APPLICATION DESIGN TEMU

Jefferson David Rico Ruiz	20242020108
Nelson Navarro De la Rosa	20242020116

REQUIREMENTS

- Be able to log in to the application to collect buyer data.
- Through the app, can make purchases of different products.
- Allow efficient shipping and logistics.
- Ensure correct communication between the user and the seller.
- Encourage impulse purchases through promotions.
- Maintain a correct stock of the items sold in the application.
- Facilitating international shopping.
- Register sales and payments for products.
- Allow the user to select the desired items through product sections.
- Ensure security in the payment process for products.

USER STORY

TITLE: Sort products by category	**PRIORITY:** Medium	**ESTIMATE:** 3 weeks

User Story: As a potential buyer in the app, I want to have a main interface so I can easily access the products I need through categories that show me related products. This allows for faster and easier interaction with the application.

Acceptance Criteria: Given a number of categories defined by the number of products offered and the relationship between them, when the user interacts with the categories, they will be able to access another window where the products assigned to that category are filtered.

USER STORY

TITLE: Products on Sale	**PRIORITY:** High	**ESTIMATE:** 3 weeks

User Story: As a potential buyer, I want to see a sales window showing a random product with a discount and its respective percentage when opening the app. This is intended to motivate and attract attention to buying products that are at a lower price.

Acceptance Criteria: Given a list of discounted products, choose one at random to be displayed when starting the application. The user has the option to choose whether he is interested in the offer or not. If the user is interested, when he interacts with the window offer he is redirected to the discounted products section. On the contrary, if he is not interested, he is sent to the initial interface.

USER STORY

TITLE: Product Purchase Interface	**PRIORITY:** High	**ESTIMATE:** 3 weeks

User Story: As a potential buyer, I want to be redirected to an interface that displays an image of the product, its price, description, reviews from other users, and a "Buy Now" button when selecting a desired product in any category. This allows the user to choose the product to purchase and learn more about it.

Acceptance Criteria: Depending on the selected product, they can choose the purchase option.When they click the "Buy Now" button, they will be redirected to the payment section, with payment options and shipping methods, to complete the purchase process.

USER STORY

TITLE: Payment methods	**PRIORITY:** High	**ESTIMATE:** 3 weeks

User Story: As a user of the application I want that at the time of paying for the product(s) I am allowed to choose between several payment methods to be able to pay with the method that suits me best, and to have a section to use discount coupons if I have one.

Acceptance Criteria: After choosing the payment method, at the time of entering my data, ask for my residence address to know where to send the product.
The product's value will be displayed upon checkout, and if it's on sale, the price minus the discount will be shown, followed by the final price of the product. Finally, a button that says "Checkout."

USER STORY

TITLE: sending the package	**PRIORITY:** Medium	**ESTIMATE:** 3 weeks

User Story: As a buyer, I want a space where the approximate date and time of the package's arrival is shown, accessible as many times as you want an update of its location periodically in order to plan the arrival of the product at its destination.

Acceptance Criteria: Given the approximate date and time, when this data is displayed, I will be able to interact with the confirm button which should redirect me to the main interface made up of products divided into categories.

USER STORY

TITLE: User registration and login	**PRIORITY:** Medium	**ESTIMATE:** 3 weeks

User Story: As a potential buyer, I want two interfaces, one to register and provide data such as name, age, residential address etc. for example, when entering the application for the first time, and another to log in after I have previously registered, in order to provide data to this.

Acceptance Criteria: When I first enter the application, if I click on register or log in, I will be redirected to the interfaces corresponding to those functionalities.

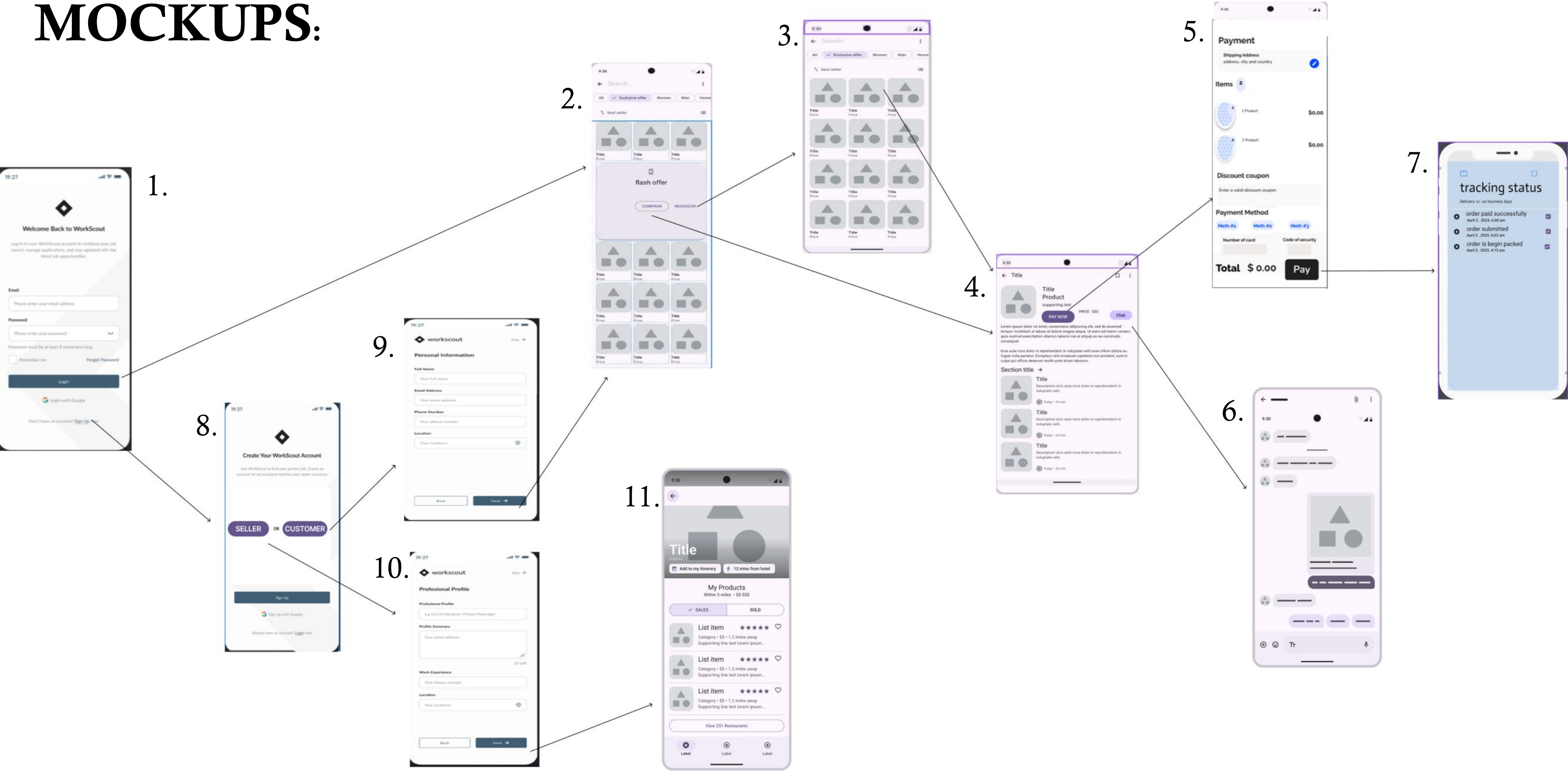
USER STORY

TITLE: Sales records	**PRIORITY:** Medium	**ESTIMATE:** 3 weeks

User Story: As a seller of app products, I want there to be an interface where the sales made are registered to have a record of the sales made.

Acceptance Criteria: The moment a product is purchased, the seller will be notified and the purchase will be automatically added to the respective interface and after that the seller will be able to enter the interface and see the product sold.

MOCKUPS:


1.
Welcome Back to WorkScout
Email
Password
Login
Login with Google
Don't have an account? Sign Up
2.
Search
All
Exclusive offer
Women
Man
Home
best seller
Title
Price
flash offer
COMPRAR
RECHAZAR
3.
4.
Title
Product
supporting text
PAY NOW
PRICE : $$$
Chat
Section title
5.
Payment
Shipping Address
address, city and country
Items 2
1 Product
2 Product
$0,00
Discount coupon
Enter a valid discount coupon
Payment Method
Meth #1
Meth #2
Meth #3
Number of card
Code of security
Total $ 0.00
Pay
6.
7.
tracking status
Delivery 12- 20 business days
order paid successfully
order submitted
order is begin packed
8.
Create Your WorkScout Account
SELLER
OR
CUSTOMER
Sign Up
9.
workscout
Personal Information
Full Name
Email Address
Phone Number
Location
Back
Next
10.
workscout
Profesional Profile
Professional Profile
Profile Summary
Work Experience
Location
Back
Next
11.
Title
Add to my itinerary
12 mins from hotel
My Products
Within 5 miles • $$-$$$
SALES
SOLD
List item
Category • $$ • 1.2 miles away
Supporting line text lorem ipsum...
View 231 Restaurants
Label

Why the mockups?

Mockpu 1.
We chose this mockup because it shows the interface for logging in and registration if the user doesn't have an account associated with the app.

Mockup 2.
We chose this design because it has window of a product offering to generate customer attraction and interest.

Mockup 3.
We chose this design because it displays the products and categories in an orderly manner, giving an organized look to the main interface.

Mockup 4.
We chose the design for mockups 4 because the interface shows the product more specifically with its description and the respective payment button and chat button.

Mockup5.
We chose the design of mockup 5 because it is very complete when it comes to presenting the payment options, the discount coupon section, and the selected product, and the price along with the pay button.

Mockup 6.
We chose this mockup because it shows the chat interface that the buyer can enter and talk to the seller of the product.

Mockup7.
We chose the design of model 6 because it shows the shipping interface and the stages through which the package passes to reach the customer.

Mockup8.
We chose this mockup because it allows us to ask the user if they want to create their account as a seller or buyer.

Mockup9.
We chose this mockup because it shows us a sign-up interface for a person who wants to use the app to buy products.

Mockup 10.
We chose this mockup because it shows a sign-up interface for a person who wants to use the app to sell products.

Mockup 11.
We chose this mockup because it shows the seller's interface where you can see the list of products they sell and the products they have sold.

CRC CARDS

Class: Seller

Responsabilities

- The seller has to publish his products on the application.
- Ensure a correct purchase and sale process with the user.
- Resolve doubts before and after the sale.
- Coordinate the shipment or delivery of the product as established in the app.
- Follow the rules and terms of use of the app.

Collaborators

Customers

- These allow the seller to have someone to promote their products and thereby generate sales and profits. The customer can also provide criticism or feedback regarding the seller's dealings and their handling of the sales process.

Product

- The seller publishes many products where they interact to achieve your purpose, allowing you to establish contact with the customer

Payment

- Receive data from this class to monitor sales

Login

- Can collect data such as name, age, etc. to validate each seller

CRC CARDS

Class: Customer

Responsabilities

- Request returns or claims, if the product does not meet what was promised.

- Order the product and finish paying for it.

- Have knowledge of the products that are for sale in the app.

- Have the ability to choose the desired product(s) across categories.

Collaborators

Seller

Provides products that the user might be interested in

Products

You can view and buy many products,
each from different sellers.

Payment

- The customer can use this class to make a purchase.

Login

- The customer can enter login information such as their name, age, etc. to validate each user.

CRC CARDS

Class: Product

Responsabilities

- Have a photo, description, and price to provide information to the buyer
- Know what type of item it is and, therefore, what category it should be associated with.
- Receive customer ratings after receiving the product
- Going on offer eventually.

Collaborators

Seller

- This person controls the product's features so they can interact with the customer. They can implement offers, modify prices, and images.

Customer

- The customer can view, rate, and purchase the product.

Payment

- The product can pass data to the Payment class, such as its price and image.

CRC CARDS

Class: Payment	
Responsabilities	Collaborators
- Record the sale of a certain product - Validate the data according to the card number, security number and amount of money on the card and discount the value of the product. - Calculate the new value to be paid depending on the discount percentage, if there is one.	Customer - Can provide information such as their name, home address, etc., for the payment process Product - Can provide the price and image of the item Seller - Receives payments made for this class and stores them in the sales record

CRC CARDS

Class: Profile	
Responsabilities	Collaborators
- Store the data supplied by the user.	Seller - Supplies the data to the login class for storage. Customer - Supplies the data to the login class for storage.

New Changes

We added a new requirement that involves collecting user data through a user and seller profile. This information is used to process payments, for example, or identify a seller for a customer.

We also added two user stories where the seller can view reports on their sales and payments made by the customer, and the registration and login section for anyone who enters the application. Including new mockups representing the new user stories

This requires creating new mockups of the login and registration interfaces for both the seller and the customer, as well as the sales and payment reporting interface.

Finally, we added two important classes: the profile and payment classes, with their respective responsibilities and collaborators.

How does our application apply the OOP concepts?

Inheritance

Using the UML diagrams, we were able to demonstrate that we can inherit some attributes from the Create Profile and Login classes, such as name, age, and address, to the Customer and Seller classes.

Also, from the Seller class, we can, for example, inherit the Add Product, Assign Price, Description, and Product Name methods to this class.

Encapsulation

For encapsulation, we have examples of attributes that we do not want to be public, but rather private to ensure the security of user data, such as the password, the card number, the security code, and everything related to payment or bank details.

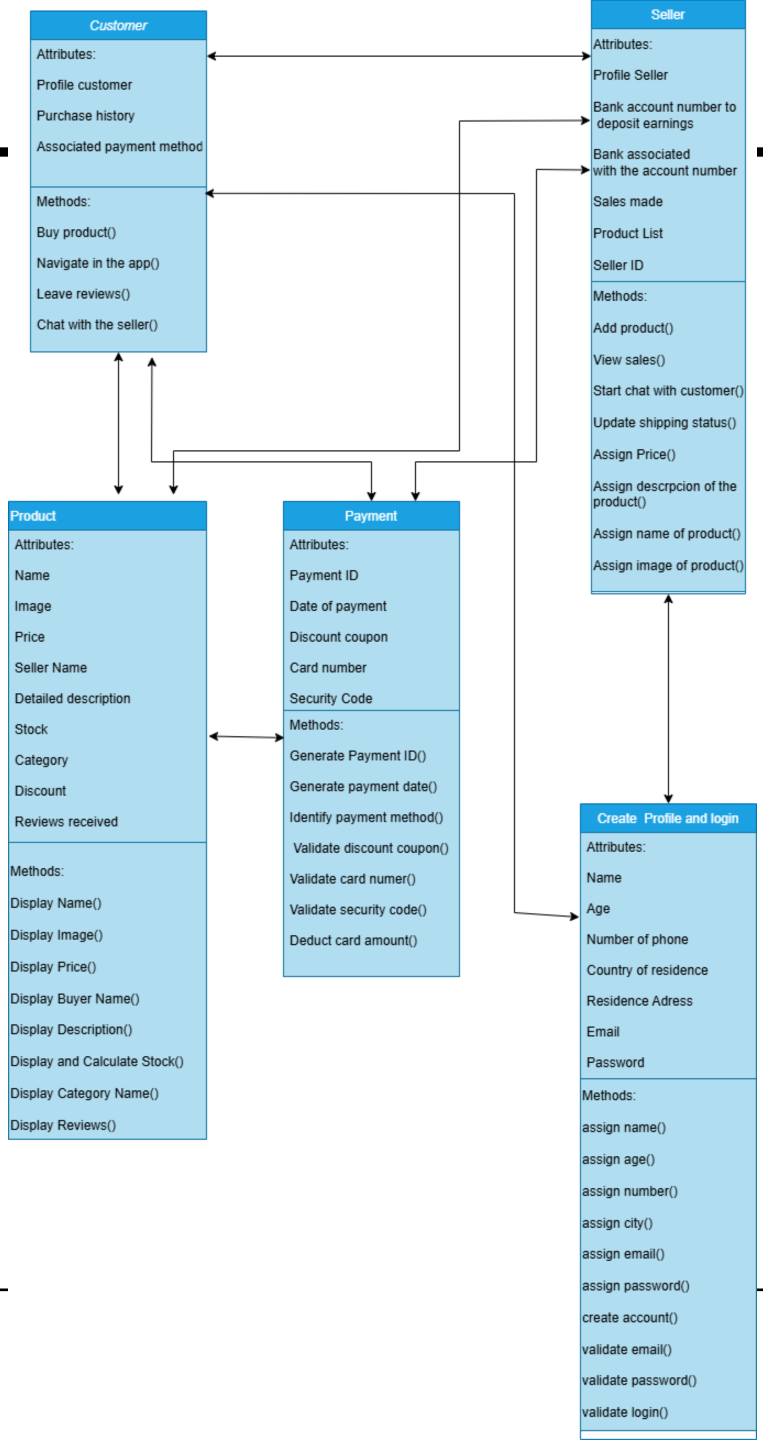
How does our application apply the OOP concepts?

For example, you could request the password with a getter and try not to expose user information and only perform a validation to be able to execute the setters or perform a specific function. Similarly, this can be applied when performing, for example, card discounting or number validation.

Polymorphism

Based on the diagram, we could not find any class to which we could apply either overloading or inheritance polymorphism.

UML DIAGRAM



Customer

Attributes:

Profile customer

Purchase history

Associated payment method

Methods:

Buy product()

Navigate in the app()

Leave reviews()

Chat with the seller()

Seller

Attributes:

Profile Seller

Bank account number to deposit earnings

Bank associated with the account number

Sales made

Product List

Seller ID

Methods:

Add product()

View sales()

Start chat with customer()

Update shipping status()

Assign Price()

Assign descrpcion of the product()

Assign name of product()

Assign image of product()

Product

Attributes:

Name

Image

Price

Seller Name

Detailed description

Stock

Category

Discount

Reviews received

Methods:

Display Name()

Display Image()

Display Price()

Display Buyer Name()

Display Description()

Display and Calculate Stock()

Display Category Name()

Display Reviews()

Payment

Attributes:

Payment ID

Date of payment

Discount coupon

Card number

Security Code

Methods:

Generate Payment ID()

Generate payment date()

Identify payment method()

Validate discount coupon()

Validate card numer()

Validate security code()

Deduct card amount()

Create Profile and login

Attributes:

Name

Age

Number of phone

Country of residence

Residence Adress

Email

Password

Methods:

assign name()

assign age()

assign number()

assign city()

assign email()

assign password()

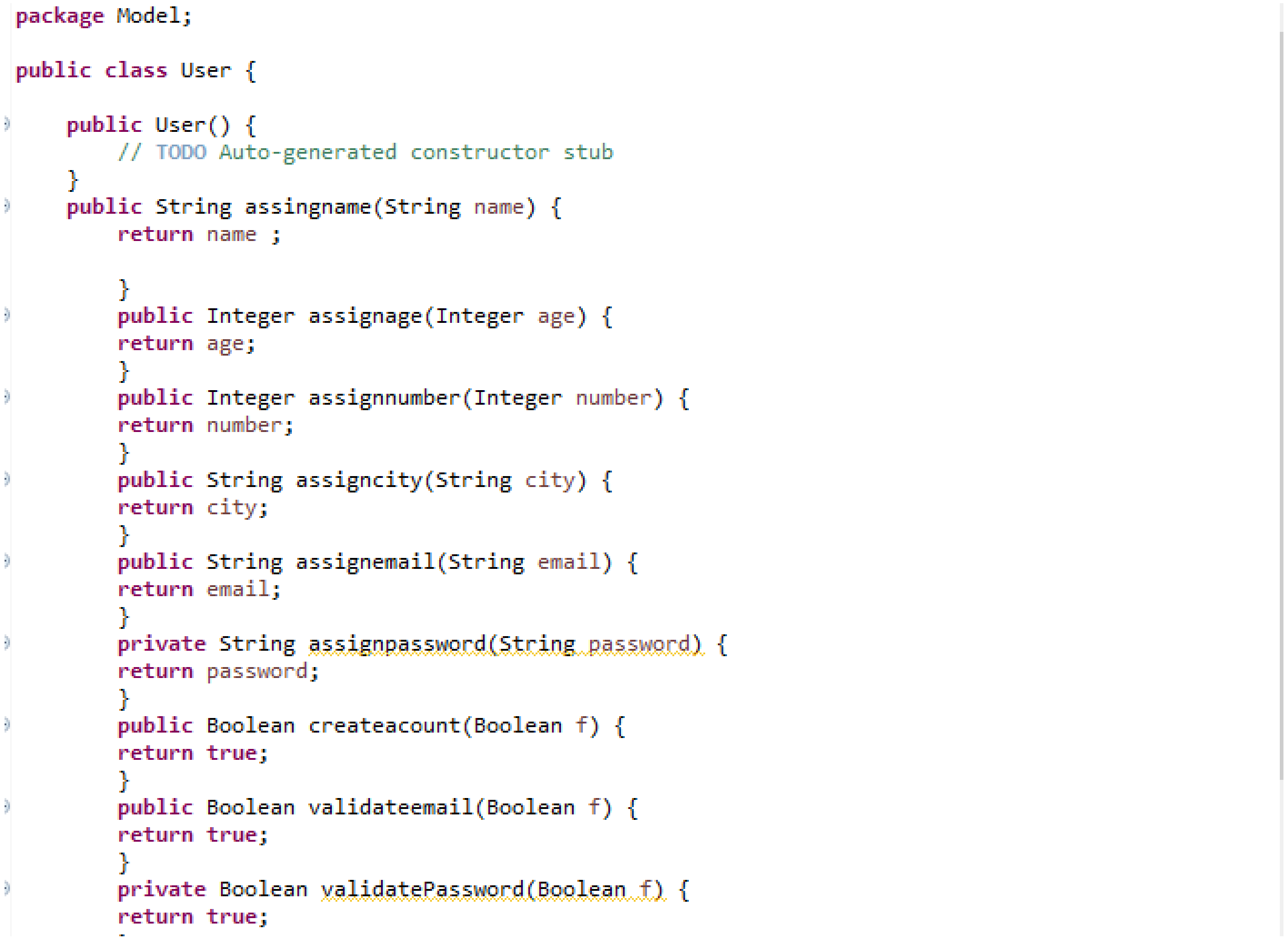
create account()

validate email()

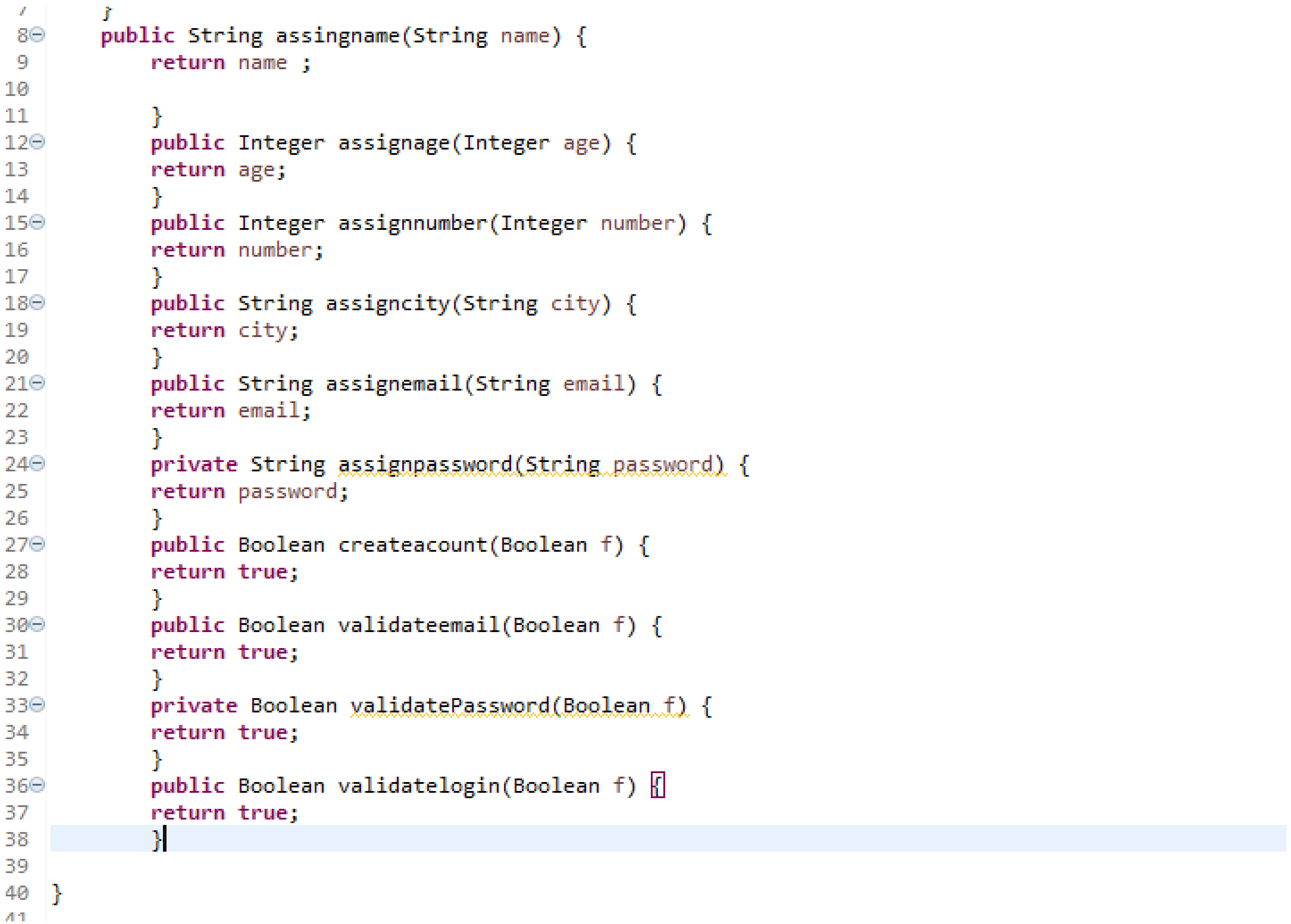
validate password()

validate login()

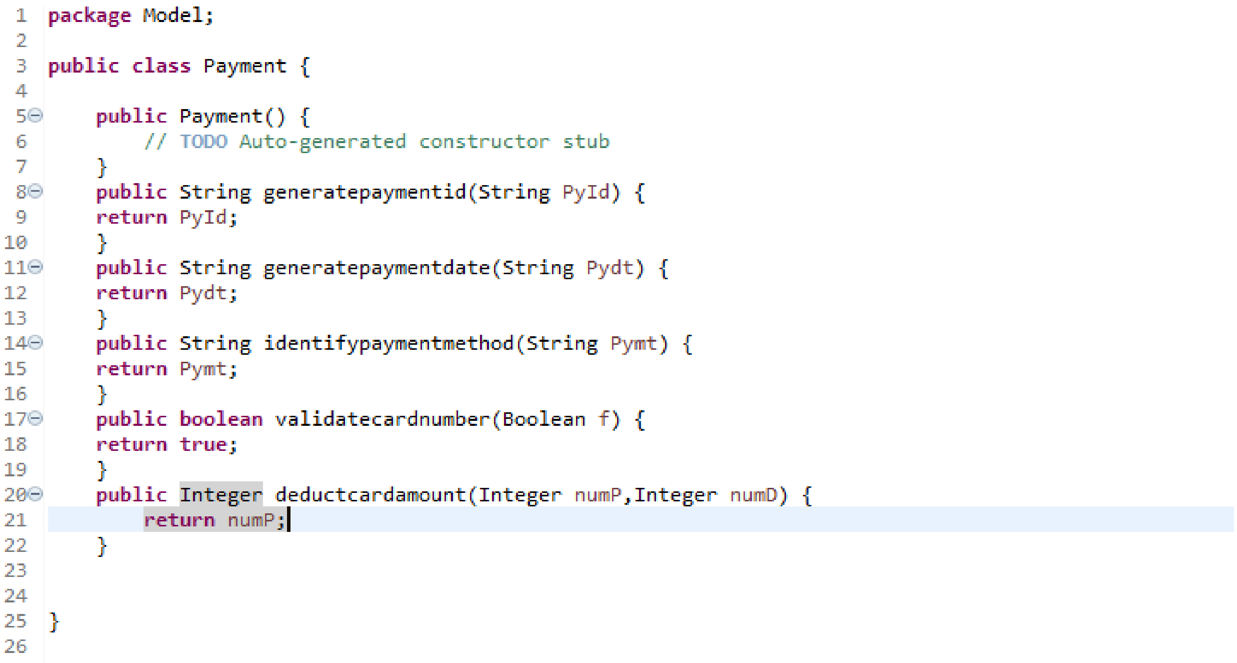
FRAGMENTOS DE CODIGO:



```java
package Model;

public class User {

    public User() {
        // TODO Auto-generated constructor stub
    }
    public String assingname(String name) {
        return name ;

        }
        public Integer assignage(Integer age) {
        return age;
        }
        public Integer assignnumber(Integer number) {
        return number;
        }
        public String assigncity(String city) {
        return city;
        }
        public String assignemail(String email) {
        return email;
        }
        private String assignpassword(String password) {
        return password;
        }
        public Boolean createacount(Boolean f) {
        return true;
        }
        public Boolean validateemail(Boolean f) {
        return true;
        }
        private Boolean validatePassword(Boolean f) {
        return true;
```



```
    }
    public String assingname(String name) {
        return name ;

        }
        public Integer assignage(Integer age) {
        return age;
        }
        public Integer assignnumber(Integer number) {
        return number;
        }
        public String assigncity(String city) {
        return city;
        }
        public String assignemail(String email) {
        return email;
        }
        private String assignpassword(String password) {
        return password;
        }
        public Boolean createacount(Boolean f) {
        return true;
        }
        public Boolean validateemail(Boolean f) {
        return true;
        }
        private Boolean validatePassword(Boolean f) {
        return true;
        }
        public Boolean validatelogin(Boolean f) {
        return true;
        }

}
```



```java
package Model;

public class Payment {

    public Payment() {
        // TODO Auto-generated constructor stub
    }
    public String generatepaymentid(String PyId) {
    return PyId;
    }
    public String generatepaymentdate(String Pydt) {
    return Pydt;
    }
    public String identifypaymentmethod(String Pymt) {
    return Pymt;
    }
    public boolean validatecardnumber(Boolean f) {
    return true;
    }
    public Integer deductcardamount(Integer numP,Integer numD) {
        return numP;
    }


}

```

How do code snippets relate to the UML diagram?

Code snippets are related to UML diagrams since UML diagrams represent the classes and methods that we are going to have at the code level.