“UNIVERSIDADDE LAS FUERZAS ARMADAS” ESPE

SOFTWARE ENGINEER

<u>OOP</u>

Proyect first partial
SMC-STYLE-IRELIA

STUDENTS:
Joel Arguello
Alex Bedón
Luis Burbano
Widinson Caiza

GROUP:
BettaCoders or first group

GUARDIAN IN CHARGE:
PHD. EDISON LASCANO

NRC:
4680

DELIVERY DATE:

SEDE MATRIZ SANGOLQUÍ
QUITO-ECUADOR
2022

SMC Stylist Irelia

For the project we use the Java Language that helps us and facilitates us due to its object-oriented paradigm, with them we use basic concepts such as Inheritance, Polymorphism, Encapsulation, Abstraction that help us above all to reuse code and that the coding is efficient.

Abstraction

Representation through its properties and methods of a reality based on its characteristics.
abstraction allows me to more comfortably organize a model or a real problem by breaking down the global into smaller components and thus making it easier.

Encapsulation

Encapsulation manages to manage the properties and methods that are going to be exposed to the outside, Hiding the desired part of its implementation, this mechanism allows us to make modifications to a class without affecting the rest of the program

Inheritance

The concept is based on creating more detailed objects from other existing models, one or more classes benefit from the properties and methods of an existing class so that instead of starting from scratch, they can simply specialize in certain aspects.
Inheritance is implemented by assigning to the prototype of a class (child class) an instance of the class from which it is desired to inherit (parent class).

Polymorphism

Polymorphism is related to inheritance and is based on the fact that 2 objects of different types can share the same interface so that you can work with that relationship regardless of the type of object being worked on, one of the possibilities that its implementation gives us in JavaScript is counting on so-called subtyped or classic polymorphism. This type of polymorphism allows us to work with an object at a certain level without having to worry about its exact type, thanks to working with it at a higher level (parent class) which guarantees that thanks to inheritance we always have the methods to the ones we call, either the originals in the parent class or the overridden versions in the child classes,

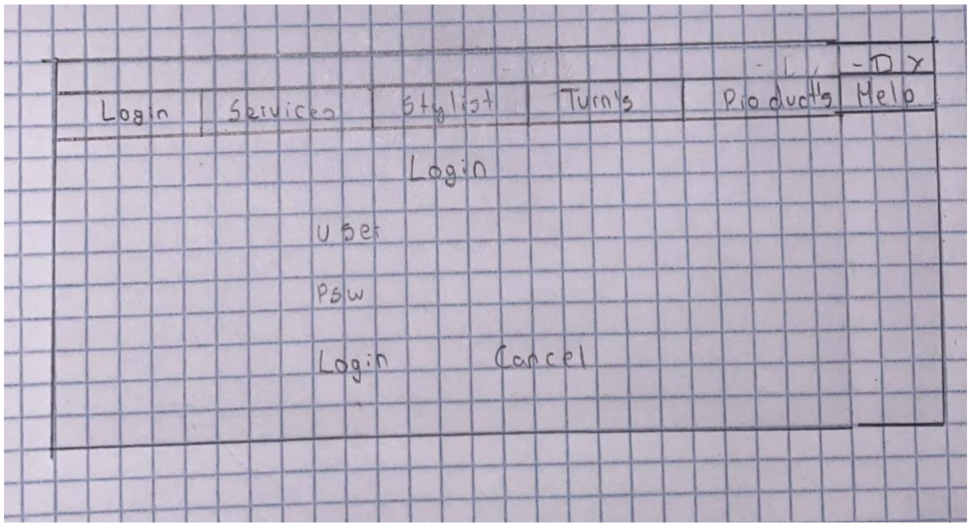
GUI

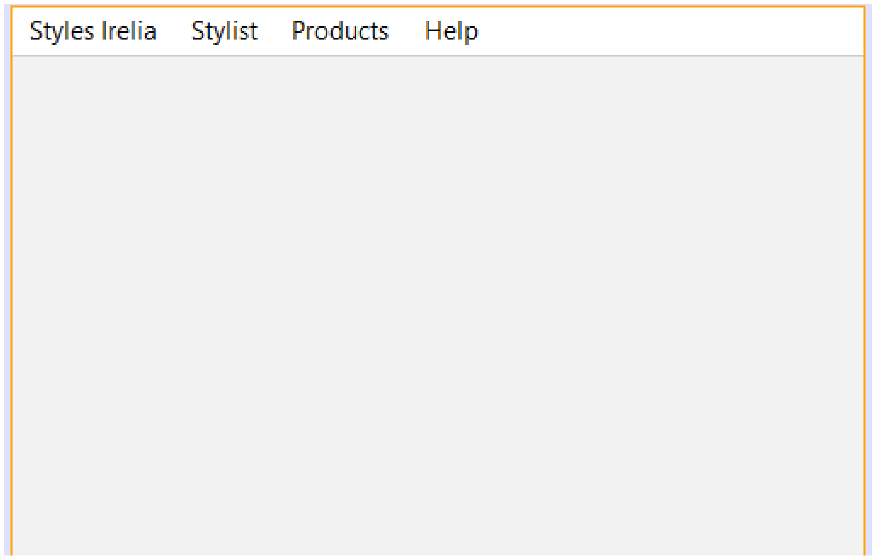
Java Swing is a lightweight graphical user interface (GUI) tool that includes a rich set of widgets. It includes a package that allows you to create GUI components for your Java applications and is platform independent.

For user facilities we can use various GUI options:

Menus:

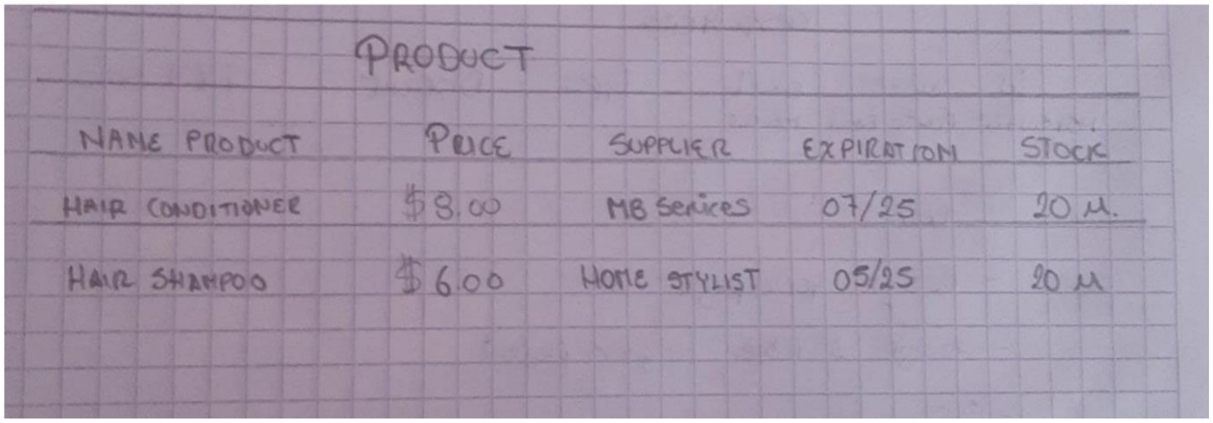
Main Options of your system





TABLES:

Component graphical to show tabulated information a list

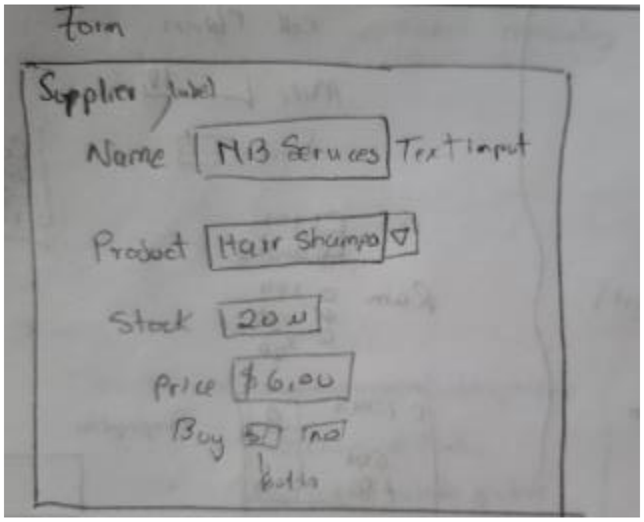


PRODUCT

NAME PRODUCT	PRICE	SUPPLIER	EXPIRATION	STOCK
HAIR CONDITIONER	$8.00	MB Serices	07/25	20 u.
HAIR SHAMPOO	$6.00	HOME STYLIST	05/25	20 u

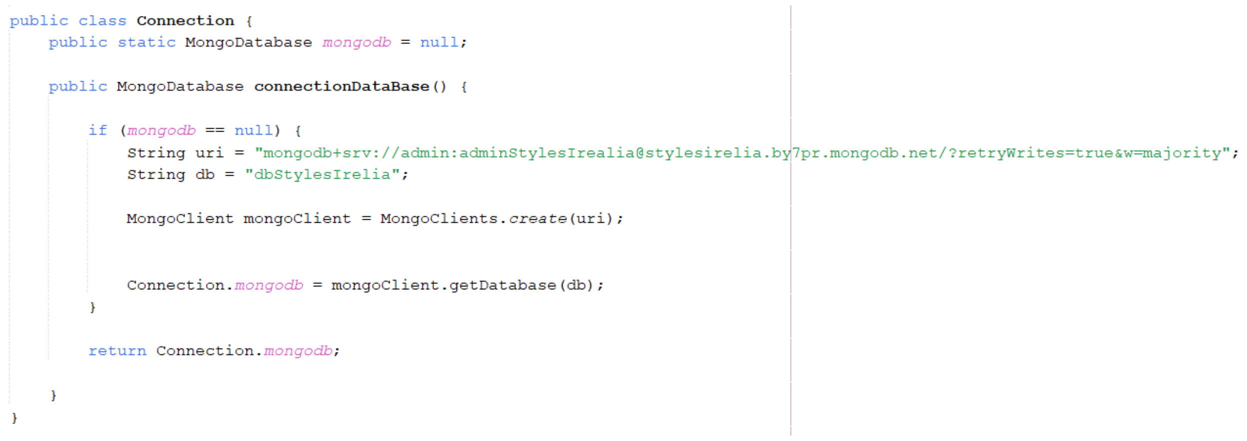
User input validation:

Control what user enters before it is processed and saved in the database



MongoDB

CONNECTION WITH THE DATABASE:



```
public class Connection {
    public static MongoDatabase mongodb = null;

    public MongoDatabase connectionDataBase() {

        if (mongodb == null) {
            String uri = "mongodb+srv://admin:adminStylesIrealia@stylesirelia.by7pr.mongodb.net/?retryWrites=true&w=majority";
            String db = "dbStylesIrelia";

            MongoClient mongoClient = MongoClients.create(uri);


            Connection.mongodb = mongoClient.getDatabase(db);
        }

        return Connection.mongodb;

    }
}
```

Insert a Single Document

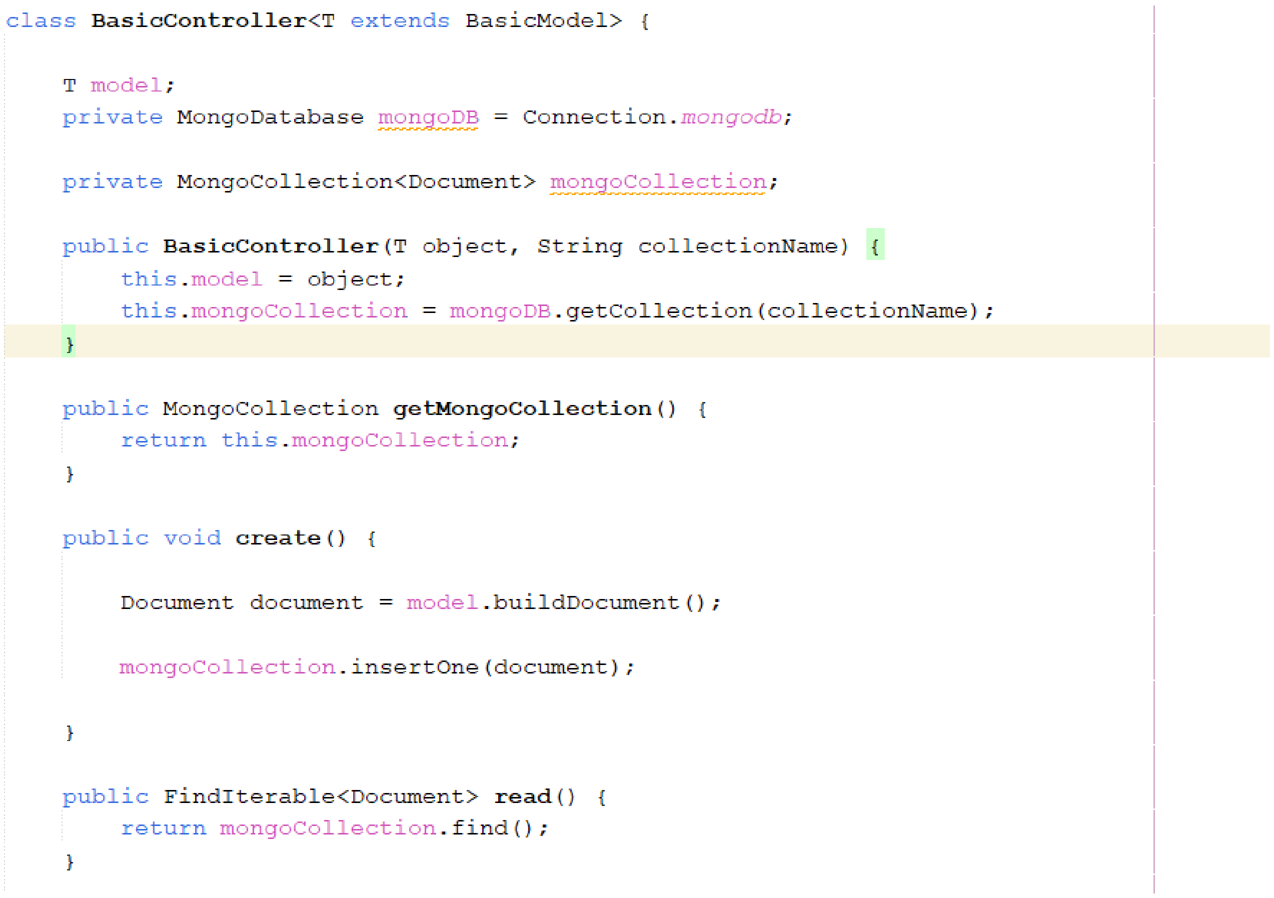
db.collection.insertOne() inserts a single document into a collection.

Update a document

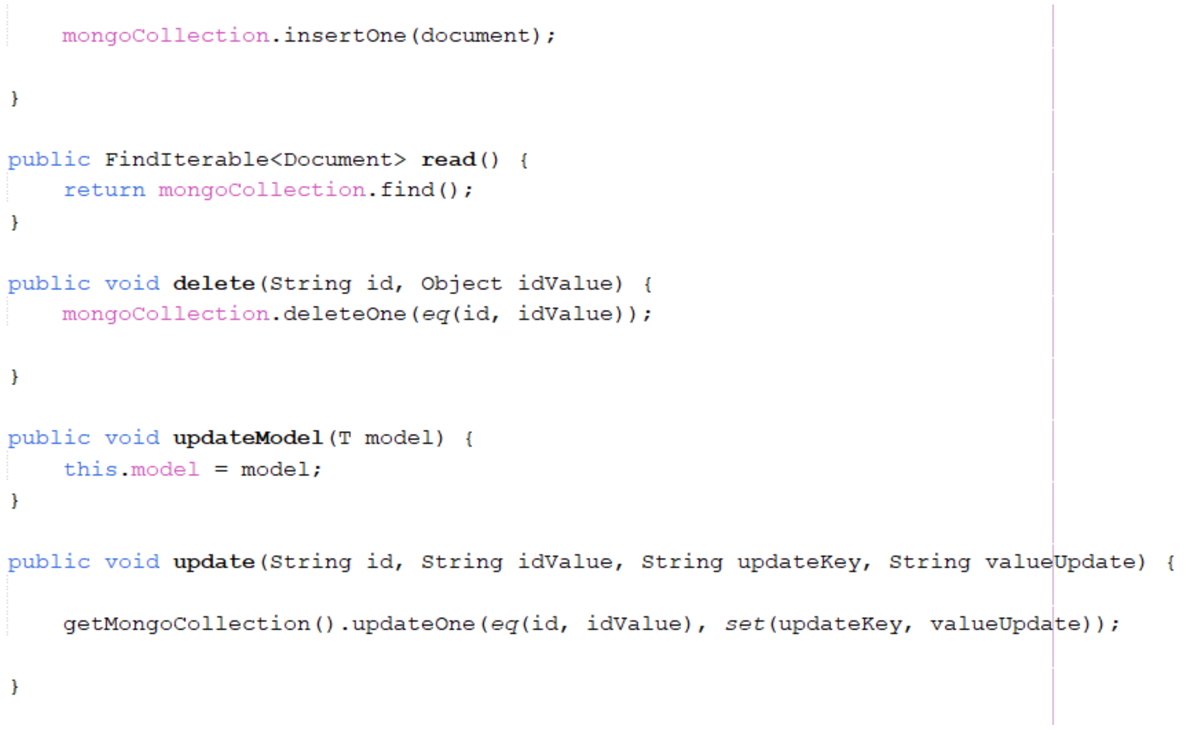
MongoDB provides update operators, such as $set, to modify field values.

Delete Documents

This page uses the following mongosh methods:
db.collection.deleteMany()
db.collection.deleteOne()



```
class BasicController<T extends BasicModel> {

    T model;
    private MongoDatabase mongoDB = Connection.mongodb;

    private MongoCollection<Document> mongoCollection;

    public BasicController(T object, String collectionName) {
        this.model = object;
        this.mongoCollection = mongoDB.getCollection(collectionName);
    }

    public MongoCollection getMongoCollection() {
        return this.mongoCollection;
    }

    public void create() {

        Document document = model.buildDocument();

        mongoCollection.insertOne(document);

    }

    public FindIterable<Document> read() {
        return mongoCollection.find();
    }
```



```
        mongoCollection.insertOne(document);

    }

    public FindIterable<Document> read() {
        return mongoCollection.find();
    }

    public void delete(String id, Object idValue) {
        mongoCollection.deleteOne(eq(id, idValue));

    }

    public void updateModel(T model) {
        this.model = model;
    }

    public void update(String id, String idValue, String updateKey, String valueUpdate) {

        getMongoCollection().updateOne(eq(id, idValue), set(updateKey, valueUpdate));

    }
```