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

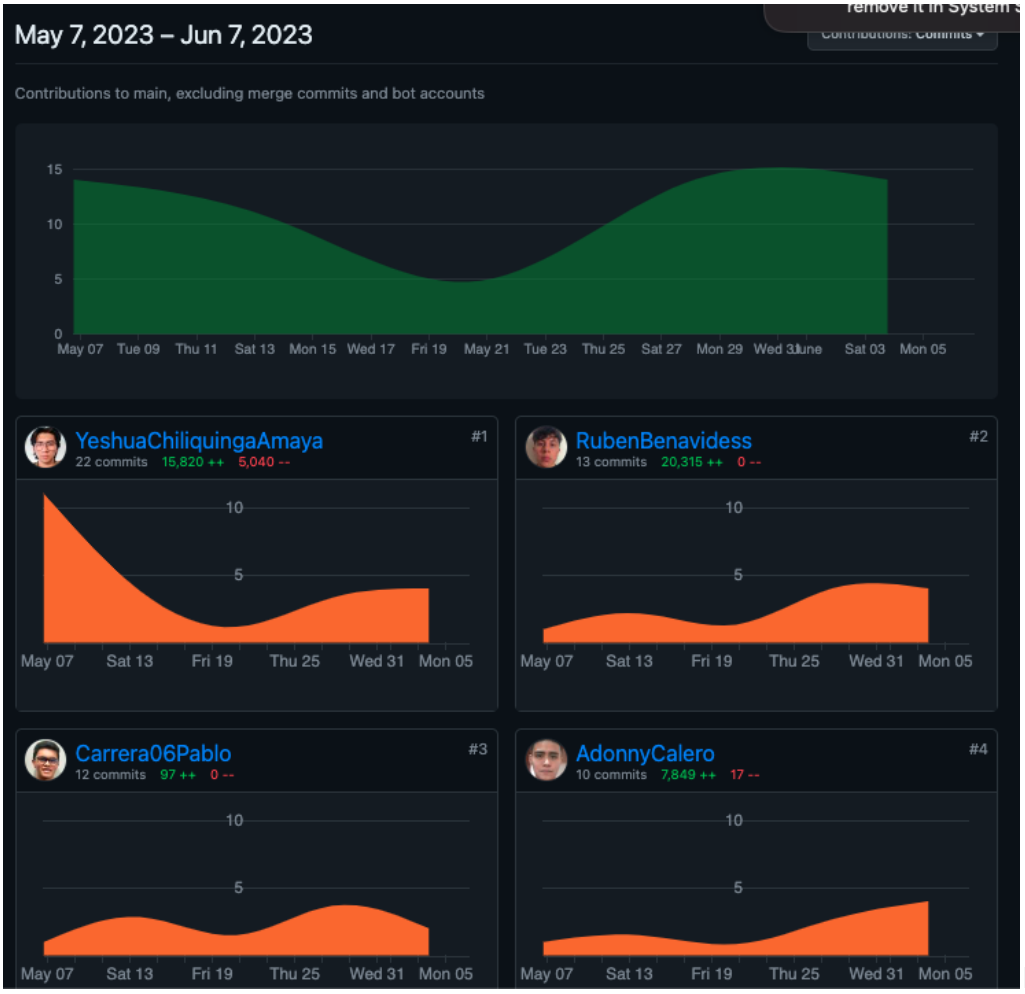
1. RUBEN DARIO BENAVIDES MACIAS
2. ADONNY MATEO CALERO ARGUELLO
3. PABLO ESTEBAN CARRERA QUINDE
4. YESHUA AMADOR CHILIQUINGA AMAYA
   Presentation: 7/10

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

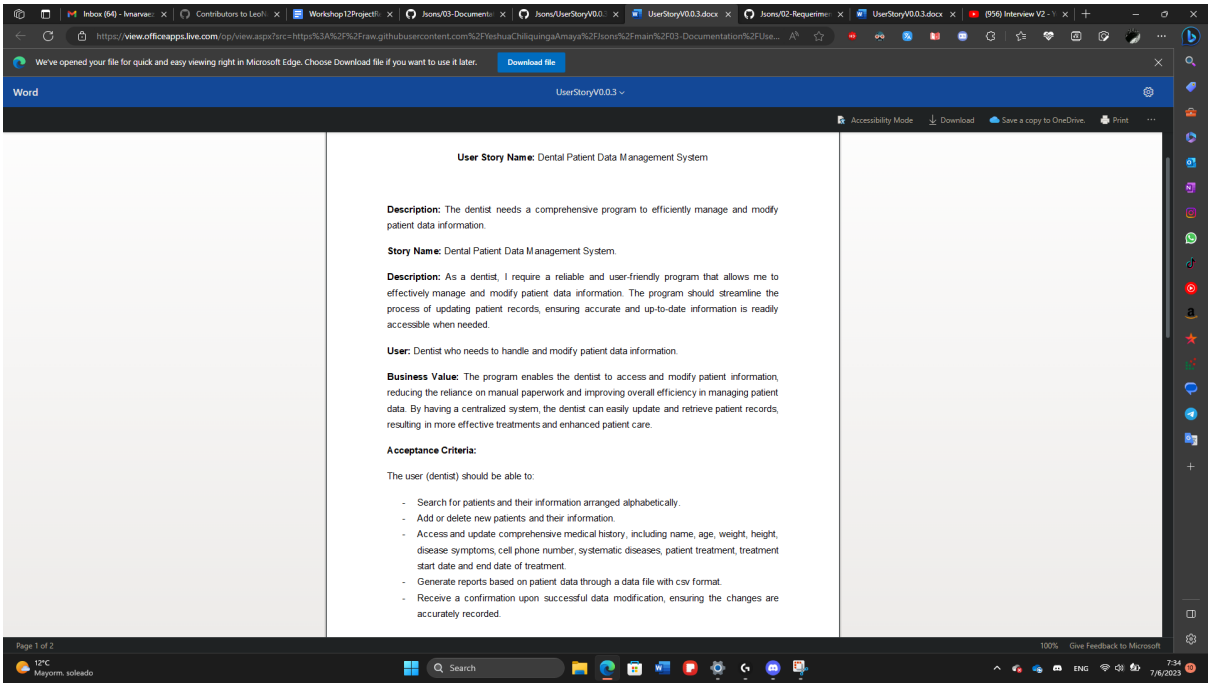


b

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.

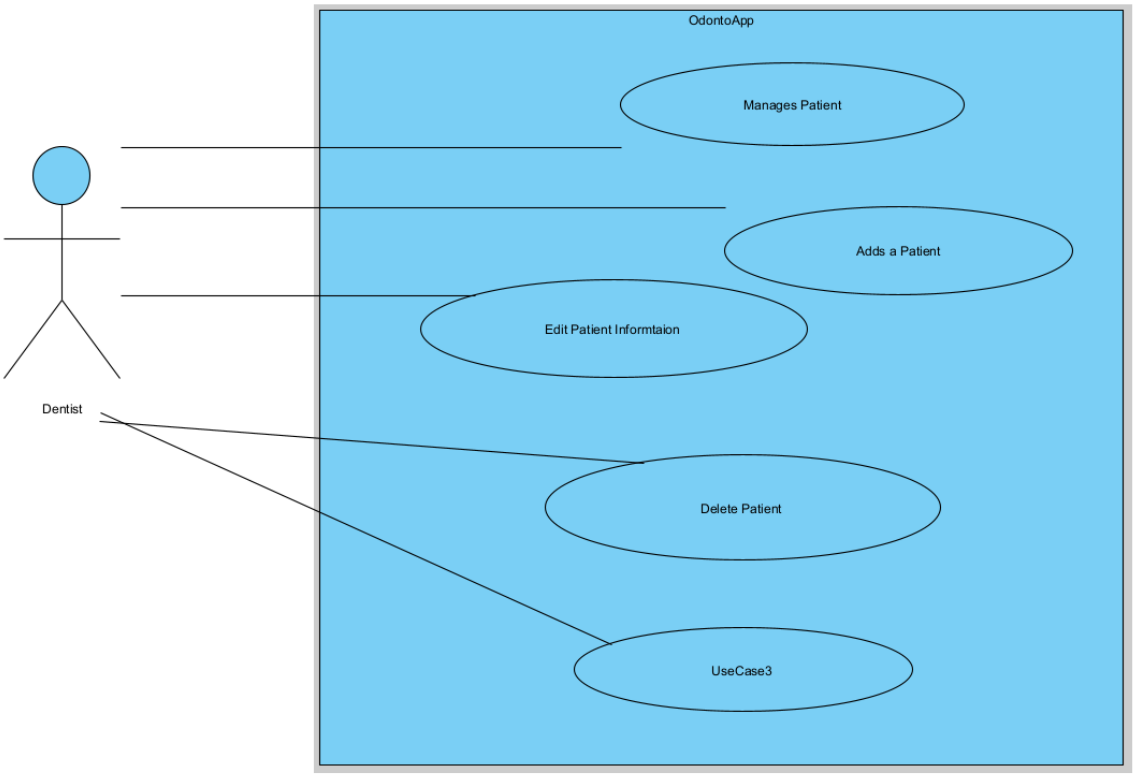
3.- Software Requirements

Requirement Identifier	FR02
Scope in the system	Buy Services Menu
Requirement Definition	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: ----Choose a service(s) or go back---- 1. Profilaxis + Fluorizacion (especialmente para ni&#65533;os): 20 dolares 2. Restauraciones simples: 20 dolares 3. Restauraciones complejas: 25

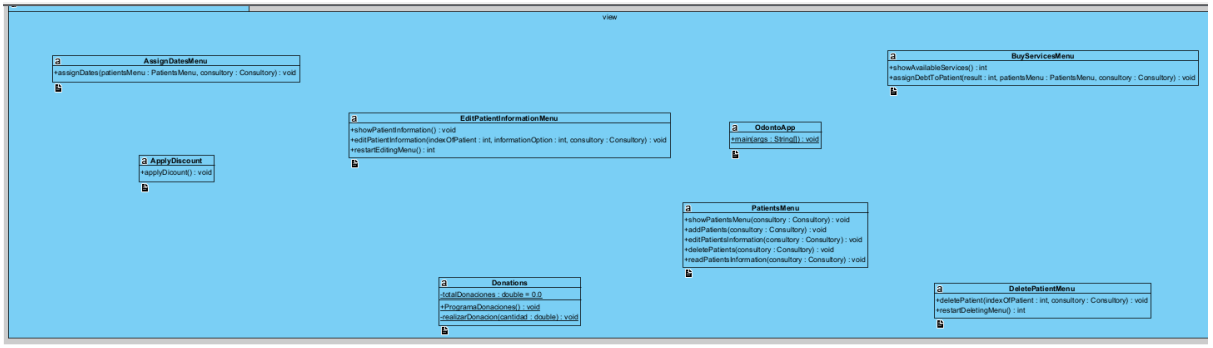
	4. Restauraciones para ni&#65533;os (empastes para ni&#65533;os): 25 dolares 5. Extracciones (extraccion de un diente): 20 dolares 6. Extracciones complicadas: 30 dolares 7. Tratamiento de un conducto (incisivos o dientes frontales): 60 dolares 8. Tratamiento de tres conductos (molares): 100 dolares 9. Go back 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

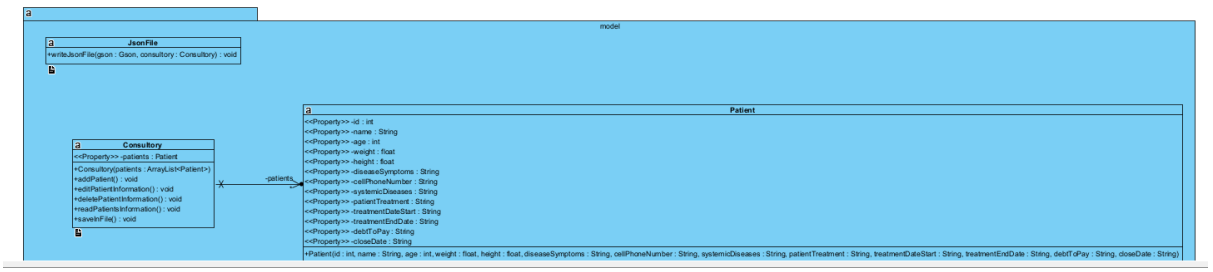
Requirement Identifier	FR02
Scope in the system	Donations Menu
Requirement Definition	When entering the donations option of the principal menu, the system will display another menu with the following structure: ----- Donations Menu ----- 1. Do a donation 2. Total of donations 3. Exit Enter an option: The system will check if there are limitations when entering data and acting respectively if these have been exceeded.

4.- Use Case

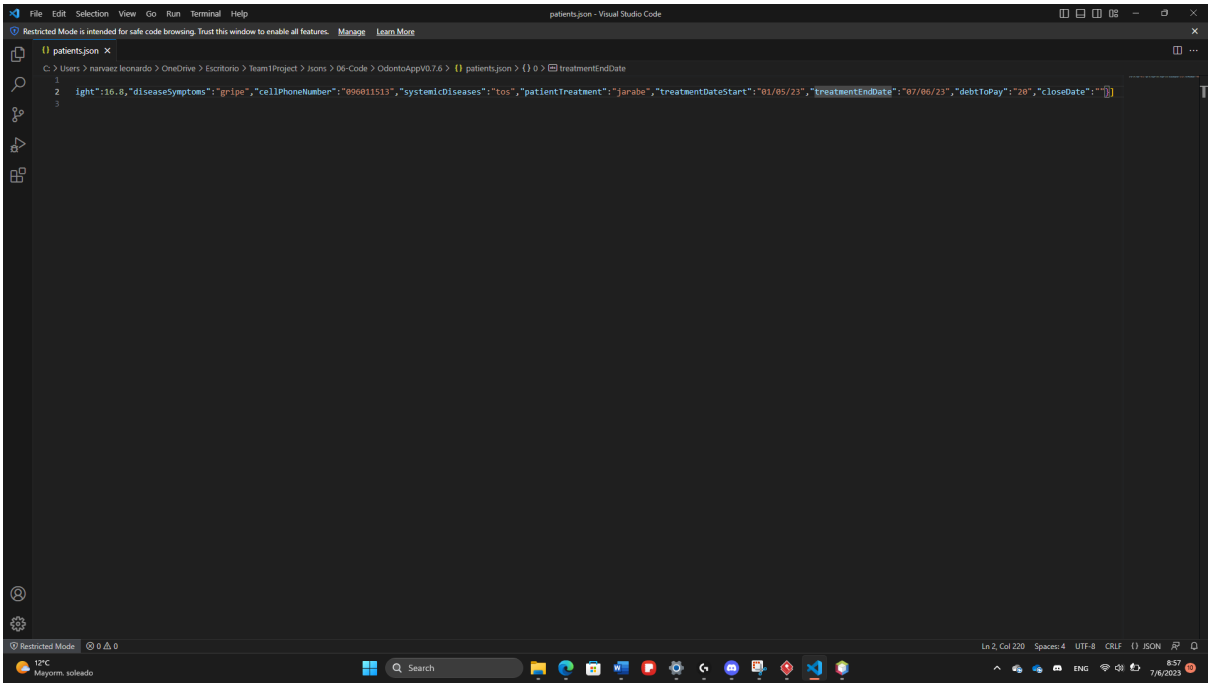


5.- Class Diagram

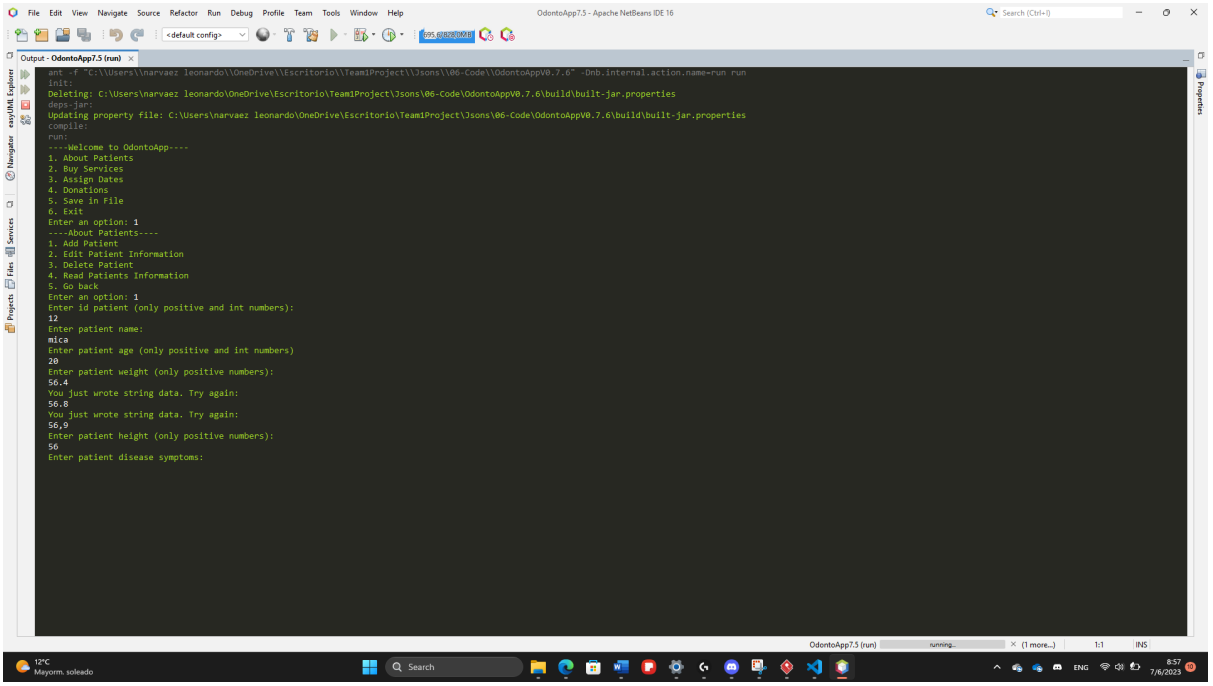




6.- Data files



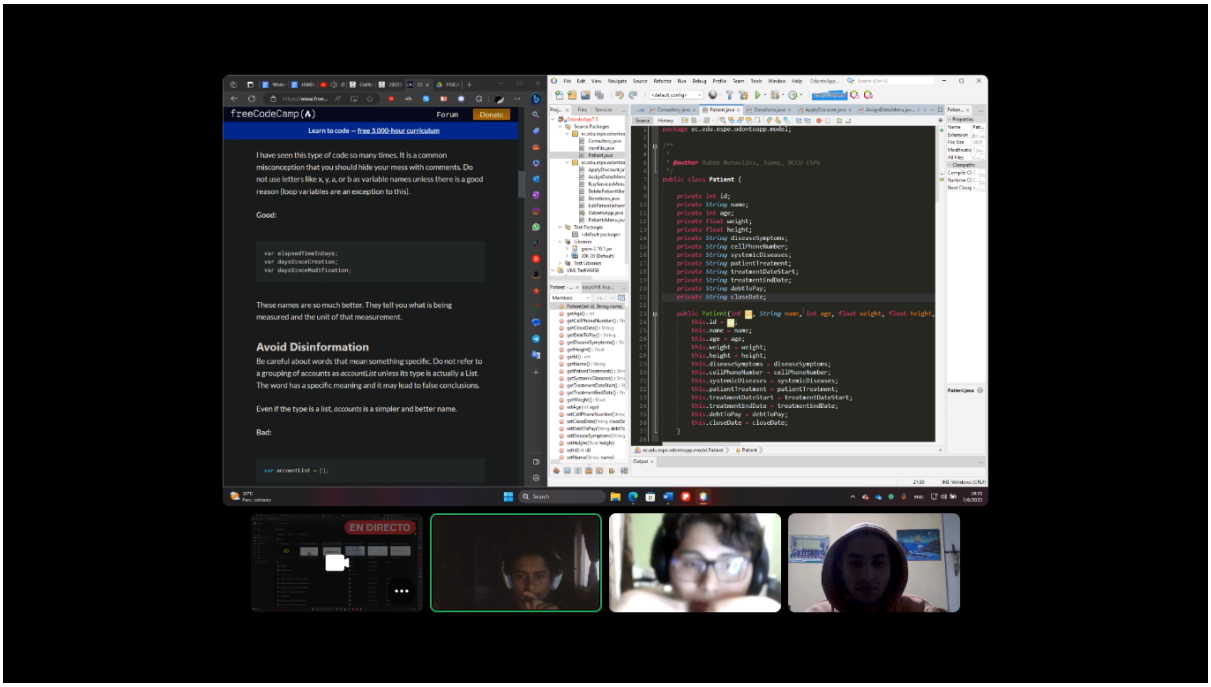
7.- Execution



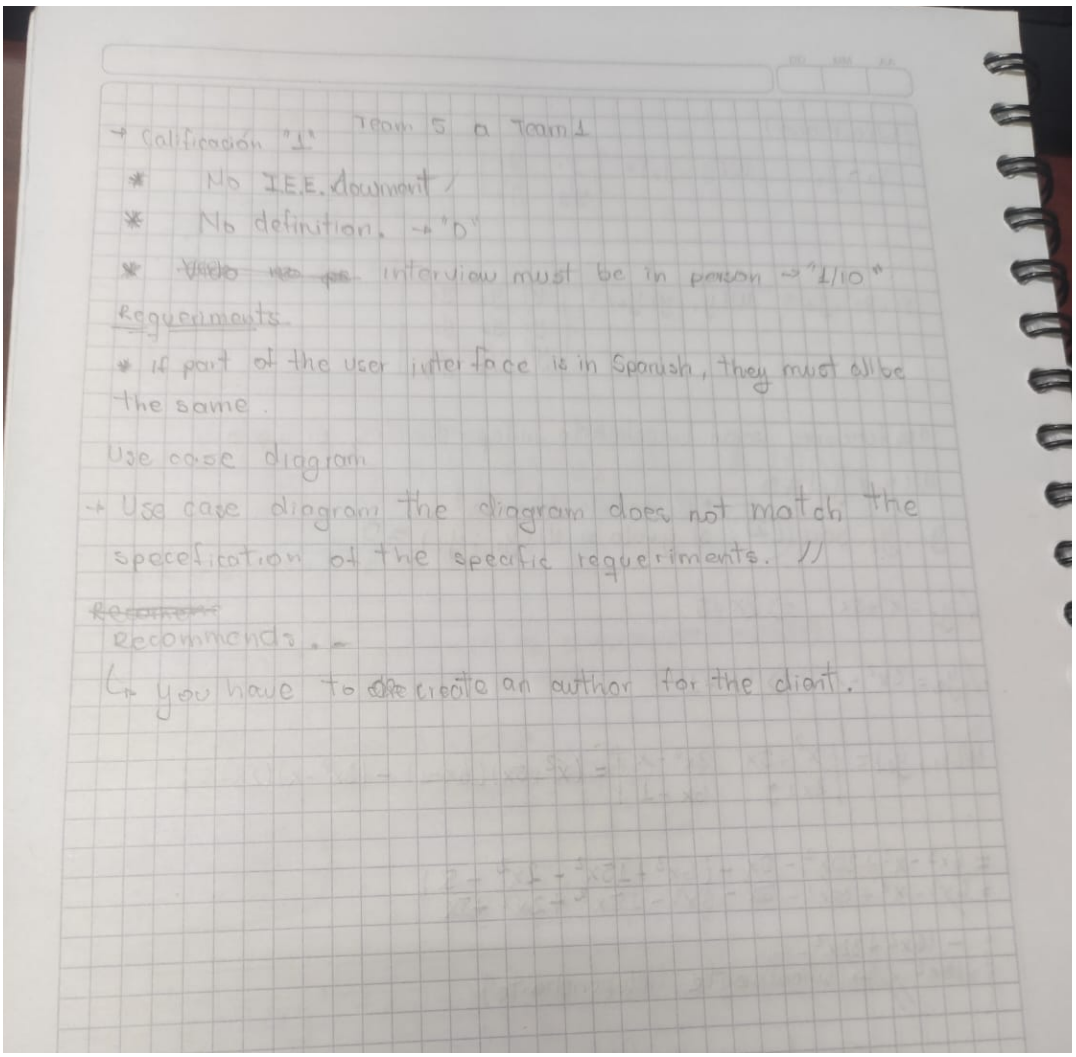
8-10.- Clean Code

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

```java
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();


ivate 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

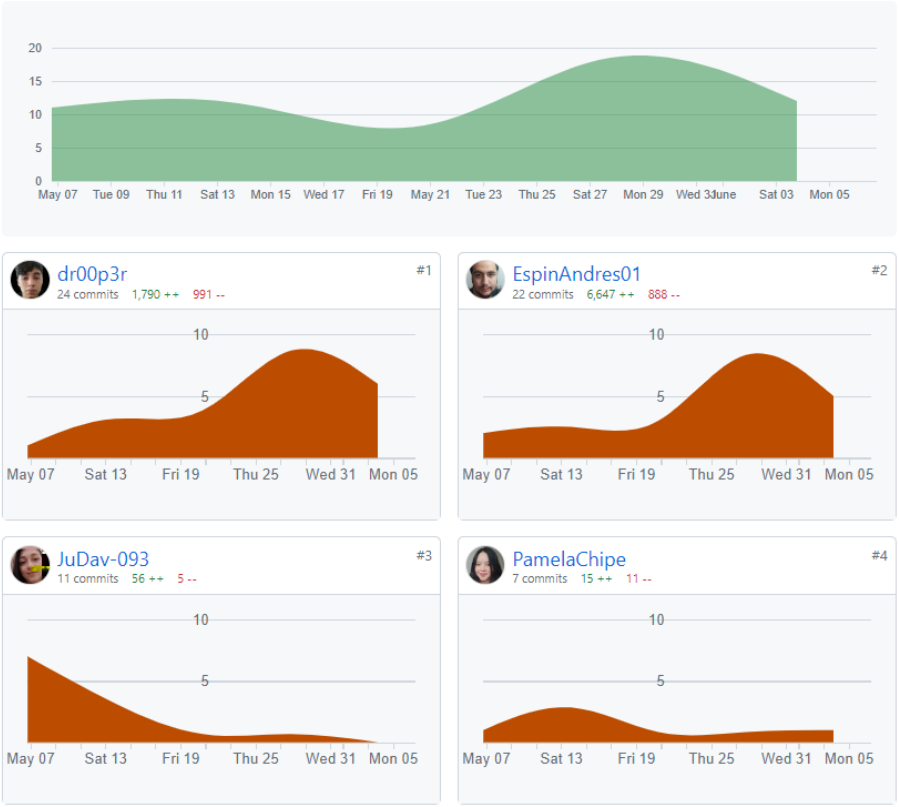
Presentation

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 eco, 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

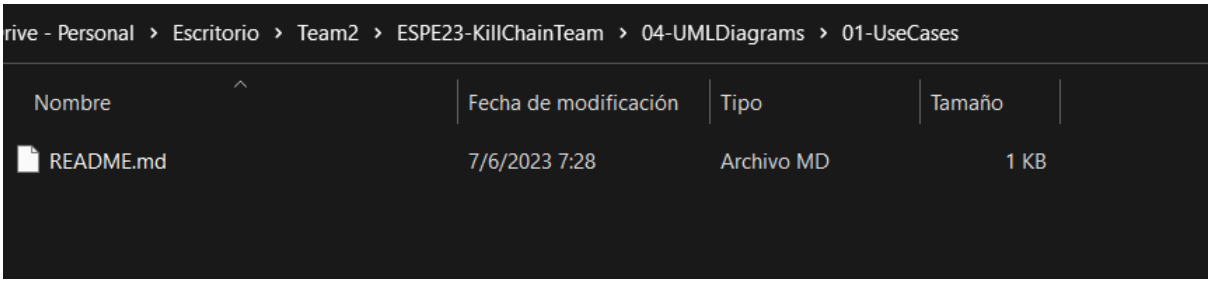
3.1 External interfaces

- 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.

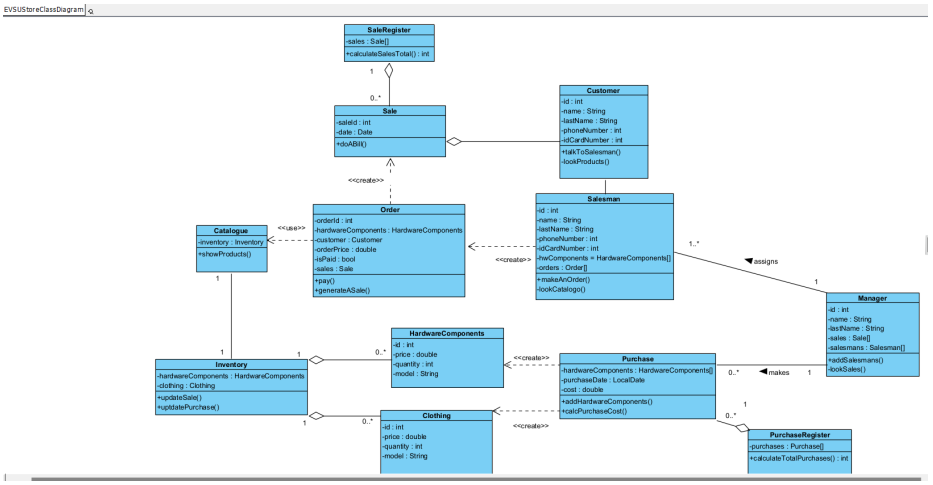
3.2 Functions

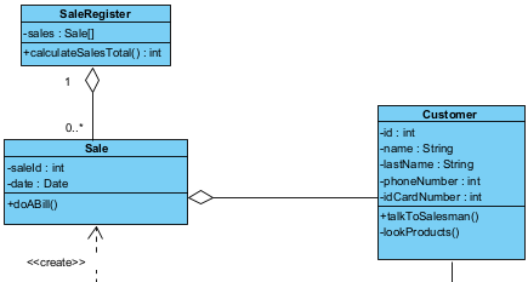
- Inventory management: add, remove and update products.
- Sales Tracking: record sales and generate reports.
- Customer Information Management: Store and update customer information.
- Vendor Recommendation: Suggest vendors based on component compatibility and quality.
- Algorithm to generate compatibility 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

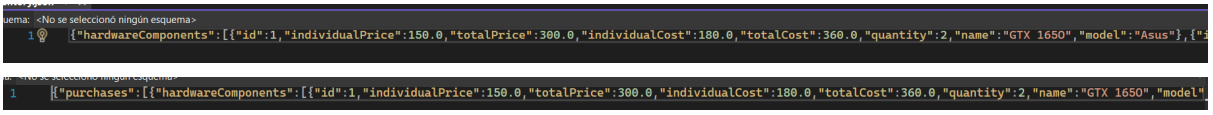


5.- Class Diagram

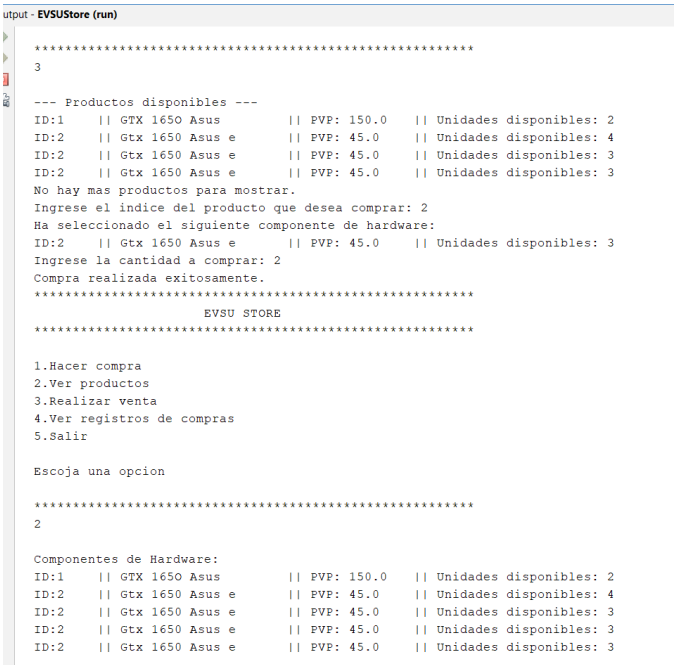




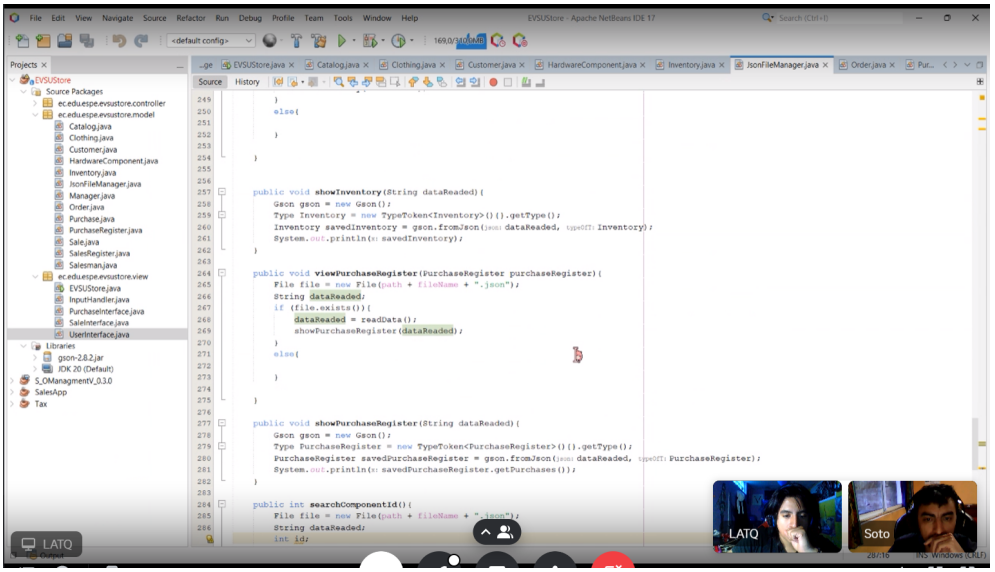
6.- Data files



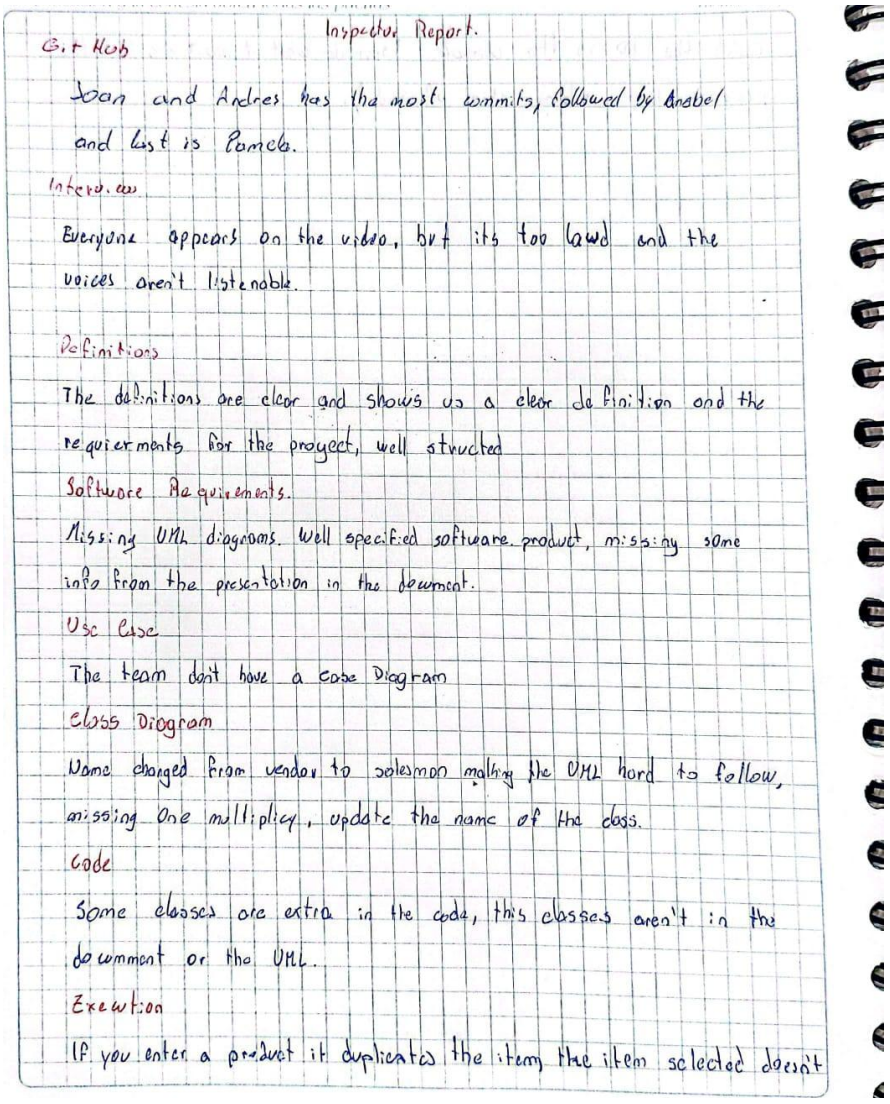
7.- Execution



8-10.- Clean Code



Evidence Notes:



Inspector Report.

Git Hub

Joan and Andres has the most commits, followed by Anabel and last is Pamela.

Interview.

Everyone appears on the video, but its too lawd and the voices aren't listenable.

Definitions

The definitions are clear and shows us a clear definition and the requirements for the project, well structed

Software Requirements.

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

Use Case

The team don't have a case Diagram

Class Diagram

Name changed from vendor to salesman making the UML hard to follow, missing One multiplicy, 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

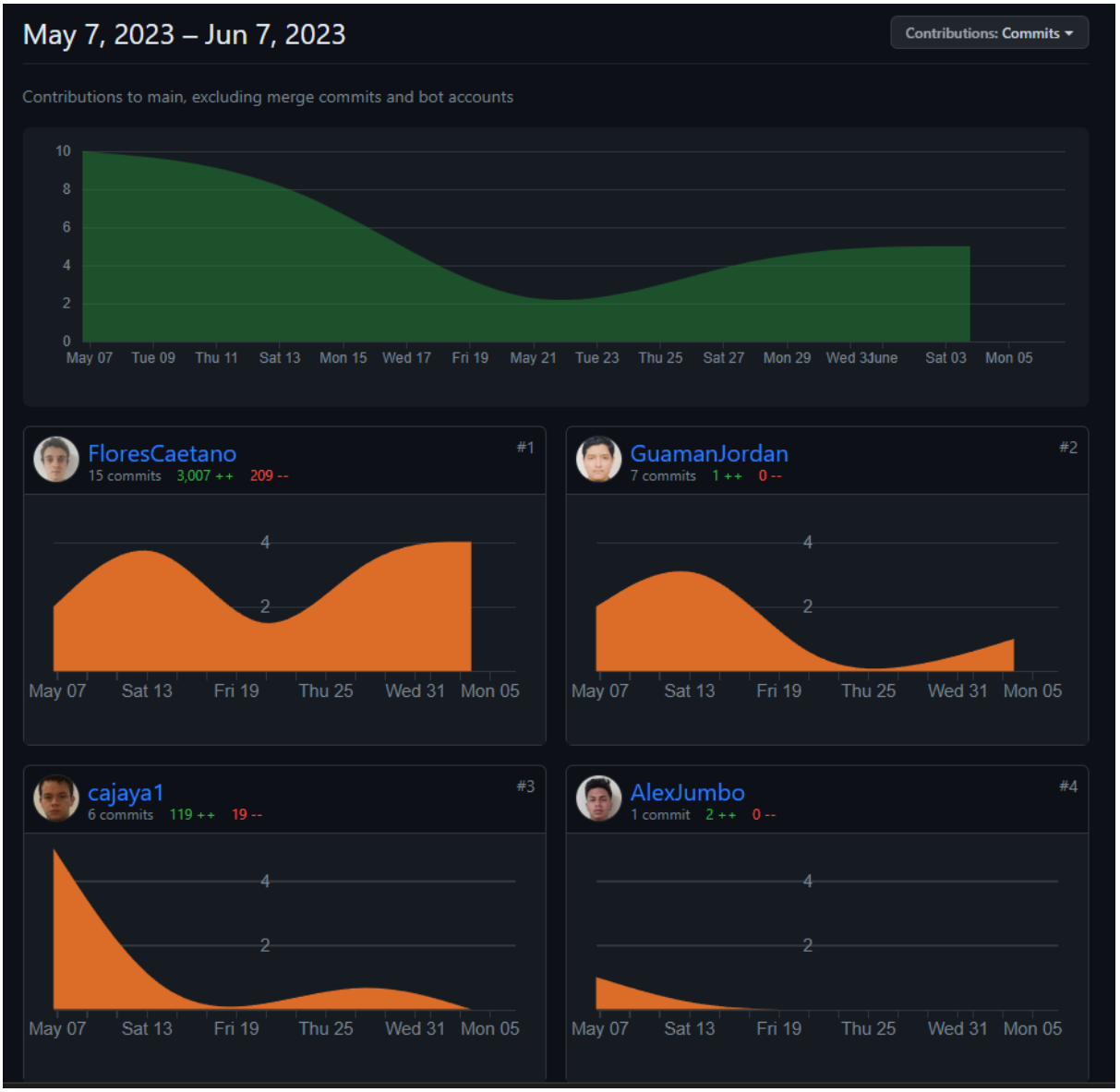
Rubric

Presentation

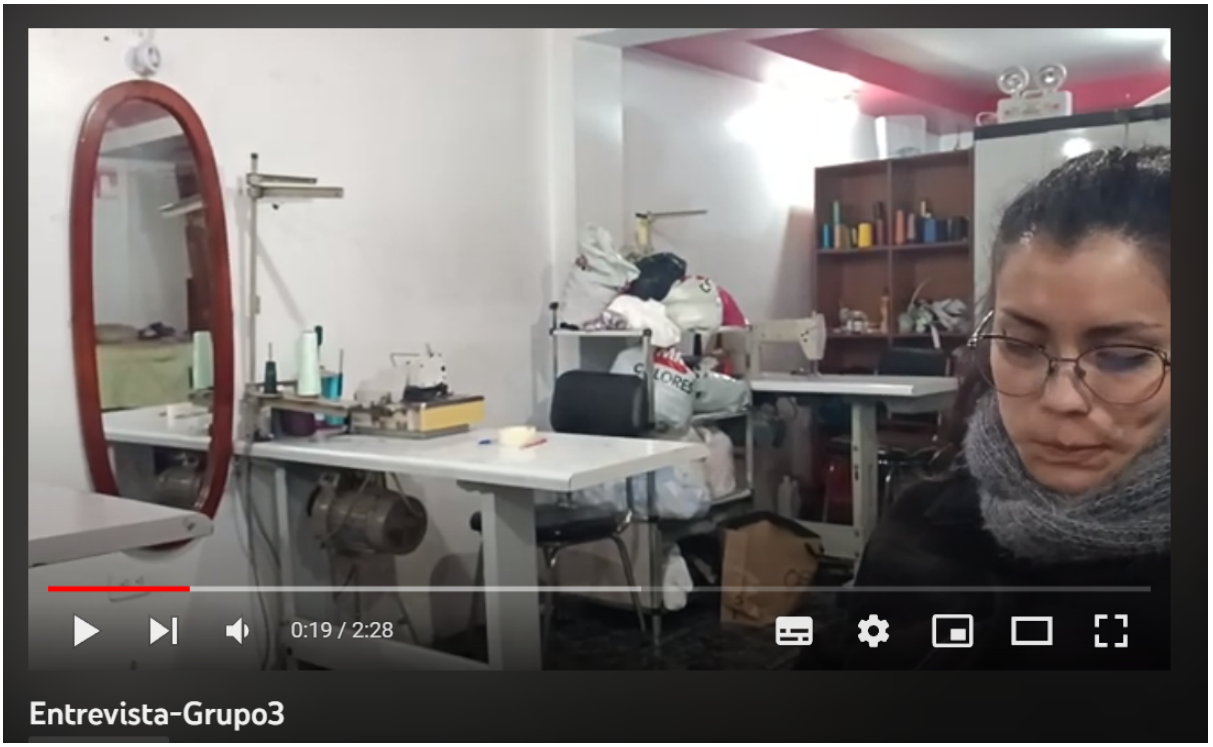
0.- GitHub	/100
1.- Interview and Presentation evidence	4.5/10 no appear the members, Only exists file txt of
2.- Project definition	8.5/10 Missing business rules, just define a CRUD.
3.- Software Requirements declared.	8/10 Missing diagrams, the terms are not defined, just
4.- Use Case action.	9.5/10 The diagram is just missing the generated report
5.- Class Diagram class attributes that don't exist in the diagram	8.5/10 Missing 1 multiplicities, there are methods with
6.- Data files	7/10 Just 1 json file and just has 1 class involved.
7.- Execution spaces in the strings reading.	7/10 Not handling exceptions, not considering the blank
8-10.- Clean Code	25/30
TOTAL	78/100

Evidence

GitHub



1.- Interview and Presentation



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.

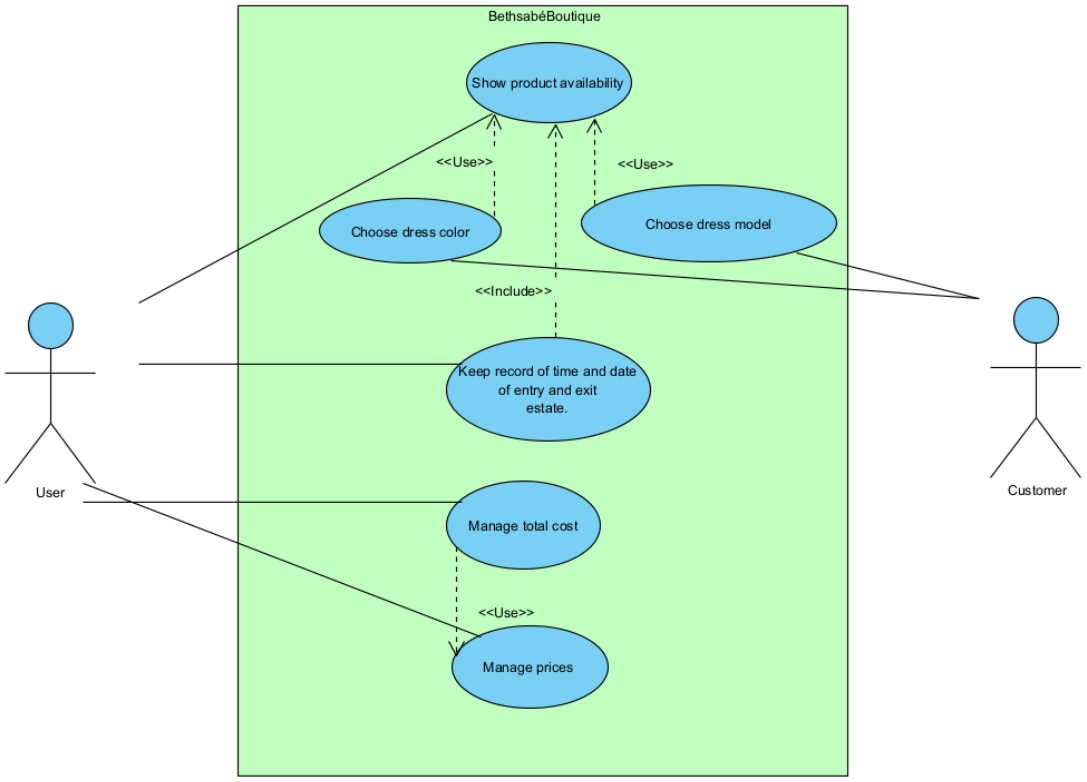
1.1. Purpose

- 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)

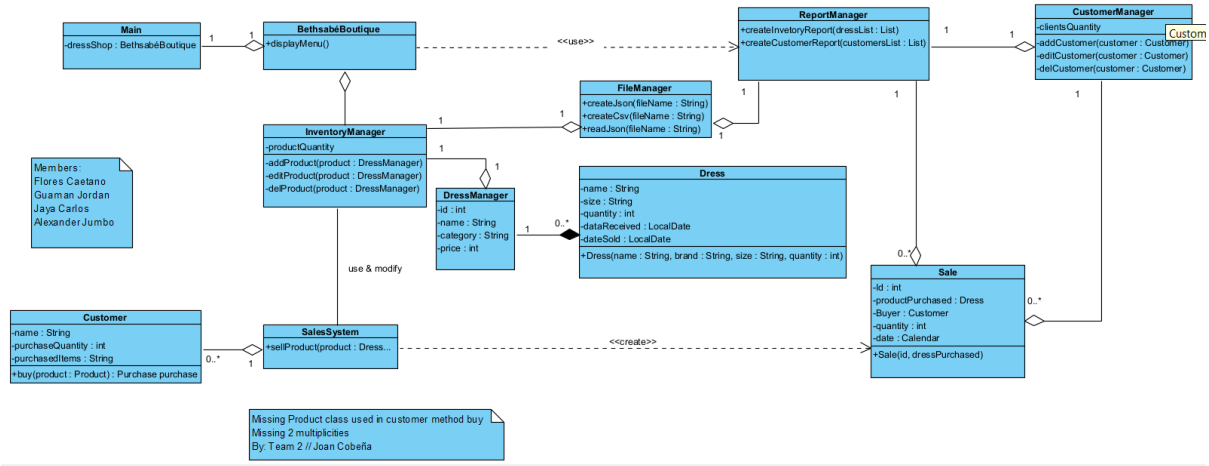
1.2. Scope

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



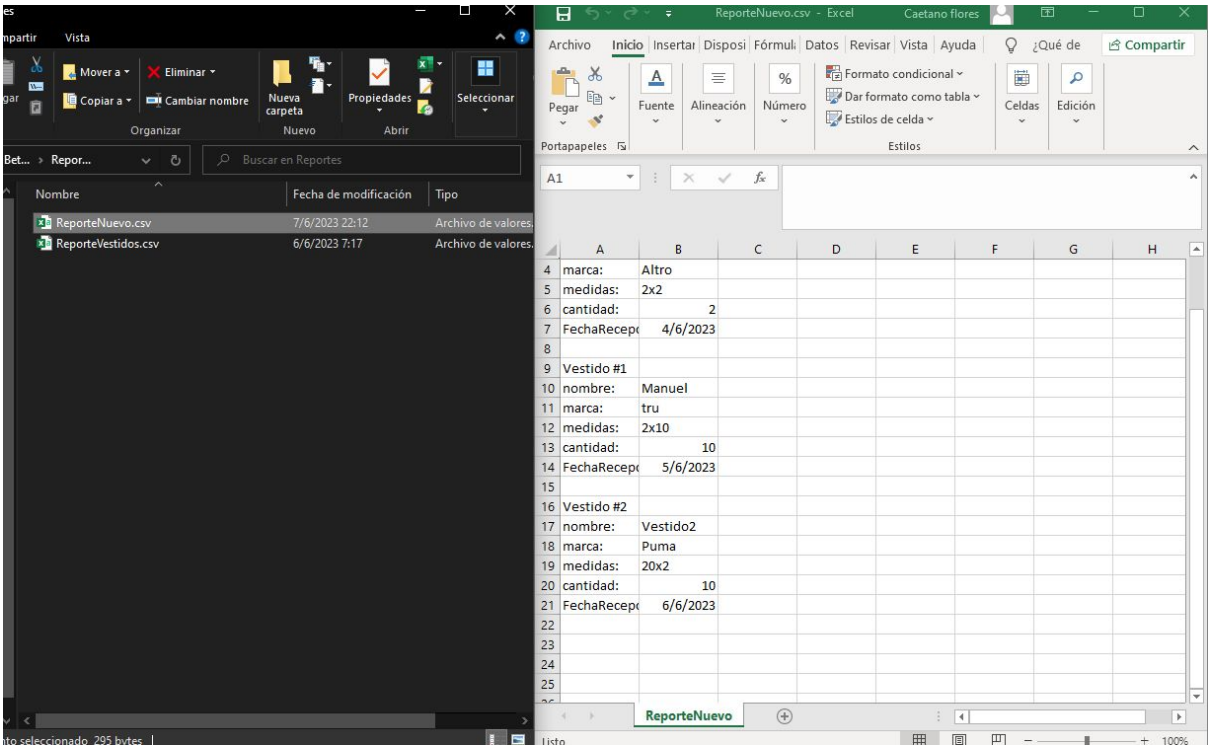
6.- Data files

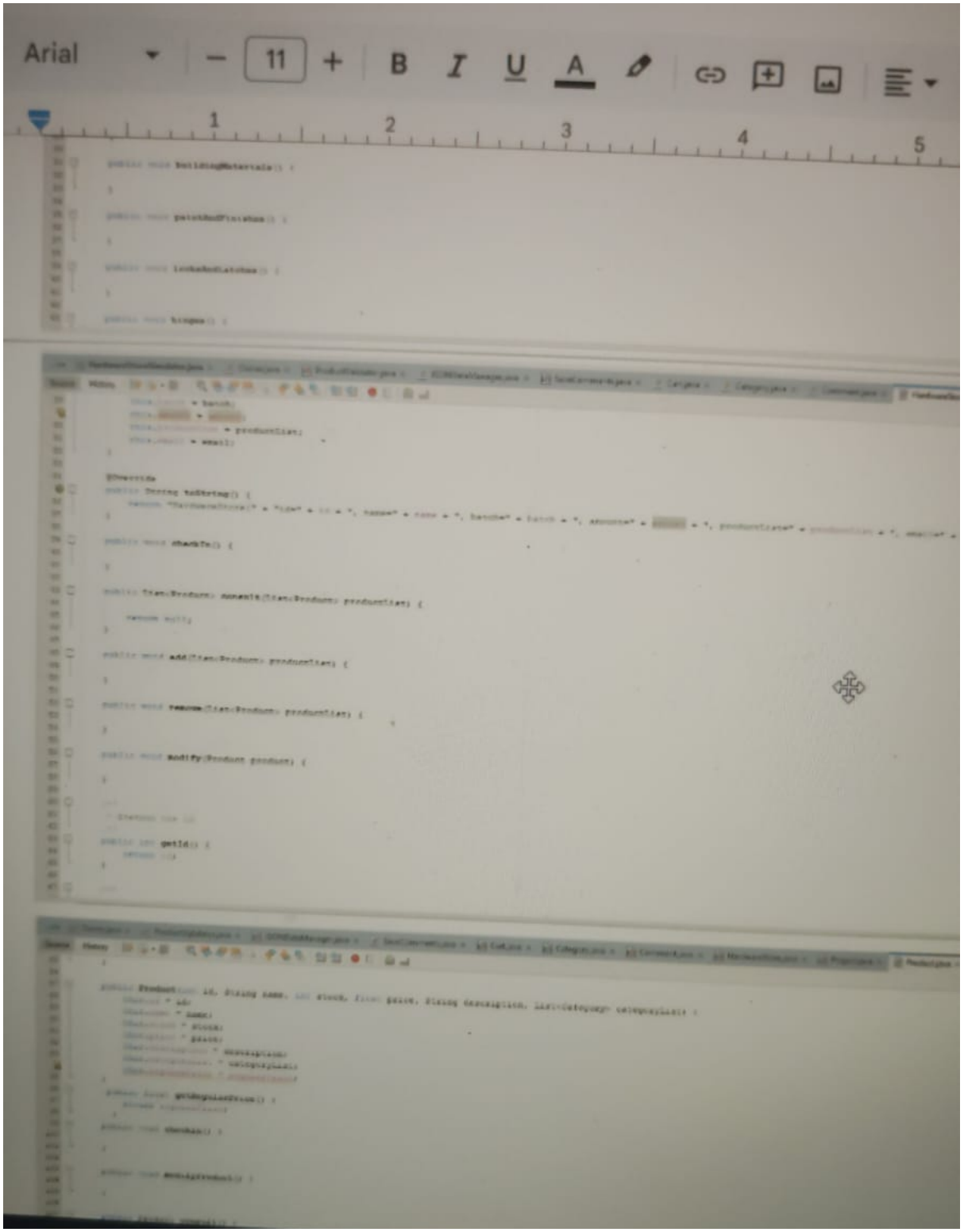
7.- Execution

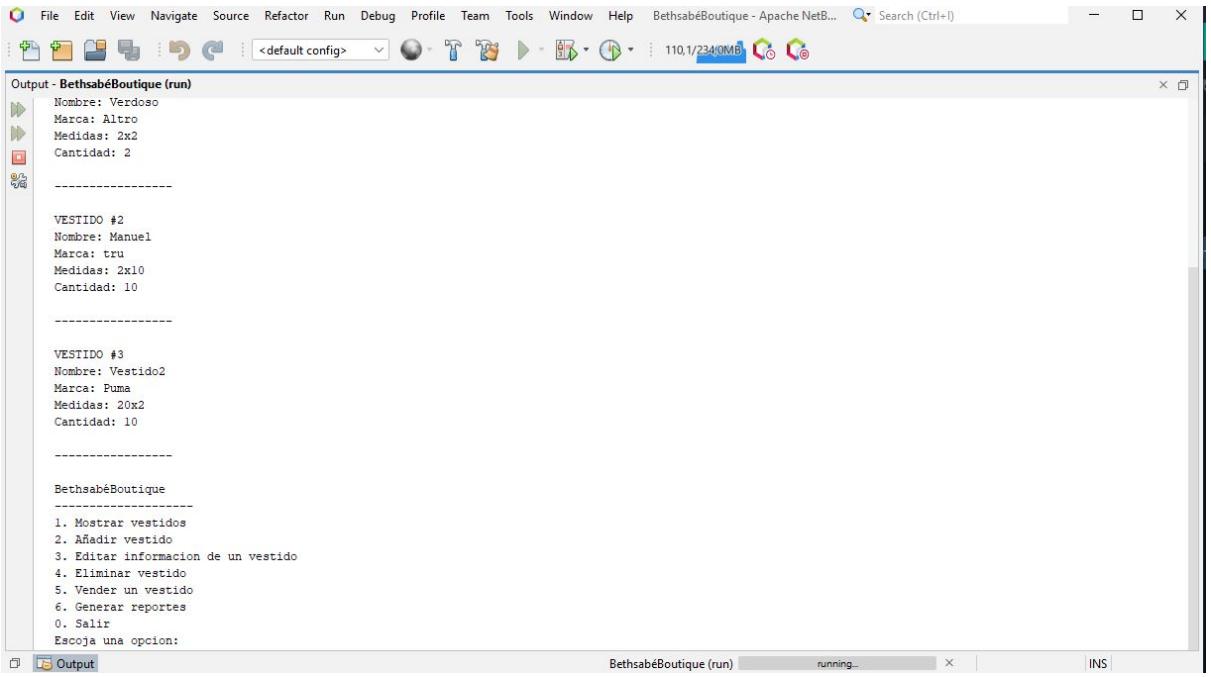
```
Bethsab?Boutique
--------------------
1. Mostrar vestidos
2. A?adir vestido
3. Editar informacion de un vestido
4. Eliminar vestido
5. Vender un vestido
6. Generar reportes
0. Salir
Escoja una opcion: 2
Ingrese el nombre del vestido: Inspection
Ingrese la marca del vestido: Team 2
Ingrese las medidas del vestido: Ingrese el precio del vestido: lol
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. A?adir 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: Altro
Medidas: 2x2
Cantidad: 2

-----------------

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

8-10.- Clean Code




Arial
11
B
I
U
A



EVIDENCE NOTE

Proteger la naturaleza es defender la vida.

Project review

- Pamela Chipe
- Joan Cabeña
- Anabel Dávila
- Andrés Espín

Project review from team 2 to team 3

- Requirements list (project definition)

- 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 requerements

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

(Team 3 project performance)

- Interview and Presentation

apper the members. Only exists file txt of evidence

- Use Case

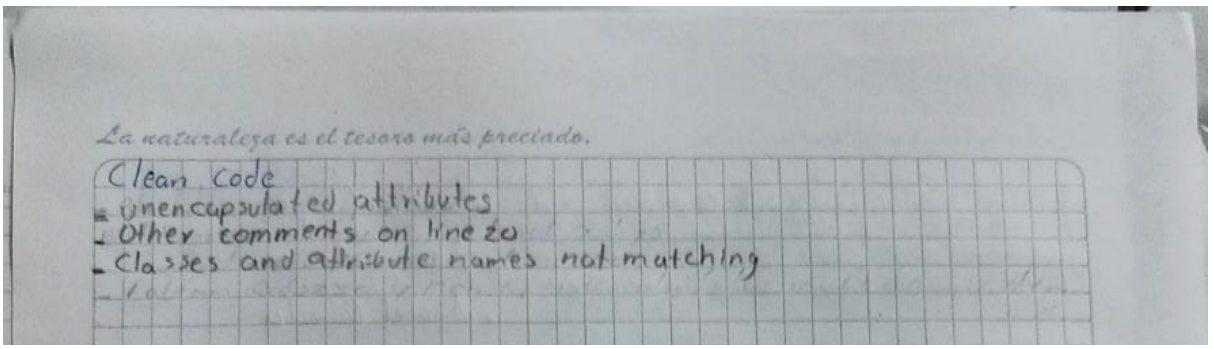
The diagram is just missing the generate report action.

- 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 black spaces in the sting 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

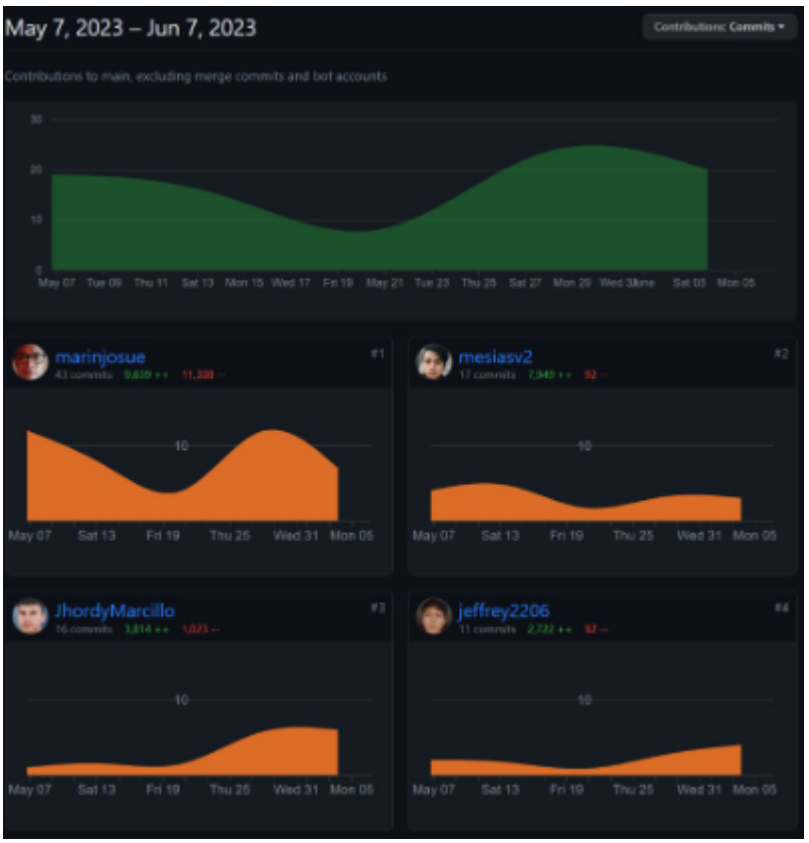
presentation: 8/10

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

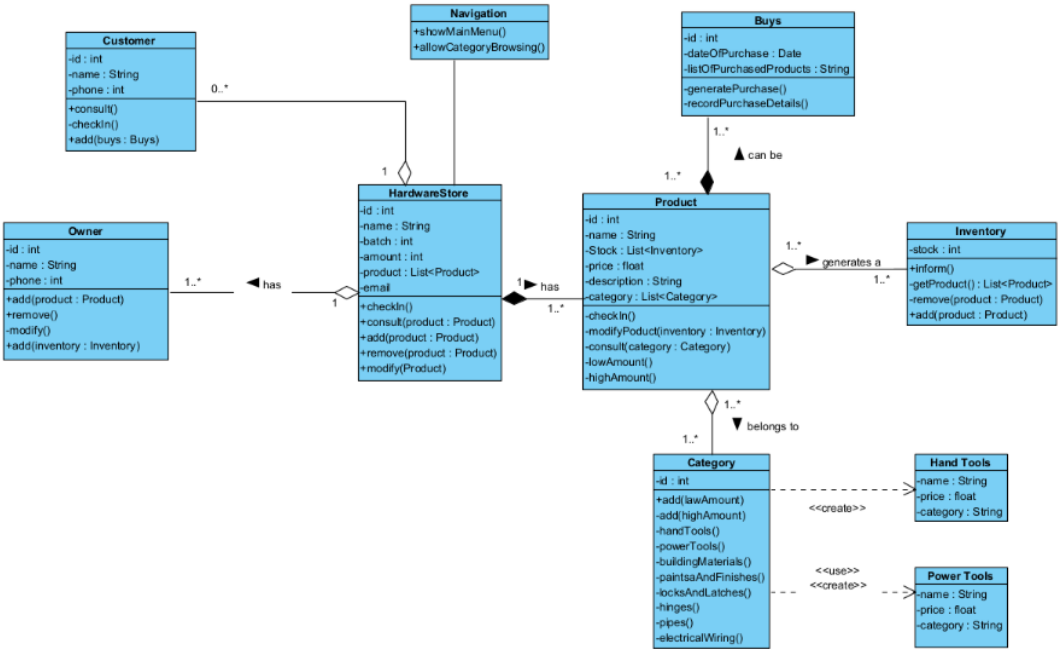
1.2 Scope of the system

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



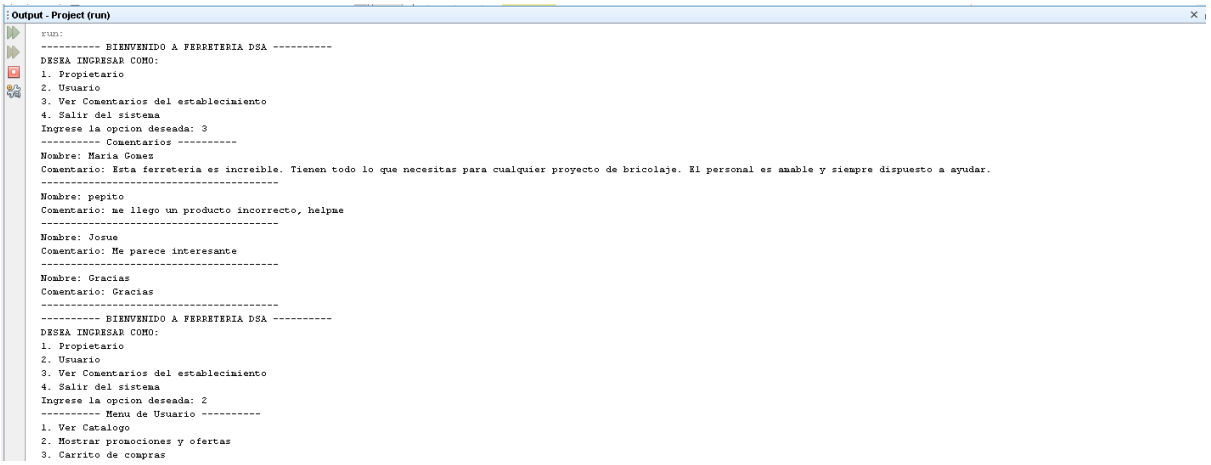
6.- Data files

data: Bloc de notas

Archivo Edición Formato Ver Ayuda

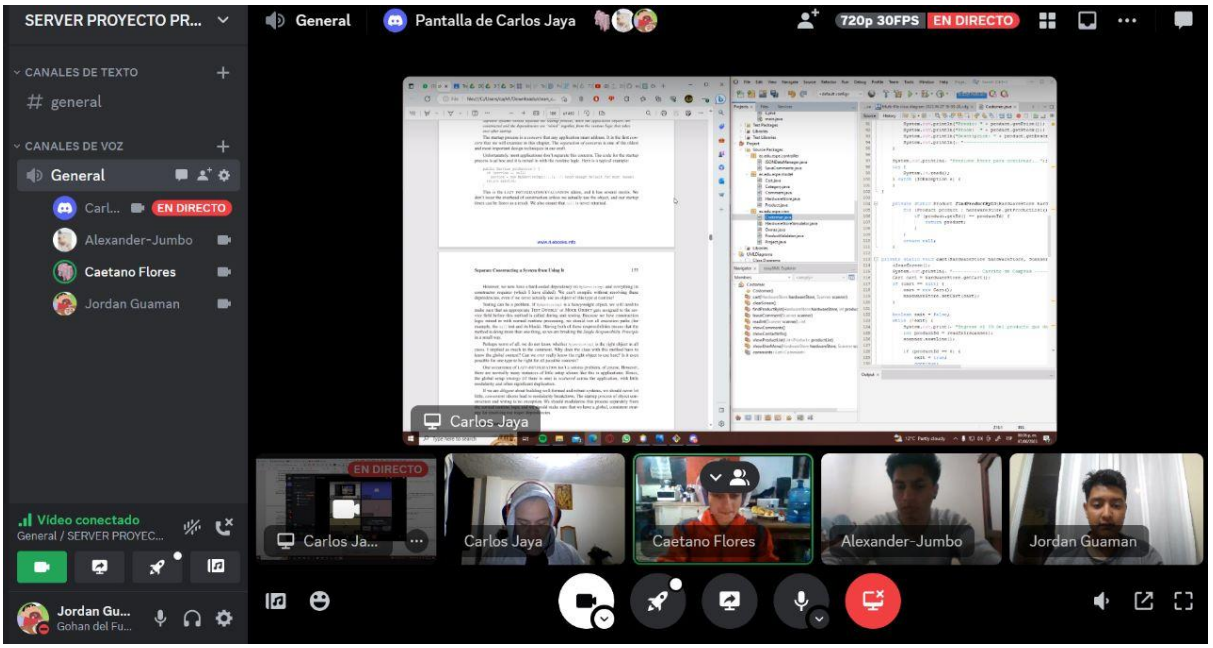

```
{
  "id": 1,
  "name": "Mi ferreteria",
  "batch": 1,
  "amount": 1,
  "productList": [
    {
      "id": 1,
      "name": "Martillo",
      "stock": 10,
      "price": 4.0,
      "description": "El martillo es una herramienta de mano compuesta por una cabeza de metal, generalmente de acero, unida a un mango o mango de madera, plastico o fibra de vi
      "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, elavorada para facilitar la aplicacion uniforme de pintura sobre diversas superficies.",
      "regularPrice": 0.0,
      "discount": 0.0,
      "discountPercentage": 0.0
    },
```

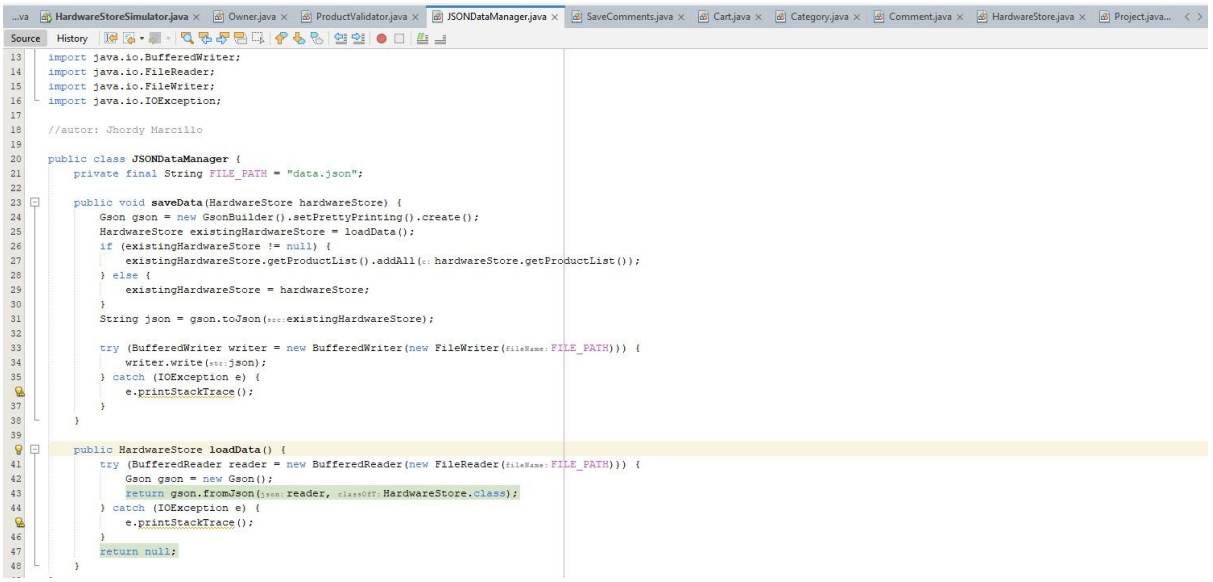
7.- Execution



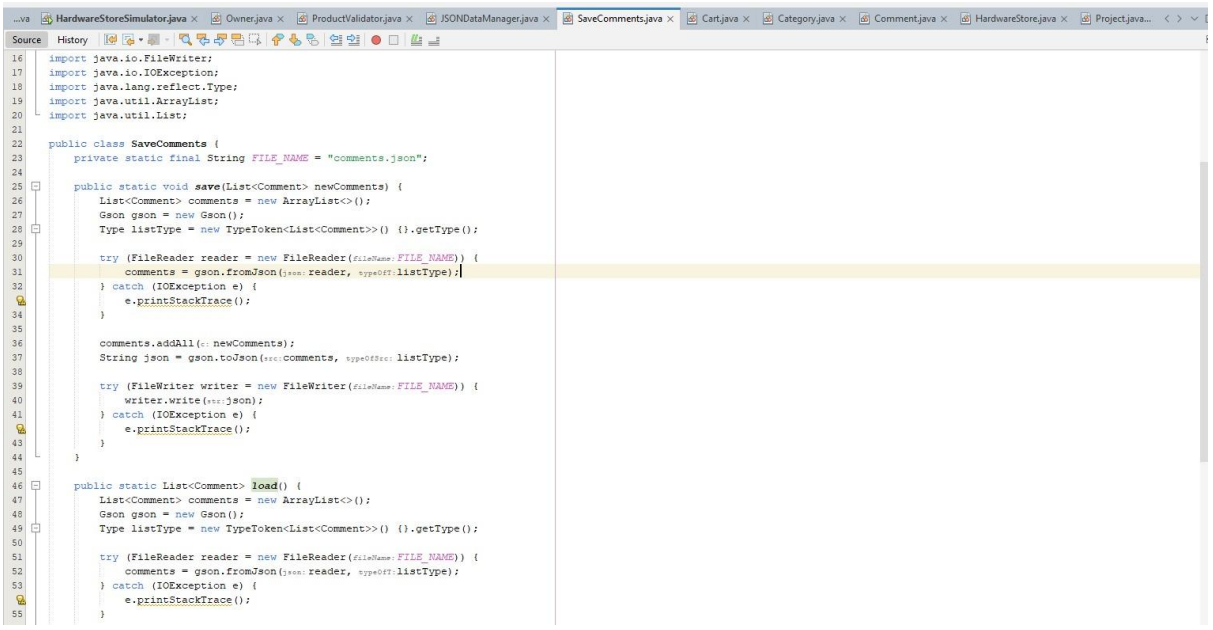
```
Output - Project (run)
run:
---------- BIENVENIDO A FERRETERIA DSA ----------
DESEA INGRESAR COMO:
1. Propietario
2. Usuario
3. Ver Comentarios del establecimiento
4. Salir del sistema
Ingrese la opcion deseada: 3
---------- Comentarios ----------
Nombre: Maria Gomez
Comentario: Esta ferreteria es 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: Josue
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
Ingrese la opcion deseada: 2
---------- Menu de Usuario ----------
1. Ver Catalogo
2. Mostrar promociones y ofertas
3. Carrito de compras
```

8-10.- Clean Code

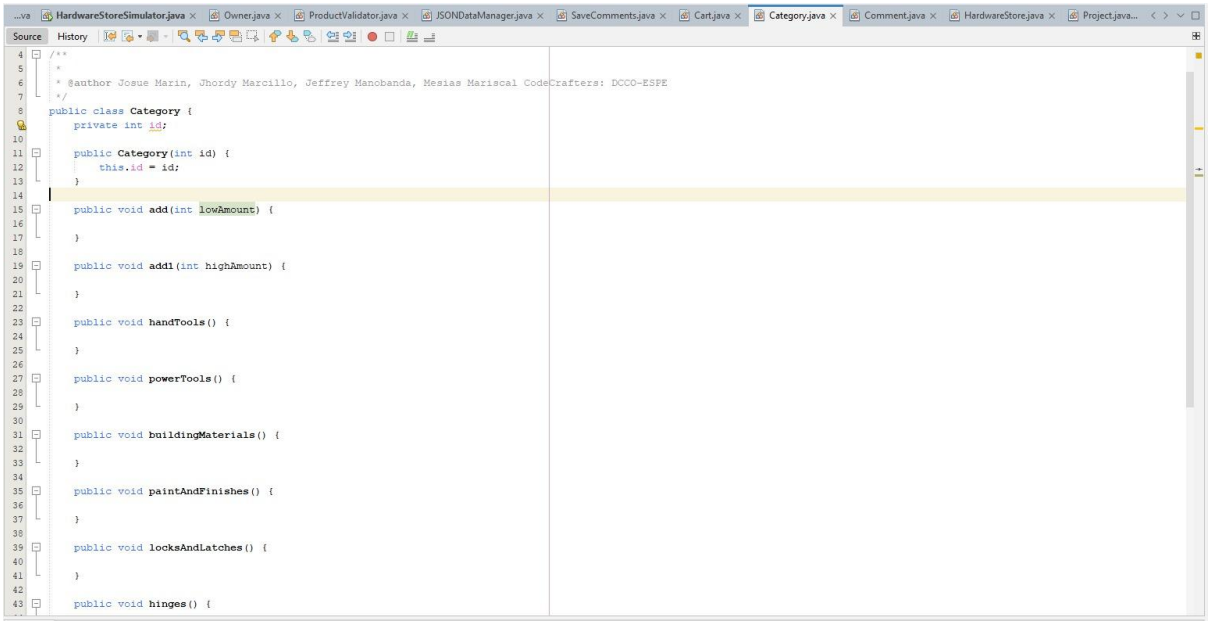




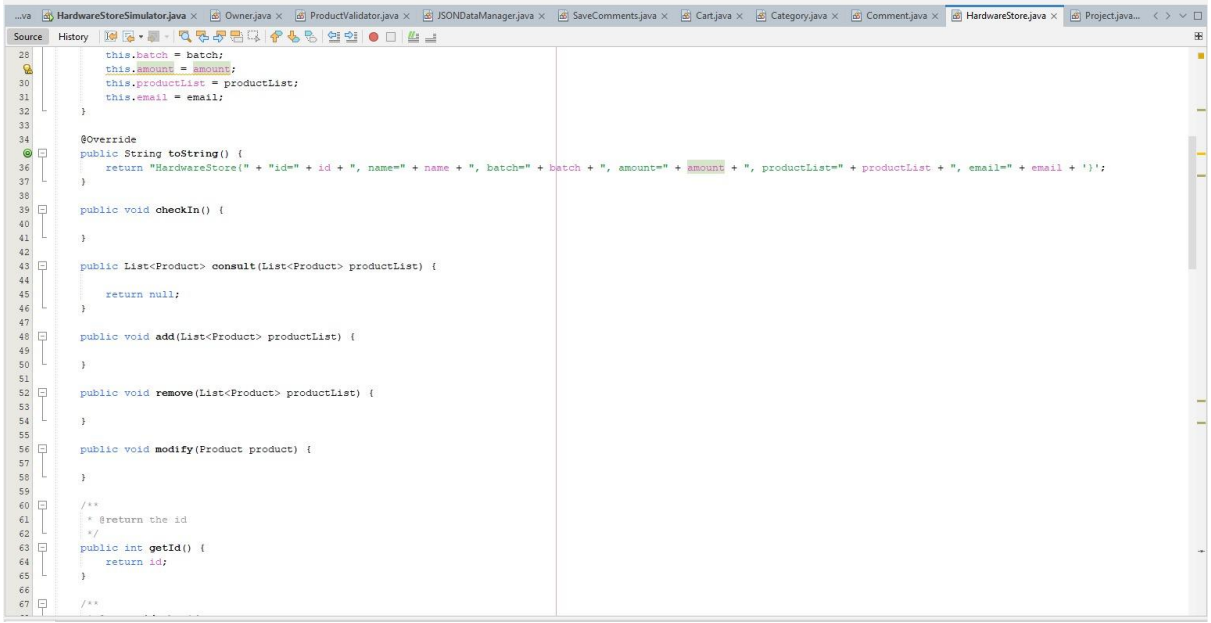
```java
import java.io.BufferedWriter;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;

//autor: Jhordy Marcillo

public class JSONDataManager {
    private final String FILE_PATH = "data.json";

    public void saveData(HardwareStore hardwareStore) {
        Gson gson = new GsonBuilder().setPrettyPrinting().create();
        HardwareStore existingHardwareStore = loadData();
        if (existingHardwareStore != null) {
            existingHardwareStore.getProductList().addAll(c: hardwareStore.getProductList());
        } else {
            existingHardwareStore = hardwareStore;
        }
        String json = gson.toJson(src:existingHardwareStore);

        try (BufferedWriter writer = new BufferedWriter(new FileWriter(fileName:FILE_PATH))) {
            writer.write(str:json);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public HardwareStore loadData() {
        try (BufferedReader reader = new BufferedReader(new FileReader(fileName:FILE_PATH))) {
            Gson gson = new Gson();
            return gson.fromJson(json: reader, classOfT:HardwareStore.class);
        } catch (IOException e) {
            e.printStackTrace();
        }
        return null;
    }
```



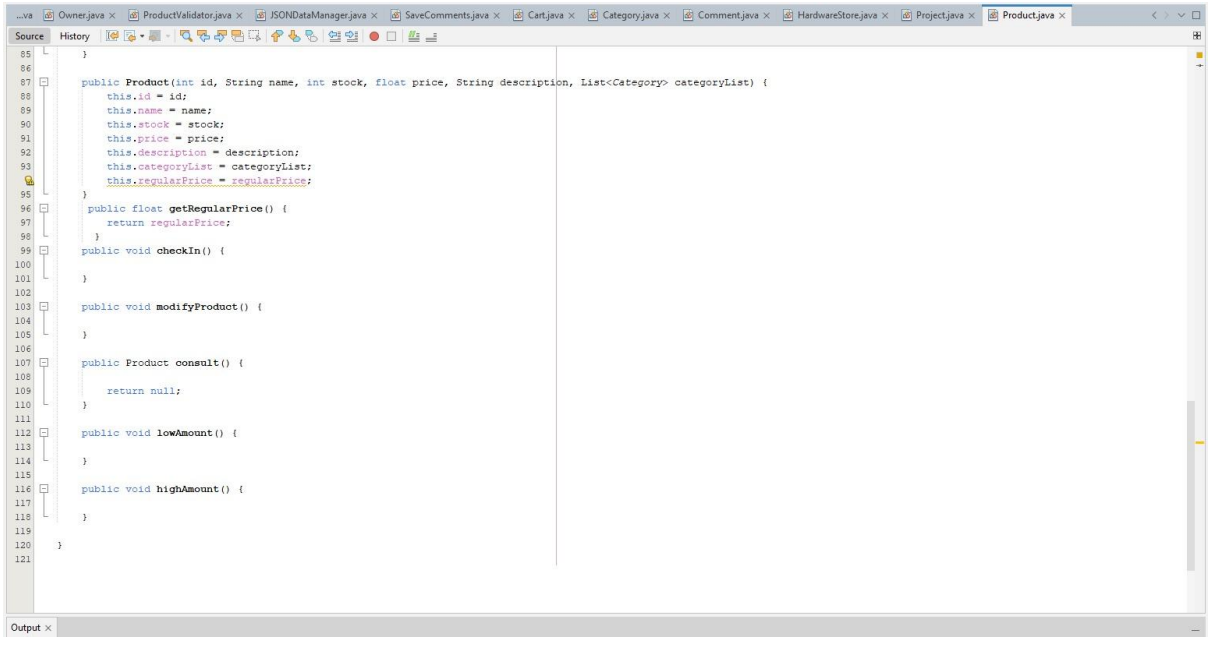
```java
import java.io.FileWriter;
import java.io.IOException;
import java.lang.reflect.Type;
import java.util.ArrayList;
import java.util.List;

public class SaveComments {
    private static final String FILE_NAME = "comments.json";

    public static void save(List<Comment> newComments) {
        List<Comment> comments = new ArrayList<>();
        Gson gson = new Gson();
        Type listType = new TypeToken<List<Comment>>() {}.getType();

        try (FileReader reader = new FileReader(fileName:FILE_NAME)) {
            comments = gson.fromJson(json: reader, typeOfT:listType);
        } catch (IOException e) {
            e.printStackTrace();
        }

        comments.addAll(c: newComments);
        String json = gson.toJson(src:comments, typeOfSrc: listType);

        try (FileWriter writer = new FileWriter(fileName:FILE_NAME)) {
            writer.write(str:json);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public static List<Comment> load() {
        List<Comment> comments = new ArrayList<>();
        Gson gson = new Gson();
        Type listType = new TypeToken<List<Comment>>() {}.getType();

        try (FileReader reader = new FileReader(fileName:FILE_NAME)) {
            comments = gson.fromJson(json: reader, typeOfT:listType);
        } catch (IOException e) {
            e.printStackTrace();
        }
```



```java
/**
 *
 * @author Josue Marin, Jhordy Marcillo, Jeffrey Manobanda, Mesias Mariscal CodeCrafters: DCCO-ESPE
 */
public class Category {
    private int id;

    public Category(int id) {
        this.id = id;
    }

    public void add(int lowAmount) {

    }

    public void add1(int highAmount) {

    }

    public void handTools() {

    }

    public void powerTools() {

    }

    public void buildingMaterials() {

    }

    public void paintAndFinishes() {

    }

    public void locksAndLatches() {

    }

    public void hinges() {
```

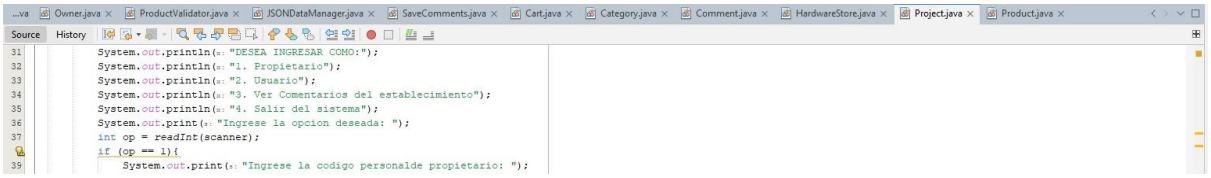


```
        this.batch = batch;
        this.amount = amount;
        this.productList = productList;
        this.email = email;
    }

    @Override
    public String toString() {
        return "HardwareStore{" + "id=" + id + ", name=" + name + ", batch=" + batch + ", amount=" + amount + ", productList=" + productList + ", email=" + email + '}';
    }

    public void checkIn() {

    }

    public List<Product> consult(List<Product> productList) {

        return null;
    }

    public void add(List<Product> productList) {

    }

    public void remove(List<Product> productList) {

    }

    public void modify(Product product) {

    }

    /**
     * @return the id
     */
    public int getId() {
        return id;
    }

    /**
```



```
    }

    public Product(int id, String name, int stock, float price, String description, List<Category> categoryList) {
        this.id = id;
        this.name = name;
        this.stock = stock;
        this.price = price;
        this.description = description;
        this.categoryList = categoryList;
        this.regularPrice = regularPrice;
    }
     public float getRegularPrice() {
        return regularPrice;
      }
    public void checkIn() {

    }

    public void modifyProduct() {

    }

    public Product consult() {

        return null;
    }

    public void lowAmount() {

    }

    public void highAmount() {

    }

}

```

Output ×



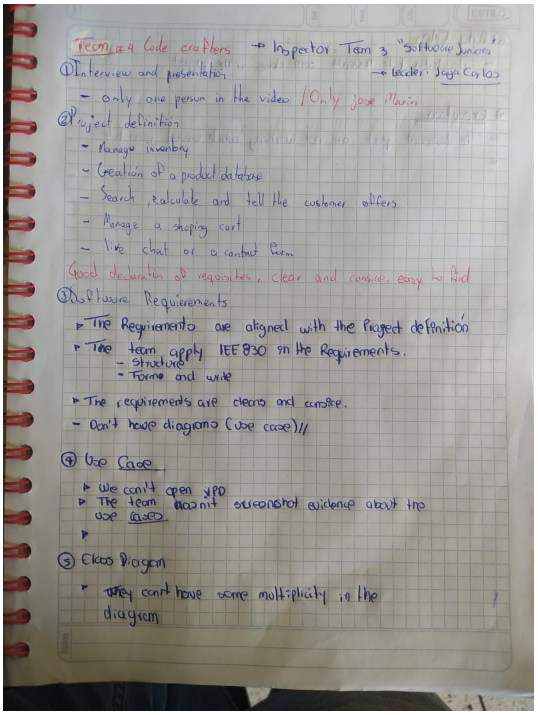
```
        System.out.println(x: "DESEA INGRESAR COMO:");
        System.out.println(x: "1. Propietario");
        System.out.println(x: "2. Usuario");
        System.out.println(x: "3. Ver Comentarios del establecimiento");
        System.out.println(x: "4. Salir del sistema");
        System.out.print(s: "Ingrese la opcion deseada: ");
        int op = readInt(scanner);
        if (op == 1){
            System.out.print(s: "Ingrese la codigo personalde propietario: ");
```

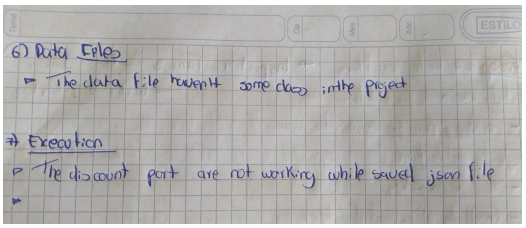
...dg | Customer.java | HardwareStoreSimulator.java | Owner.java | ProductValidator.java | JSONDataManager.java | SaveComments.j

Source | History

```
1
2   package ec.edu.espe.view;
3
4   /**
5    *
6    * @author Josue Marin, Jhordy Marcillo, Jeffrey Manobanda, Mesias Mariscal CodeCrafters: DCCO-ESPE
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  public class HardwareStoreSimulator {
13
14      public static void main(String[] args) {
15
16
17          JSONDataManager jsonDataManager = new JSONDataManager();
18          HardwareStore hardwareStore = jsonDataManager.loadData();
19          showMainMenu(hardwareStore);
20      }
21      }
```

EVIDENCE NOTE:





The FAMSE Project.

1. **Abstract.**

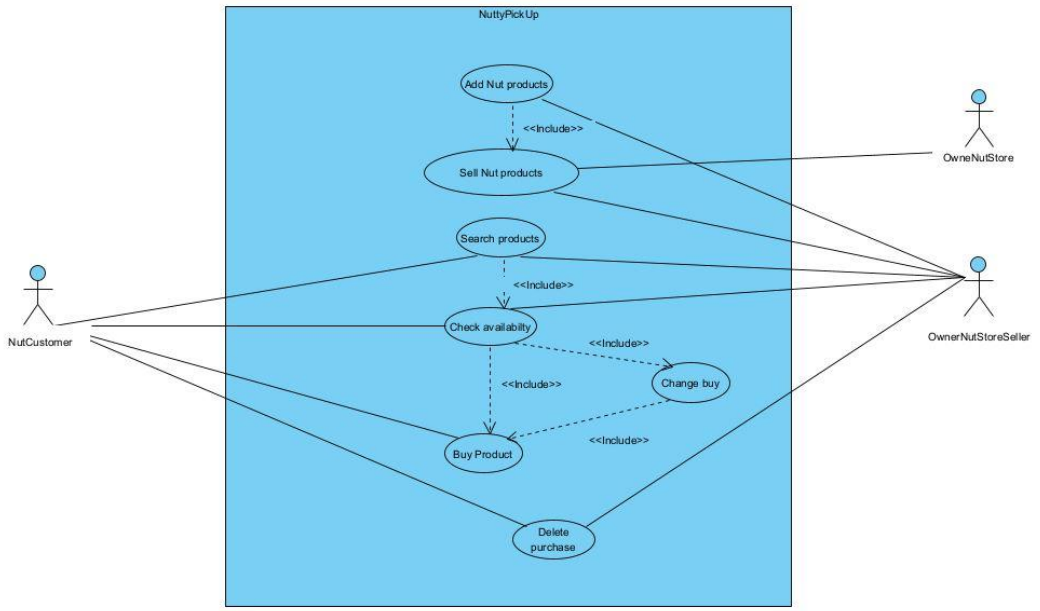
This project aims to create a product reservation system for a nut store that will allow customers to make their purchases faster and more comfortable. This system will have several functional and non-functional requirements, as well as include an intuitive and non-complex interface for customers to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

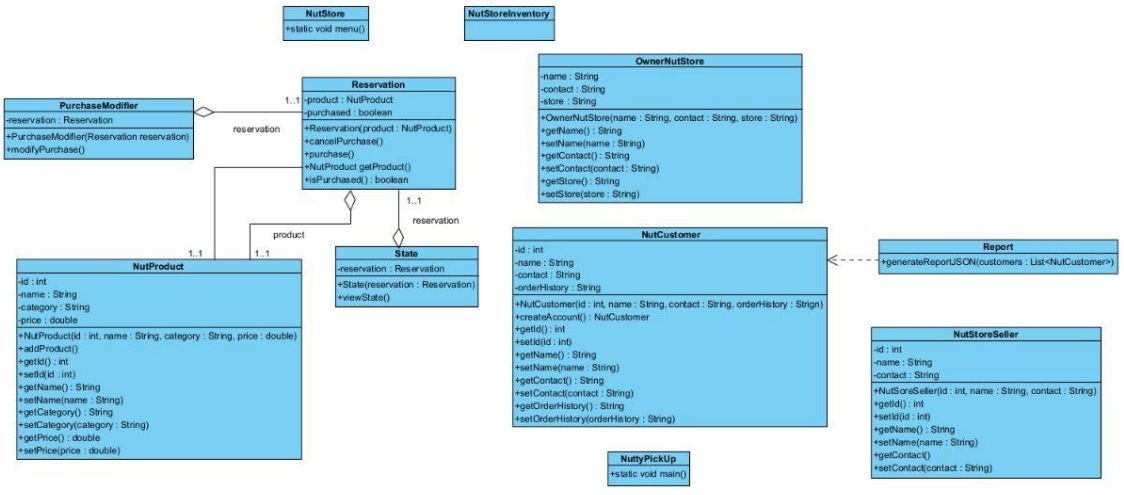
3.- Software Requirements
Bad enumeration
Does not contain diagrams

Contents

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

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

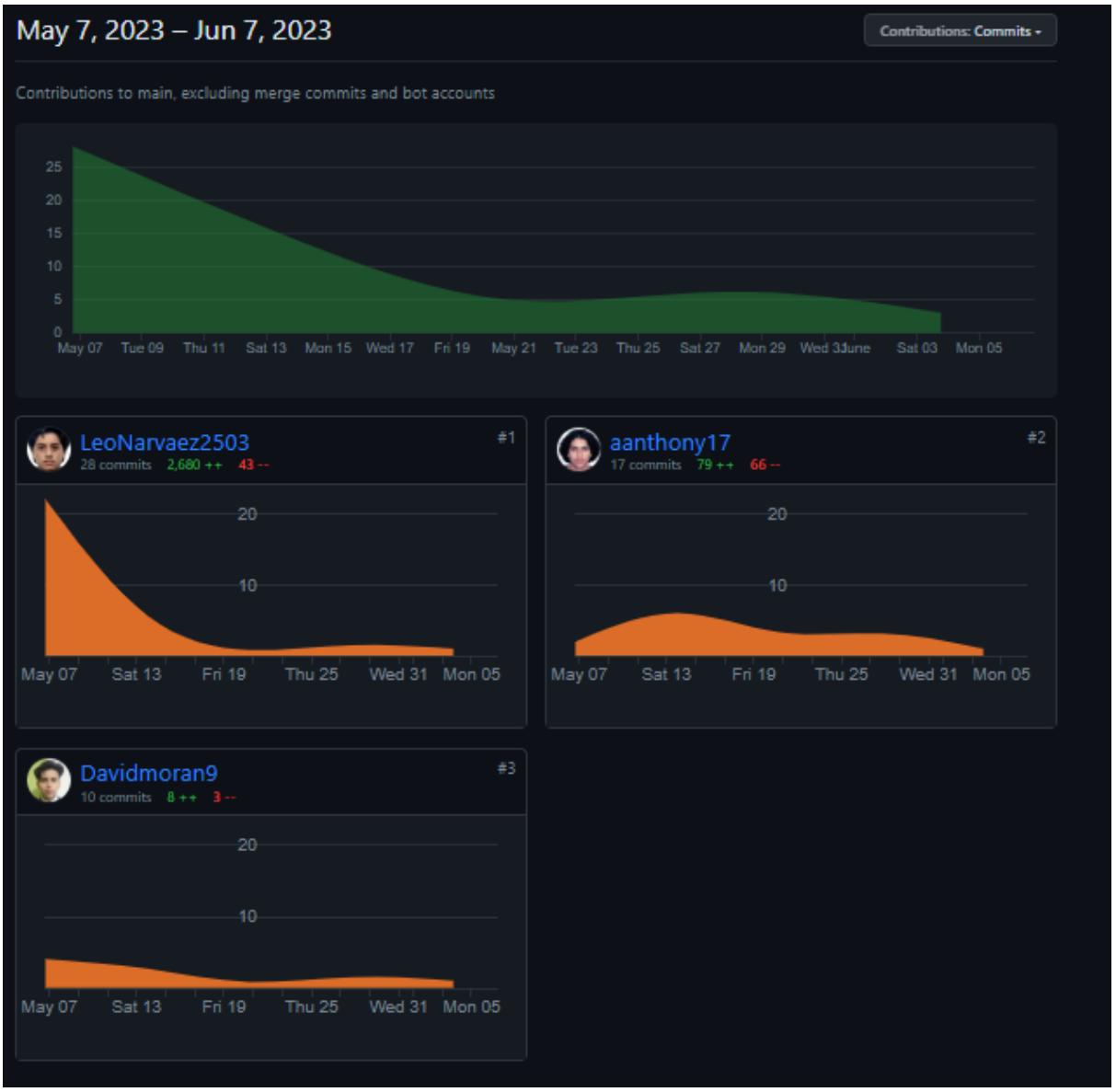
presentation: 6.5/10

Rubric

Presentation	
0.- GitHub	/100
1.- Interview and Presentation	8/10 (Only one member of the group appears)
2.- Project definition	6/10 (Definition does not specify the mentioned requirements)
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 (Missing associations and the one of "Use" and "Create" poorly structured.) (Put dependencies, do not leave classes alone.)	7/10
6.- Data files	6/10 (a single JSON file)
7.- Execution	7/10(Added products are not saved.)
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	24/30
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

D 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

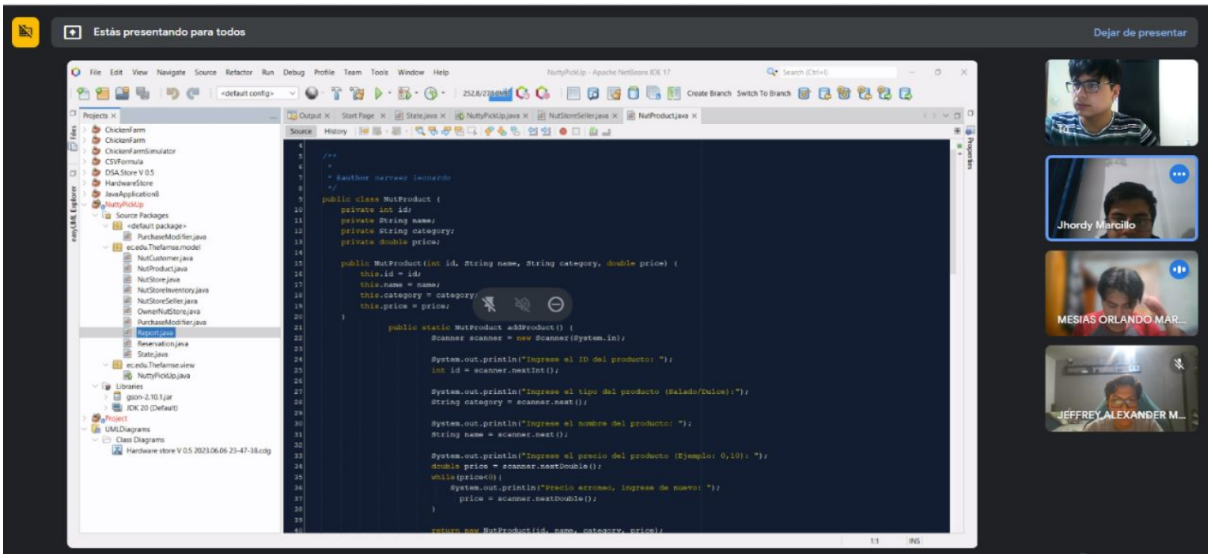
7.- Execution

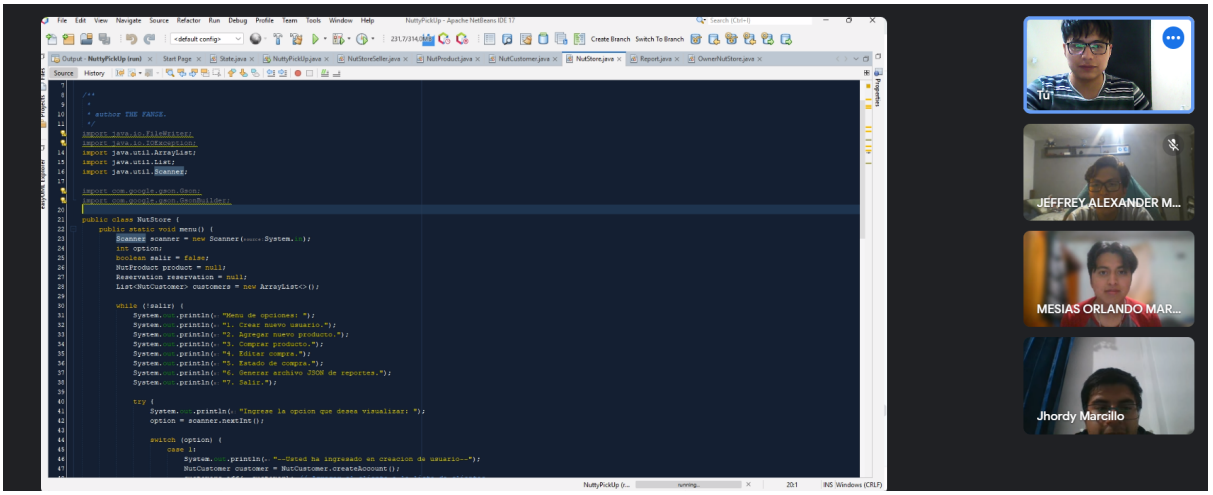
No product catalog
Added products are not saved.
Does not generate purchases

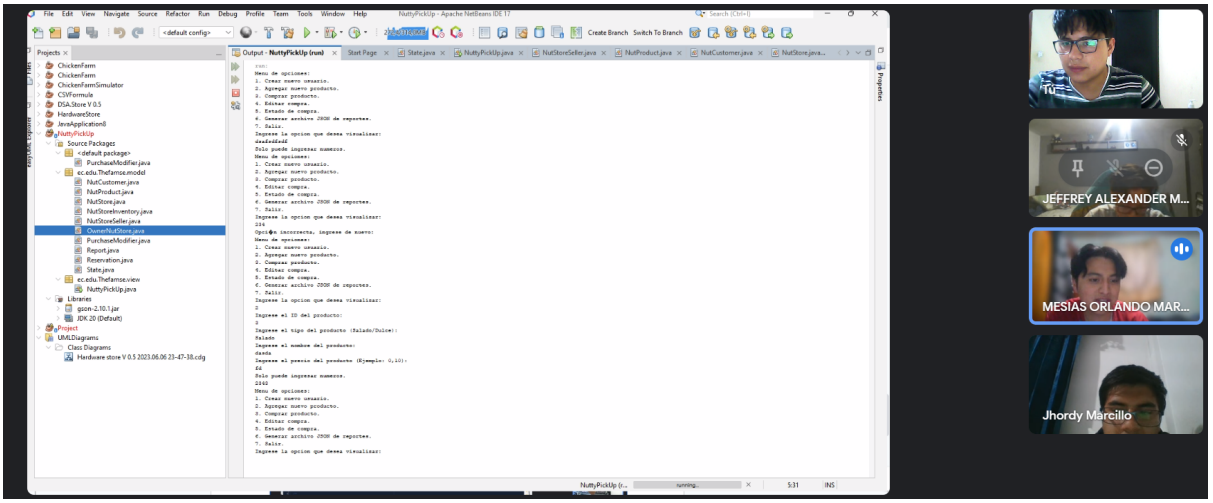
```
Menu de opciones:
1. Crear nuevo usuario.
2. Agregar nuevo producto.
3. Comprar producto.
4. Editar compra.
5. Estado de compra.
6. Generar archivo JSON de reportes.
7. Salir.
Ingrese la opcion que desea visualizar:
2
Ingrese el ID del producto:
4
Ingrese el tipo del producto (Salado/Dulce):
Dalsado
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:
```

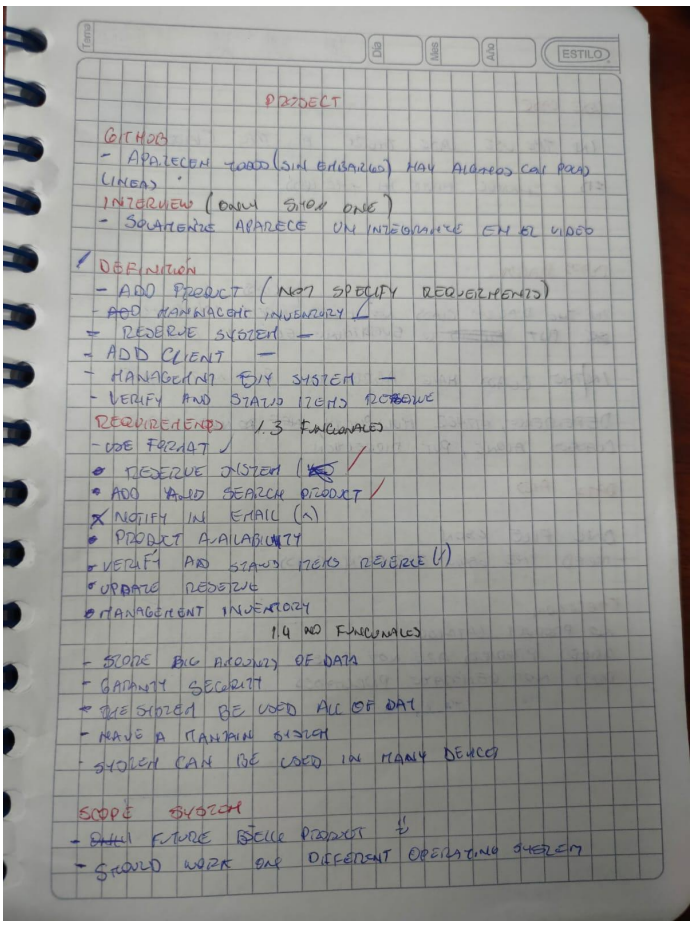
8-10.- Clean Code
Classes that do not fulfill any function
Comments in Spanish and unnecessary
Unused floating packages
Functions that do not operate correctly




TU
JEFFREY ALEXANDER M...
MESIAS ORLANDO MAR...
Jhordy Marcillo


TU
JEFFREY ALEXANDER M...
MESIAS ORLANDO MAR...
Jhordy Marcillo

Notes

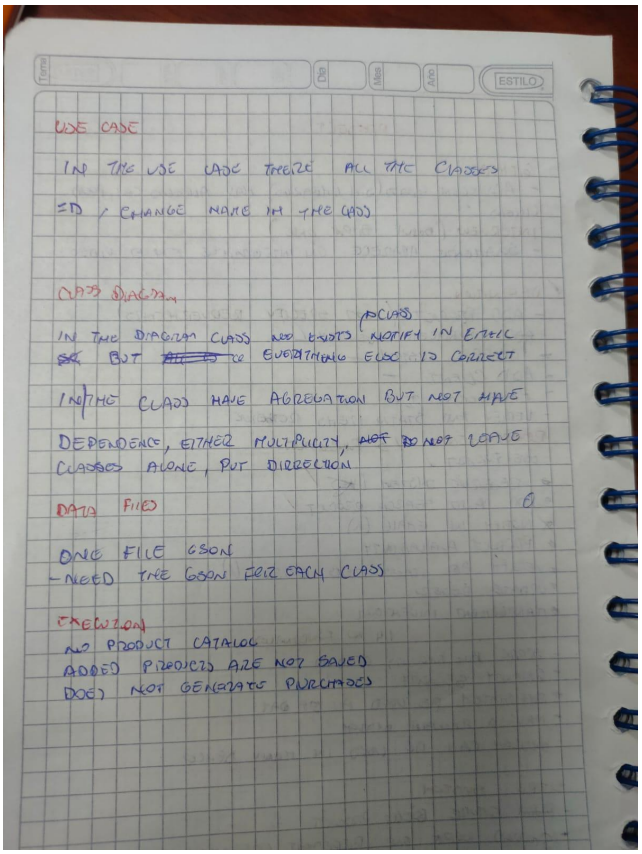


PROJECT

GITHUB
- APARECEN TODOS (SIN EXCEPCIÓN) HAY ALGUNOS CAL POR LINEAS
INTERVIEW (ONLY SIDE ONE)
- SOLAMENTE APARECE UN INTEGRANTE EN EL VIDEO

DEFINITION
- ADD PRODUCT (NOT SPECIFY REQUIREMENTS)
- ~~ADD~~ MANAGMENT INVENTORY ✓
- RESERVE SYSTEM —
- ADD CLIENT —
- MANAGMENT ENY SYSTEM —
- VERIFY AND STATUS ITEMS RESERVE
REQUIREMENTS 1.3 FUNCIONALES
- USE FORMAT ✓
• RESERVE SYSTEM (~~NO~~) ✓
• ADD AND SEARCH PRODUCT ✓
✗ NOTIFY IN EMAIL (✗)
• PRODUCT AVAILABILITY
• VERIFY AND STATUS ITEMS RESERVE (✓)
• UPDATE RESERVE
• MANAGEMENT INVENTORY
1.4 NO FUNCIONALES
- STORE BIG AMOUNTS OF DATA
- GARANTY SECURITY
- THE SYSTEM BE USED ALL OF DAY
- HAVE A MAINTAIN SYSTEM
- SYSTEM CAN BE USED IN MANY DEVICES

SCOPE SYSTEM
- ~~SHALL~~ FUTURE DELETE PRODUCT ✓
- SHOULD WORK ON DIFFERENT OPERATING SYSTEM



USE CASE

IN THE USE CASE FREEZE ALL THE CLASSES

=D / CHANGE NAME IN THE CASO

CLASS DIAGRAM

IN THE DIAGRAM CLASS (CLASS) NOTIFY IN EMAIL NO ENDS
~~SO~~ BUT ~~ALL~~ THE EVERYTHING ELSE IS CORRECT

IN THE CLASS HAVE AGREGATION BUT NOT HAVE

DEPENDENCE, EITHER MULTIPLICITY, ~~NOT~~ DO NOT LEAVE CLASSES ALONE, PUT DIRECTION

DATA FILES

ONE FILE GSON
- NEED THE GSON FOR EACH CLASS

EXCEPTION
NO PRODUCT CATALOG
ADDED PRODUCTS ARE NOT SAVED
DOES NOT GENERATE PURCHASES

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

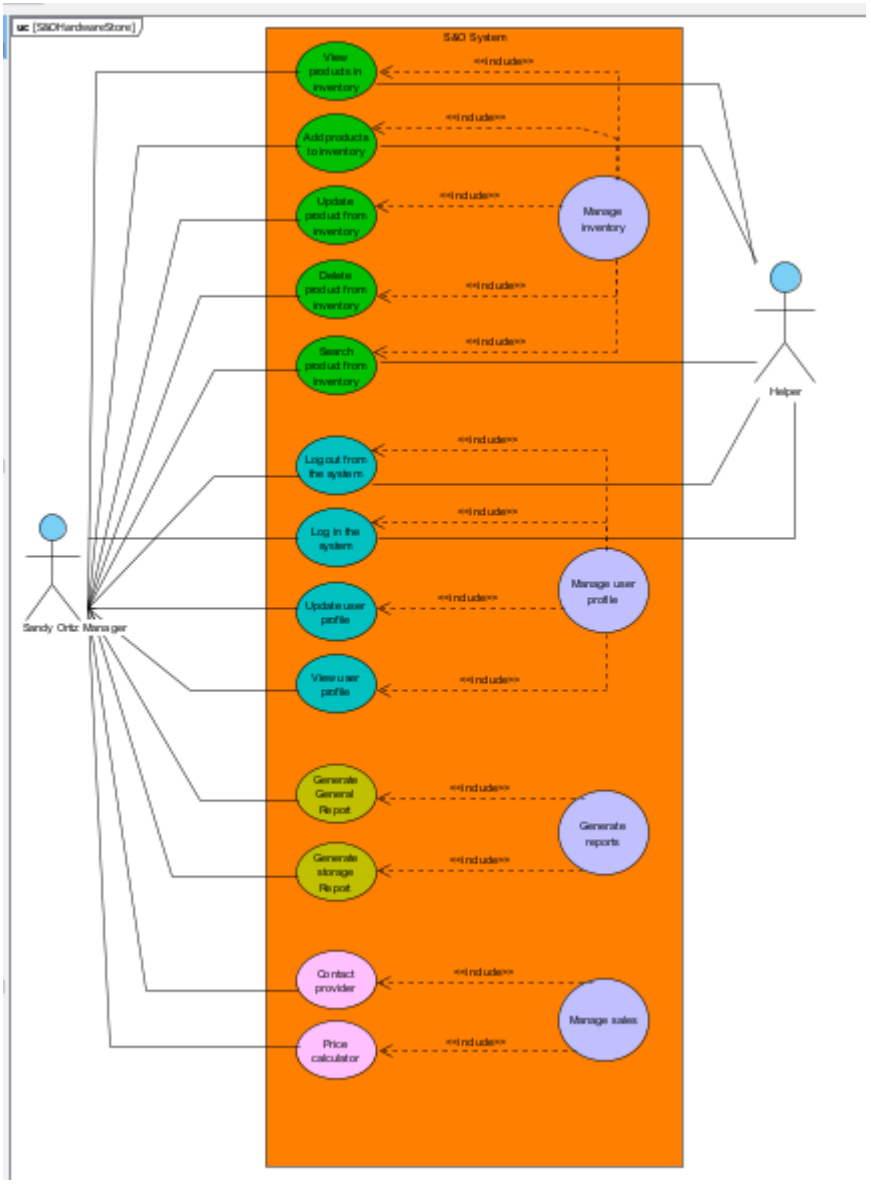
2.- Project definition

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.

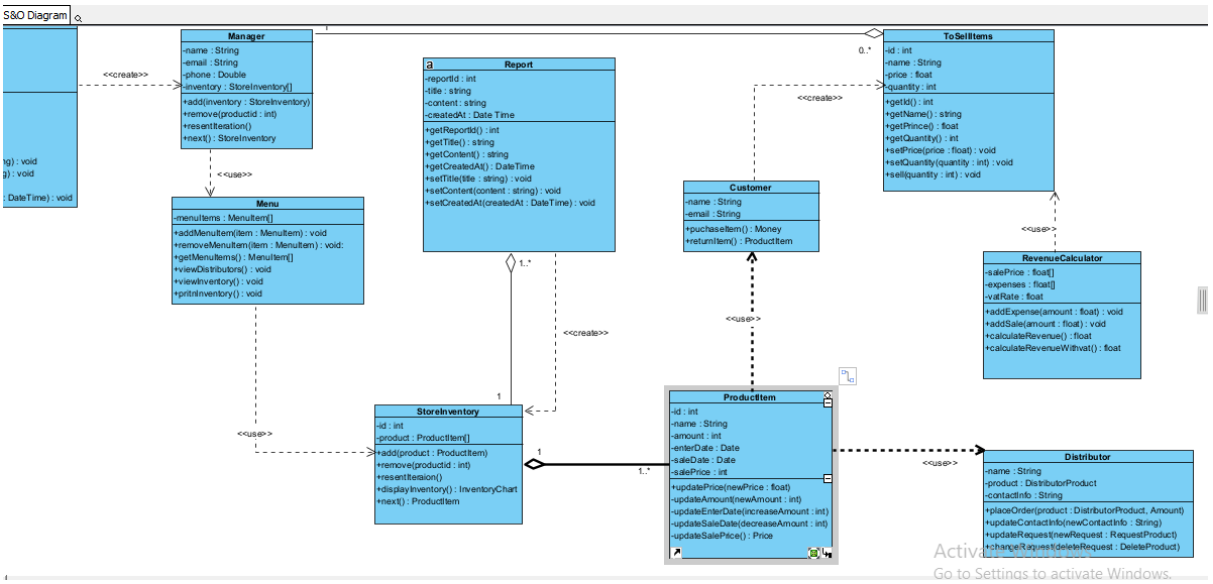
3.- Software Requirements

Table of Contents

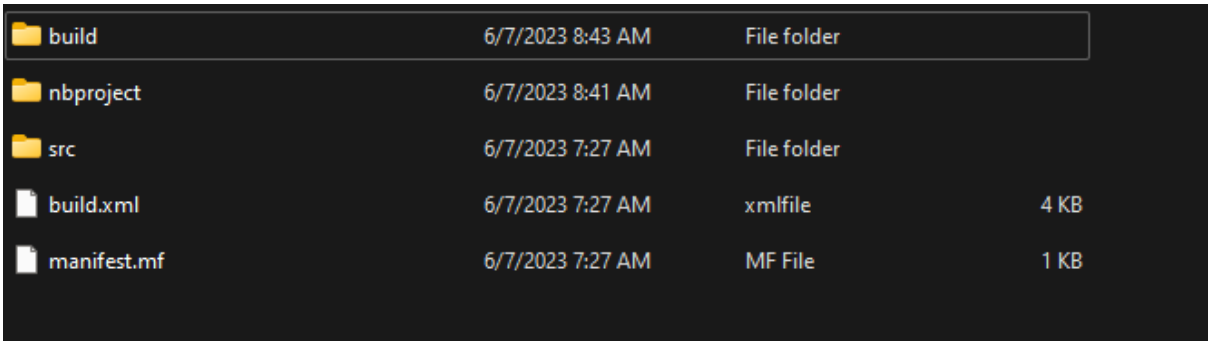
4.- Use Case



5.- Class Diagram



6.- Data files



7.- Execution

```
Nombre de usuario: username
Contrase�a: password
Inicio de sesi�n exitoso. �Bienvenido!
----- Men� -----
1. Crear usuarios
2. Ingresar productos
3. Vender productos
4. Ver reporte de ventas
5. Contactos de proveedores
6. Salir
Ingrese la opci�n: 1
Se han creado los usuarios
----- Men� -----
1. Crear usuarios
2. Ingresar productos
3. Vender productos
4. Ver reporte de ventas
5. Contactos de proveedores
6. Salir
Ingrese la opci�n: 1
Se han creado los usuarios
----- Men� -----
1. Crear usuarios
2. Ingresar productos
3. Vender productos
4. Ver reporte de ventas
5. Contactos de proveedores
6. Salir
Ingrese la opci�n:
```

8-10.- Clean Code

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

            do {
                System.out.println(x: "----- Menú -----");
                System.out.println(x: "1. Crear usuarios");
                System.out.println(x: "2. Ingresar productos");
                System.out.println(x: "3. Vender productos");
                System.out.println(x: "4. Ver reporte de ventas");
                System.out.println(x: "5. Contactos de proveedores");
                System.out.println(x: "6. Salir");
                System.out.print(s: "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(x: "Saliendo del programa...");
                    default -> System.out.println(x: "Opción inválida");
                }
            } while (option != 6);
        }
    }

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

```java
/**
 *
 * @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(x: "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) {
```

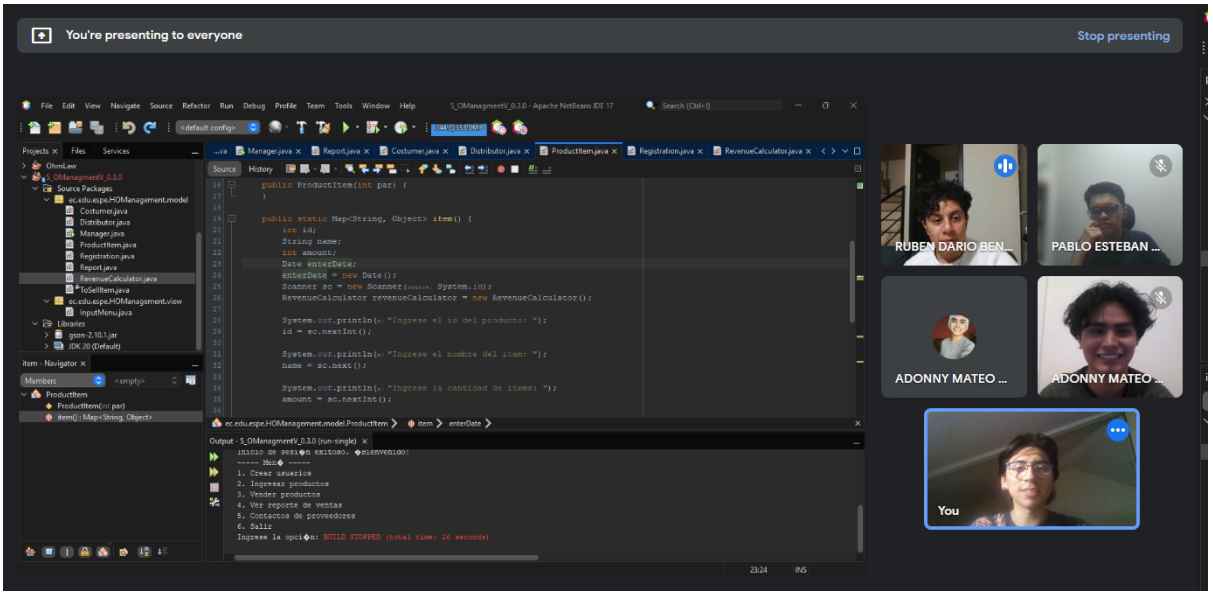
```java
package ec.edu.espe.HOManagement.model;

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

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

```java
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();
```

```java
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;
}
```

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



Evidence Notes:

Report of Inspection

Team: 1 Leader: Yestiva Amador Chilquinga Anaya

Name: Jsons

Team Inspected: 6

Team Inspected Name: Code Warriors

0. GitHub

Screenshot of activity in the repository

1.- Interview

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 stored that the information was saved in Excel during the presentation.

3.- Requirements

We reviewed that the document complies with the IEEE 830 standard of SRS and all of its parts. Result: 7,5, because the requirements do not follow the format of numbering IEEE 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 Jsons, 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: 9 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 is empty); after reviewing the requirements the class wasn't found, also there are some attributes names that are verbs like enterDate, requestedSellPercentage, and a method that is not a verb "SupplierContacts". Finally there is a Scanner so that is not used

GAMA