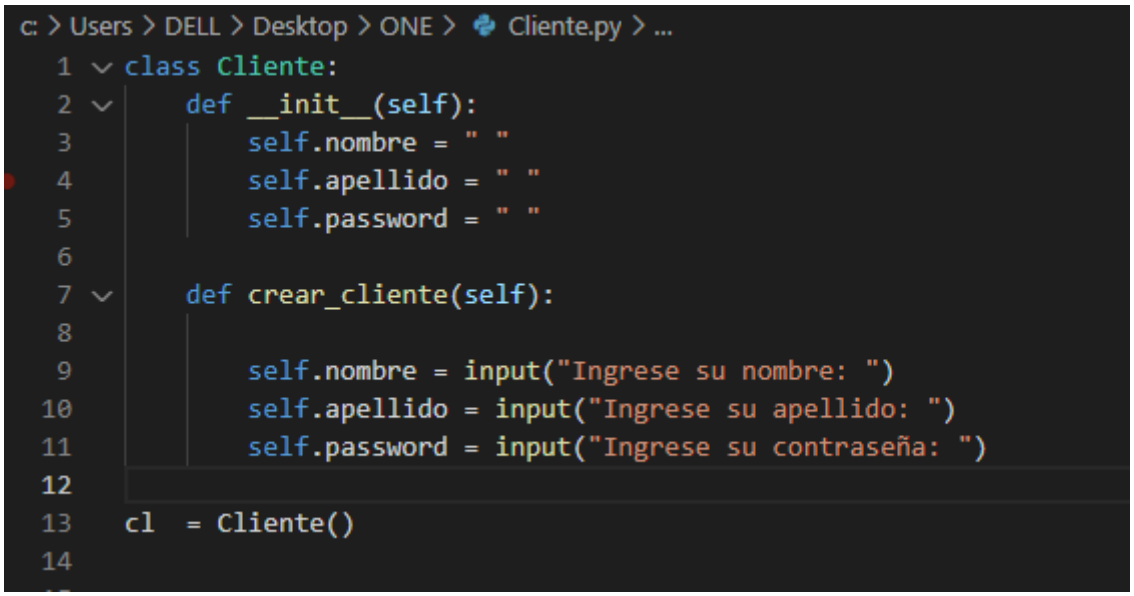

Ilustración 17: Codificación.- Clase Cliente

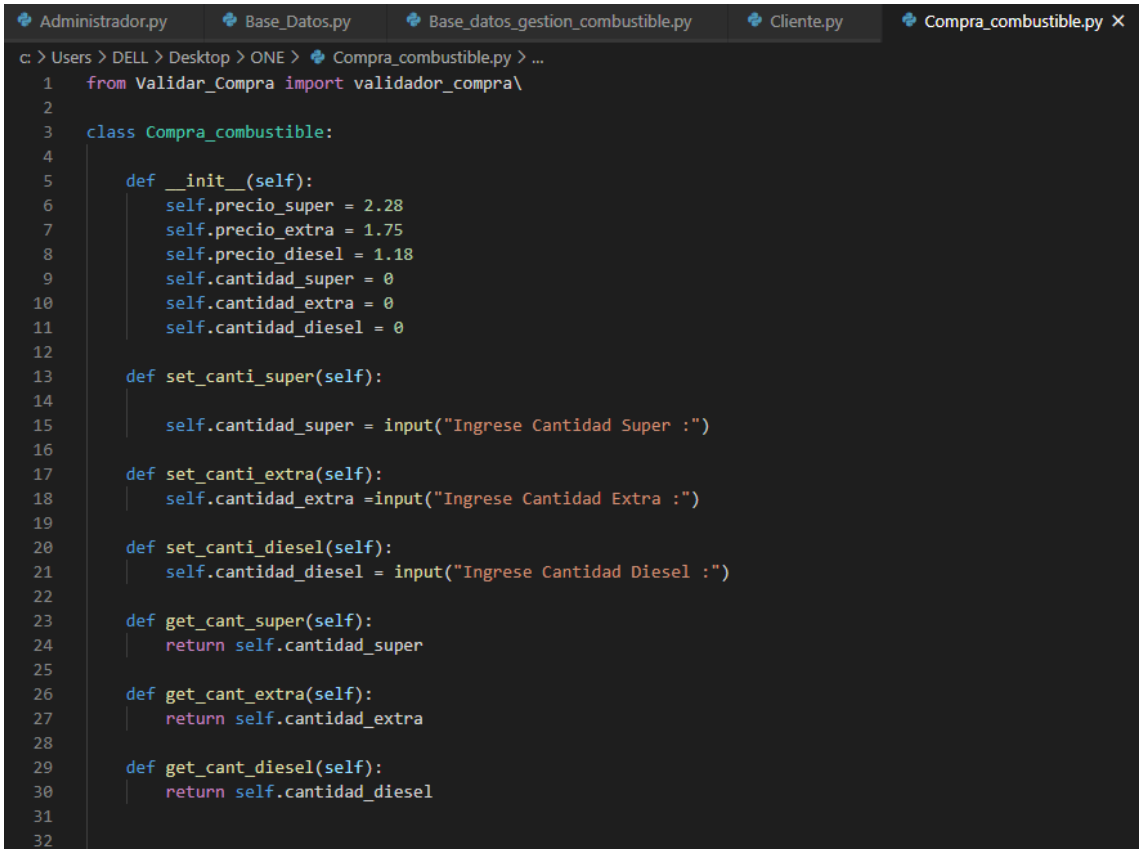

Ilustración 18: Codificación.- Clase Compra Combustible

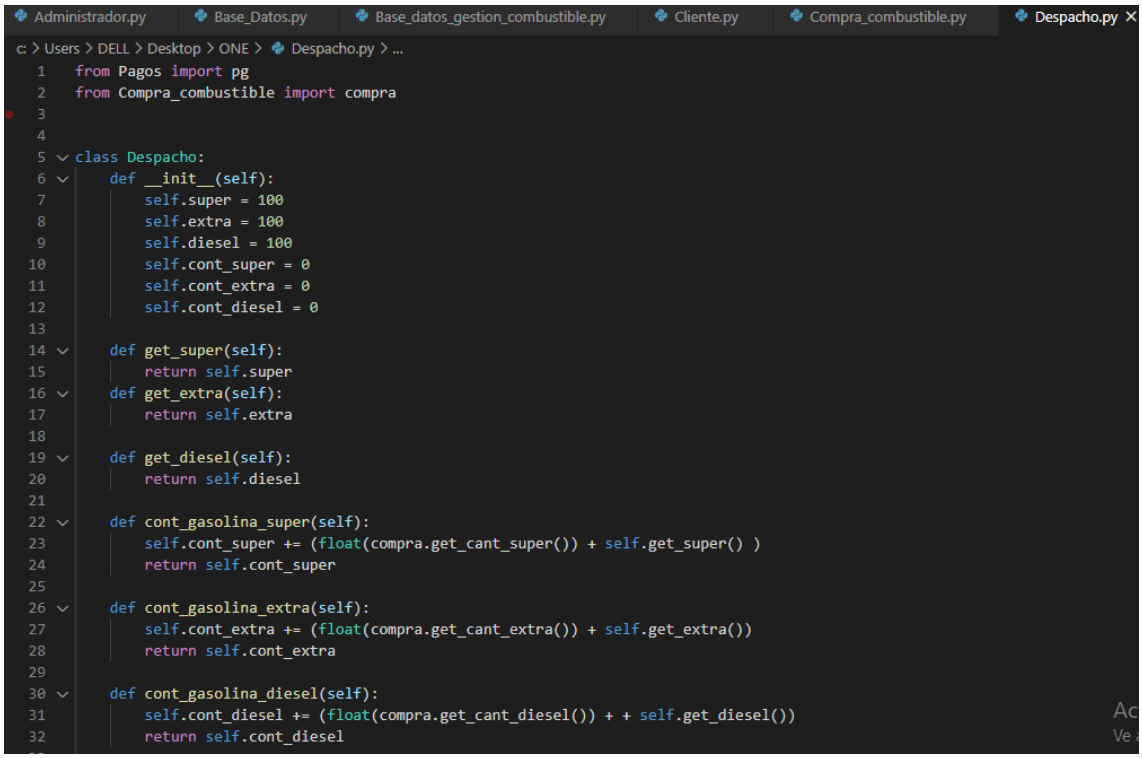

Ilustración 19: Codificación. –Clase Despacho

```python
#from Pagos import pg
from Validar_venta import validador_venta

class Facturacion :

    def __init__(self):
        self.precio_super = 2.28
        self.descuento_super = 0.1
        self.precio_extra = 1.75
        self.descuento_extra = 0.05
        self.precio_diesel = 1.18
        self.descuento_diesel = 0.05

    def formato_super(self, nombre, apellido ,cantidad, total):

        print("\n               ---------------- FACTURA SUPER ---------------")

        print(f"{nombre} {apellido} ha comprado {cantidad} litros ")
        print(f"El monto total de pagar de SUPER es {round(total,4)} $")

    def formato_extra(self, nombre, apellido ,cantidad, total):

        print("\n               ---------------- FACTURA EXTRA ---------------")

        print(f"{nombre} {apellido} ha comprado {cantidad} litros de Extra")
        print(f"El monto total de pagar de EXTRA es {round(total,4)} $")

    def formato_diesel(self,nombre, apellido ,cantidad, total):

        print("\n               ---------------- FACTURA DIESEL ---------------")

        print(f"{nombre} {apellido} ha comprado {cantidad} litros de Diesel")
```

Ilustración 20: Codificación.- Clase Facturación

c: > Users > DELL > Desktop > ONE > Gestor_combustible.py > ...

```python
from Cliente import cl
from Administrador import ad
from Base_datos_gestion_combustible import ingresar_datos , visualizar_datos, buscar_cliente , buscar_cliente_loguearse
from Validar_datos import validar
from Venta_combustible import vent
from Despacho import despa
from Compra_combustible import compra


def menu_prinipal():

    print("                          ----------------Sistema de Gestión de Combustible-------------")
    print("1. Administrador\n"+
        "2. Cliente\n"
        + "3. Salir\n"

    )

def controlador(opcion):

    while (opcion > 3 or opcion < 1):
        print("Solo opciones: 1 al 3")
        break

def menu_administrador():
    print("------------ Menu Administrador ------------- ")
    print("1. Compra Combustible\n"
        + "2. Ver Despacho de Combustible\n"
        + "3. Reporte de ventas\n"
        + "4. Reporte de Combustible\n"
        + "5. Salir\n" )

```

Ilustración 21: Codificación.-Clase Gestor Combustible

Base_Datos.py | Base_datos_gestion_combustible.py | Cliente.py | Compra_combustible.py | Despacho.py | Factur

c: > Users > DELL > Desktop > ONE > Base_Datos.py > ...

```python
#Conexion

database = mysql.connector.connect(
    host = "localhost",
    user = "root",
    passwd = "",
    database = "sistema_gestor_combustibles"
)


#print(database)
# CREAR TABLA DE DATOS

cursor = database.cursor(buffered= True)
"""
cursor.execute("CREATE DATABASE IF NOT EXISTS sistema_gestor_combustibles")


#cursor.execute("SHOW DATABASEs")
"""

# Cear Tablas
#nombre = kevinsaq;
#cursor.execute("INSERT INTO clientes VALUES(null,'Kevin','Saquinga','123')")


clien = [
    ('xxxx','xxxx','xxxx'),
]
"""
cursor.executemany("INSERT INTO clientes VALUES(null,%s,%s,%s)", clien)
database.commit()
```

Ilustración 22: Clase Base Datos

```python
import base64
try:
    database = mysql.connector.connect(
    host = "localhost",
    user = "root",
    passwd = "",
    database = "gestion_combustible"
    )
    print("------------------ Conexion Exitosa ------------------------")
except :
    print("-------------- Conexion Fallida --------------------")


def ingresar_datos(nombre,apellido,password):


    cursor = database.cursor(buffered= True)

    clientes = [
        (nombre,apellido,password)
        ]

    cursor.executemany("INSERT INTO clientes VALUES(null,%s,%s,%s)", clientes)
    database.commit()




def visualizar_datos():
    cursor = database.cursor(buffered= True)
    print("----- Cliente ------")
    """
    cursor.execute("SELECT *FROM clientes  ")
```

Ilustración 23: Codificación. - Clase Base de Datos Gestión de Combustible

```
----------------Sistema de Gestión de Combustible-------------
1. Administrador
2. Cliente
3. Salir

Ingresar Opcion: 1

--------- Iniciar Sesion -------
Ingrese su nombre: Kevin
Ingrese su apellido: Saquinga
Ingrese su contraseña: 123
Kevin
Dato encontrado
------------ Menu Administrador -------------
1. Compra Combustible
2. Ver Despacho de Combustible
3. Reporte de ventas
4. Reporte de Combustible
5. Salir

Ingresar Opcion:
```

Ilustración 24: Codificación. -Funcionamiento del Proyecto