**UNIVERSIDAD DE LAS FUERZAS ARMADAS**

**ESPE**

****

**PRUEBA DE CAJA BLANCA**

**“Aplicativo de gestión y administración de información de MUNDOTV-EC”**

**VERSIÓN 6**

**ASIGNATURA:**

Metodología de Desarrollo de Software

**ESTUDIANTES:**

Carlos Alejandro Pullas Herrera

Michael Alexander Villacrés Pabón

**NRC 9899**

**Ing. Jenny A. Ruiz**

**Sangolquí - Ecuador, 01 de agosto de 2023**

**Login de ingreso el aplicativo**

## **Caso de Prueba 1**

**CÓDIGO FUENTE:**

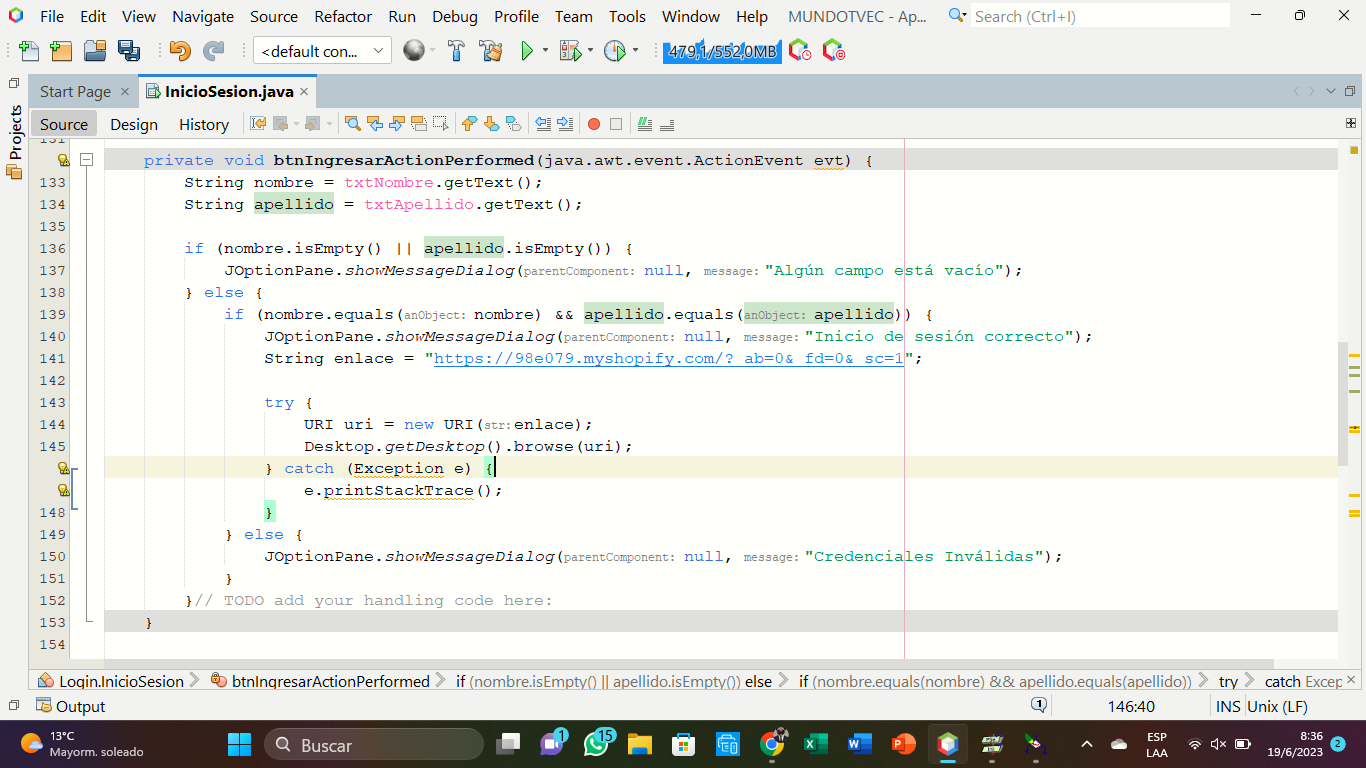


Ilustración 1. Código Login ingreso del aplicativo

**DIAGRAMA DE FLUJO:**

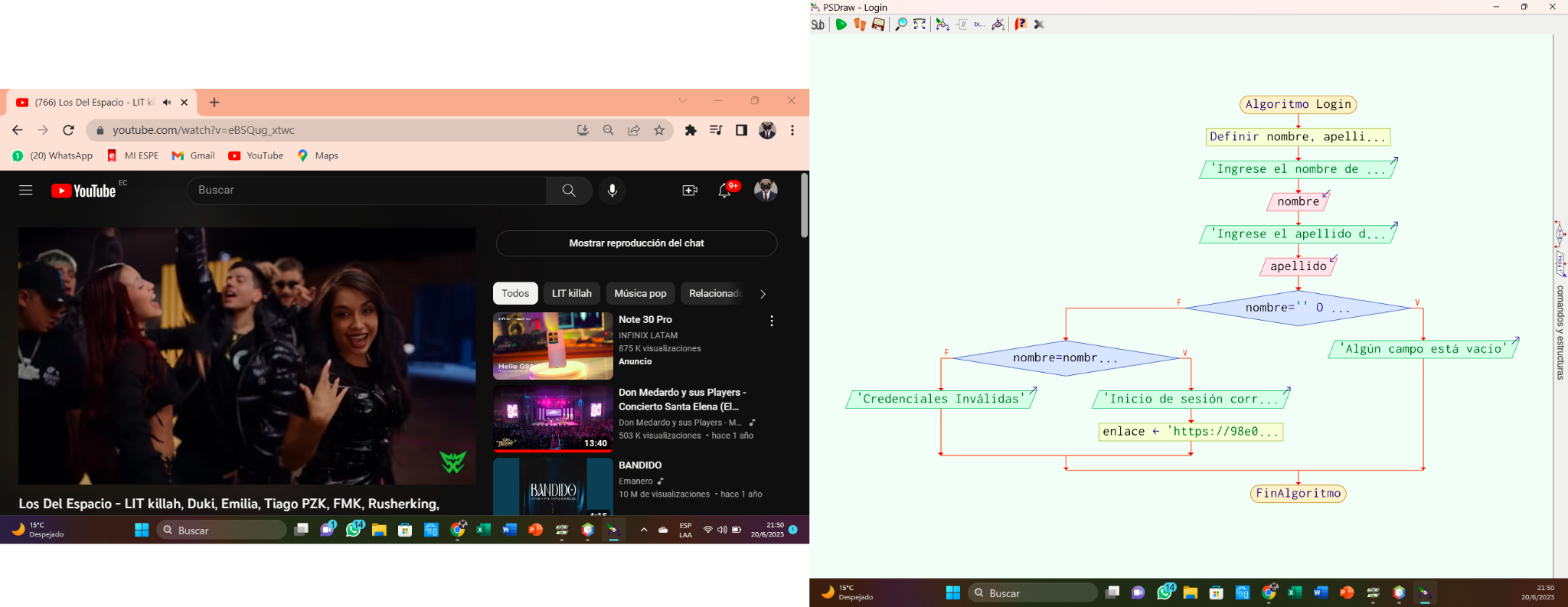


Ilustración 2. Diagrama de flujo login

**DIAGRAMA DE GRAFOS:**

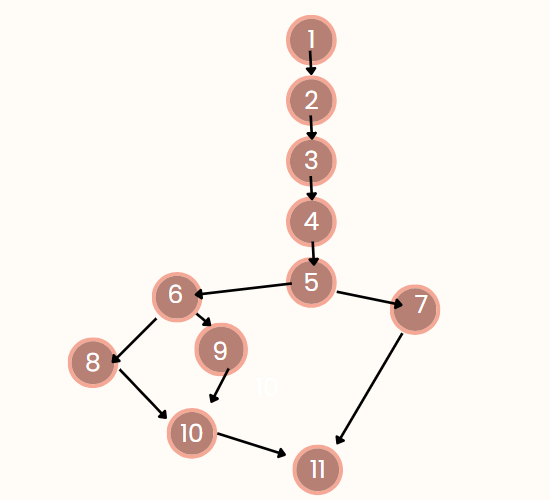


Ilustración 3. Diagrama de Grafos login

**RUTAS**

**R1:** 1,2,3,4,5,7,11

**R2:** 1,2,3,4,5,6,9,10,11

**R3:** 1,2,3,4,5,6,8,10,11

**COMPLEJIDAD CICLOMÁTICA**

Se puede calcular de las siguientes formas:

●   V(G) = número de nodos predicados(decisiones) 2 +1 = 3

●   V(G) = E – N + 2 = 12 – 11 + 2 = 3

**DONDE:**

**P:** Número de nodos predicados

**E:** Número de aristas

**N:** Número de nodos

**P:** 2

**E:** 12

**N:** 11

**Opción buscar productos en el aplicativo**

## **Caso de Prueba 2**

**CÓDIGO FUENTE:**

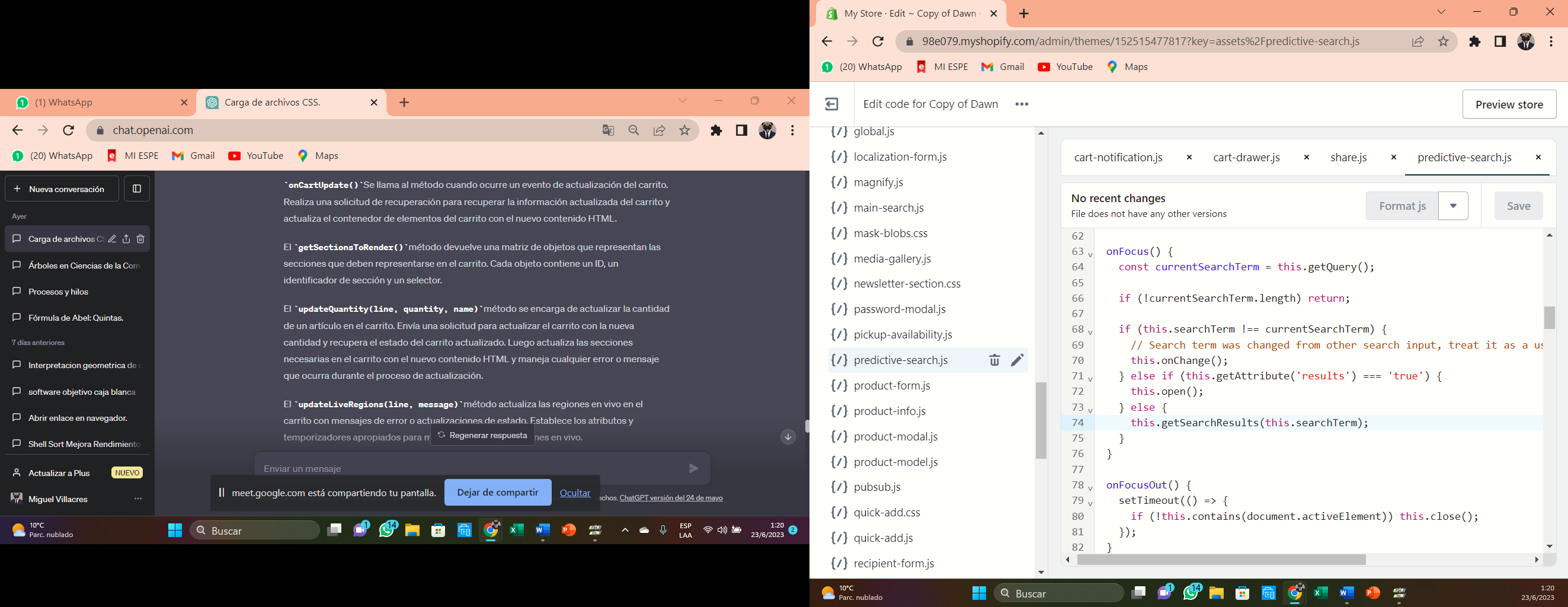


Ilustración 4. Código opción buscar

**DIAGRAMA DE FLUJO:**

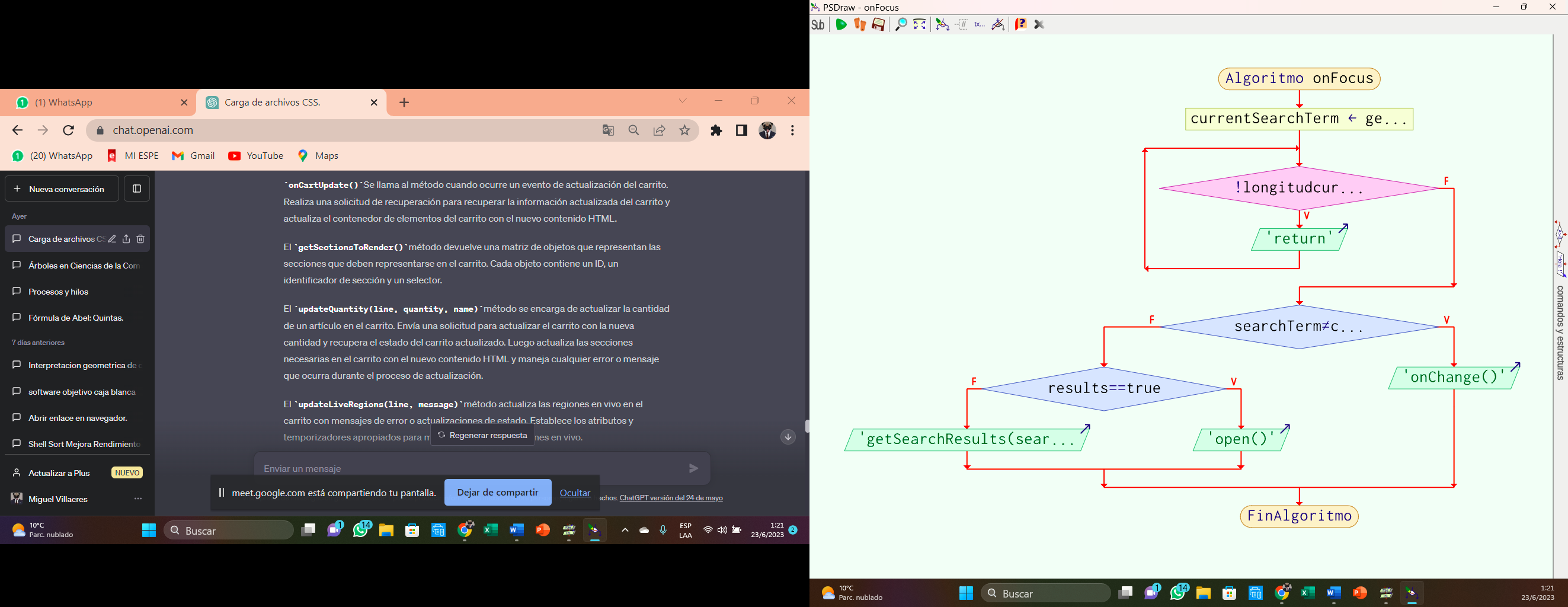


Ilustración 5. Diagrama de flujo opción buscar

**DIAGRAMA DE GRAFOS:**

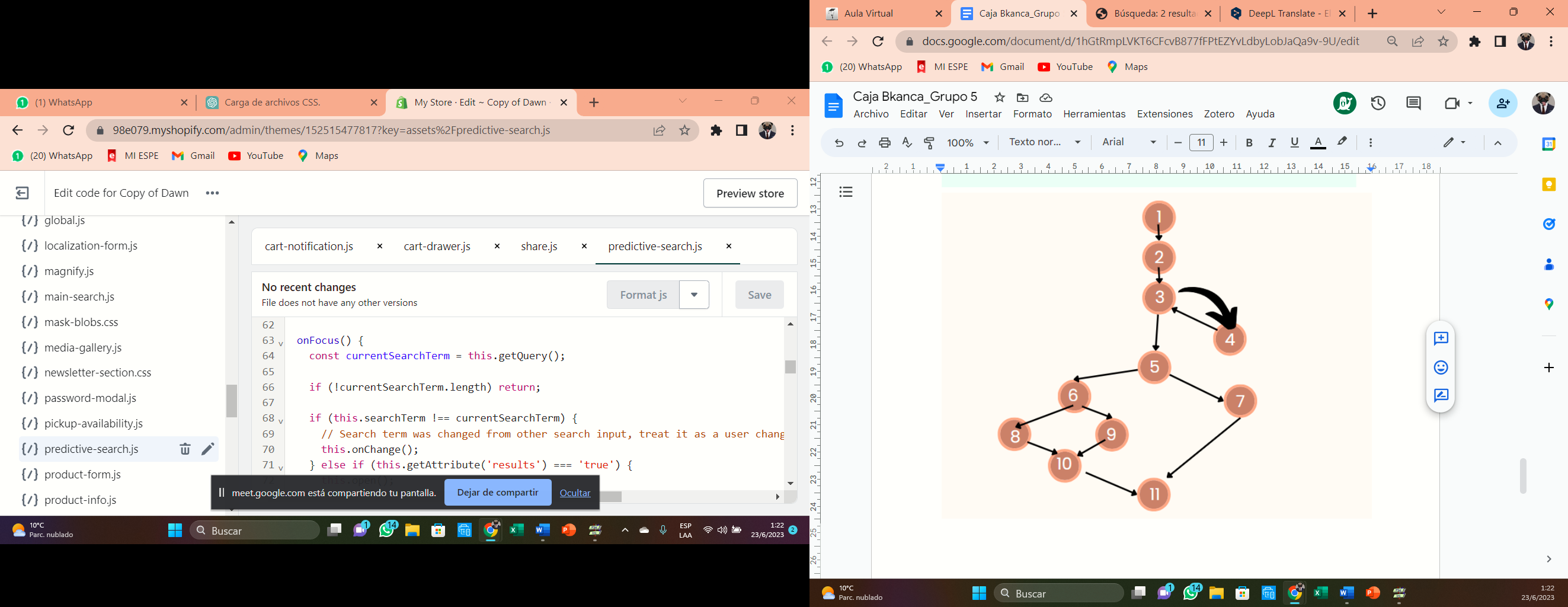


Ilustración 6. Diagrama de Grafos opción buscar

**RUTAS**

**R1:** 1,2,3,4

**R2:** 1,2,3,5,7,11

**R3:** 1,2,3,5,6,9,10,11

**R4:** 1,2,3,5,6,8,10,11

**COMPLEJIDAD CICLOMÁTICA**

Se puede calcular de las siguientes formas:

●   V(G) = número de nodos predicados(decisiones) 3 +1 = 4

●   V(G) = E – N + 2 = 13– 11 + 2 = 4

**DONDE:**

**P:** Número de nodos predicados

**E:** Número de aristas

**N:** Número de nodos

**P:** 3

**E:** 13

**N:** 11

**Manejo del enfoque y la interacción con los elementos del carrito de compras**

## **Caso de Prueba 3**

**CÓDIGO FUENTE:**

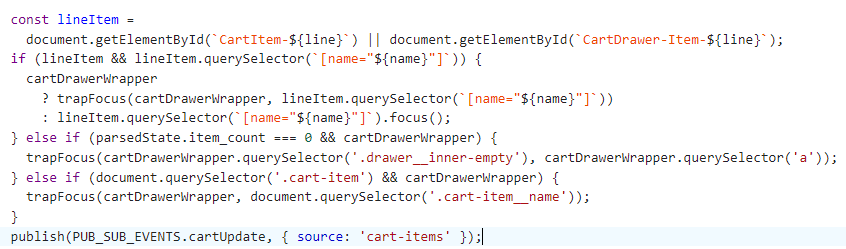
****

Ilustración 7. Código de interacción con los elementos del carrito de compras

**DIAGRAMA DE FLUJO:**

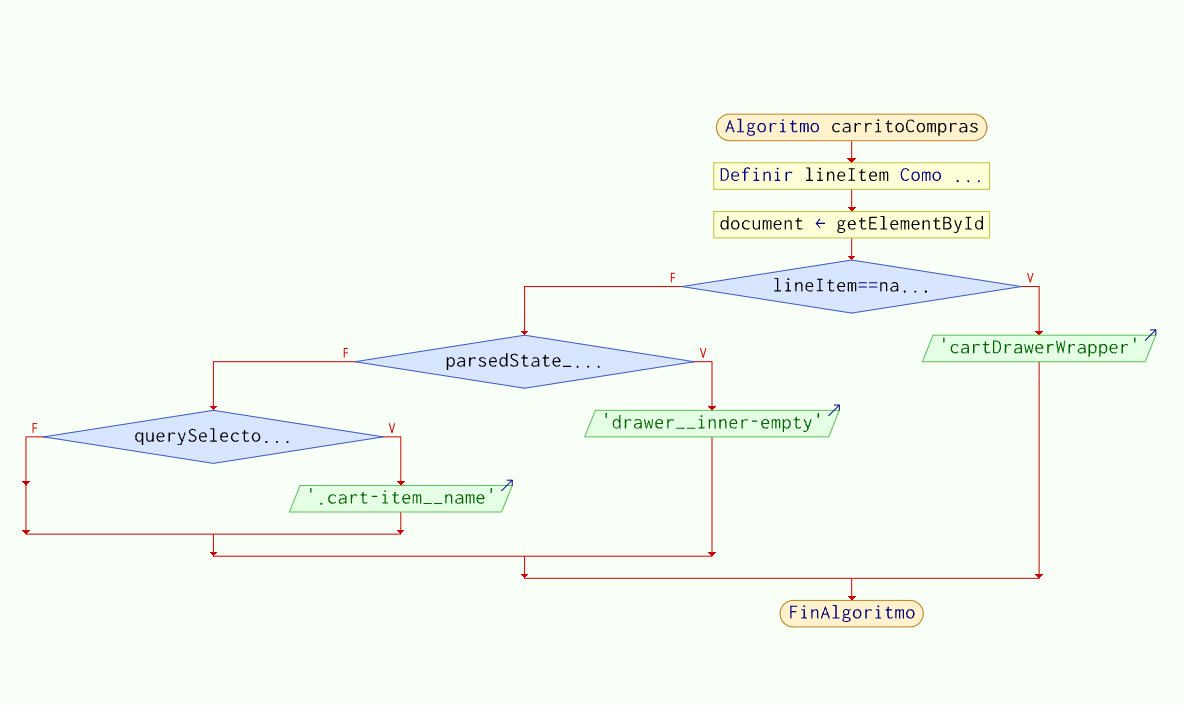
****

Ilustración 8. Diagrama de flujo de la interacción con los productos del carrito de compra

**DIAGRAMA DE GRAFOS:**

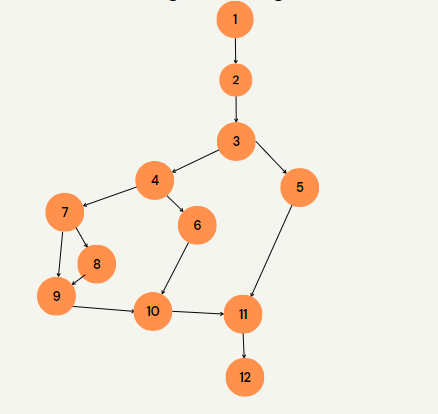
****

Ilustración 9. Diagrama de Grafos interacción de los elementos del carrito de compra

**RUTAS:**

**R1:** 1,2,3,5,11,12

**R2:** 1,2,3,4,6,10,11,12

**R3:** 1,2,3,4,7,8,9,10,11,12

**R4:** 1,2,3,4,7,9,10,11,12

**COMPLEJIDAD CICLOMÁTICA**

●      V(G) = número de nodos predicados(decisiones) 3 +1 = 4

●      V(G) = E – N + 2 = 14– 12 + 2 = 4

**DONDE:**

**P:** Número de nodos predicados

**E:** Número de aristas

**N:** Número de nodos

**P:** 4

**E:** 14

**N:** 12

**Mostrar los productos en oferta**

## **Caso de Prueba 4**

**CÓDIGO FUENTE:**

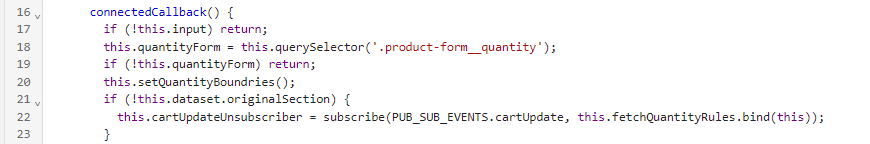
****

Ilustración 10. Código de Mostrar los productos en oferta

**DIAGRAMA DE FLUJO:**

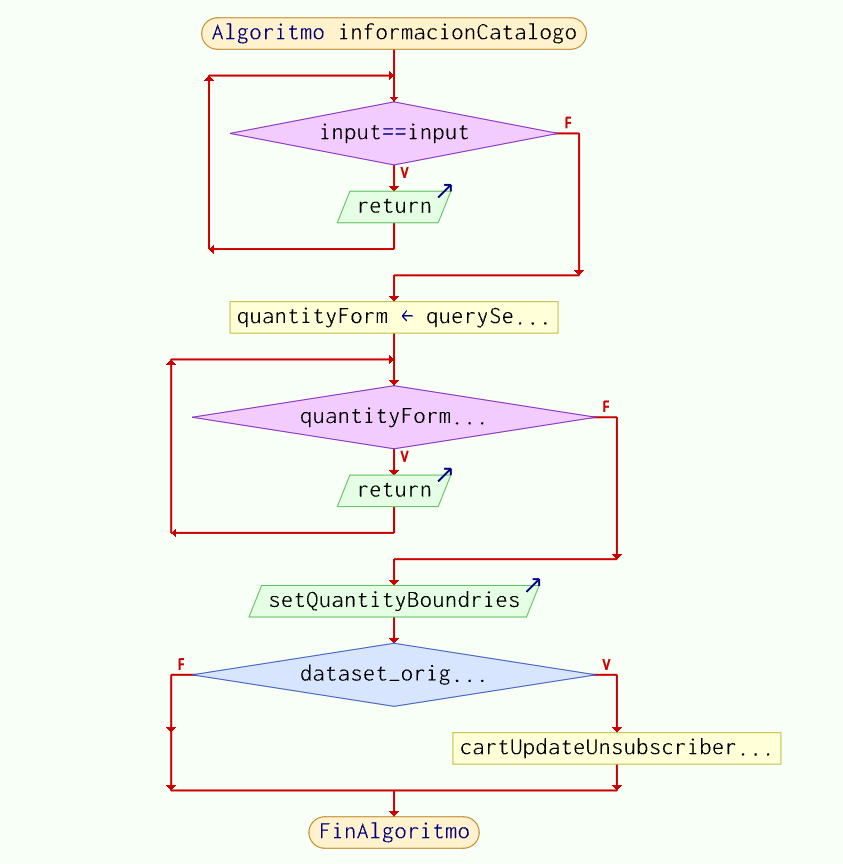
****

Ilustración 11. Diagrama de flujo de Mostrar los productos en oferta

**DIAGRAMA DE GRAFOS:**

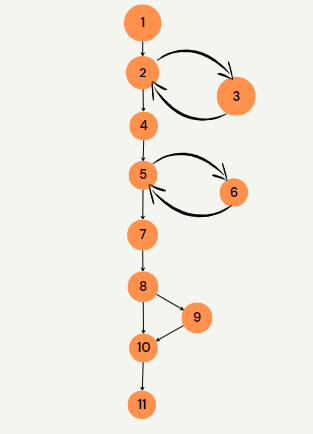
****

Ilustración 12. Diagrama de Grafos Mostrar los productos en oferta

**RUTAS**

**R1:** 1,2,3

**R2:** 1,2,4,5,6

**R3:** 1,2,4,5,7,8,9.10,11

**R4:** 1,2,4,5,7,8.10,11

**COMPLEJIDAD CICLOMÁTICA**

●      V(G) = número de nodos predicados(decisiones) 3 +1 = 4

●      V(G) = E – N + 2 = 13– 11+ 2 = 4D

**DONDE:**

**P:** Número de nodos predicados

**E:** Número de aristas

**N:** Número de nodos

**P:** 4

**E:** 13

**N:** 11

**Mensaje de errores en el ingreso de datos de la sección de reporte de problemas**

## **Caso de Prueba 5**

**CÓDIGO FUENTE:**

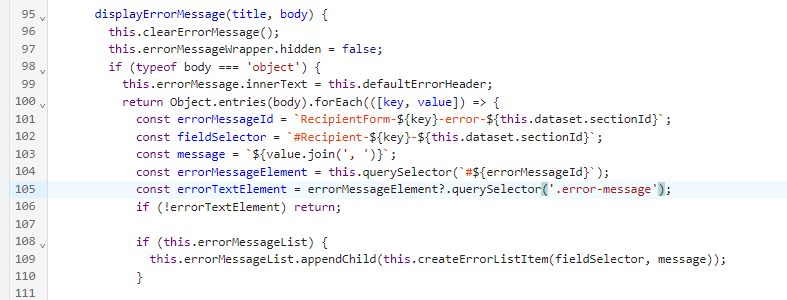
****

Ilustración 13. Código de Mensaje de errores en el ingreso de datos.

**DIAGRAMA DE FLUJO:**

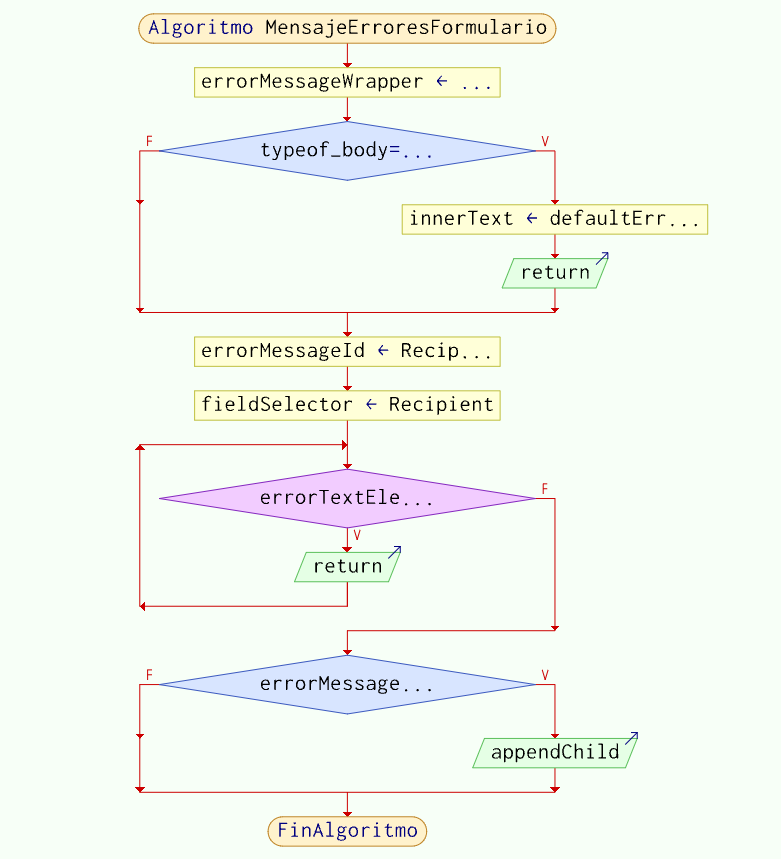
****

Ilustración 14. Diagrama de flujo de Mensaje de errores en el ingreso de datos.

**DIAGRAMA DE GRAFOS:**

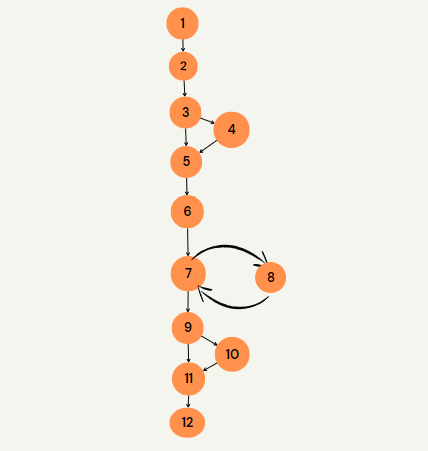
****

Ilustración 15. Diagrama de Grafos Mensaje de errores en el ingreso de datos.

**RUTAS**

**R1:** 1,2,3,4,5,6,7,8

**R2:** 1,2,3,5,6,7,8

**R3:** 1,2,3,5,6,7,9,10,11,12

**R4:** 1,2,3,5,6,7,9,11,12

**COMPLEJIDAD CICLOMÁTICA**

●      V(G) = número de nodos predicados(decisiones) 3 +1 = 4

●      V(G) = E – N + 2 = 14– 12+ 2 = 4

**DONDE:**

**P:** Número de nodos predicados

**E:** Número de aristas

**N:** Número de nodos

**P:** 4

**E:** 14

**N:** 12

## **Caso de Prueba 6**

**CÓDIGO FUENTE:**

****

Ilustración 16. Código de Mensaje de error.

**DIAGRAMA DE FLUJO:**

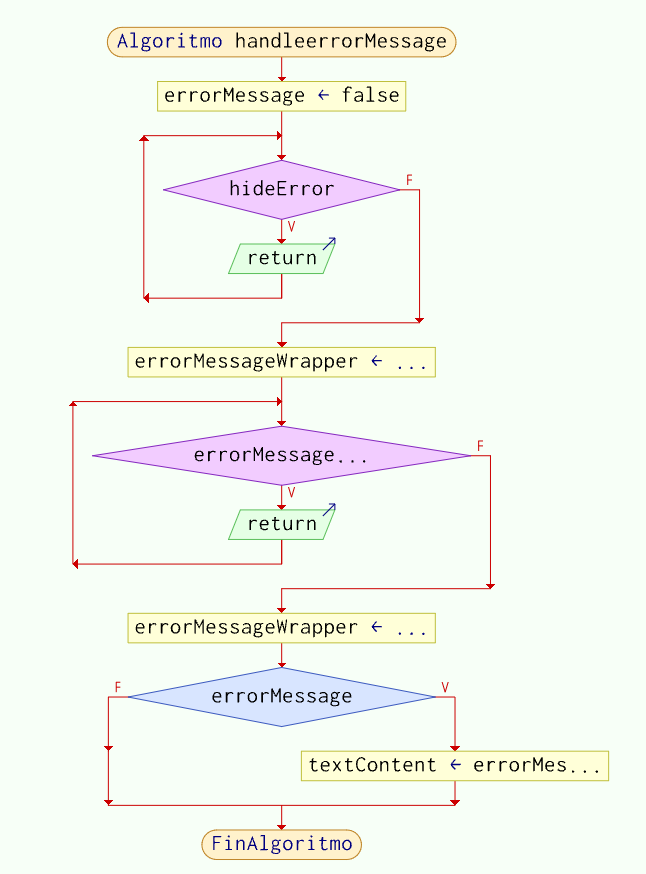
****

Ilustración 17. Diagrama de flujo de Mensaje de error

**DIAGRAMA DE GRAFOS:**

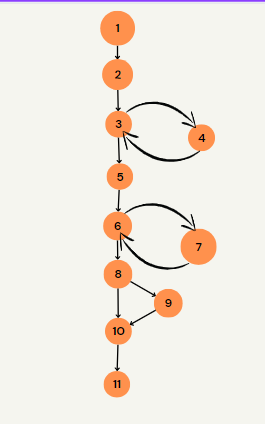
****

Ilustración 18. Diagrama de Grafos Mensaje de error.

**RUTAS**

**R1:** 1, 2. 3, 4

**R2:** 1, 2. 3, 5, 6, 7

**R3:** 1, 2. 3, 5, 6, 8, 9, 10 ,11

**R4:** 1, 2. 3, 5, 6, 8, 10, 11

**COMPLEJIDAD CICLOMÁTICA**

●      V(G) = número de nodos predicados(decisiones) 3 +1 = 4

●      V(G) = E – N + 2 = 13 – 11+ 2 = 4

**DONDE:**

**P:** Número de nodos predicados

**E:** Número de aristas

**N:** Número de nodos

**P:** 3

**E:** 13

**N:** 11

## **Caso de Prueba 7**

**CÓDIGO FUENTE:**

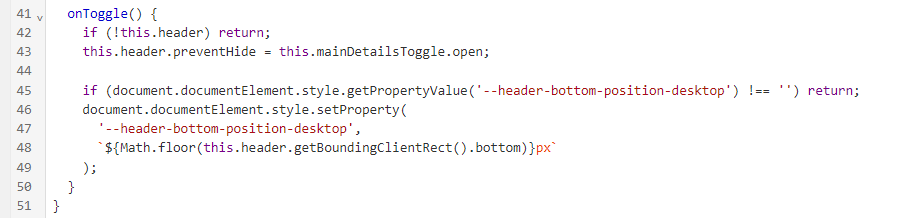
****

Ilustración 19. Código de alternancia

**DIAGRAMA DE FLUJO:**

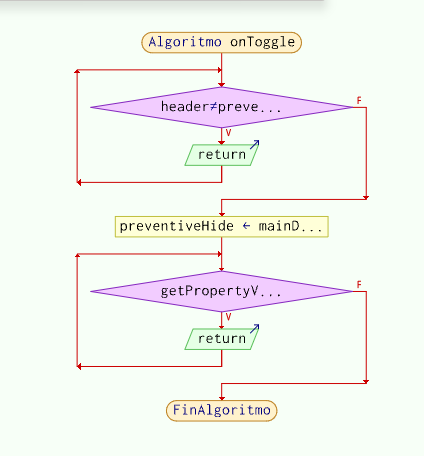
****

Ilustración 20. Diagrama de flujo de alternancia

**DIAGRAMA DE GRAFOS:**

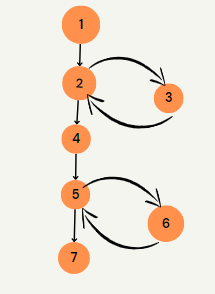
****

Ilustración 21. Diagrama de alternancia

**RUTAS**

**R1:** 1, 2, 3

**R2:** 1, 2, 4, 5, 6

**R3:** 1, 2. 4, 5, 7

**COMPLEJIDAD CICLOMÁTICA**

●      V(G) = número de nodos predicados(decisiones) 2 + 1 = 3

●      V(G) = E – N + 2 = 8 – 7 + 2 = 3

**DONDE:**

**P:** Número de nodos predicados

**E:** Número de aristas

**N:** Número de nodos

**P:** 2

**E:** 8

**N:** 7

**CONCLUSIONES:**

* Al aplicar la técnica de prueba de software que se centra en examinar y evaluar la estructura interna del código de un programa junto con la complejidad ciclomática se ha determinado la cantidad adecuada de todos los caminos de ejecución posibles.
* La realización de pruebas de caja blanca no solo implica la evaluación del comportamiento del usuario en la interfaz, sino también la detección de errores en el código fuente. No es factible asegurar que un software o sistema nunca falle, solamente se pueden llevar a cabo pruebas que reduzcan dicho riesgo.