

UNIVERSIDAD DE LAS FUERZAS ARMADAS
ESPE



Object-oriented programming

NRC: 9642

Workshop N° 12

Topic: Project Review

ING. Jorge Edison Lascano.

Rubric

Presentation

0.- GitHub	/100	7:20-7:25
1.- Interview and Presentation	/10	7:25-7:30
2.- Project definition	/10	
3.- Software Requirements	/10	7:47-7:57
4.- Use Case	/10	8:00-8:10
5.- Class Diagram	/10	8:15-8:25
6.- Data files	/10	
7.- Execution	/10	
8-10.- Clean Code	/30	
TOTAL	/100	

Evidence

GitHub

- 1.- Interview and Presentation
- 2.- Project definition
- 3.- Software Requirements
- 4.- Use Case
- 5.- Class Diagram
- 6.- Data files
- 7.- Execution
- 8-10.- Clean Code

Team 1: Jsons

Project's name: OdontoApp

Leader: Yeshua Amador Chiliquinga Amaya

GitHub: <https://github.com/YeshuaChiliquingaAmaya/Jsons.git>

Inspector: Team 5, Leonardo Narvaez

Interview video: <https://youtu.be/BsEjTRw4V0U>

1. RUBEN DARIO BENAVIDES MACIAS
2. ADONNY MATEO CALERO ARGUELLO
3. PABLO ESTEBAN CARRERA QUINDE
4. YESHUA AMADOR CHILINUINGA AMAYA

Presentation: 7/10

Rubric

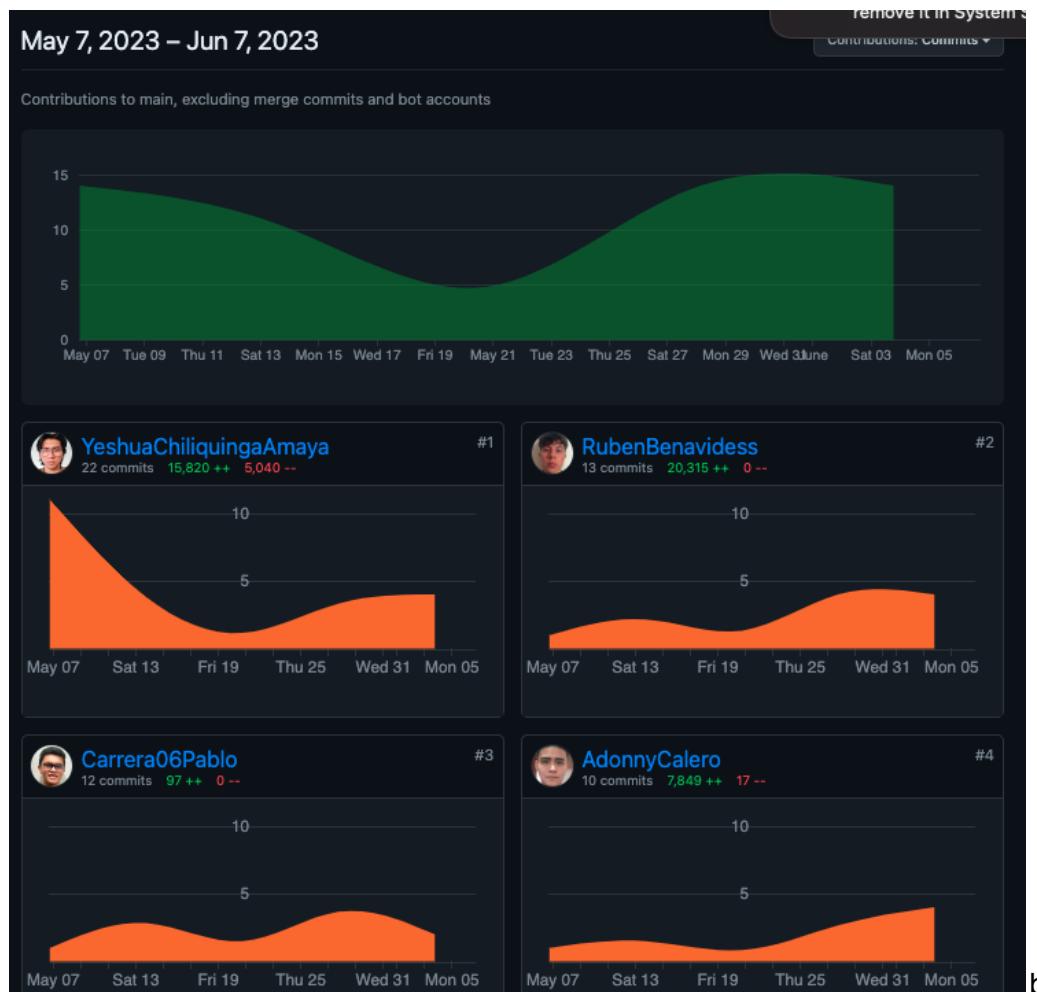
Presentation

0.- GitHub	100/100
1.- Interview and Presentation	1/10
2.- Project definition	7/10 (There is no definition, there are only objectives.)

3.- Software Requirements	7.5/10 (If a part of the user interface is going to be in Spanish, all of it should be in Spanish or vice versa, Missing UML)
4.- Use Case	2/10 (The use case diagram does not match well to the specific requirements, as a recommendation you have to create an author for the client.)
5.- Class Diagram	2/10 (Too many classes that do not use, lack of associations, no multiplicity in the single association.)
6.- Data files	10/10
7.- Execution	10/10
8-10.- Clean Code	25/30 (If a menu is going to be in Spanish all of them should be in Spanish or vice versa, all the variables in English there are some in Spanish.)
TOTAL	64.5/100

Evidence

GitHub



1.- Interview and Presentation



The interview had to be face-to-face, not online.

2.- Project definition

User Story Name: Dental Patient Data Management System

Description: The dentist needs a comprehensive program to efficiently manage and modify patient data information.

Story Name: Dental Patient Data Management System

Description: As a dentist, I require a reliable and user-friendly program that allows me to effectively manage and modify patient data information. The program should streamline the process of updating patient records, ensuring accurate and up-to-date information is readily accessible when needed.

User: Dentist who needs to handle and modify patient data information.

Business Value: The program enables the dentist to access and modify patient information, reducing the reliance on manual paperwork and improving overall efficiency in managing patient data. By having a centralized system, the dentist can easily update and retrieve patient records, resulting in more effective treatments and enhanced patient care.

Acceptance Criteria:

The user (dentist) should be able to:

- Search for patients and their information arranged alphabetically.
- Add or delete new patients and their information.
- Access and update comprehensive medical history, including name, age, weight, height, disease symptoms, cell phone number, systematic diseases, patient treatment, treatment start date and end date of treatment.
- Generate reports based on patient data through a data file with csv format.
- Receive a confirmation upon successful data modification, ensuring the changes are accurately recorded.

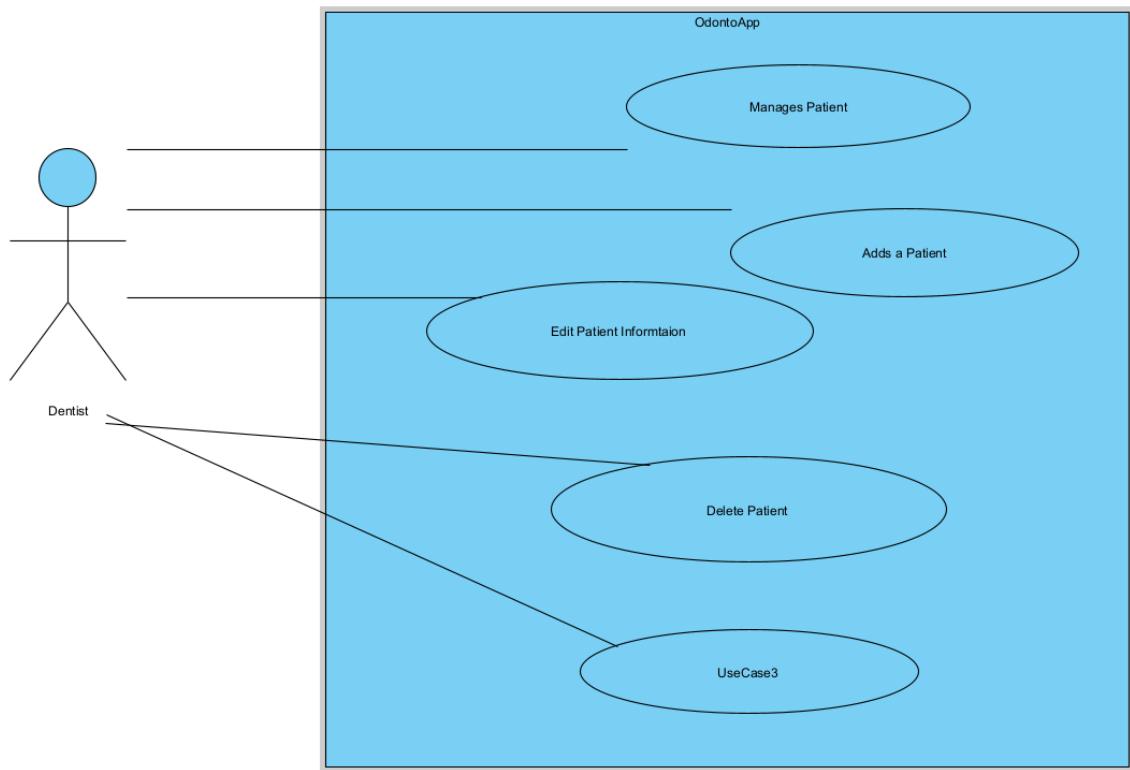
The project definition is not specified.

3.- Software Requirements

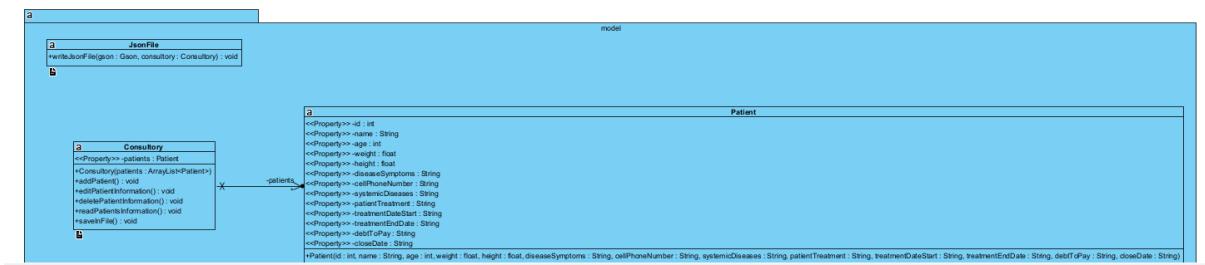
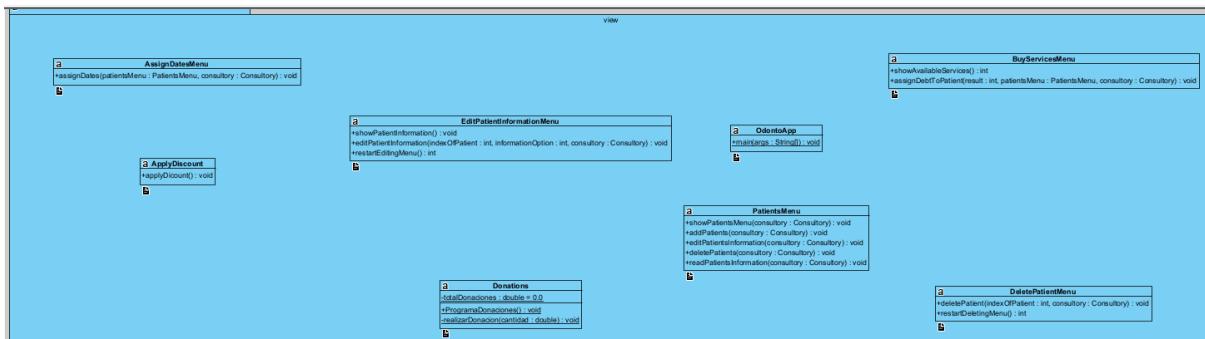
Requirement Identifier	FR02	
Scope in the system	Buy Services Menu	
Requirement Definition	<p>When entering the buy services option of the principal menu, the system will first display a menu with the following structure and these available options:</p> <p>----Choose a service(s) or go back----</p> <ol style="list-style-type: none"> 1. Profilaxis + Fluorizacion (especialmente para niños): 20 dolares 2. Restauraciones simples: 20 dolares 3. Restauraciones complejas: 25 	
	<ol style="list-style-type: none"> 4. Restauraciones para niños (empastes para niños): 25 dolares 5. Extracciones (extraccion de un diente): 20 dolares 6. Extracciones complicadas: 30 dolares 7. Tratamiento de un conducto (incisivos o dientes frontales): 60 dolares 8. Tratamiento de tres conductos (molares): 100 dolares 9. Go back <p>You can choose as many services as you want (if the purchase is more than 200 dollars, patient will get a discount), and then go back, also when entering the option, the system will check if there are limitations when entering</p>	

Requirement Identifier	FR02
Scope in the system	Donations Menu
Requirement Definition	<p>When entering the donations option of the principal menu, the system will display another menu with the following structure:</p> <p>----- Donations Menu -----</p> <ol style="list-style-type: none"> 1. Do a donation 2. Total of donations 3. Exit <p>Enter an option:</p> <p>The system will check if there are limitations when entering data and acting respectively if these have been exceeded.</p>

4.- Use Case



5.- Class Diagram



6.- Data files

```
patients.json - Visual Studio Code
File Edit Selection View Go Run Terminal Help
Restricted Mode is intended for safe code browsing. Trust this window to enable all features Manage Learn More
patients.json
C:\Users\narvaez.leonardo\OneDrive\Escritorio\Team1Project\JSONs\06-Code\OdontoAppV0.7.6\patients.json
1
2 "ight":16.8,"diseaseSymptoms":"gripe","cellPhoneNumber":"0966011513","systemicDiseases": "tos","patientTreatment": "jarabe","treatmentStartDate": "01/05/23","treatmentEndDate": "07/06/23","debtToPay": "20","closeDate": ""}]
```

Ln 2, Col 220 Spaces: 4 CRLF () JSON 8:57 7/6/2023

7.- Execution

```
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help OdontoApp7.5 - Apache NetBeans IDE 16
Output - OdontoApp7.5 (run) Search (Ctrl+I)
Output Explorer Navigator Services Projects Files
Output - OdontoApp7.5 (run)
init:
Deleting: C:\Users\narvaez.leonardo\OneDrive\Escritorio\Team1Project\JSONs\06-Code\OdontoAppV0.7.6\build\internal.action.name=run run
deps-jar:
Updating property file: C:\Users\narvaez.leonardo\OneDrive\Escritorio\Team1Project\JSONs\06-Code\OdontoAppV0.7.6\build\built-jar.properties
compile:
run:
---Welcome to OdontoApp---
1. Add Patients
2. Buy Services
3. Assign Dates
4. Donations
5. Save in File
6. Exit
Enter an option: 1
---About Patients---
1. Add Patient
2. Read Patient Information
3. Delete Patient
4. Read Patients Information
5. Go back
Enter an option: 1
Enter id patient (only positive and int numbers):
12
Enter patient name:
nica
Enter patient age (only positive and int numbers):
20
Enter patient weight (only positive numbers):
50
You just wrote string data. Try again:
56.8
You just wrote string data. Try again:
56.9
Enter patient height (only positive numbers):
56
Enter patient disease symptoms:
```

12C Mayorm. soleado 8:57 7/6/2023

8-10.- Clean Code

```
public void assignDebtToPatient(int result, P
    int idOption;
    int validator = 0;
    double descuento;
    double totalDescuent;
```

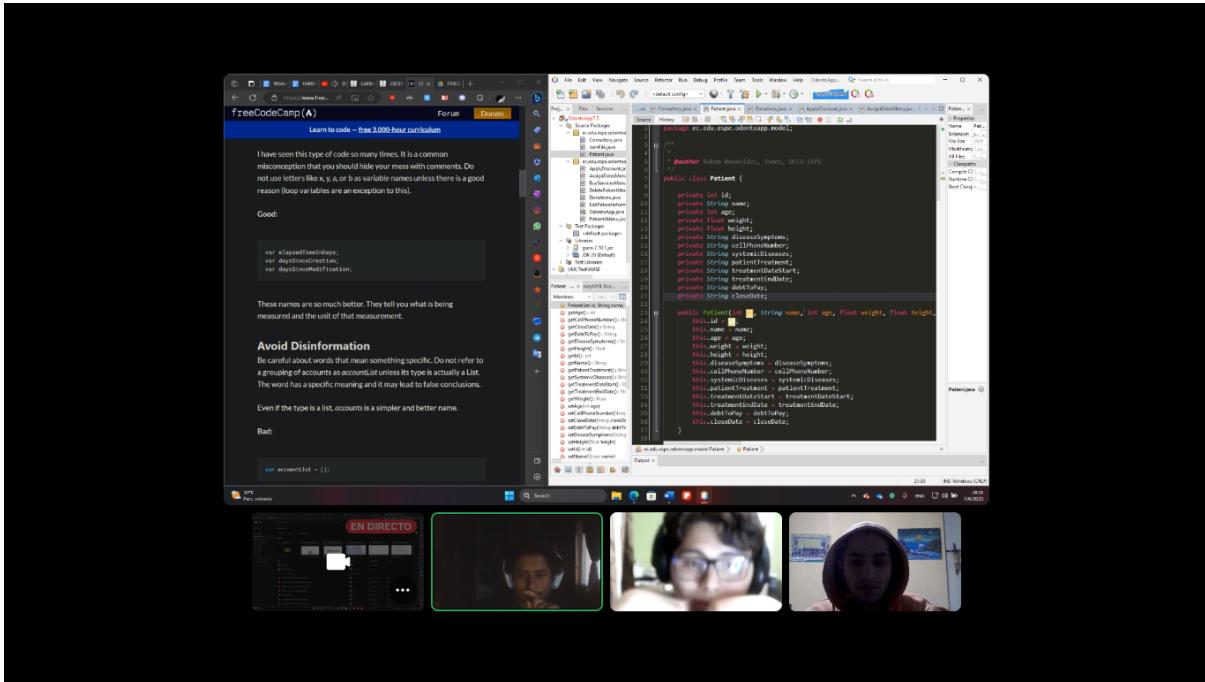
```
Scanner scanner = new Scanner( source: System.in);

boolean salir = false;
while (!salir) {
    System.out.println( x: "----- Menú de Donaciones -----");
    System.out.println( x: "1. Realizar una donación");
    System.out.println( x: "2. Ver total de donaciones");
    System.out.println( x: "3. Salir");
    System.out.print( s: "Seleccione una opción: ");
    int opcion = scanner.nextInt();

    switch (opcion) {
        case 1:
            System.out.print( s: "Ingrese la cantidad de la donación: ");
            double cantidad = scanner.nextDouble();
            realizarDonacion(cantidad);
            System.out.println( x: "¡Donación realizada con éxito!");
            break;
        case 2:
            System.out.println("El total de donaciones recibidas es: $" + totalDonaciones);
            break;
        case 3:
            salir = true;
            System.out.println( x: "¡Gracias por utilizar el programa de donaciones!");
            System.exit( status:0);
            break;
        default:
            System.out.println( x: "Opción inválida. Por favor, seleccione una opción válida.");
            break;
    }
    System.out.println();
}

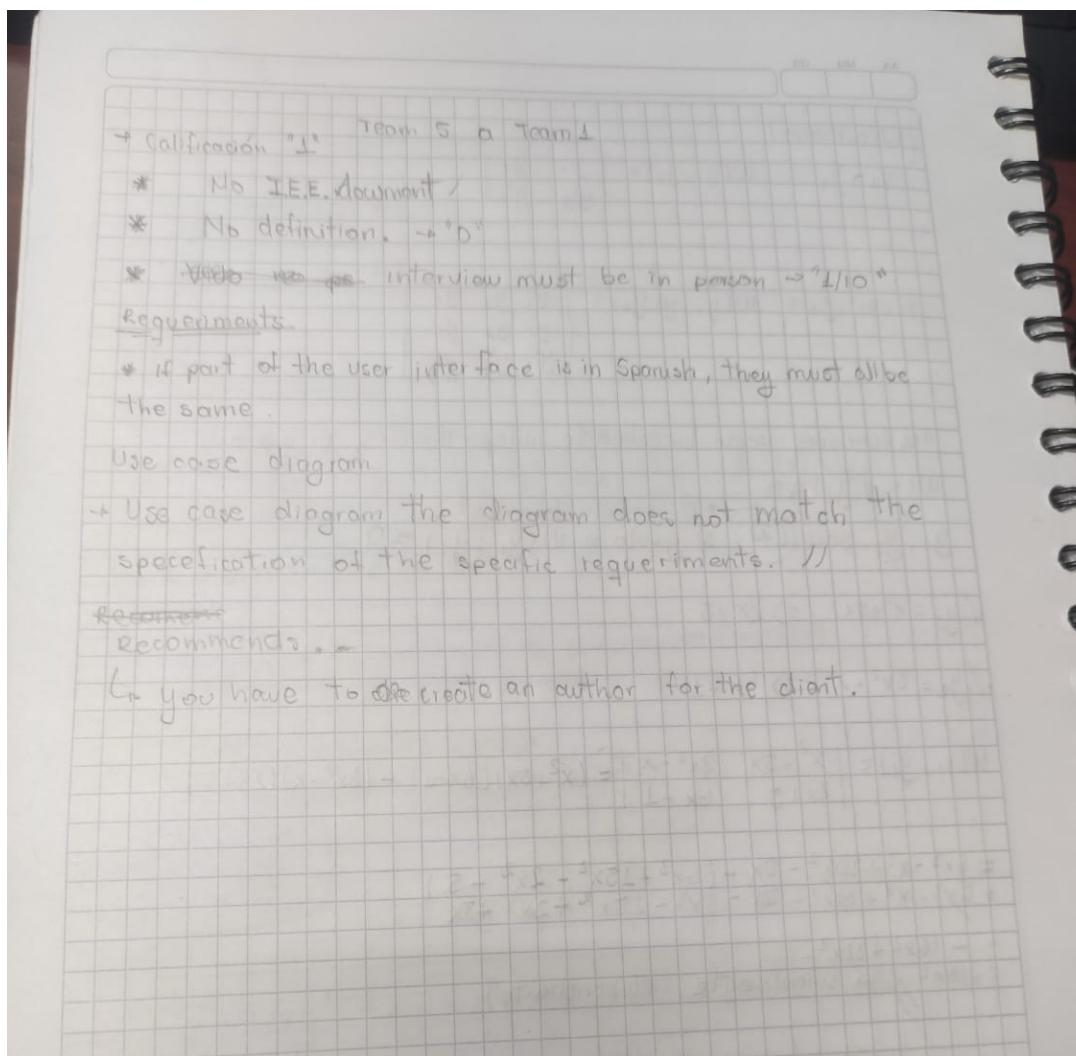
scanner.close();

private static void realizarDonacion(double cantidad) {
    totalDonaciones += cantidad;
```



- EVIDENCE NOTES

-



Team2: KillChain

Project's Name: EVSOSTore

Leader: ANABEL DAVILA MOLINA

GitHub: <https://github.com/JuDav-093/ESPE23-KillChainTeam>

Interview Video: <https://youtu.be/7rRxT3tVh-g>

Inspector: Team 6, Edison Verdesoto

5. PAMELA NAOMI CHIPE ZAMBRANO
6. JOAN OSWALDO COBEÑA ZAMBRANO
7. ANABEL DAVILA MOLINA
8. ANDRES ALEXANDER ESPIN ANDRADE

Presentation 9/10

Rubric

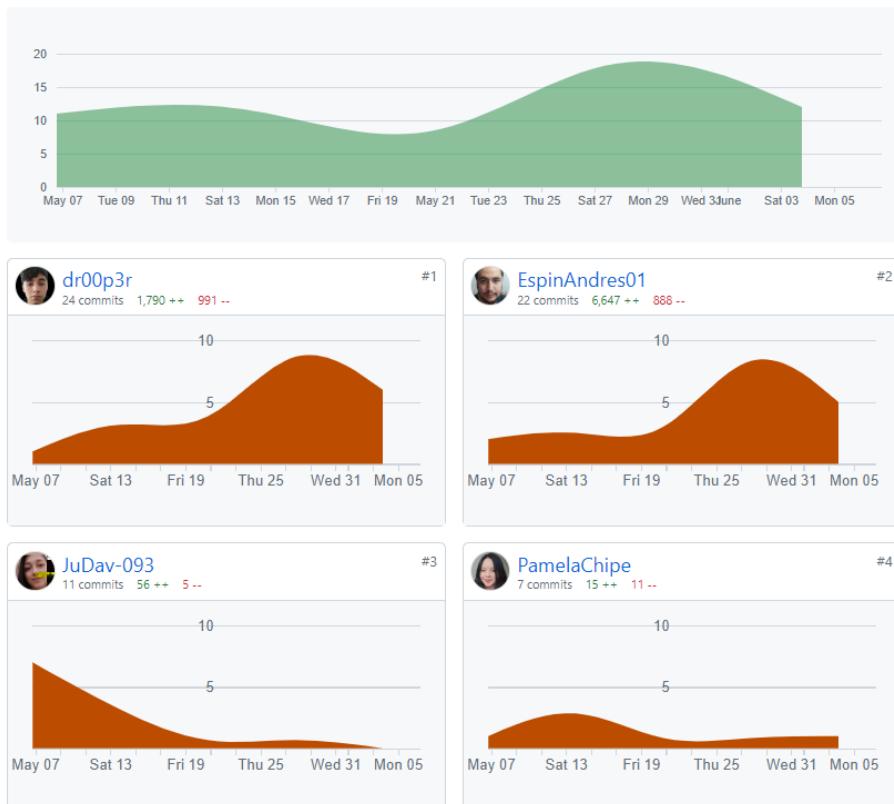
Presentation

0.- GitHub	100/100	7:20-7:25
1.- Interview and Presentation	8.5/10	the video is too loud and has a lot of echo, voices are not clear.

2.- Project definition comparison	8.2/10 requirements sets but missing analyst
3.- Software Requirements	8/10 missing UML diagrams
4.- Use Case	0/10 no Use Case Diagram
5.- Class Diagram	9/10 missing a multiplicity
6.- Data files	9/10 the program generate just 2 Json file
7.- Execution	9/10 the items selected by keyboard don't match the ID in the console
8-10.- Clean Code	30/30 complies with Clean Code standards
TOTAL	81.7/100

Evidence

GitHub



1.- Interview and Presentation



2.- Project definition

Problem

We need a system to manage the operations of a store called EVSU Store. This system should allow us to manage a catalog of products for sale, make sales and purchases, manage inventories, record sales, generate invoices, and manage customers, suppliers, and managers.

General description

In the field of retail, it is essential to maintain effective control over stocks, sales and purchases. Additionally, it is crucial to have a clear record of customer-employee interactions. This system will allow the EVSU Store to perform all of these tasks efficiently and effectively.

3.- Software Requirements

3. Specific requirements

3.1 External interfaces

- **Api:** It is planned to use text generation APIs to create reviews of the products sold in the store.
- User Interface: Intuitive and easy to use design.
- Database interface: interaction with the MongoDB database to store and retrieve information.

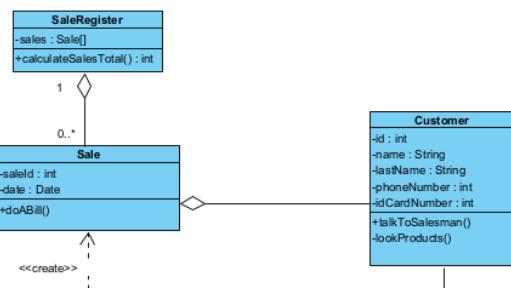
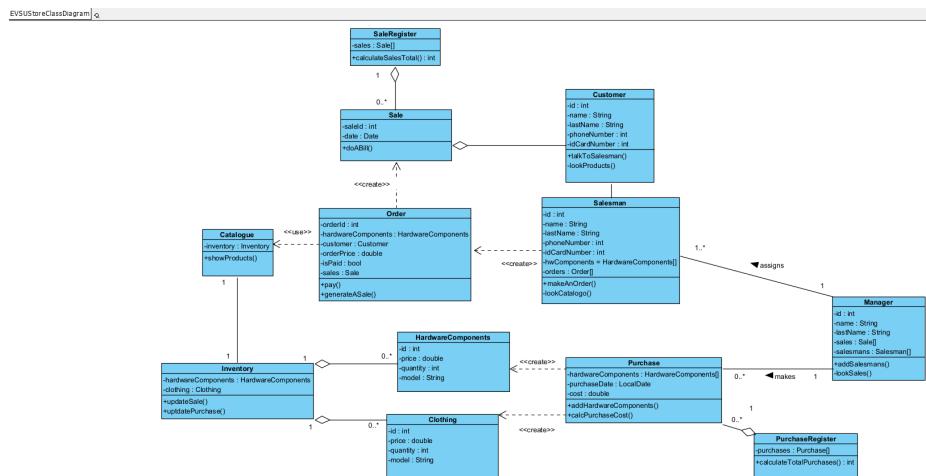
3.2 Functions

- Inventory management: add, remove and update products.
- Sales Tracking: record sales and generate reports.
- Customer Information Management: Store and update customer information.
- Vendor Recommendation: Suggest vendors based on component compatibility and quality.
- Algorithm to generate compatibility according to the components you buy.
- Implement a payment system.
- Implement a text generator to automate the creation of reviews and thus improve the business.

4.- Use Case

rive - Personal > Escritorio > Team2 > ESPE23-KillChainTeam > 04-UMLDiagrams > 01-UseCases			
Nombre	Fecha de modificación	Tipo	Tamaño
 README.md	7/6/2023 7:28	Archivo MD	1 KB

5.- Class Diagram



6.- Data files

```
1 <No se seleccionó ningún esquema>
1 1 {"hardwareComponents": [{"id":1,"individualPrice":150.0,"totalPrice":300.0,"individualCost":180.0,"totalCost":360.0,"quantity":2,"name":"GTX 1650","model":"Asus"}, {"id":2,"individualPrice":45.0,"totalPrice":90.0,"individualCost":40.0,"totalCost":80.0,"quantity":4,"name":"Gtx 1650 Asus e","model":"Asus e"}, {"id":3,"individualPrice":45.0,"totalPrice":90.0,"individualCost":40.0,"totalCost":80.0,"quantity":3,"name":"Gtx 1650 Asus e","model":"Asus e"}]}
1 <No se seleccionó ningún esquema>
1 1 {"purchases": [{"hardwareComponents": [{"id":1,"individualPrice":150.0,"totalPrice":300.0,"individualCost":180.0,"totalCost":360.0,"quantity":2,"name":"GTX 1650","model":"Asus"}, {"id":2,"individualPrice":45.0,"totalPrice":90.0,"individualCost":40.0,"totalCost":80.0,"quantity":4,"name":"Gtx 1650 Asus e","model":"Asus e"}, {"id":3,"individualPrice":45.0,"totalPrice":90.0,"individualCost":40.0,"totalCost":80.0,"quantity":3,"name":"Gtx 1650 Asus e","model":"Asus e"}]}]}
```

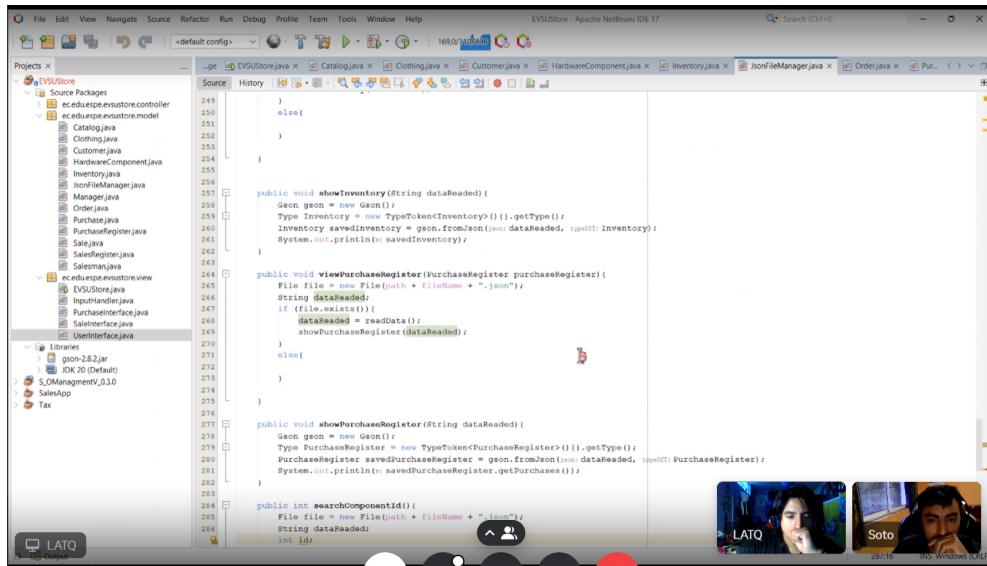
7.- Execution

```
Output - EVSUSStore (run)
*****
3
-----
--- Productos disponibles ---
ID:1  || GTX 1650 Asus      || PVP: 150.0    || Unidades disponibles: 2
ID:2  || Gtx 1650 Asus e    || PVP: 45.0     || Unidades disponibles: 4
ID:2  || Gtx 1650 Asus e    || PVP: 45.0     || Unidades disponibles: 3
ID:2  || Gtx 1650 Asus e    || PVP: 45.0     || Unidades disponibles: 3
No hay mas productos para mostrar.
Ingrese el indice del producto que desea comprar: 2
Ha seleccionado el siguiente componente de hardware:
ID:2  || Gtx 1650 Asus e    || PVP: 45.0     || Unidades disponibles: 3
Ingrese la cantidad a comprar: 2
Compra realizada exitosamente.
*****
EVSU STORE
*****
1.Hacer compra
2.Ver productos
3.Realizar venta
4.Ver registros de compras
5.Salir

Escoja una opcion
*****
2

Componentes de Hardware:
ID:1  || GTX 1650 Asus      || PVP: 150.0    || Unidades disponibles: 2
ID:2  || Gtx 1650 Asus e    || PVP: 45.0     || Unidades disponibles: 4
ID:2  || Gtx 1650 Asus e    || PVP: 45.0     || Unidades disponibles: 3
ID:2  || Gtx 1650 Asus e    || PVP: 45.0     || Unidades disponibles: 3
ID:2  || Gtx 1650 Asus e    || PVP: 45.0     || Unidades disponibles: 3
```

8-10.- Clean Code



The screenshot shows the Apache NetBeans IDE interface with the project 'EVSISotre' open. The code editor displays a Java file with several methods: `showInventory`, `viewPurchaseRegister`, `showPurchaseRegister`, and `searchComponentId`. The code uses modern Java features like `Gson` for JSON parsing and `TypeToken` for type safety. The code is annotated with line numbers (249 to 284) and includes comments explaining the logic. The IDE's toolbar and menu bar are visible at the top.

Evidence Notes:

Git Hub
Joan and Andres has the most commits, followed by Andrei and last is Pamela.

Interview
Everyone appears on the video, but it's too loud and the voices aren't listenable.

Definitions
The definitions are clear and shows us a clear definition and the requirements for the project, well structured.

Software Requirements
Missing UML diagrams. Well specified software product, missing some info from the presentation in the document.

UML Case
The team don't have a case diagram.

Class Diagram
Name changed from vendor to salesmen making the UML hard to follow, missing one multiplicity, update the name of the class.

Code
Some classes are extra in the code, this classes aren't in the document or the UML.

Execution
If you enter a product it duplicates the item, the item selected doesn't



Team3: Software Juniors

Project's Name: Bethsabe's Boutique Management

Leader: Carlos Jaya

GitHub: <https://github.com/cajaya1/MGOOP-SOFTWAREJUNIORS>

Inspector: Team 2 ANABEL DAVILA MOLINA

9. ALEJANDRO CAETANO FLORES MAYA
10. JORDAN ALEXANDER GUAMAN ALCIVAR
11. CARLOS ANDRES JAYA HERRERA
12. BRYAN ALEXANDER JUMBO SALCEDO

presentation: 10/10

Rubric

Presentation

0.- GitHub /100
1.- Interview and Presentation 4.5/10 no appear the members, Only exists file txt of evidence

2.- Project definition 8.5/10 Missing business rules, just define a CRUD.

3.- Software Requirements 8/10 Missing diagrams, the terms are not defined, just declared.

4.- Use Case 9.5/10 The diagram is just missing the generated report action.

5.- Class Diagram 8.5/10 Missing 1 multiplicities, there are methods with class attributes that don't exist in the diagram

6.- Data files 7/10 Just 1 json file and just has 1 class involved.

7.- Execution 7/10 Not handling exceptions, not considering the blank spaces in the strings reading.

8-10.- Clean Code 25/30

TOTAL 78/100

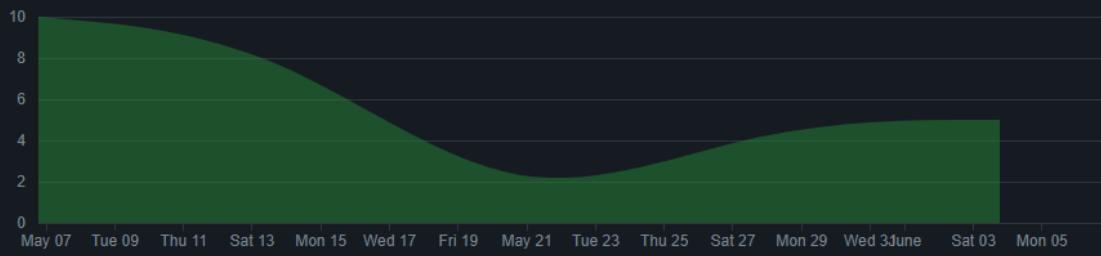
Evidence

GitHub

May 7, 2023 – Jun 7, 2023

Contributions: Commits ▾

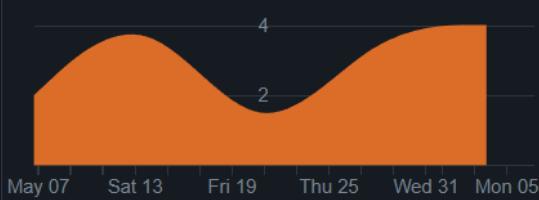
Contributions to main, excluding merge commits and bot accounts



FloresCaetano

15 commits 3,007 ++ 209 --

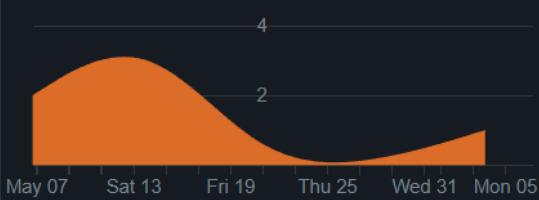
#1



GuamanJordan

7 commits 1 ++ 0 --

#2



cajaya1

6 commits 119 ++ 19 --

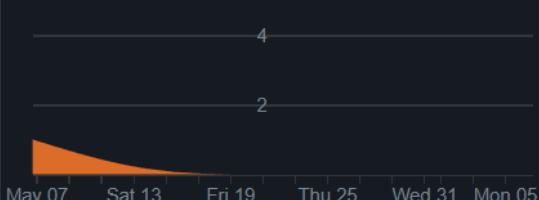
#3



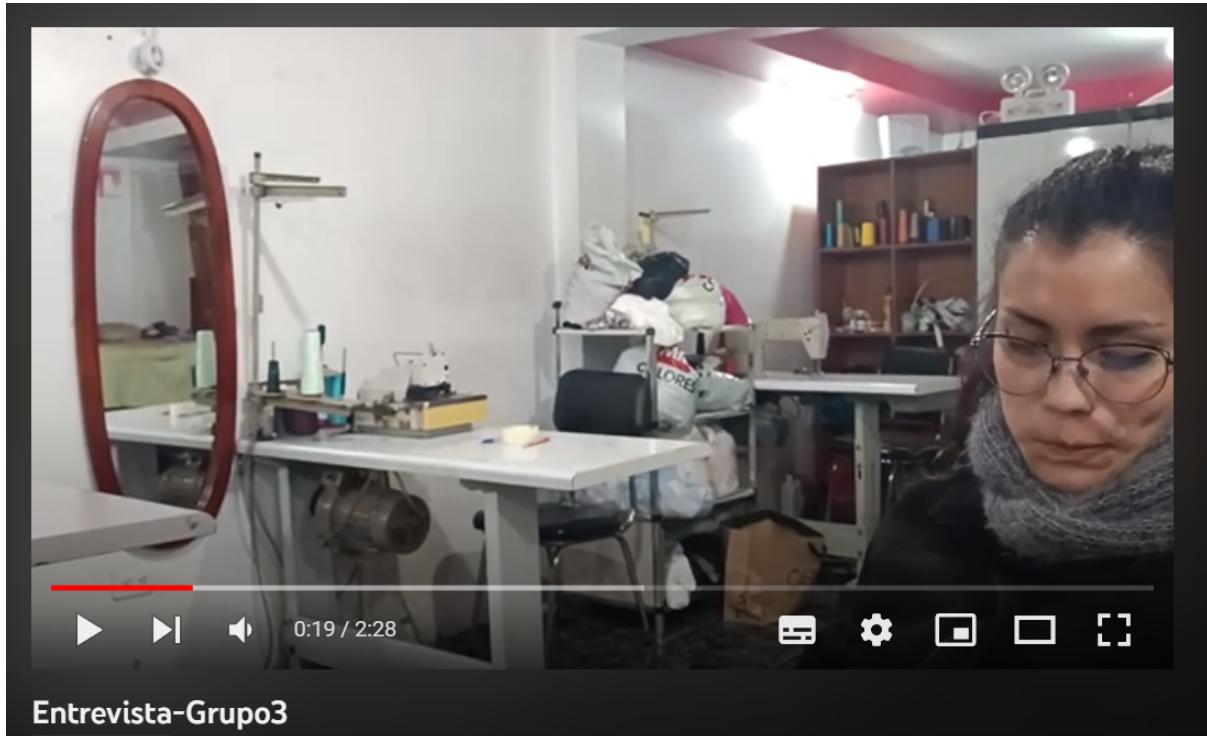
AlexJumbo

1 commit 2 ++ 0 --

#4



1.- Interview and Presentation



Entrevista-Grupo3

2.- Project definition

Problem Definitions (BethsabéBoutique)

Members Team:

Flores Caetano, Guaman Jordan, Jaya Carlos, Jumbo Alexander

Needs more business
rules in the project
definition

Problem:

We need a program that acts as a manager; that can perform actions such as showing the dresses in stock, add new dresses, be able to edit the information of the entered dresses, delete a dress and that manages to generate a readable report with the input data.

Overview:

Small stores are beginning to implement the use of administrative software to have better data management and automate processes that were done manually not long ago.

Store managers look for solidity and confidence in the software that will be applied in store management.

3.- Software Requirements

1. INTRODUCTION

***Details:**

The clothing store "Bethsabé Boutique" is a family business that is in Quito, Ecuador.

***Problem:**

After the interview and based on the answers, it can be said that the business has a problem in the financial administration, merchandise and sales control and statistical sales reports.

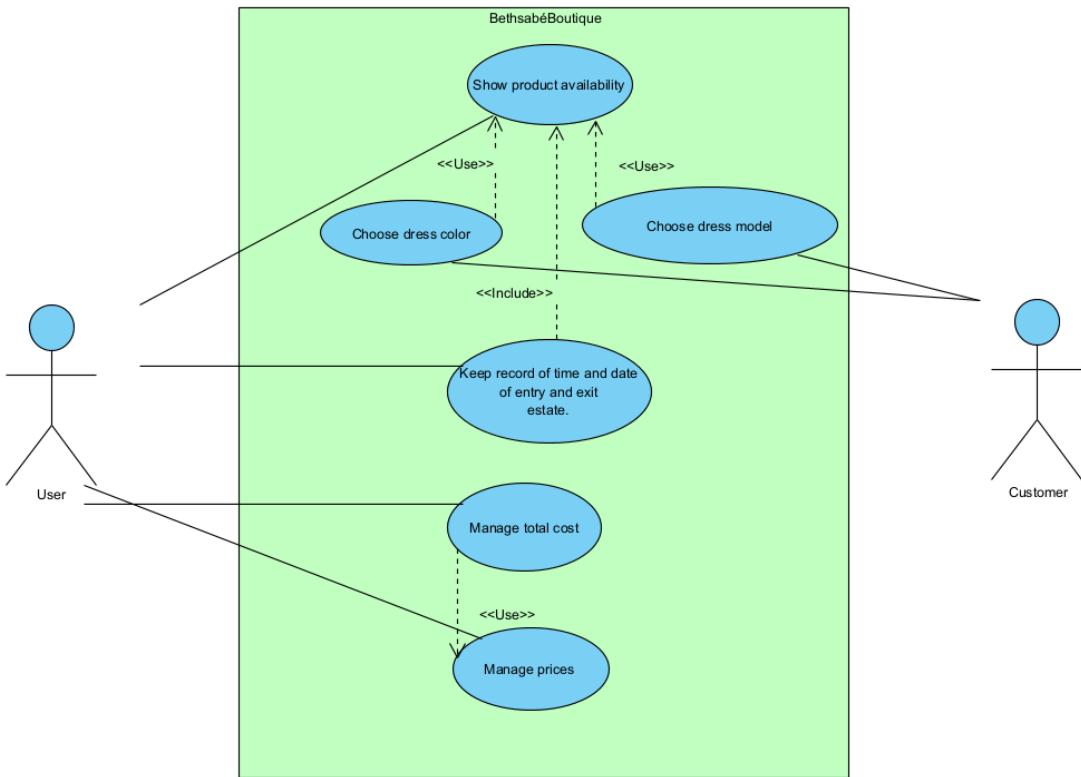
1.1. Purpose

- To optimize the human and technological resources of the clothing store.
- Develop a customer-friendly tool, knowing that customers and store staff have basic notions of technological tools.
- That the system manages to cover all the information and that the errors are minimal
- Automate processes: Simplify and streamline administrative tasks (now only inventory)

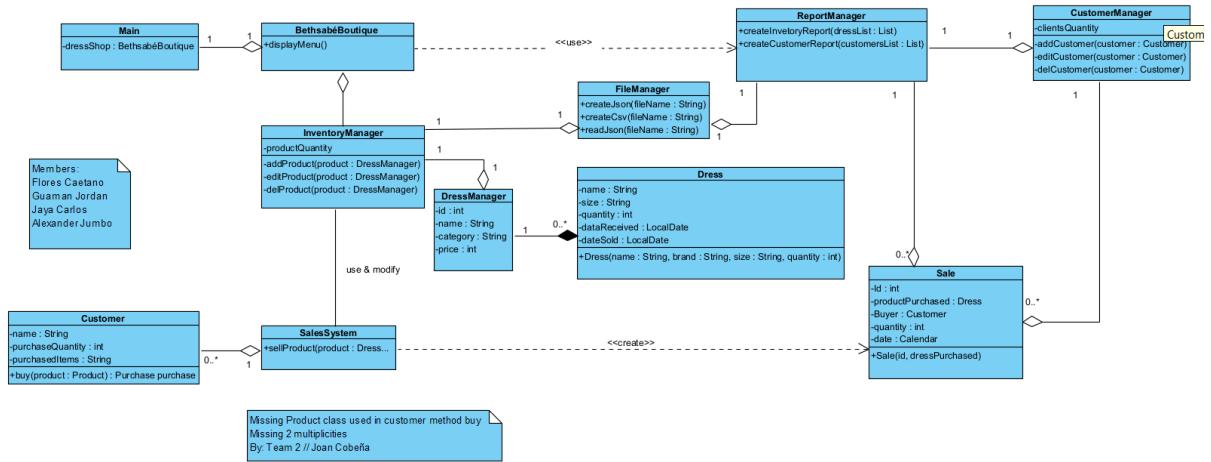
1.2. Scope

Due to our knowledge in development through code, we can make use of multiple tools provided by the structure, syntax, and libraries of the chosen programming language to develop the final program (Java), to give examples, we will use data input and output, storage, editing and deletion of data, and printing of data outside the console (Json/CSV).

4.- Use Case



5.- Class Diagram



6.- Data files

7.- Execution

```
Bethsab@Boutique
-----
1. Mostrar vestidos
2. Añadir vestido
3. Editar informacion de un vestido
4. Eliminar vestido
5. Vender un vestido
6. Generar reportes
0. Salir
Escoja una opcion: 2
Ingrese el nombre del vestido: Inspection
Ingrese la marca del vestido: Team 2
Ingrese las medidas del vestido: Ingrese el precio del vestido: 10
Exception in thread "main" java.util.InputMismatchException
    at java.base/java.util.Scanner.throwFor(Scanner.java:947)
    at java.base/java.util.Scanner.next(Scanner.java:1602)
    at java.base/java.util.Scanner.nextFloat(Scanner.java:2505)
    at ec.edu.espe.BethsabeBoutique.controller.DressManager.getDressInformation(DressManager.java:35)
    at ec.edu.espe.BethsabeBoutique.controller.InventoryManager.addDress(InventoryManager.java:26)
    at ec.edu.espe.BethsabeBoutique.view.Bethsab@Boutique.displayMenu(Bethsab@Boutique.java:42)
    at ec.edu.espe.BethsabeBoutique.view.Main.main(Main.java:13)
C:\Users\joanc\AppData\Local\NetBeans\Cache\17\executor-snippets\run.xml:111: The following error occurred while executing this line:
C:\Users\joanc\AppData\Local\NetBeans\Cache\17\executor-snippets\run.xml:68: Java returned: 1
BUILD FAILED (total time: 1 minute 31 seconds)
```

```
0. Salir
Escoja una opcion: 3
Ingrese el nombre del vestido: Manuel
Enter New Dress Name: Inspection2
Enter New Dress Brand: KillChain
Enter New Dress Size: S
Enter New Price: 13
Enter New Dress Quantity: 3
Vestido -Inspection2- editado exitosamente
Bethsab@Boutique
-----
1. Mostrar vestidos
2. Añadir vestido
3. Editar informacion de un vestido
4. Eliminar vestido
5. Vender un vestido
6. Generar reportes
0. Salir
Escoja una opcion: 1
Inventario de vestidos:
-----
VESTIDO #1
Nombre: Verdoso
Marca: Otro
Medidas: 2x2
Cantidad: 2

-----
VESTIDO #2
Nombre: Manuel
Marca: tru
Medidas: 2x10
Cantidad: 10
```

8-10.- Clean Code

The screenshot shows a Microsoft Excel window with the title "ReporteNuevo.csv - Excel". The ribbon menu is visible at the top, showing tabs like Archivo, Inicio, Insertar, etc. The main area displays a table with columns: Nombre, Fecha de modificación, and Tipo. Two rows are listed:

Nombre	Fecha de modificación	Tipo
ReporteNuevo.csv	7/6/2023 22:12	Archivo de valores
ReporteVestidos.csv	6/6/2023 7:17	Archivo de valores

Below the table, the content of the "ReporteNuevo.csv" file is displayed as a series of key-value pairs in a grid format:

A	B	C	D	E	F	G	H
4	marca:	Altro					
5	medidas:	2x2					
6	cantidad:	2					
7	FechaRecep	4/6/2023					
8							
9	Vestido #1						
10	nombre:	Manuel					
11	marca:	tru					
12	medidas:	2x10					
13	cantidad:	10					
14	FechaRecep	5/6/2023					
15							
16	Vestido #2						
17	nombre:	Vestido2					
18	marca:	Puma					
19	medidas:	20x2					
20	cantidad:	10					
21	FechaRecep	6/6/2023					
22							
23							
24							
25							

Arial - | + | **11** | **B** **I** **U** **A** **¶** | **o** **+** **□** | **≡**

1 2 3 4 5

```
public void initialize() {
}

public void printHello() {
}
```

File Edit View Insert Project Tools Window Help

src main java com.technocrafts.productlist

```
public class ProductList {
    private String name;
    private String price;
    private String description;
    private Category category;

    public ProductList(String name, String price, String desc, Category category) {
        this.name = name;
        this.price = price;
        this.description = desc;
        this.category = category;
    }

    public void add(Product product) {
        products.add(product);
    }

    public void remove(Product product) {
        products.remove(product);
    }

    public void notify(Product product) {
        for (ProductListener listener : listeners) {
            listener.onUpdate(product);
        }
    }

    public int getSize() {
        return products.size();
    }
}
```

File Edit View Insert Project Tools Window Help

src main java com.technocrafts

```
public class Product {
    private int id;
    private String name;
    private String price;
    private String description;
    private Category category;

    public Product(int id, String name, String price, String desc, Category category) {
        this.id = id;
        this.name = name;
        this.price = price;
        this.description = desc;
        this.category = category;
    }

    public void update() {
        listeners.forEach(listener -> listener.onUpdate(this));
    }

    public void remove() {
        listeners.forEach(listener -> listener.onRemove(this));
    }

    public void notify() {
        listeners.forEach(listener -> listener.onUpdate(this));
    }
}
```

The screenshot shows a Windows desktop environment. In the foreground, there is a command-line interface window titled "Output - BethsabéBoutique (run)". The window displays the following text:

```
Nombre: Verdoso
Marca: Otro
Medidas: 2x2
Cantidad: 2

-----
VESTIDO #2
Nombre: Manuel
Marca: tru
Medidas: 2x10
Cantidad: 10

-----
VESTIDO #3
Nombre: Vestido2
Marca: Puma
Medidas: 20x2
Cantidad: 10

-----
BethsabéBoutique
1. Mostrar vestidos
2. Añadir vestido
3. Editar informacion de un vestido
4. Eliminar vestido
5. Vender un vestido
6. Generar reportes
0. Salir
Escoja una opcion:
```

Below the command-line window, the taskbar shows the application is running under "Apache NetBeans".

In the background, a file explorer window is open, showing a file named "Inventory.json" with the extension ".json". The file content is partially visible:

```
[{"name": "Verdoso", "brand": "Otro", "size": "2x2", "price": 100.0, "quantity": 2, "dateReceived": "2023-06-04"}, {"name": "Manuel", "brand": "tru", "size": "2x10", "price": 10.0, "quantity": 10}]
```

EVIDENCE NOTE

Destacar la matemática en defensa de vida.

Project review

Project review from team 2 to team 3

- Pamela Chipe

- Joan Cabeña

- Anabel Dávila

Requirements list (project definition) • Andrés Espín

CRUD dresses

Generates information reports when entering dresses

Small store project

The project help with store control

Note: missing business rules in project Definition

Project definition

Missing business rules, just define a CRUD.

(Requer) Software requirements

Missing diagrams, missing index, the terms are not defined just declared.

(Team 3 project performance)

Interview and Presentation

upper the members, Only exists file txt of evidence

Use Case

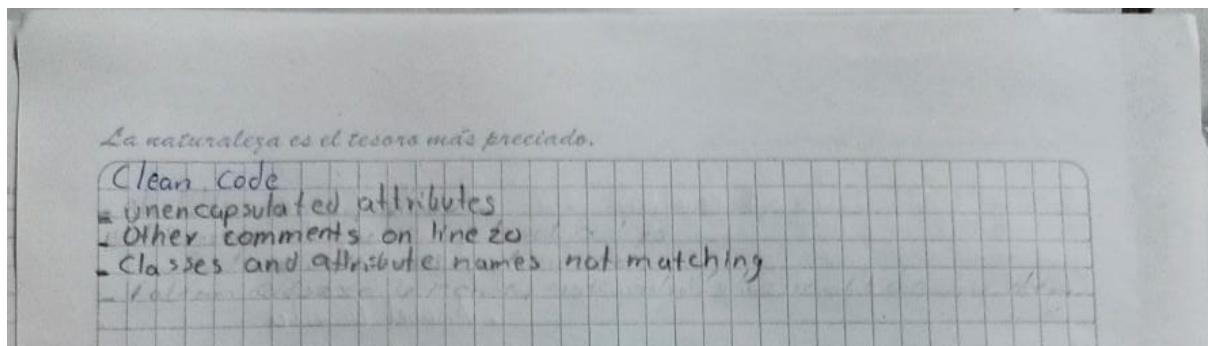
The diagram is just missing the generate report action.

Class Diagram

Missing 2 multiplicities, there are methods with class attributes that don't exist in the diagram.

Execution

Handling exceptions for characters in numeric values, not considering the blank spaces in the string's reading,



Team 4:Code Crafters

Project's name: Hardware Store

Leader:Josue Isaac Marin Alquinga

GitHub: <https://github.com/marinjosue/ESPE2305-CodeCrafters.git>

Video:<https://www.youtube.com/watch?v=sVhJWFINGwg>

Inspector: Team 3 Leader: CARLOS ANDRES JAYA HERRERA

13. JEFFREY ALEXANDER MANOBANDA CHABLA

14. JHORDY PAUL MARCILLO MORA

15. JOSUE ISAAC MARIN ALQUINGA

16. MESIAS ORLANDO MARISCAL OÑA

presentation: 8/10

Rubric

Presentation

0.- GitHub	/100
1.- Interview and Presentation	8/10
2.- Project definition	10/10
3.- Software Requirements	8/10 no class or use case diagrams on the document
4.- Use Case	8/10 Missing the shopping cart part
5.- Class Diagram	8/10 Missing some multiplicity and relations
6.- Data files	8/10 The data files are missing some classes from the project.
7.- Execution	9/10 the discount parts are not working,
8-10.- Clean Code	21/30 FILE_PATH and FILE_NAME variable is not on / trash code: useless methods on Category, HardwareStore, Product / Bad indentation on line 21 in HardwareStoreSimulator / Incorrect name on "Costomer" class and a missing space on line 39 in Project.
TOTAL	80/100

Evidence

GitHub



1.- Interview and Presentation



2.- Project definition

- Inventory: An inventory management system will be required that allows users to see the available quantity of each product in real time. This system must be accurate and constantly updated so that customers can make informed decisions about their purchases.
- Product Catalog: The product catalog functionality involves the creation of a product database with details such as images, descriptions, prices, among others. To implement this feature, it is necessary to have a content management system that allows the creation, update and deletion of products easily.
- Promotions and offers: To offer promotions and special offers in the application, a promotion management system will be required that allows users to view and apply discounts to selected products. This system should be easy to use and ensure discounts are applied correctly.
- Shopping Cart: Shopping cart functionality involves allowing customers to add products to a virtual cart and make a purchase. To implement this, a shopping cart management system will be required that allows adding, removing and updating selected products before finalizing the purchase.
- Customer Service: Customer service functionality involves providing a way for customers to contact customer service with questions or problems. This can be accomplished through a live chat system or a contact form that allows customers to submit questions and receive responses in real time.

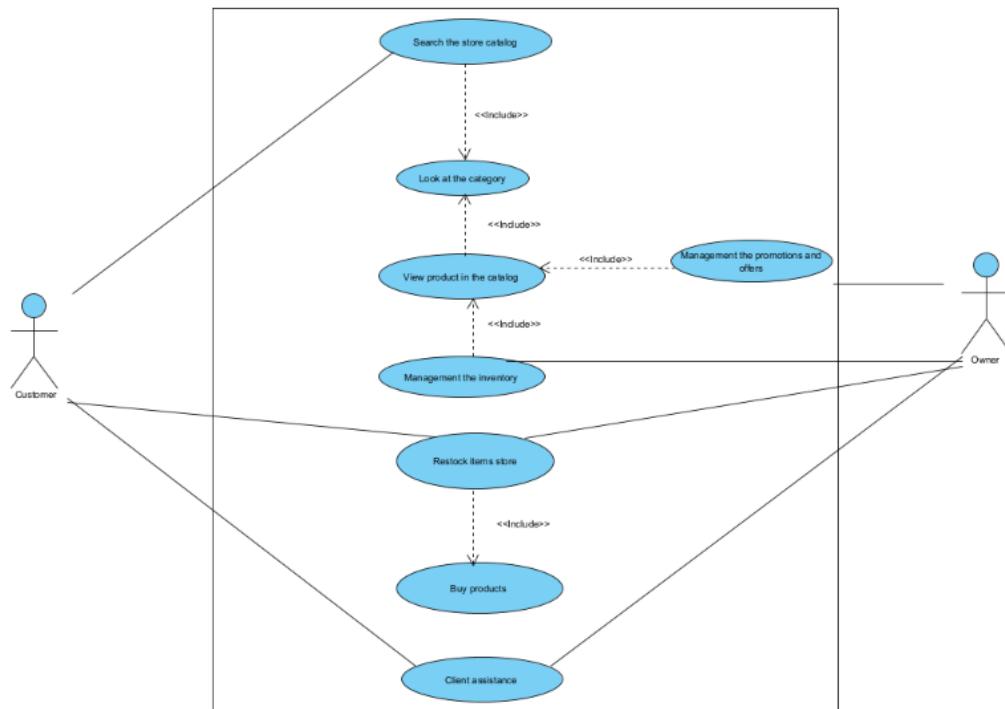
3.- Software Requirements

1.2 Scope of the system

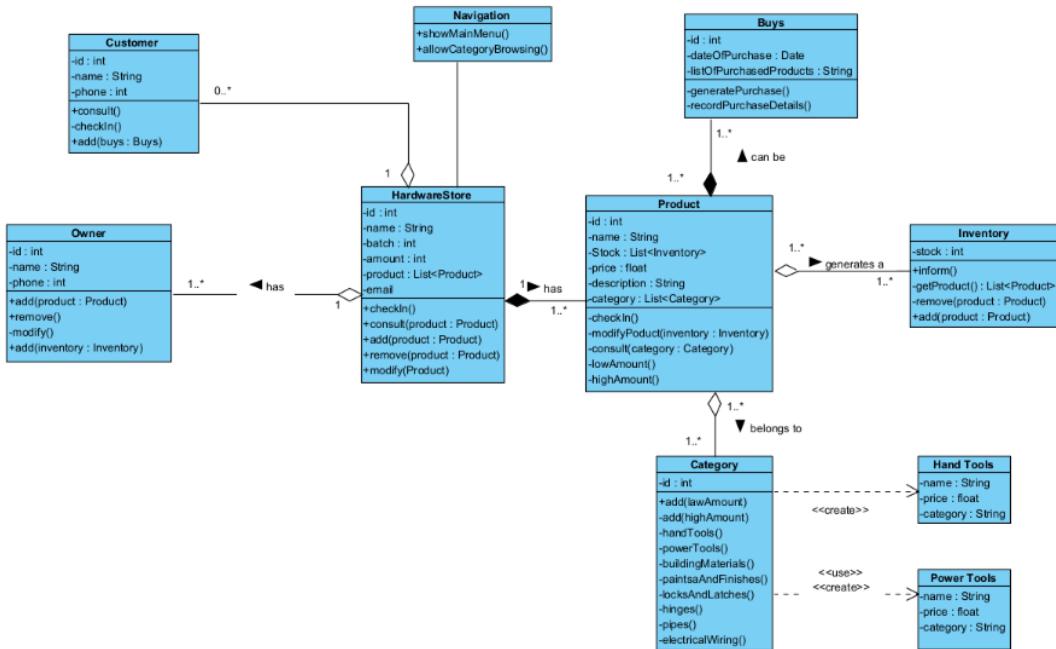
The application will include the following features:

- Inventory: An inventory management system will be required that allows users to see the available quantity of each product in real time. This system must be accurate and constantly updated so that customers can make informed decisions about their purchases.
- Product Catalog: The product catalog functionality involves the creation of a product database with details such as images, descriptions, prices, among others. To implement this feature, it is necessary to have a content management system that allows the creation, update and deletion of products easily.
- Promotions and offers: To offer promotions and special offers in the application, a promotion management system will be required that allows users to view and apply discounts to selected products. This system should be easy to use and ensure discounts are applied correctly.
- Shopping Cart: Shopping cart functionality involves allowing customers to add products to a virtual cart and make a purchase. To implement this, a shopping cart management system will be required that allows adding, removing and updating selected products before finalizing the purchase.
- Customer Service: Customer service functionality involves providing a way for customers to contact customer service with questions or problems. This can be accomplished through a live chat system or a contact form that allows customers to submit questions and receive responses in real time.

4.- Use Case



5.- Class Diagram



6.- Data files

```

data: Bloc de notas
Archivo Edición Formato Ver Ayuda
{
    "id": 1,
    "name": "Mi ferreteria",
    "batch": 1,
    "amount": 1,
    "productList": [
        {
            "id": 1,
            "name": "Martillo",
            "stock": 10,
            "price": 4.0,
            "description": "El martillo es una herramienta de mano compuesta por una cabeza de metal, generalmente de acero, unida a un mango o mango de madera, plastico o fibra de vidrio.",
            "regularPrice": 0.0,
            "discount": 0.0,
            "discountPercentage": 0.0
        },
        {
            "id": 2,
            "name": "Clavo",
            "stock": 21,
            "price": 0.75,
            "description": "Los clavos son elementos de fijacion fabricados en metal, generalmente acero, que se utilizan para unir y asegurar materiales de manera firme y duradera",
            "regularPrice": 0.0,
            "discount": 0.0,
            "discountPercentage": 0.0
        },
        {
            "id": 3,
            "name": "Brocha",
            "stock": 20,
            "price": 2.0,
            "description": "La brocha es una herramienta de pintura versatil y duradera, elaborada para facilitar la aplicacion uniforme de pintura sobre diversas superficies.",
            "regularPrice": 0.0,
            "discount": 0.0,
            "discountPercentage": 0.0
        }
    ],
}

```

7.- Execution

```
Output - Project (run)
run:
----- BIENVENIDO A FERRETERIA DSA -----
DESEA INGRESAR COMO:
1. Propietario
2. Usuario
3. Ver Comentarios del establecimiento
4. Salir del sistema
Ingresse la opcion deseada: 3
----- Menu de Comentario -----
Nombre: Maria Gomez
Comentario: Esta ferreteria es increible. Tienen todo lo que necesitas para cualquier proyecto de bricolaje. El personal es amable y siempre dispuesto a ayudar.

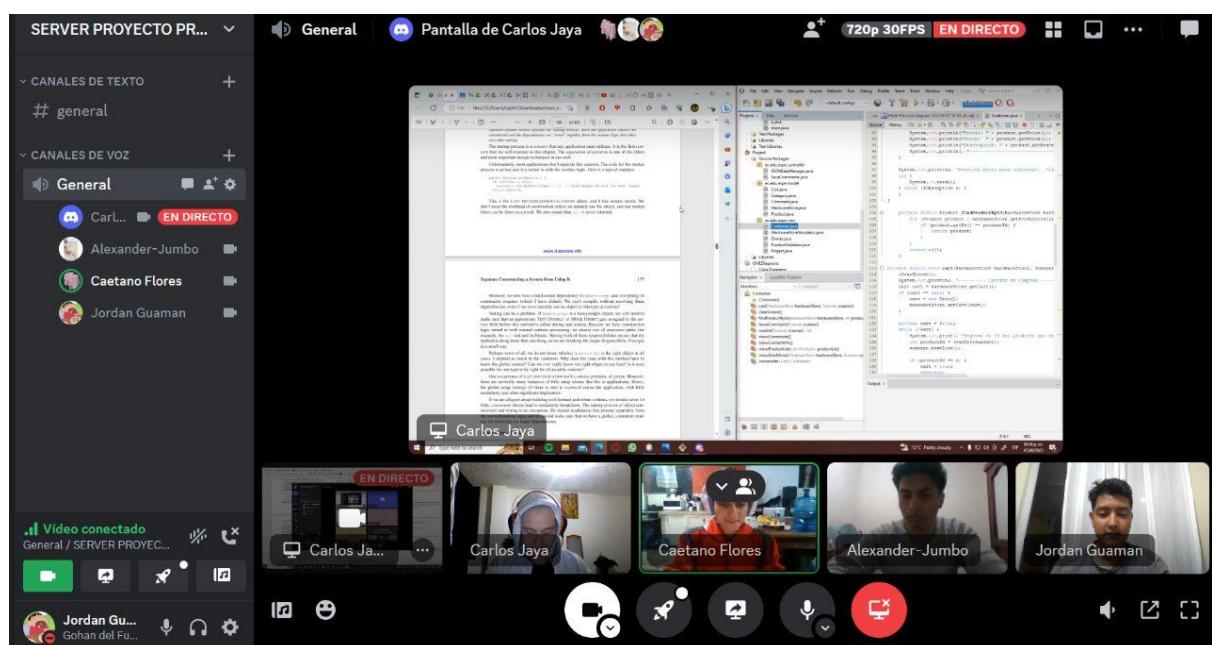
Nombre: pepito
Comentario: me llego un producto incorrecto, helpme

Nombre: Jesus
Comentario: Me parece interesante

Nombre: Gracias
Comentario: Gracias

----- BIENVENIDO A FERRETERIA DSA -----
DESEA INGRESAR COMO:
1. Propietario
2. Usuario
3. Ver Comentarios del establecimiento
4. Salir del sistema
Ingresse la opcion deseada: 2
----- Menu de Usuario -----
1. Ver Catalogo
2. Mostrar promociones y ofertas
3. Carrito de compras
```

8-10.- Clean Code



The image shows three vertically stacked Java code editors from an IDE interface. Each editor has a toolbar at the top with various icons for file operations, code navigation, and search.

Top Editor: JSONDataManager.java

```
13 import java.io.BufferedReader;
14 import java.io.FileReader;
15 import java.io.FileWriter;
16 import java.io.IOException;
17
18 //autor: Jhordy Marcillo
19
20 public class JSONDataManager {
21     private final String FILE_PATH = "data.json";
22
23     public void saveData(HardwareStore hardwareStore) {
24         Gson gson = new GsonBuilder().setPrettyPrinting().create();
25         HardwareStore existingHardwareStore = loadData();
26         if (existingHardwareStore != null) {
27             existingHardwareStore.addAll(hardwareStore.getProductList());
28         } else {
29             existingHardwareStore = hardwareStore;
30         }
31         String json = gson.toJson(existingHardwareStore);
32
33         try (BufferedWriter writer = new BufferedWriter(new FileWriter(FILE_PATH))) {
34             writer.write(json);
35         } catch (IOException e) {
36             e.printStackTrace();
37         }
38     }
39
40     public HardwareStore loadData() {
41         try (BufferedReader reader = new BufferedReader(new FileReader(FILE_PATH))) {
42             Gson gson = new Gson();
43             return gson.fromJson(reader, HardwareStore.class);
44         } catch (IOException e) {
45             e.printStackTrace();
46         }
47         return null;
48     }
49 }
```

Middle Editor: SaveComments.java

```
16 import java.io.FileWriter;
17 import java.io.IOException;
18 import java.lang.reflect.Type;
19 import java.util.ArrayList;
20 import java.util.List;
21
22 public class SaveComments {
23     private static final String FILE_NAME = "comments.json";
24
25     public static void save(List<Comment> newComments) {
26         List<Comment> comments = new ArrayList<>();
27         Gson gson = new Gson();
28         Type listType = new TypeToken<List<Comment>>() {}.getType();
29
30         try (FileReader reader = new FileReader(FILE_NAME)) {
31             comments = gson.fromJson(reader, listType);
32         } catch (IOException e) {
33             e.printStackTrace();
34         }
35
36         comments.addAll(newComments);
37         String json = gson.toJson(comments, listType);
38
39         try (FileWriter writer = new FileWriter(FILE_NAME)) {
40             writer.write(json);
41         } catch (IOException e) {
42             e.printStackTrace();
43         }
44     }
45
46     public static List<Comment> load() {
47         List<Comment> comments = new ArrayList<>();
48         Gson gson = new Gson();
49         Type listType = new TypeToken<List<Comment>>() {}.getType();
50
51         try (FileReader reader = new FileReader(FILE_NAME)) {
52             comments = gson.fromJson(reader, listType);
53         } catch (IOException e) {
54             e.printStackTrace();
55         }
56     }
57 }
```

Bottom Editor: Category.java

```
4 /**
5 * @author Jessie Marin, Jhordy Marcillo, Jeffrey Mancbanda, Mesias Mariscal
6 * CodeCrafters: DCCO-ESPE
7 */
8 public class Category {
9     private int id;
10
11     public Category(int id) {
12         this.id = id;
13     }
14
15     public void add(int lowAmount) {
16     }
17
18     public void add(int highAmount) {
19     }
20
21     public void handTools() {
22     }
23
24     public void powerTools() {
25     }
26
27     public void buildingMaterials() {
28     }
29
30     public void paintAndFinishes() {
31     }
32
33     public void locksAndLatches() {
34     }
35
36     public void hinges() {
37     }
38 }
```

HardwareStoreSimulator.java

```
28     this.batch = batch;
29     this.amount = amount;
30     this.productList = productList;
31     this.email = email;
32 }
33
34 @Override
35 public String toString() {
36     return "HardwareStore(" + id + ", name=" + name + ", batch=" + batch + ", amount=" + amount + ", productList=" + productList + ", email=" + email + ')';
37 }
38
39 public void checkIn() {
40 }
41
42 public List<Product> consult(List<Product> productList) {
43     return null;
44 }
45
46 public void add(List<Product> productList) {
47 }
48
49 public void remove(List<Product> productList) {
50 }
51
52 public void modify(Product product) {
53 }
54
55 /**
56 * @return the id
57 */
58 public int getId() {
59     return id;
60 }
61
62 /**
63 */
64 public int getBatch() {
65     return batch;
66 }
67
68 /**
69 */
70 public void setBatch(int batch) {
71     this.batch = batch;
72 }
73
74 /**
75 */
76 public void setAmount(int amount) {
77     this.amount = amount;
78 }
79
80 /**
81 */
82 public void setProductList(List<Product> productList) {
83     this.productList = productList;
84 }
85
86 /**
87 */
88 public void setEmail(String email) {
89     this.email = email;
90 }
91
92 /**
93 */
94 public void setOwner(Owner owner) {
95     this.owner = owner;
96 }
97
98 /**
99 */
100 public void setCart(Cart cart) {
101     this.cart = cart;
102 }
103
104 /**
105 */
106 public void setCategoryList(List<Category> categoryList) {
107     this.categoryList = categoryList;
108 }
109
110 /**
111 */
112 public void lowAmount() {
113 }
114
115 /**
116 */
117 public void highAmount() {
118 }
119
120 }
121
```

Product.java

```
85 }
86
87 public Product(int id, String name, int stock, float price, String description, List<Category> categoryList) {
88     this.id = id;
89     this.name = name;
90     this.stock = stock;
91     this.price = price;
92     this.description = description;
93     this.categoryList = categoryList;
94     this.regularPrice = regularPrice;
95 }
96 public float getRegularPrice() {
97     return regularPrice;
98 }
99 public void checkIn() {
100 }
101
102 public void modifyProduct() {
103 }
104
105 public Product consult() {
106     return null;
107 }
108
109 public void lowAmount() {
110 }
111
112 public void highAmount() {
113 }
114
115 }
116
117 }
118
119 }
120
121 }
```

Output

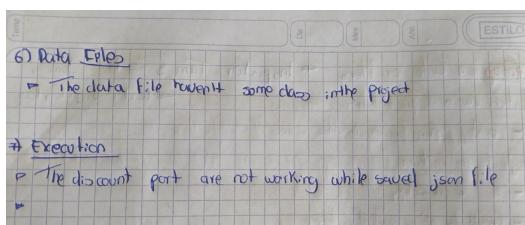
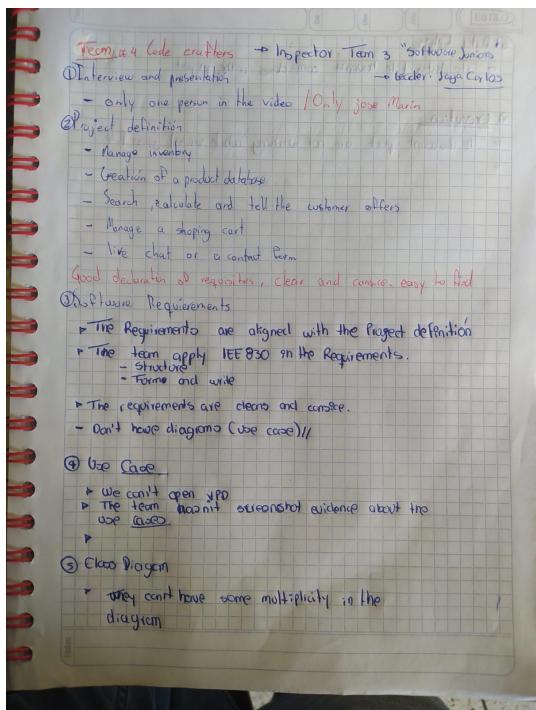
```
31 System.out.println("DESEA INGRESAR COMO:");
32 System.out.println("1. Propietario");
33 System.out.println("2. Usuario");
34 System.out.println("3. Ver Comentarios del establecimiento");
35 System.out.println("4. Salir del sistema");
36 System.out.print("Ingrese la opcion deseada: ");
37 int op = readInt(scanner);
38 if (op == 1) {
39     System.out.print("Ingrese la codigo personalde propietario: ");
```

The screenshot shows a Java code editor interface with the following details:

- Project Structure:** The tabs at the top show files: ...dg, Customer.java, HardwareStoreSimulator.java (which is the active tab), Owner.java, ProductValidator.java, JSONDataManager.java, and SaveComments.java.
- Source Tab:** The "Source" tab is selected, indicating the current view of the code.
- Code Content:** The code is as follows:

```
1 package ec.edu.espe.view;
2
3 /**
4  * @author Josue Marin, Jhordy Marcillo, Jeffrey Manobanda, Mesias Mariscal
5  * @version 1.0
6  * @since 2023-09-01
7  */
8
9 import ec.edu.espe.controller.JSONDataManager;
10 import ec.edu.espe.model.HardwareStore;
11 import static ec.edu.espe.view.Project.showMainMenu;
12
13 public class HardwareStoreSimulator {
14
15     public static void main(String[] args) {
16
17         JSONDataManager jsonDataManager = new JSONDataManager();
18         HardwareStore hardwareStore = jsonDataManager.loadData();
19         showMainMenu(hardwareStore);
20     }
21 }
```

EVIDENCE NOTE:



The FAMSE Project.

1. Abstract.

This project aims to create a product reservation system for a nut store that will allow customers to make their purchases faster and more comfortable. This system will have several functional and non-functional requirements, as well as include an intuitive and non-complex interface for customers to use.

This project will involve a software development team and the store owner, this implementation of the reservation system is expected to improve the store's customer experience and improve inventory and production management.

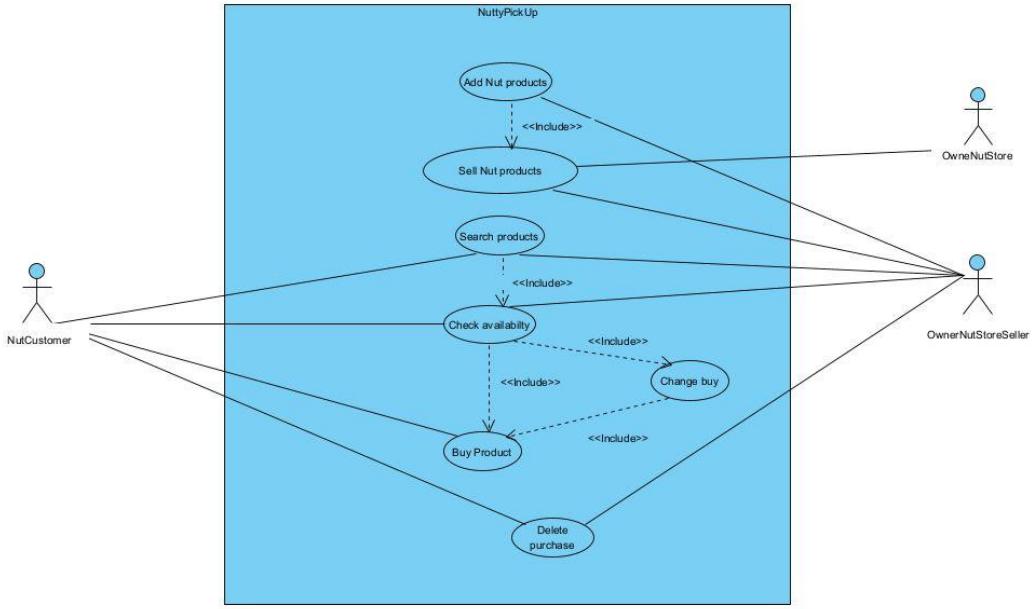
3.- Software Requirements

Bad enumeration

Does not contain diagrams

Contents

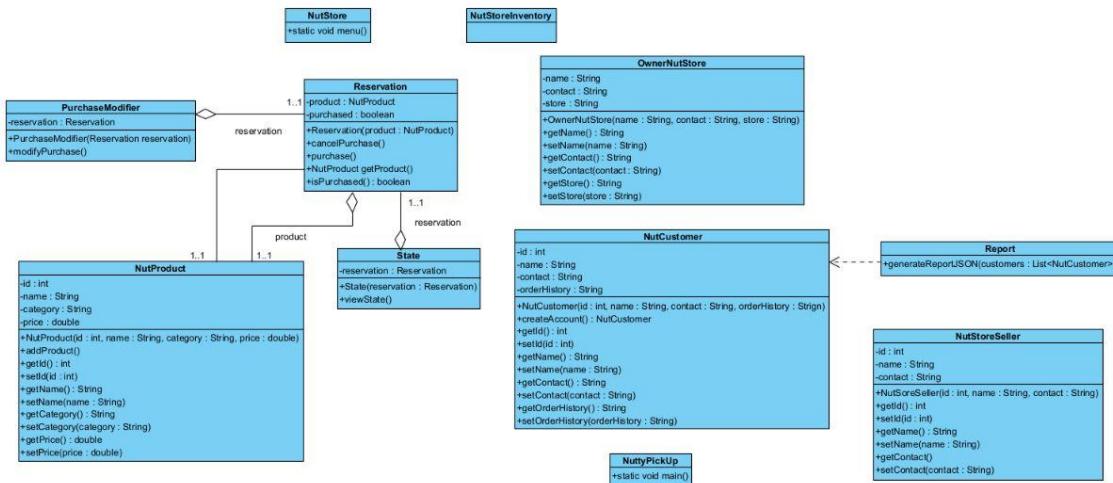
1.1 Purpose.....	5
1.2 Reach.....	5
1.3 Definitions, Acronyms and Abbreviations.....	5
1.4 Staff involved.....	6
1.5 Reference.....	6
1.6 Abstract	6
1.7 System Perspective	7
1.8 Product Functionality.....	7
1.9 User Characteristics	7
1.10 Restrictions.....	8
1.11 Assumptions and Dependence.....	8
1.12 Foreseeable Evolution of the System	8
1.13 Functional Requirements	8
1.14 Non-Functional Requirements.....	10
1.14.1 Performance Requirements.	10
1.14.2 Security.....	10
1.14.3 Availability	10
1.14.4 Maintainability	10
1.14.5 Portability	10
1.15 Other Requirements	10
4.- Use Case	



5.- Class Diagram

Missing associations and the one of "Use" and "Create"
poorly structured

Put dependencies, do not leave classes alone



6.- Data files

JSON is not updated
a single JSON file
you need a JSON for each Class

```
neDrive > Escritorio > ESPE > 1_2_3 level > OOP_9642 > Calificar > The_FAMSE > 06-Code > NuttyPickUp > {} leo.json
1  {
2    "id": 1750573733,
3    "name": "leo",
4    "contact": "096011513",
5    "orderHistory": ""
6 }
```

Team 5: The FAMSE

Project's name: Nutty Pick Up

Leader: Leonardo Narvaez

GitHub: https://github.com/LeoNarvaez2503/The_FAMSE.git

Inspector: Team 4: Josue Isaac Marin Alquinga

17. ANTHONY ALAIN MORALES CARVAJAL

18. LENIN DAVID MORAN PALOMO

19. LEONARDO VINICIO NARVAEZ CRIOLLO

presentation: 6.5/10

Rubric

Presentation

0.- GitHub	/100
1.- Interview and Presentation	8/10 (Only one member of the group appears)
2.- Project definition requirements)	6/10 (Definition does not specify the mentioned
3.- Software Requirements	8/10 (missing UML diagrams, missing some information from the presentation in the document)
4.- Use Case	8/10

5.- Class Diagram 7/10

(Missing associations and the one of "Use" and "Create"

poorly structured.)

(Put dependencies, do not leave classes alone.)

6.- Data files 6/10 (a single JSON file)

7.- Execution 7/10(Added products are not saved.)

8-10.- Clean Code 24/30

Classes that do not fulfill any function

Comments in Spanish and unnecessary

Unused floating packages

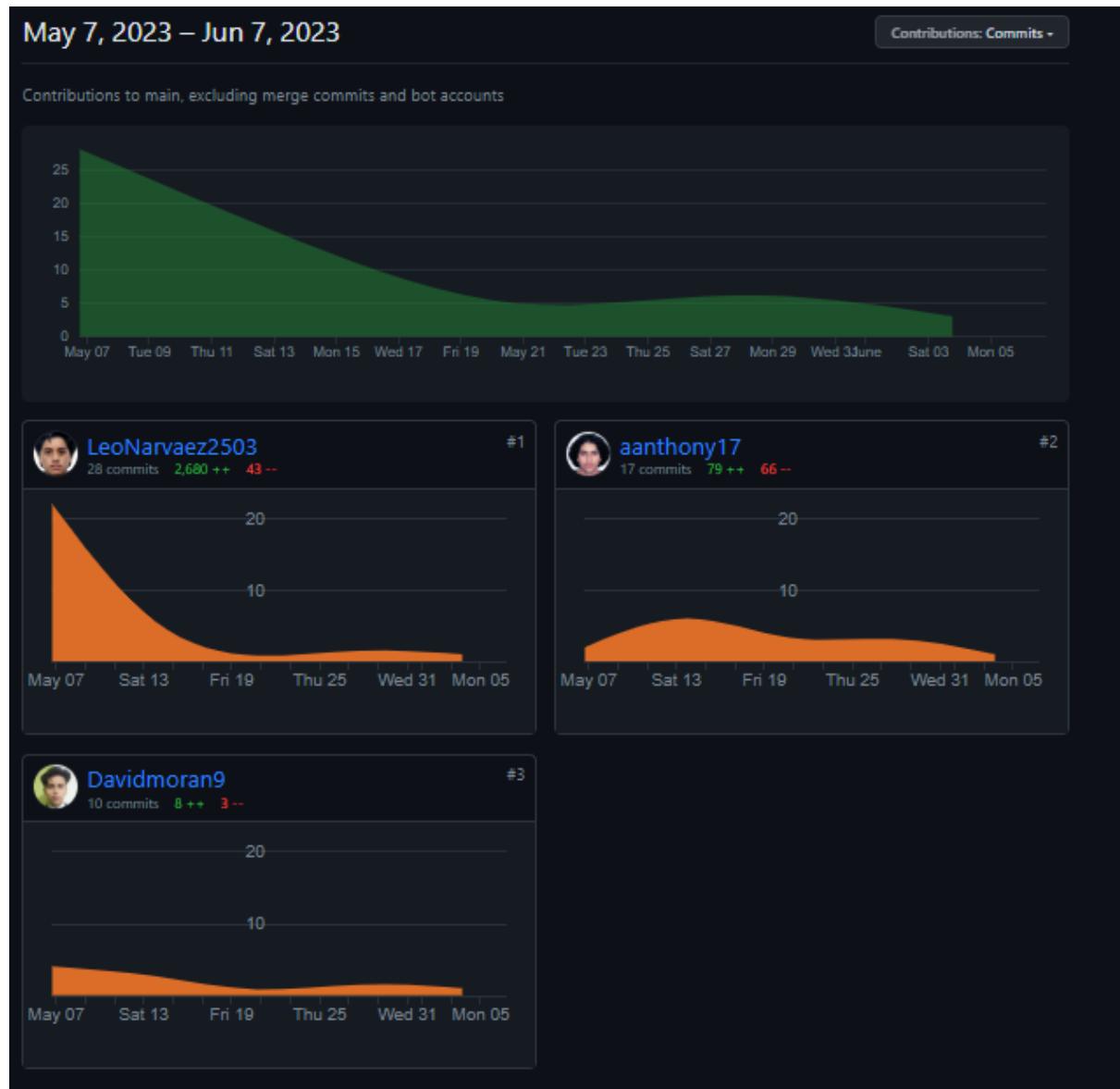
Functions that do not operate correctly

TOTAL 74/100

Evidence

GitHub

<https://www.youtube.com/watch?v=gRtJKr3Djs4>



1.- Interview and Presentation

ANTHONY ALAIN MORALES CARVAJAL:0/10

LENIN DAVID MORAN PALOMO: 10/10

LEONARDO VINICIO NARVAEZ CRIOLLO:0/10



Interview for the Software project "THE FANSE"

[No listado](#)



DAVID LEONEL
2 suscriptores

[Suscribirse](#)

[0](#)



[Compartir](#)

[Guardar](#)



2.- Project definition

	build	07/06/2023 8:40	Carpeta de archivos	
	nbproject	07/06/2023 8:40	Carpeta de archivos	
	src	07/06/2023 8:40	Carpeta de archivos	
	Anthony	07/06/2023 8:40	Archivo de valores...	1 KB
	build.xml	07/06/2023 8:40	Archivo XML	4 KB
	leo	07/06/2023 8:40	Archivo de origen ...	1 KB
	manifest.mf	07/06/2023 8:40	Archivo MF	1 KB

7.- Execution

No product catalog

Added products are not saved.

Does not generate purchases

```
Menu de opciones:  
1. Crear nuevo usuario.  
2. Agregar nuevo producto.  
3. Comprar producto.  
4. Editar compra.  
5. Estado de compra.  
6. Generar archivo JSON de reportes.  
7. Salir.  
Ingrese la opcion que desea visualizar:  
2  
Ingrese el ID del producto:  
4  
Ingrese el tipo del producto (Salado/Dulce):  
Dalsado  
Ingrese el nombre del producto:  
CACAO  
Ingrese el precio del producto (Ejemplo: 0,10):  
hola  
Solo puede ingresar numeros.  
333  
Menu de opciones:  
1. Crear nuevo usuario.  
2. Agregar nuevo producto.  
3. Comprar producto.  
4. Editar compra.  
5. Estado de compra.  
6. Generar archivo JSON de reportes.  
7. Salir.  
Ingrese la opcion que desea visualizar:  
3  
No se ha seleccionado ning n producto.
```

```

Menu de opciones:
1. Crear nuevo usuario.
2. Agregar nuevo producto.
3. Comprar producto.
4. Editar compra.
5. Estado de compra.
6. Generar archivo JSON de reportes.
7. Salir.

Ingrese la opcion que desea visualizar:
5
No se ha realizado ninguna compra.

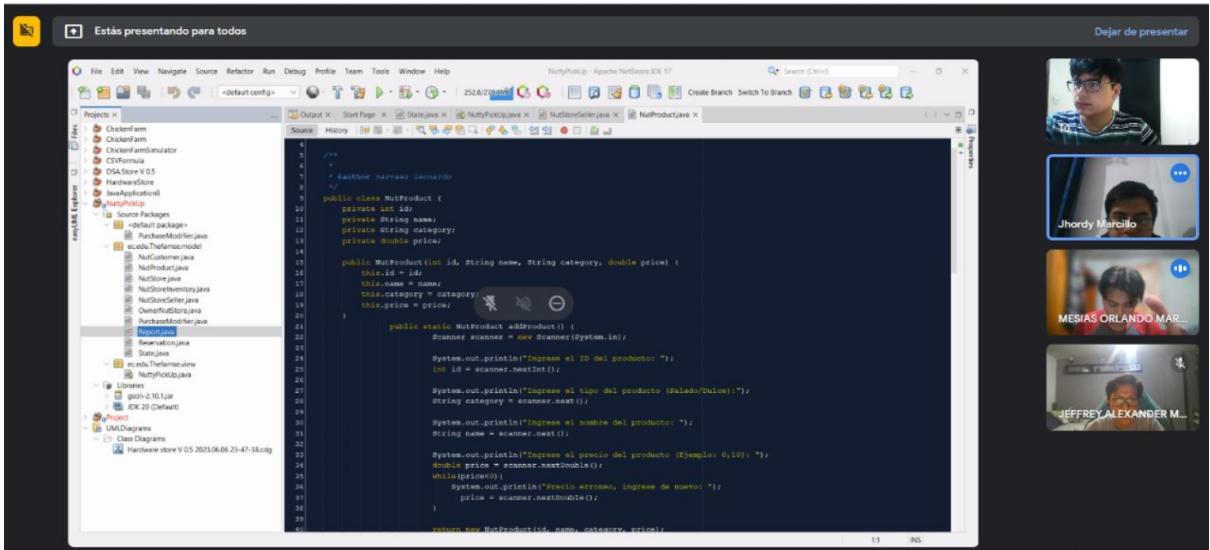
Menu de opciones:
1. Crear nuevo usuario.
2. Agregar nuevo producto.
3. Comprar producto.
4. Editar compra.
5. Estado de compra.
6. Generar archivo JSON de reportes.
7. Salir.

Ingrese la opcion que desea visualizar:
...

```

8-10.- Clean Code

Classes that do not fulfill any function
Comments in Spanish and unnecessary
Unused floating packages
Functions that do not operate correctly



The screenshot shows the NetBeans IDE interface. On the left, the Project Explorer displays several Java files under the 'src' and 'Libraries' sections. In the center, the code editor shows a Java class named 'NutProduct'. The code contains several comments in Spanish and some unnecessary floating packages. On the right side of the screen, there are four video call windows showing students in a virtual meeting.

```

/*
 * Author: Leonardo
 */
public class NutProduct {
    private int id;
    private String name;
    private String category;
    private double price;
}

public NutProduct(int id, String name, String category, double price) {
    this.id = id;
    this.name = name;
    this.category = category;
    this.price = price;
}

public static NutProduct addProduct() {
    Scanner scanner = new Scanner(System.in);

    System.out.println("Ingrese el ID del producto:");
    int id = scanner.nextInt();

    System.out.println("Ingrese el tipo del producto (Salado/Dulce):");
    String category = scanner.next();

    System.out.println("Ingrese el nombre del producto:");
    String name = scanner.next();

    System.out.println("Ingrese el precio del producto (Ejemplo: 0,10):");
    double price = scanner.nextDouble();
    while(price < 0) {
        System.out.println("Precio erroneo, ingrese de nuevo:");
        price = scanner.nextDouble();
    }

    return new NutProduct(id, name, category, price);
}

```

The screenshot shows the Apache NetBeans IDE interface. On the left, there's a code editor with Java code for a NuttyPickUp application. The code includes imports for Scanner, ArrayList, NutProduct, NutReservation, and NutCustomer. It defines a NutCustomer class and a main menu loop with options for creating a new user, adding a product, deleting a product, editing a product, viewing purchases, generating a JSON report, and exiting. On the right, there are four video feeds of team members: JEFFREY ALEXANDER M..., MESIAS ORLANDO MAR..., and Jhordy Marcillo, along with a fourth feed that is mostly blacked out.

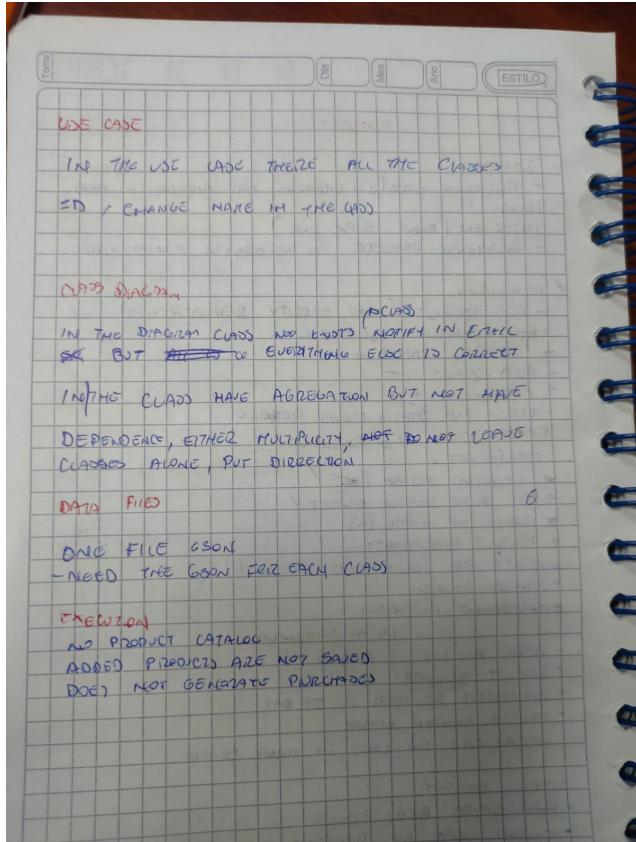
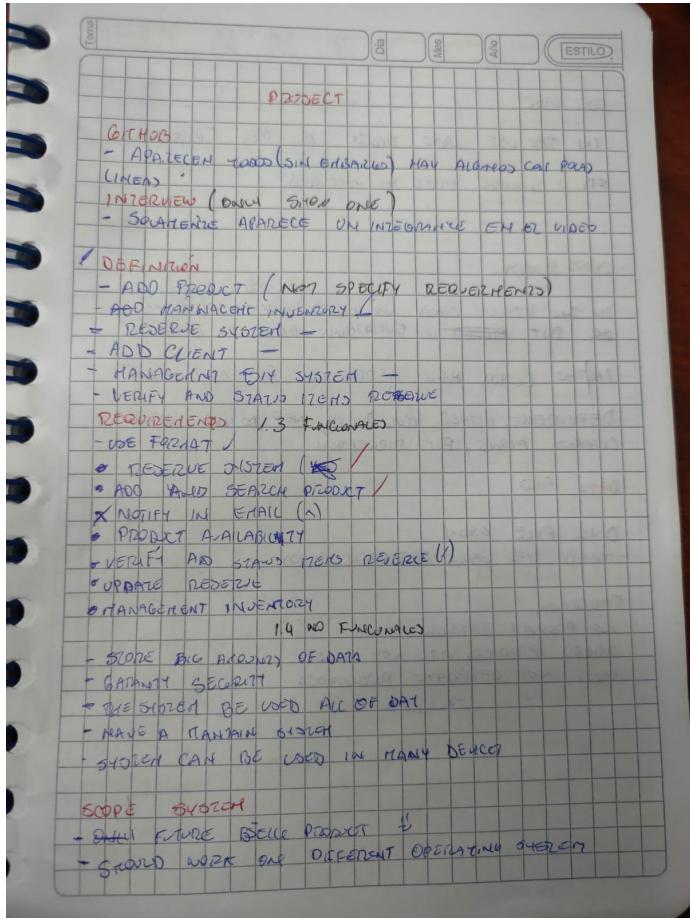
```

    /**
 * author THE FANGE.
 */
import java.util.ArrayList;
import java.util.Scanner;
import java.util.List;
import java.util.Random;
import com.google.gson.Gson;
import com.google.gson.GsonBuilder;
public class NutCustomer {
    Scanner scanner = new Scanner(System.in);
    int option;
    NutCustomer customer = null;
    Reservation reservation = null;
    List<NutCustomer> customers = new ArrayList<>();
    while (true) {
        System.out.println("Menú de opciones:");
        System.out.println("1. Crear nuevo usuario.");
        System.out.println("2. Agregar nuevo producto.");
        System.out.println("3. Eliminar producto.");
        System.out.println("4. Editar producto.");
        System.out.println("5. Estado de compra.");
        System.out.println("6. Generar archivo JSON de reportes.");
        System.out.println("7. Salir.");
        try {
            System.out.print("Ingrese la opción que desea visualizar: ");
            option = scanner.nextInt();
        } catch (Exception e) {
            scanner.nextLine();
            System.out.println("--Usted ha ingresado en creación de usuario--");
            NutCustomer customer = NutCustomer.createCustomer();
        }
        switch (option) {
            case 1:
                System.out.println("--Usted ha ingresado en creación de usuario--");
                NutCustomer customer = NutCustomer.createCustomer();
                break;
            case 2:
                System.out.println("Ingrese la opción que desea visualizar:");
                break;
            case 3:
                System.out.println("Ingrese la opción que desea visualizar:");
                break;
            case 4:
                System.out.println("Ingrese la opción que desea visualizar:");
                break;
            case 5:
                System.out.println("Ingrese la opción que desea visualizar:");
                break;
            case 6:
                System.out.println("Ingrese la opción que desea visualizar:");
                break;
            case 7:
                System.out.println("Saliendo...");
                break;
            default:
                System.out.println("Opción incorrecta, ingrese de nuevo:");
                System.out.println("1. Crear nuevo usuario.");
                System.out.println("2. Agregar nuevo producto.");
                System.out.println("3. Eliminar producto.");
                System.out.println("4. Editar producto.");
                System.out.println("5. Estado de compra.");
                System.out.println("6. Generar archivo JSON de reportes.");
                System.out.println("7. Salir.");
                System.out.print("Ingrese la opción que desea visualizar:");
                break;
        }
    }
}

```

This screenshot shows the Apache NetBeans IDE with the project structure visible on the left. The project 'NuttyPickUp' contains several source packages and files, including PurchaseModel.java, PurchaseModel.modd, State.java, NutCustomer.java, NutStore.java, NutCustomerInventory.java, NutCustomerOrder.java, NutCustomerOrderItem.java, PurchaseModel.java, PurchaseModel.modd, Reservation.java, State.java, NutCustomer.java, and NutCustomerOrderItem.java. The right side of the screen shows the same four video feeds as the previous screenshot.

Notes



Team 6:Code Warriors

Project's name: S&O Hardware Store

Leader: Edison Damian Verdesoto Segovia

GitHub: https://github.com/EDVerdesoto/T06_SandO-System.git

Inspector: Team 1 , Yeshua Chiliquinga

20. JOHN MATEO QUISHPE LLUMIQUINGA

21. LEONEL ALEJANDRO TIPAN QUINTO

22. EDISON DAMIAN VERDESOTO SEGOVIA

Presentation: 8.5/10

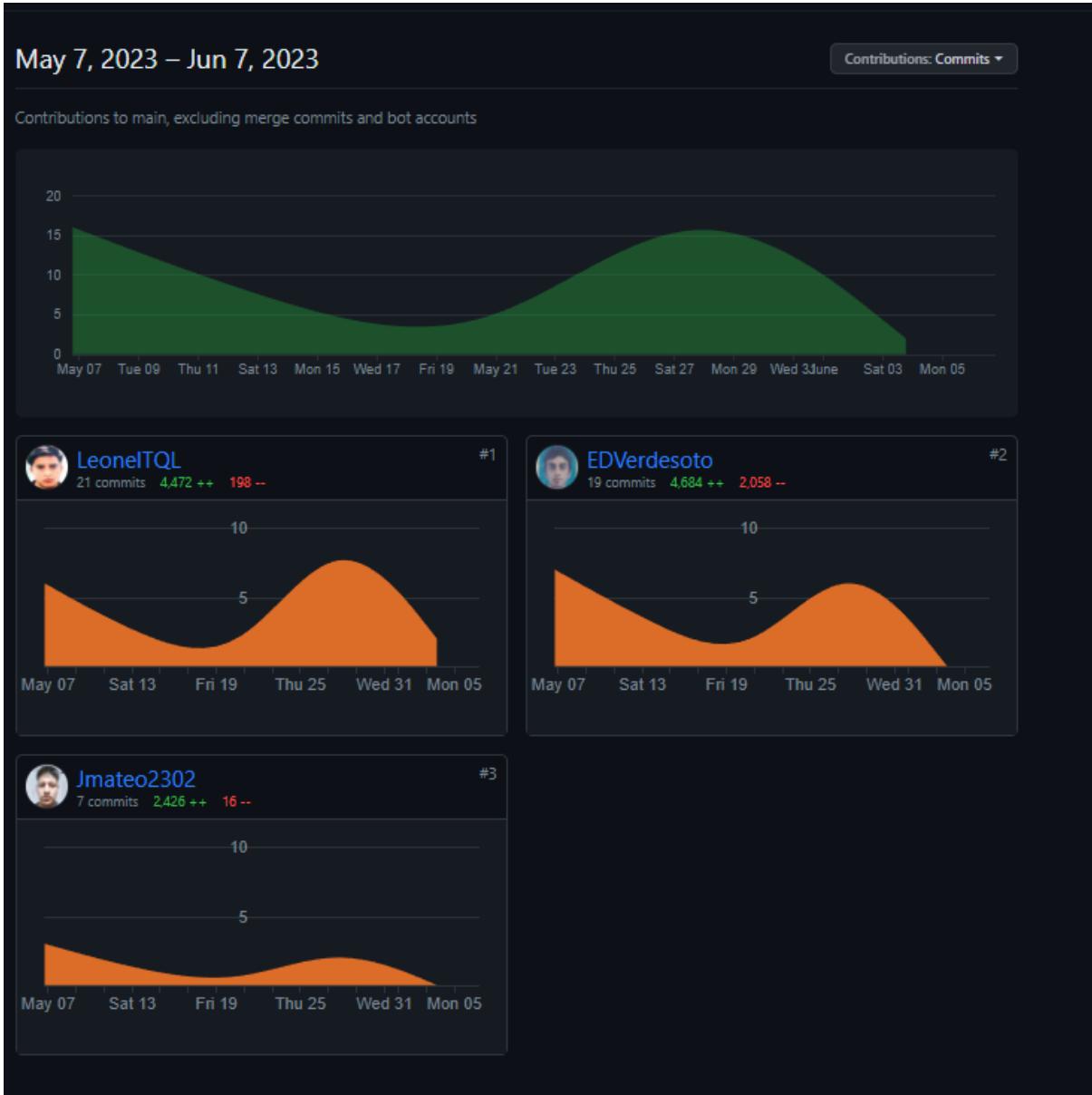
Rubric

Presentation

0.- GitHub	/100
1.- Interview and Presentation	10/10 (All the members appear in the video)
2.- Project definition	9/10 (In the presentation the team said that the program should save data in csv/excel files)
3.- Software Requirements	7.5/10 (The requirements do not follow the format of numbering IEEE-830 in the functional requirements and there are no images of the Class Diagram nor the Use Case Diagram)
4.- Use Case	10/10(Everything is following the specifications)
5.- Class Diagram	10/10(The diagram is according with the Requirements and the use case)
6.- Data files	1/10(The program does not generate the json file)
7.- Execution	9/10(The program executes very well but it has one bug detected)
8-10.- Clean Code	24/30 (The class name “ToSellItem” is incorrect, because it is a verb and there is no code [it is empty]; after reviewing the requirements this class wasn’t found, also there are some attributes names that are verbs like enterDate, request, sellPercentage, and a method that is not a verb “supplierContacts”. Finally there is a Scanner sc that is not used.)
TOTAL	80.5/100

Evidence

GitHub



1.- Interview and Presentation

https://www.youtube.com/watch?v=4yvGGJaW0A4&t=4s&ab_channel=DamianVerdesoto



2.- Project definition

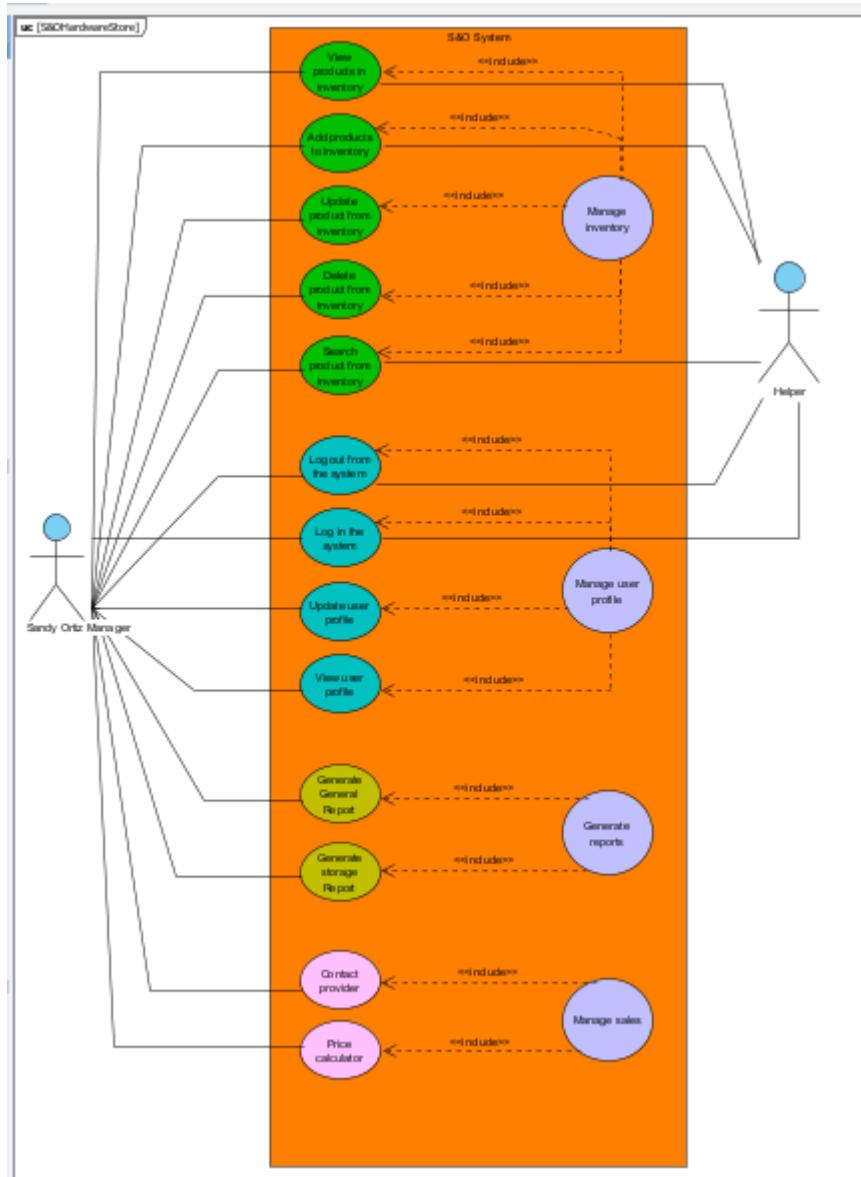
- a. Who will be the main user?
 - i. Just me, to owner.
- b. What are your expectations of use of the software?
 - i. To be easy, practical and intuitive, it must also give me accurate data and make it easy to search for products
- c. What type of information will the software get?
 - i. Items.
 - ii. Product price.
 - iii. VAT on products 30% and for faucets 20%
 - iv. What kind of application would you like to have?
 - v. Application for desktop PC with an eighth generation i3 processor, a memory of 4 GB of RAM with Windows 11 of 64 beats. What type of documentation would you like the software to give?
 - vi. The report of the product in stock, the taxes and the sale price. It should give me the option to download and print the report.
- d. What interfaces do you need?
 - i. A tab that allows me to view all the inventory with the supplier prices, their contacts, amount of the product and sale price.

3.- Software Requirements

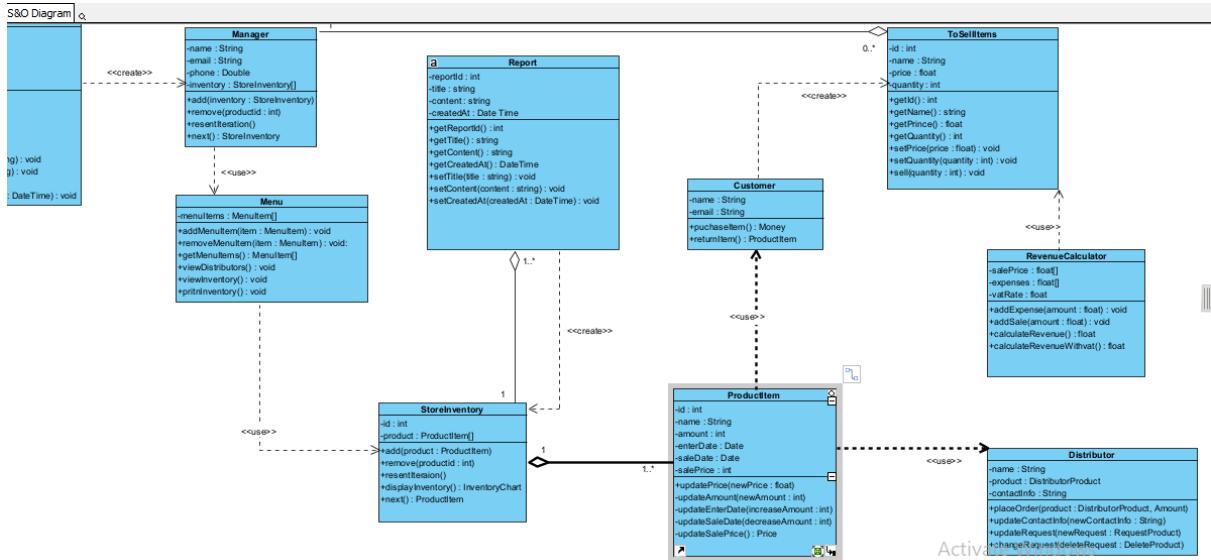
Table of Contents

Table of Contents.....	ii
Revision History	iii
1. Introduction	1
1.1 Purpose	1
1.2 Product Scope.....	1
1.3 Definitions, accronyms, abbreviations.....	1
1.4 References	1
1.5 Overview.....	1
2. General Description	1
2.1 Product Perspective	1
2.2 Product Functions	2
2.3 User Classes and Characteristics	2
2.4 Operating Enviroment	2
2.5 Desing and Implementation Constraints	2
2.6 User documentation.....	2
2.7 Assumptions and dependencies	3
3. Specific Requirements	4
3.1 Functional Requirements	4
3.2 Functions.....	4
3.3 Performance Requirements.....	5
3.4 Design Contrains	5
3.5 System Atribbutes	5
3.6 Other Requirements	5

4.- Use Case



5.- Class Diagram



6.- Data files

 build	6/7/2023 8:43 AM	File folder
 nbproject	6/7/2023 8:41 AM	File folder
 src	6/7/2023 7:27 AM	File folder
 build.xml	6/7/2023 7:27 AM	xmlfile 4 KB
 manifest.mf	6/7/2023 7:27 AM	MF File 1 KB

7.- Execution

```

Nombre de usuario: username
Contraseña: password
Inicio de sesión exitoso. ✨Bienvenido!
----- Menú -----
1. Crear usuarios
2. Ingresar productos
3. Vender productos
4. Ver reporte de ventas
5. Contactos de proveedores
6. Salir
Ingrese la opción: 1
Se han creado los usuarios
----- Menú -----
1. Crear usuarios
2. Ingresar productos
3. Vender productos
4. Ver reporte de ventas
5. Contactos de proveedores
6. Salir
Ingrese la opción: 1
Se han creado los usuarios
----- Menú -----
1. Crear usuarios
2. Ingresar productos
3. Vender productos
4. Ver reporte de ventas
5. Contactos de proveedores
6. Salir
Ingrese la opción:

```

8-10.- Clean Code

```
public class InputMenu {
    public static void menu() {
        try (Scanner scanner = new Scanner(source: System.in)) {
            int option;

            do {
                System.out.println("----- Menú -----");
                System.out.println("1. Crear usuarios");
                System.out.println("2. Ingresar productos");
                System.out.println("3. Vender productos");
                System.out.println("4. Ver reporte de ventas");
                System.out.println("5. Contactos de proveedores");
                System.out.println("6. Salir");
                System.out.print("Ingrese la opción: ");
                option = scanner.nextInt();

                switch (option) {
                    case 1 -> createUsers();
                    case 2 -> enterProducts();
                    case 3 -> sellProducts();
                    case 4 -> viewSalesReport();
                    case 5 -> supplierContacts();
                    case 6 -> System.out.println("Saliendo del programa...");
                    default -> System.out.println("Opción inválida");
                }
            } while (option != 6);
        }
    }

    public static void createUsers() {
        System.out.println("Se han creado los usuarios");
    }
}
```

```
    /**
     *
     * @author Code Warriors, DCCO-ESPE
     */
    public class Costumer {

        private String name;
        private String email;
        Scanner sc = new Scanner(source: System.in);

        public void purchaseItem () {
            System.out.println("What item the client purchase?");
        }

        public void returnItem () {
        }

        /**
         * @return the name
         */
        public String getName() {
            return name;
        }

        /**
         * @param name the name to set
         */
        public void setName(String name) {
```

```
package ec.edu.espe.HOManagement.model;

    /**
     *
     * @author Code Warriors, DCCO-ESPE
     */
    public class ToSellItem {
        //TODO ingreso desde teclado, llamar a la calculadora, llamar costumer y eliminar cosas del inventario
    }
```

```
8     public class Distributor {
9         private String name;
10        private String product;
11        private String contactInfo;
12        private String request;
13        Scanner sc = new Scanner(source: System.in);
14    }
```

```
public static Map<String, Object> item() {
    int id;
    String name;
    int amount;
    Date enterDate;
    enterDate = new Date();
    Scanner sc = new Scanner(source: System.in);
    RevenueCalculator revenueCalculator = new RevenueCalculator();
```

```

28     public float calculateRevenue() {
29         System.out.println("Ingrese el precio del Item: ");
30         expenses = sc.nextFloat();
31         float sellPercentage = expenses * revenue;
32         salePrice = expenses + sellPercentage;
33         System.out.println("Este es el precio de venta, obteniendo su ganancia: " + salePrice);
34         return salePrice;
35     }
36
37     public static void supplierContacts() {
38         Distributor.menu();
39     }
40
41 }

```

You're presenting to everyone Stop presenting

Screenshot of Apache NetBeans IDE 17.0 showing code editor with Java files and a video conference interface.

Code Editor:

```

public float calculateRevenue() {
    System.out.println("Ingrese el precio del Item: ");
    expenses = sc.nextFloat();
    float sellPercentage = expenses * revenue;
    salePrice = expenses + sellPercentage;
    System.out.println("Este es el precio de venta, obteniendo su ganancia: " + salePrice);
    return salePrice;
}

public static void supplierContacts() {
    Distributor.menu();
}

}

```

IDE Interface:

- File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help
- S:OManagement/0.3.0 - Apache NetBeans IDE 17
- Search (Ctrl+F)
- Projects x Files Services
- src Manager.java ReportJava Customer.java Distributor.java ProductItem.java Registration.java RevenueCalculator.java
- Source History > - < > < > < > < >
- ProductItem.java (selected)
- 16 public ProductItem(int id) {
17 ...
18 }
19 public static Map<String, Object> item() {
20 int id;
21 String name;
22 int amount;
23 Date enterDate;
24 enterDate = new Date();
25 Scanner sc = new Scanner(System.in);
26 RevenueCalculator revenueCalculator = new RevenueCalculator();
27
28 System.out.println("Ingrese el id del producto: ");
29 id = sc.nextInt();
30
31 System.out.println("Ingrese el nombre del item: ");
32 name = sc.nextLine();
33
34 System.out.println("Ingrese la cantidad de items: ");
35 amount = sc.nextInt();
36
37 }
38
39 public void updateItem() {
40 ...
41 }
42
43 public void deleteItem() {
44 ...
45 }
46
47 public void printReport() {
48 ...
49 }
50
51 public void contactsSupplier() {
52 ...
53 }
54
55 public void menu() {
56 ...
57 }
58
59 public void reportSales() {
60 ...
61 }
62
63 public void calculateRevenue() {
64 ...
65 }
66
67 public static void supplierContacts() {
68 ...
69 }
70
71 }

- Members: ProductItem (int id)
- ProductItem (int id)
- Item : Map<String, Object>
- Libraries: gson-2.8.1.jar, Jdk 20 (Default)
- Item : Navigator
- ProductItem (int id)
- Item : Map<String, Object>
- ecaducepe:OManagement.model.ProductItem > item > enterDate
- Output - S:OManagement/0.3.0/run-single
- Introduzca la opción: 6
- 1. Crear usuarios
- 2. Ingresar productos
- 3. Ver reporte de ventas
- 4. Ver reporte de ventas
- 5. Contactos de proveedores
- 6. Salir
- Ingrese la opción: BUILD STOPPED (total time: 26 seconds)

Stop presenting

Evidence Notes:

Report of Inspection

Team: 1

Leader: Yesica Amador Chilquinga Anaya

Name: Icons

Team Inspected: 6

Team Inspected Name: Code Warriors

0. GitHub

Screenshot of activity in the repository

1. Testimony

Checked the video and if all the team participated there. Result: 10

2. Project Definition

We checked if it is coincident with the requirements.

Result 9, because it was not proved that the information was saved in Excel Sheets, the presentation.

3. Requirements

We reviewed that the document complies with the T-IEC 830 standard of SPS and all of its parts. Result: 7.5, because the requirements do not follow the format of numbering IEC 830 in the functional requirements. Also, there is no class and use case diagrams/images.

4. Use Case

The Use Case diagram has to be coincident with the Functional requirements.

Result: 10, because everything is coincident with the requirements.

5. Class Diagram

The class diagram has to be coincident with the functional requirements.

Result: 10, because everything is coincident with the requirements.

6. Data Files

About Icons, these or this has to be consistent and working well. Result:

GAMA

Because, jsons are not being created

7: Execution

Review that code is working well according to the functional requirements.

Result: a. Because has an error founded

Review if code follows a clean code format according to the good practices we saw during the unit 4.

Results: The class name "ToSellItem" is incorrect and there is no code (it's empty); after reviewing the requirements this class wasn't found; also there are some attributes names that are verbs like enterDate, request, sellPercentage, and a method that is not a verb "SupplierContacts". Finally, there is a Scanner sc that is not used.