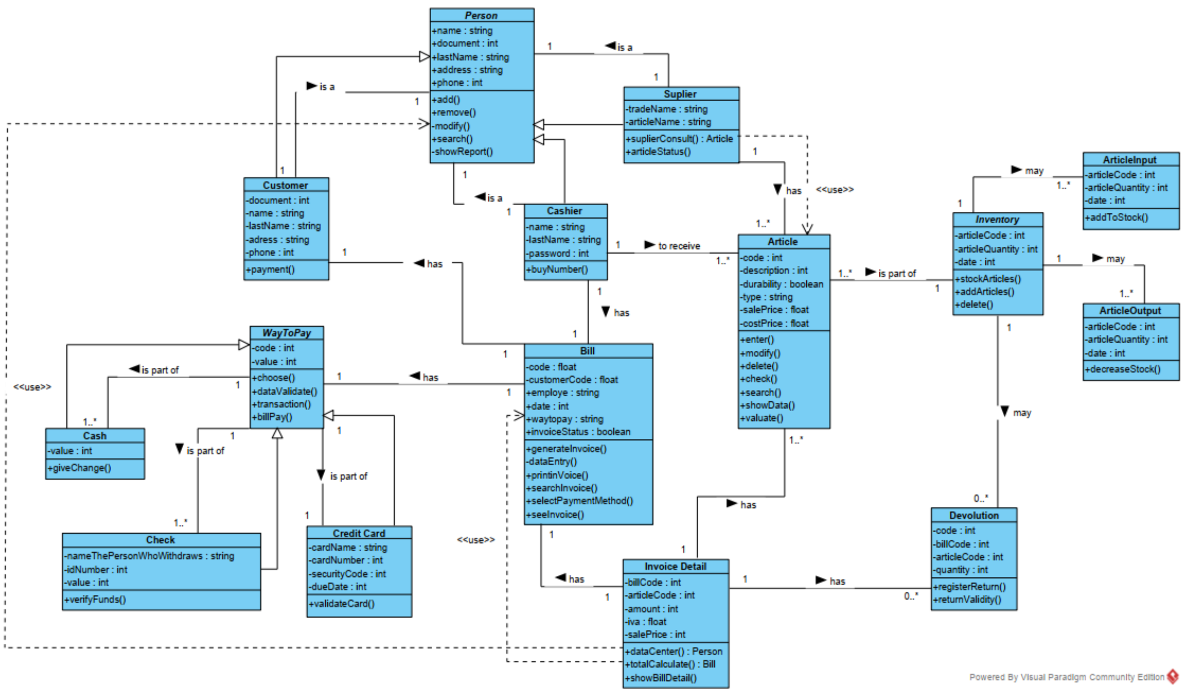
Inspector Group ---> GROUP 6 DevsJavaKND
Inspected Group ---> GROUP 1 CrSystem

1. Class diagram	4/5
2. Use case diagram	4,8/5
3. MVC pattern applied	
a. Packages (controller) and libraries/utils	**5/5**
b. Clean GUI (no logic in this layer)	**2,7/5**
c. Model (only POJO files)	**4,7/5**
4. Inheritance	**3,7/5**
5. Abstraction (abstract classes or interfaces)	**3,7/5**
6. Polymorphism	**4,3/5**
7. MongoDB	**4,85**
8. Unit tests (at least 100 cases)	**0/5**
9. GUI:	

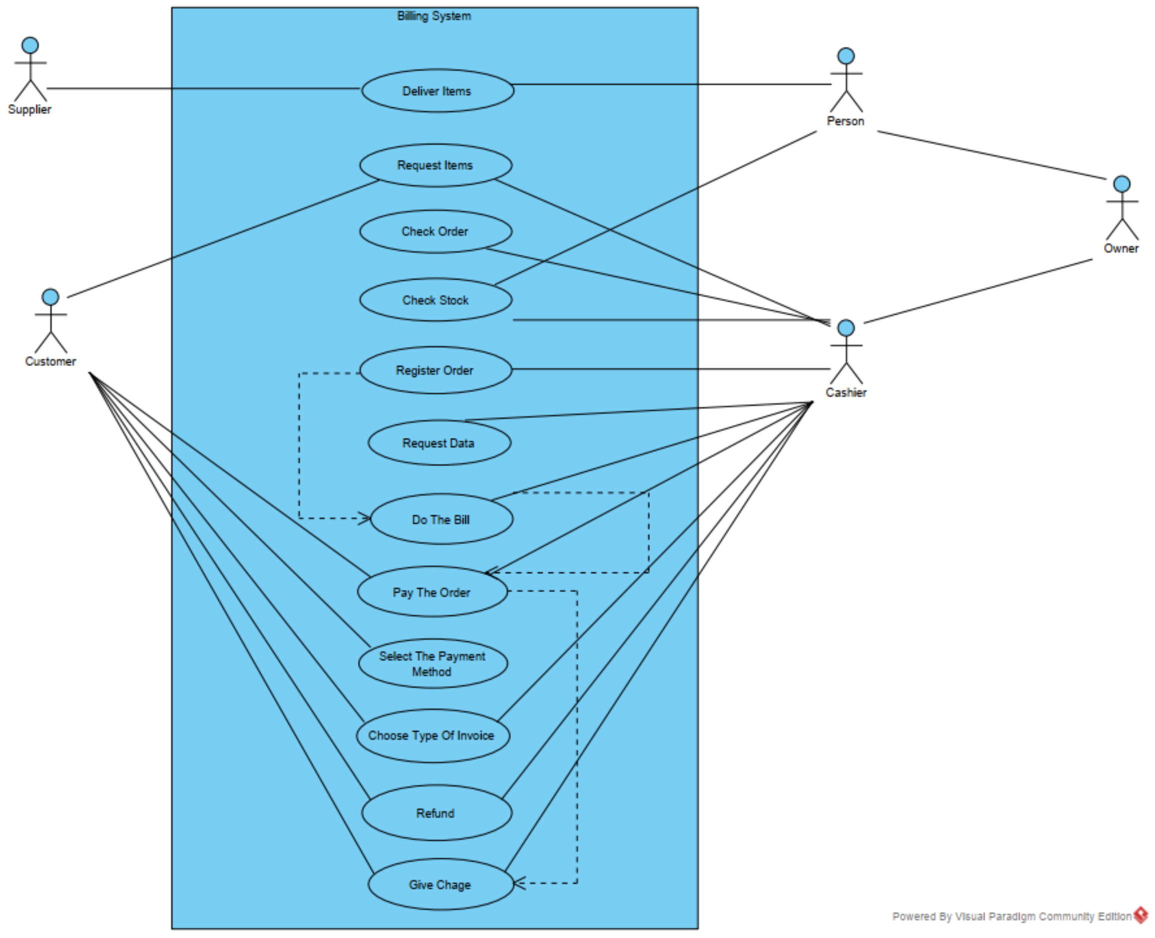
a.	login screen	9,5/10
b.	main menu	8/10
c.	forms	10/10
d.	use of tables (printing)	5,1/10
e.	Functionality	0/10
TOTAL		70.6/100
GITHUB		/100
GRADE (TOTAL+GITHUB)/2		/100

1. Class Diagram



- Some classes need to be in a package.model , package.controller. , package.view, package.utils.

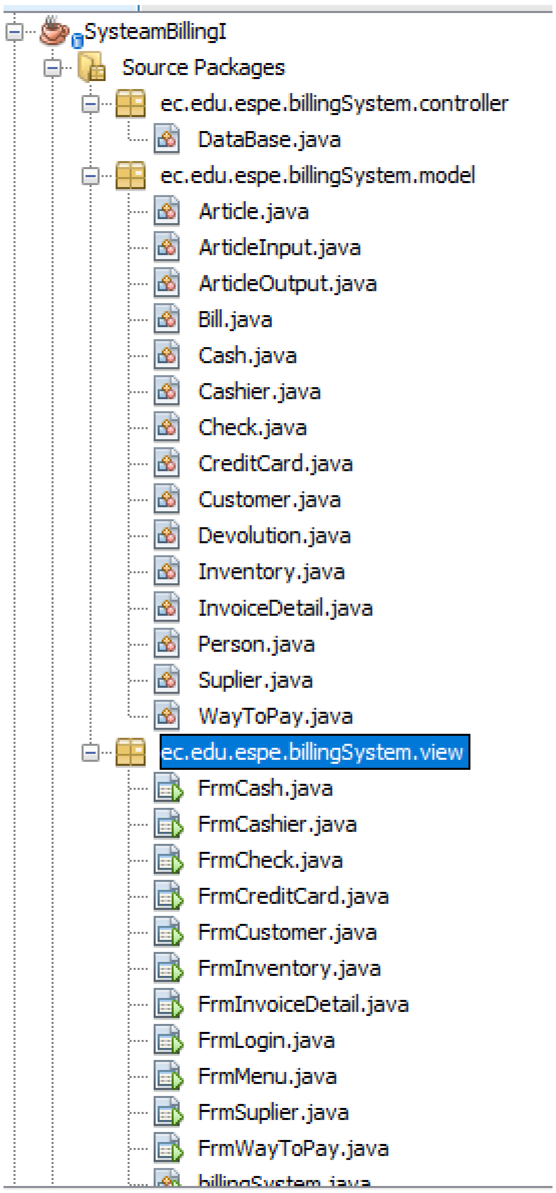
2. Use Case Diagram



- It is recommended to update the use cases and their relationship between authors(-0.2)

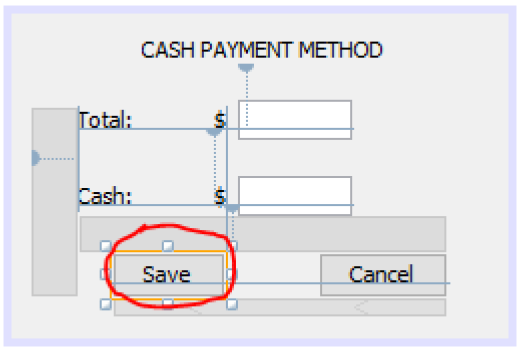
3. MVC pattern applied

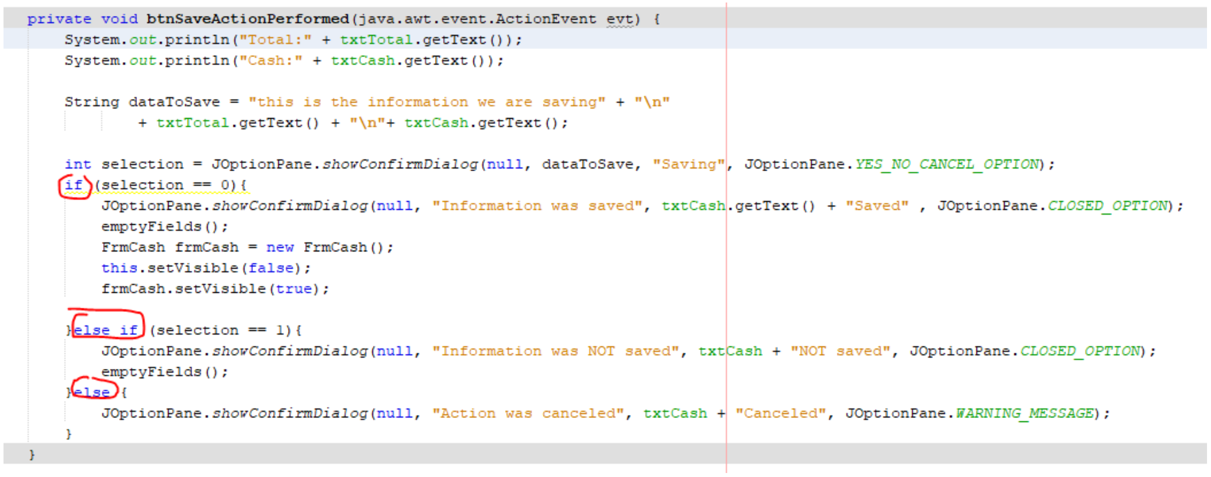
a. No interfaces have been implemented in the utils package view



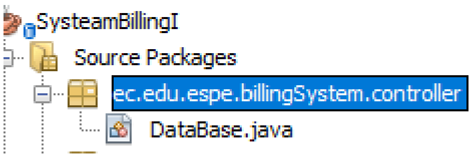
```
SysteamBillingI
  Source Packages
    ec.edu.espe.billingSystem.controller
      DataBase.java
    ec.edu.espe.billingSystem.model
      Article.java
      ArticleInput.java
      ArticleOutput.java
      Bill.java
      Cash.java
      Cashier.java
      Check.java
      CreditCard.java
      Customer.java
      Devolution.java
      Inventory.java
      InvoiceDetail.java
      Person.java
      Suplier.java
      WayToPay.java
    ec.edu.espe.billingSystem.view
      FrmCash.java
      FrmCashier.java
      FrmCheck.java
      FrmCreditCard.java
      FrmCustomer.java
      FrmInventory.java
      FrmInvoiceDetail.java
      FrmLogin.java
      FrmMenu.java
      FrmSuplier.java
      FrmWayToPay.java
      billingSystem.java
```

b. They are not fully utilizing the model view controller architecture. Since the GUI should have clean code and the forms should be controlled from a controller class in the controller package





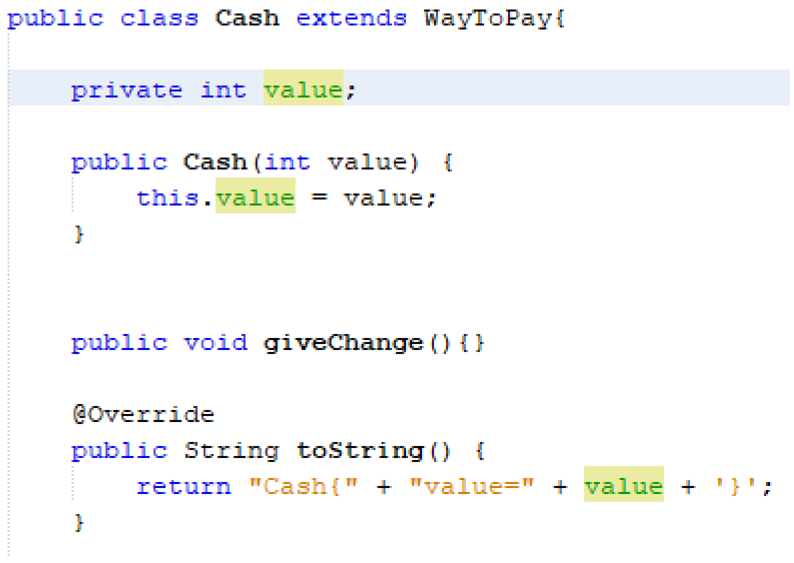
```java
private void btnSaveActionPerformed(java.awt.event.ActionEvent evt) {
    System.out.println("Total:" + txtTotal.getText());
    System.out.println("Cash:" + txtCash.getText());

    String dataToSave = "this is the information we are saving" + "\n"
            + txtTotal.getText() + "\n"+ txtCash.getText();

    int selection = JOptionPane.showConfirmDialog(null, dataToSave, "Saving", JOptionPane.YES_NO_CANCEL_OPTION);
    if(selection == 0){
        JOptionPane.showConfirmDialog(null, "Information was saved", txtCash.getText() + "Saved" , JOptionPane.CLOSED_OPTION);
        emptyFields();
        FrmCash frmCash = new FrmCash();
        this.setVisible(false);
        frmCash.setVisible(true);

    }else if (selection == 1){
        JOptionPane.showConfirmDialog(null, "Information was NOT saved", txtCash + "NOT saved", JOptionPane.CLOSED_OPTION);
        emptyFields();
    }else{
        JOptionPane.showConfirmDialog(null, "Action was canceled", txtCash + "Canceled", JOptionPane.WARNING_MESSAGE);
    }
}
```



c. Correctly.

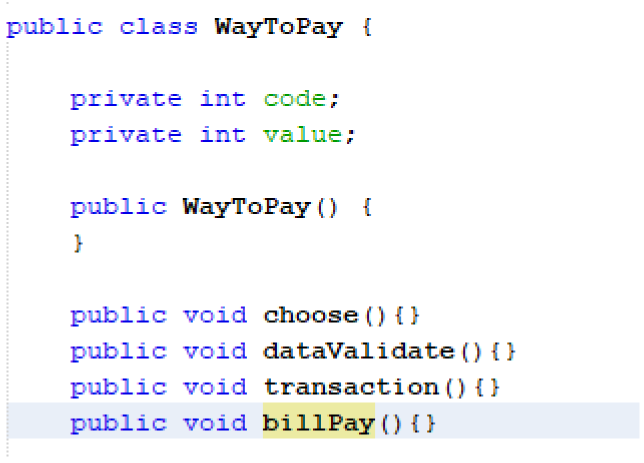
4. Inheritance

The "Cash" class is a child of the "WayToPay" class, however, it does not use any of its attributes or methods.(inheritage)



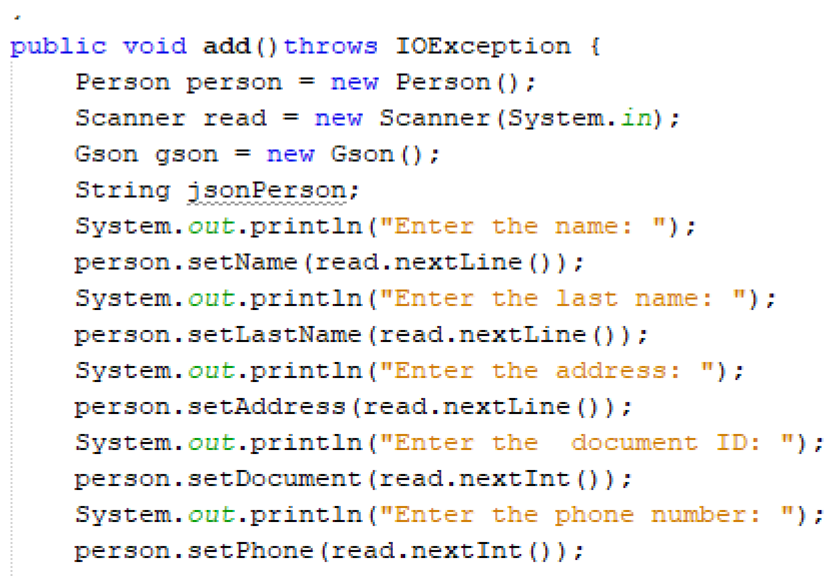
```java
public class Cash extends WayToPay{

    private int value;

    public Cash(int value) {
        this.value = value;
    }


    public void giveChange(){}

    @Override
    public String toString() {
        return "Cash{" + "value=" + value + '}';
    }
```

- There are still undefined methods that are not part of an interface

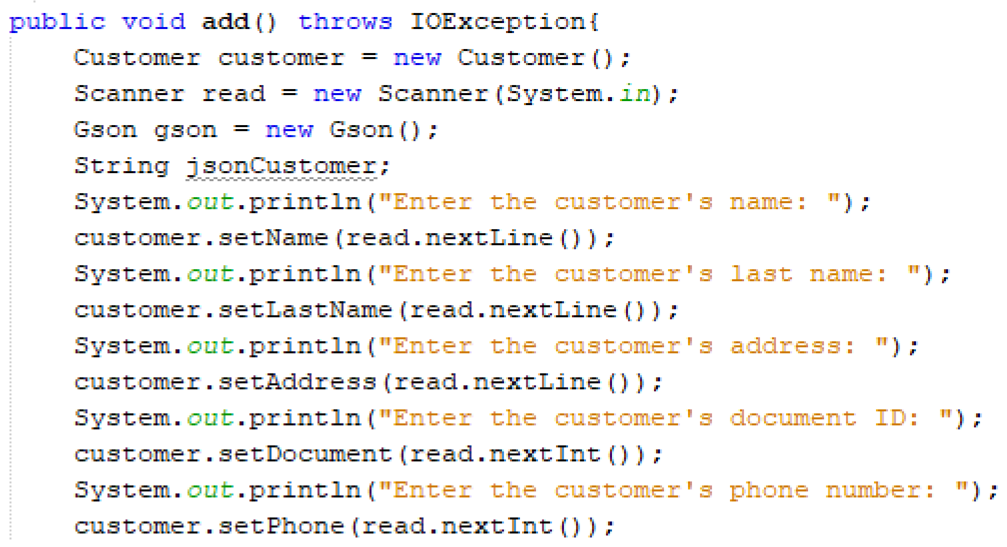


```java
public class WayToPay {

    private int code;
    private int value;

    public WayToPay() {
    }

    public void choose(){}
    public void dataValidate(){}
    public void transaction(){}
    public void billPay(){}
```

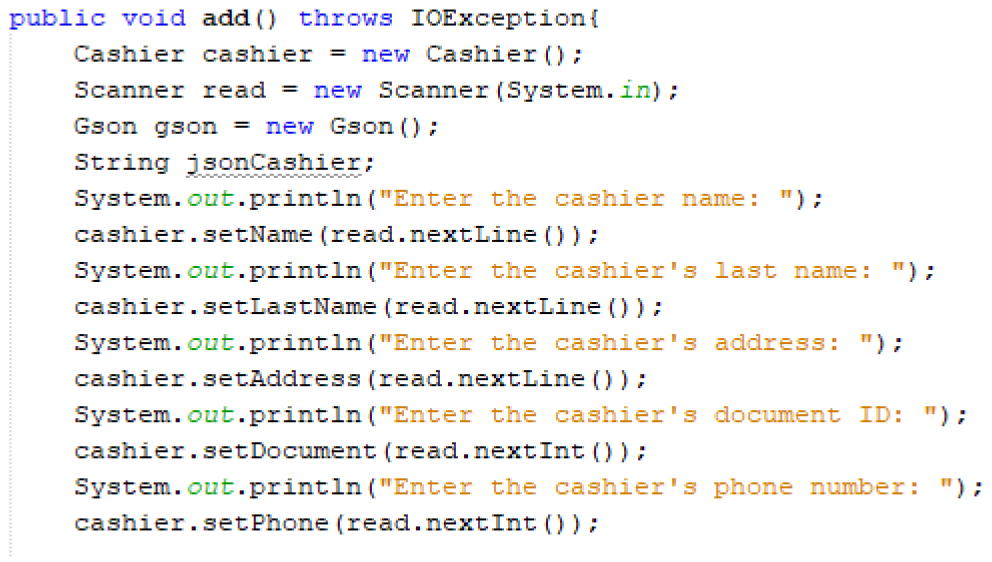
- Data entry can be transformed into an interface so as not to repeat the same code in different classes



```
public void add()throws IOException {
    Person person = new Person();
    Scanner read = new Scanner(System.in);
    Gson gson = new Gson();
    String jsonPerson;
    System.out.println("Enter the name: ");
    person.setName(read.nextLine());
    System.out.println("Enter the last name: ");
    person.setLastName(read.nextLine());
    System.out.println("Enter the address: ");
    person.setAddress(read.nextLine());
    System.out.println("Enter the  document ID: ");
    person.setDocument(read.nextInt());
    System.out.println("Enter the phone number: ");
    person.setPhone(read.nextInt());
```



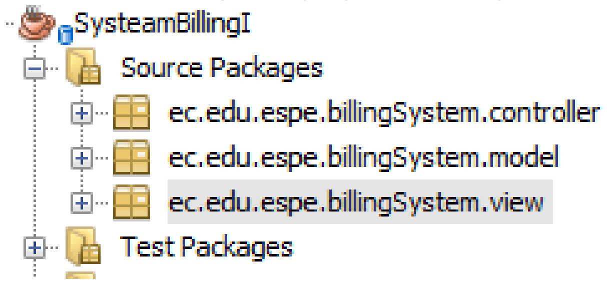
```
public void add() throws IOException{
    Customer customer = new Customer();
    Scanner read = new Scanner(System.in);
    Gson gson = new Gson();
    String jsonCustomer;
    System.out.println("Enter the customer's name: ");
    customer.setName(read.nextLine());
    System.out.println("Enter the customer's last name: ");
    customer.setLastName(read.nextLine());
    System.out.println("Enter the customer's address: ");
    customer.setAddress(read.nextLine());
    System.out.println("Enter the customer's document ID: ");
    customer.setDocument(read.nextInt());
    System.out.println("Enter the customer's phone number: ");
    customer.setPhone(read.nextInt());
```



```
public void add() throws IOException{
    Cashier cashier = new Cashier();
    Scanner read = new Scanner(System.in);
    Gson gson = new Gson();
    String jsonCashier;
    System.out.println("Enter the cashier name: ");
    cashier.setName(read.nextLine());
    System.out.println("Enter the cashier's last name: ");
    cashier.setLastName(read.nextLine());
    System.out.println("Enter the cashier's address: ");
    cashier.setAddress(read.nextLine());
    System.out.println("Enter the cashier's document ID: ");
    cashier.setDocument(read.nextInt());
    System.out.println("Enter the cashier's phone number: ");
    cashier.setPhone(read.nextInt());
```

5. Abstraction

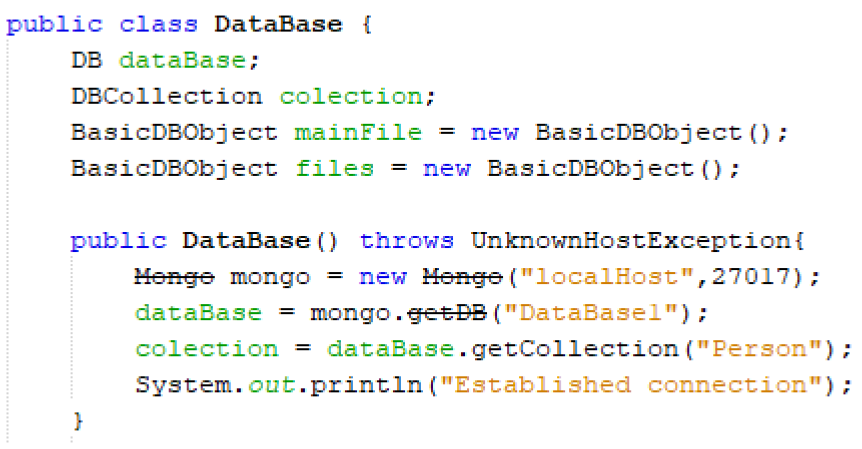
There is no utils package (abstraccion)



```
SysteamBillingI
  Source Packages
    ec.edu.espe.billingSystem.controller
    ec.edu.espe.billingSystem.model
    ec.edu.espe.billingSystem.view
  Test Packages
```

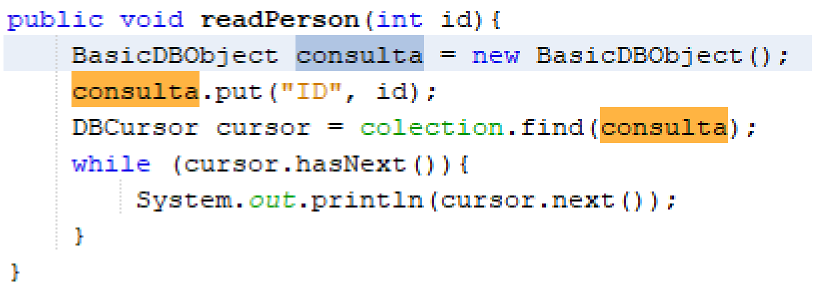
7. MongoDB (Observation)

- The connection to the mongo DB is local, when it must be in the cloud.(mongoDB).



```
public class DataBase {
    DB dataBase;
    DBCollection colection;
    BasicDBObject mainFile = new BasicDBObject();
    BasicDBObject files = new BasicDBObject();

    public DataBase() throws UnknownHostException{
        ~~Mongo~~ mongo = new ~~Mongo~~("localHost",27017);
        dataBase = mongo.~~getDB~~("DataBase1");
        colection = dataBase.getCollection("Person");
        System.out.println("Established connection");
    }
```

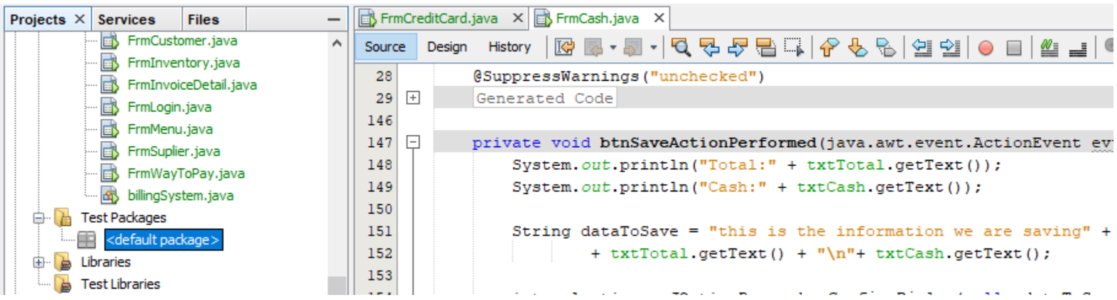
- In the DataBase class there is an attribute called "consulta" that must be in English. (0.05)



```
public void readPerson(int id){
    BasicDBObject consulta = new BasicDBObject();
    consulta.put("ID", id);
    DBCursor cursor = colection.find(consulta);
    while (cursor.hasNext()){
        System.out.println(cursor.next());
    }
}
```

8. Unit test

- They do not have unit test (unit test)



Projects | Services | Files

FrmCustomer.java
FrmInventory.java
FrmInvoiceDetail.java
FrmLogin.java
FrmMenu.java
FrmSuplier.java
FrmWayToPay.java
billingSystem.java
Test Packages
<default package>
Libraries
Test Libraries

FrmCreditCard.java | FrmCash.java

Source | Design | History

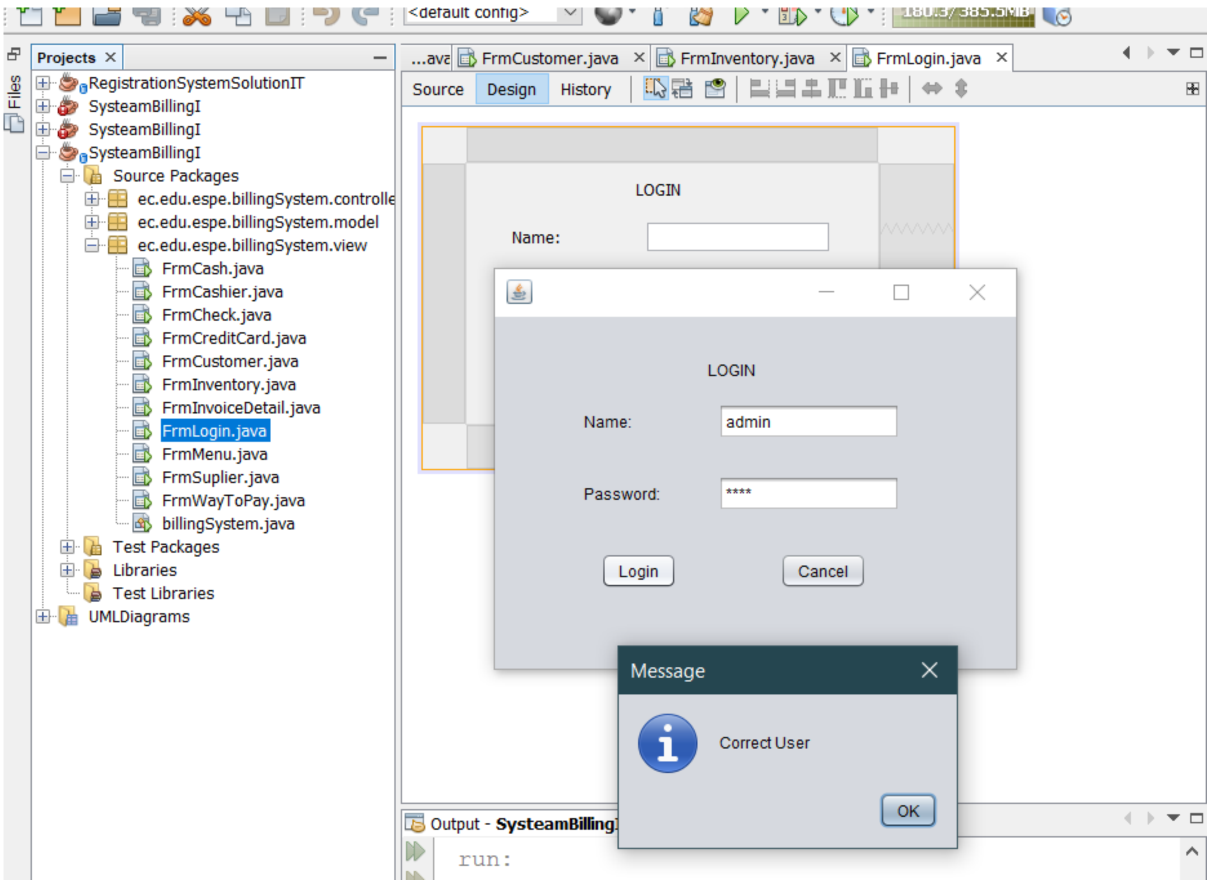
```
    @SuppressWarnings("unchecked")
    Generated Code

    private void btnSaveActionPerformed(java.awt.event.ActionEvent ev
        System.out.println("Total:" + txtTotal.getText());
        System.out.println("Cash:" + txtCash.getText());

        String dataToSave = "this is the information we are saving" +
                + txtTotal.getText() + "\n"+ txtCash.getText();
```

9. GUI

- It does not maintain the mvc architecture, it is required that the buttons do not encode any logic part. Does not comply with the clean code

a. **Login: User = admin , Password = 1234**

Correct User and Password:



Projects

RegistrationSystemSolutionIT
SysteamBillingI
SysteamBillingI
SysteamBillingI
Source Packages
ec.edu.espe.billingSystem.controlle
ec.edu.espe.billingSystem.model
ec.edu.espe.billingSystem.view
FrmCash.java
FrmCashier.java
FrmCheck.java
FrmCreditCard.java
FrmCustomer.java
FrmInventory.java
FrmInvoiceDetail.java
FrmLogin.java
FrmMenu.java
FrmSuplier.java
FrmWayToPay.java
billingSystem.java
Test Packages
Libraries
Test Libraries
UMLDiagrams

...ava | FrmCustomer.java | FrmInventory.java | FrmLogin.java

Source | Design | History

LOGIN

Name:

LOGIN

Name: admin

Password: ****

Login | Cancel

Message

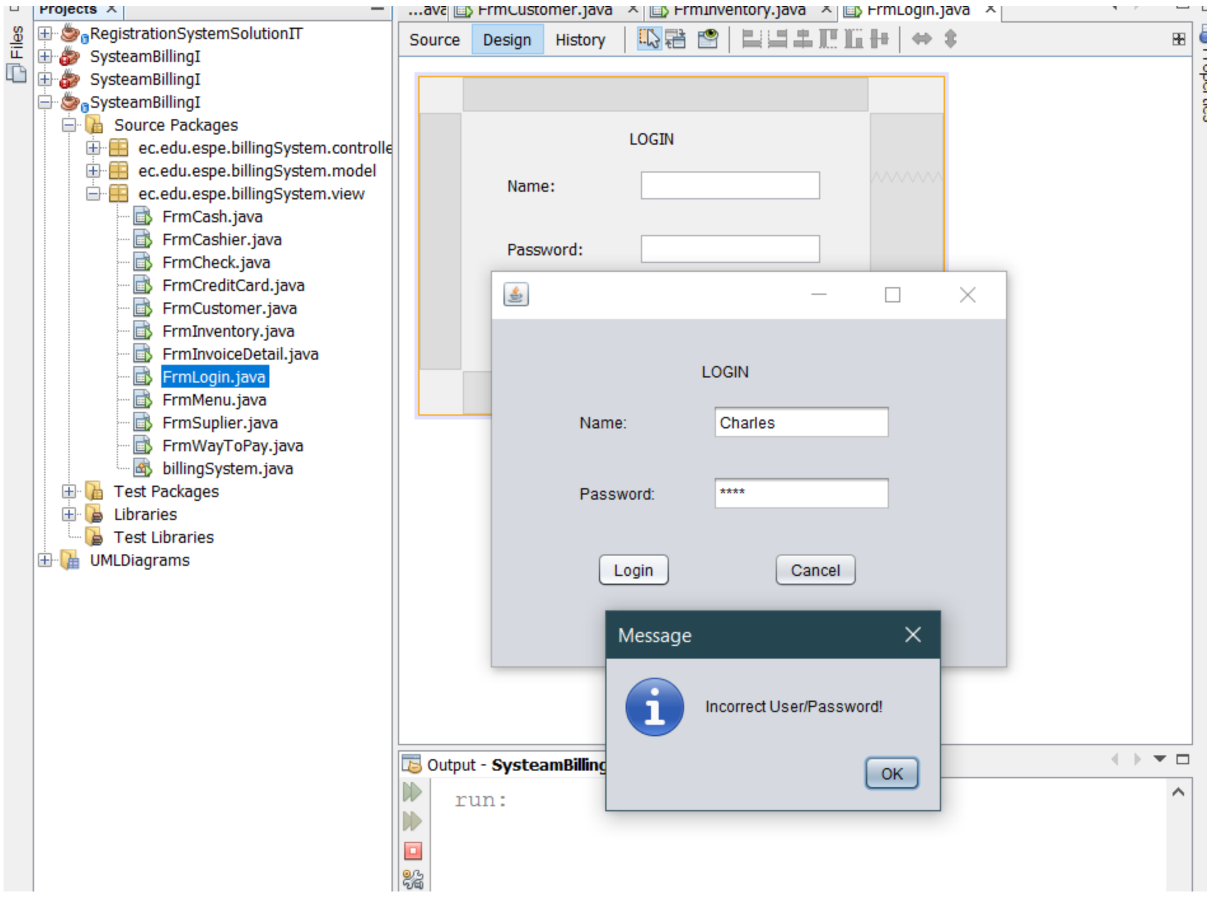
Correct User

OK

Output - SysteamBilling

```
run:
```

- Incorrect User and Correct Password:



```
RegistrationSystemSolutionIT
SysteamBillingI
SysteamBillingI
SysteamBillingI
  Source Packages
    ec.edu.espe.billingSystem.controlle
    ec.edu.espe.billingSystem.model
    ec.edu.espe.billingSystem.view
      FrmCash.java
      FrmCashier.java
      FrmCheck.java
      FrmCreditCard.java
      FrmCustomer.java
      FrmInventory.java
      FrmInvoiceDetail.java
      FrmLogin.java
      FrmMenu.java
      FrmSuplier.java
      FrmWayToPay.java
      billingSystem.java
  Test Packages
  Libraries
  Test Libraries
UMLDiagrams
```

LOGIN

Name: Charles

Password: ****

Login Cancel

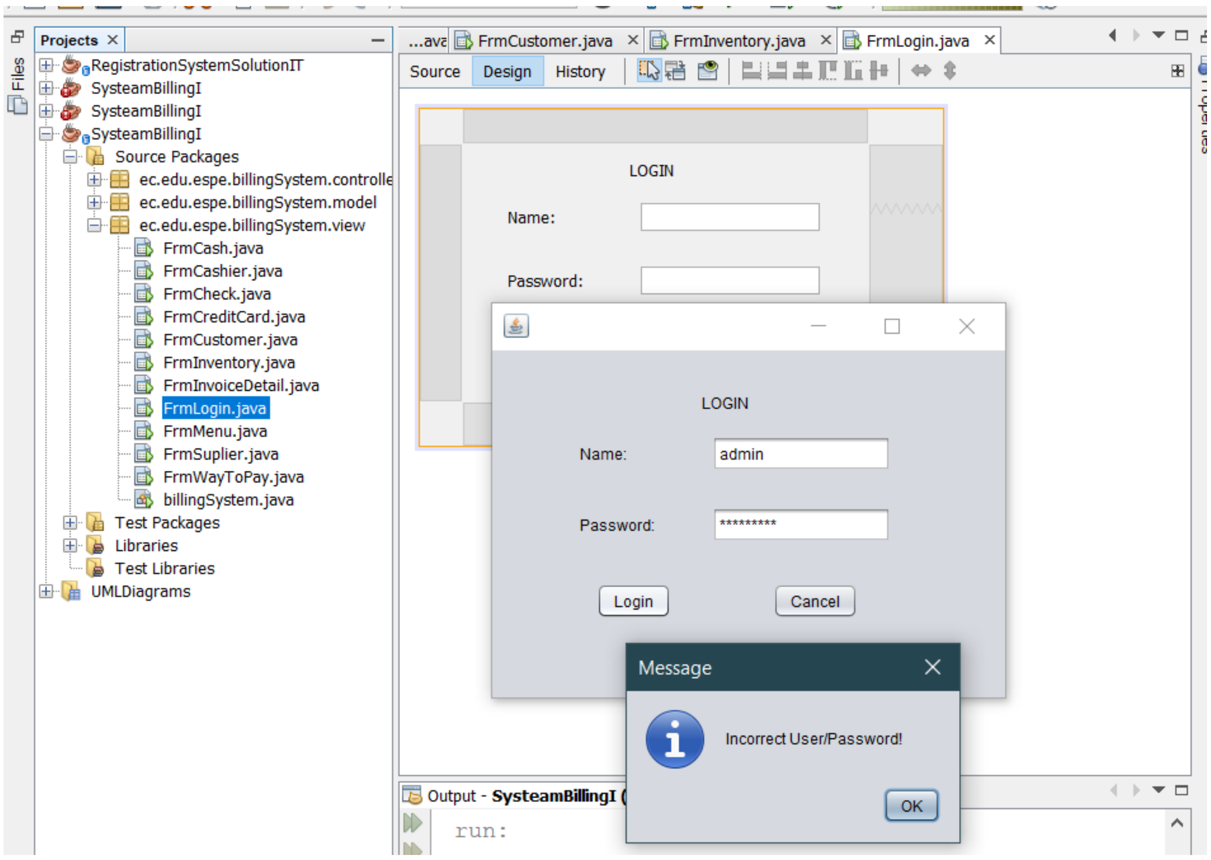
Message

Incorrect User/Password!

OK

run:

- Correct User and Incorrect Password



```
RegistrationSystemSolutionIT
SysteamBillingI
SysteamBillingI
SysteamBillingI
  Source Packages
    ec.edu.espe.billingSystem.controlle
    ec.edu.espe.billingSystem.model
    ec.edu.espe.billingSystem.view
      FrmCash.java
      FrmCashier.java
      FrmCheck.java
      FrmCreditCard.java
      FrmCustomer.java
      FrmInventory.java
      FrmInvoiceDetail.java
      FrmLogin.java
      FrmMenu.java
      FrmSuplier.java
      FrmWayToPay.java
      billingSystem.java
  Test Packages
  Libraries
  Test Libraries
UMLDiagrams
```

LOGIN

Name: admin

Password: *********

Login Cancel

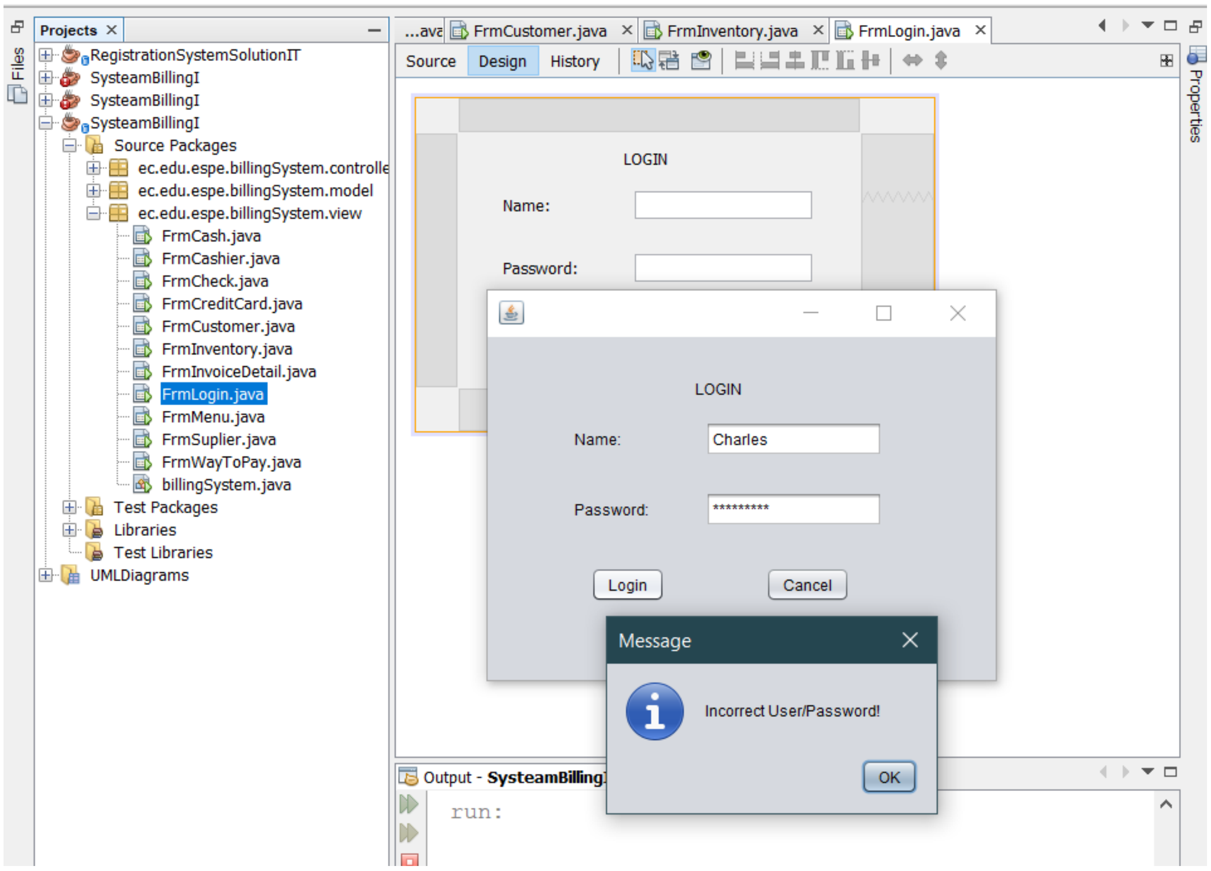
Message

Incorrect User/Password!

OK

run:

- Incorrect User and Incorrect Password:



Name: Charles

Password: *********

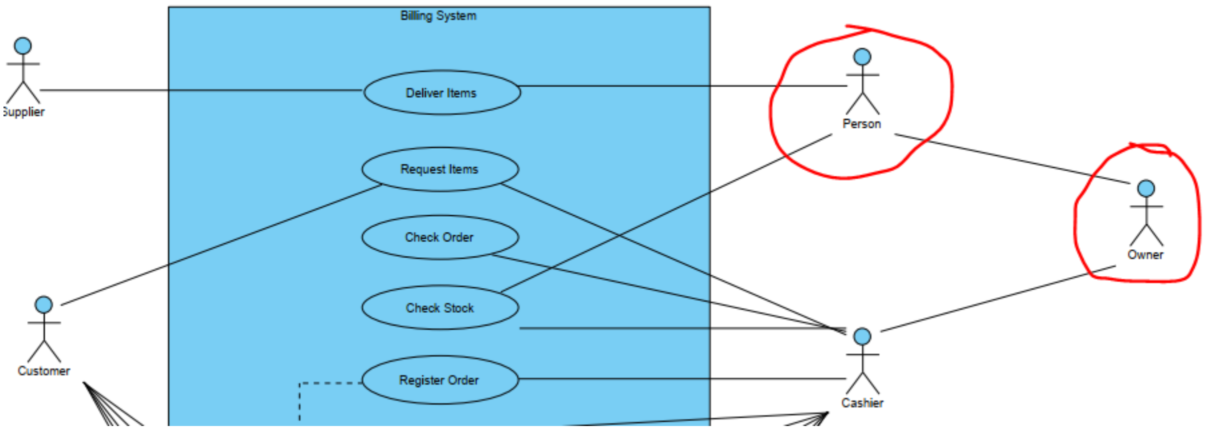
Message

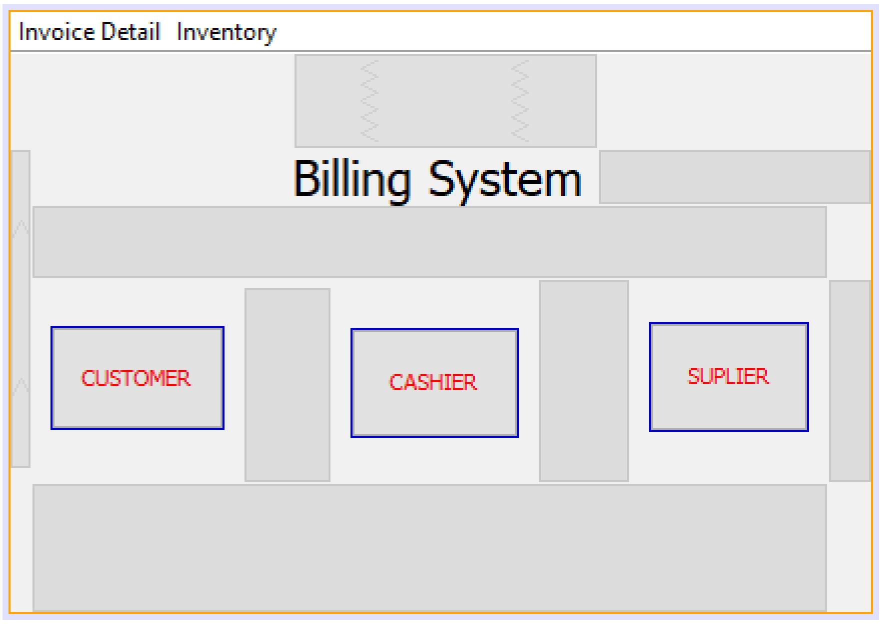
Incorrect User/Password!

```
run:
```

b. Main Menu:

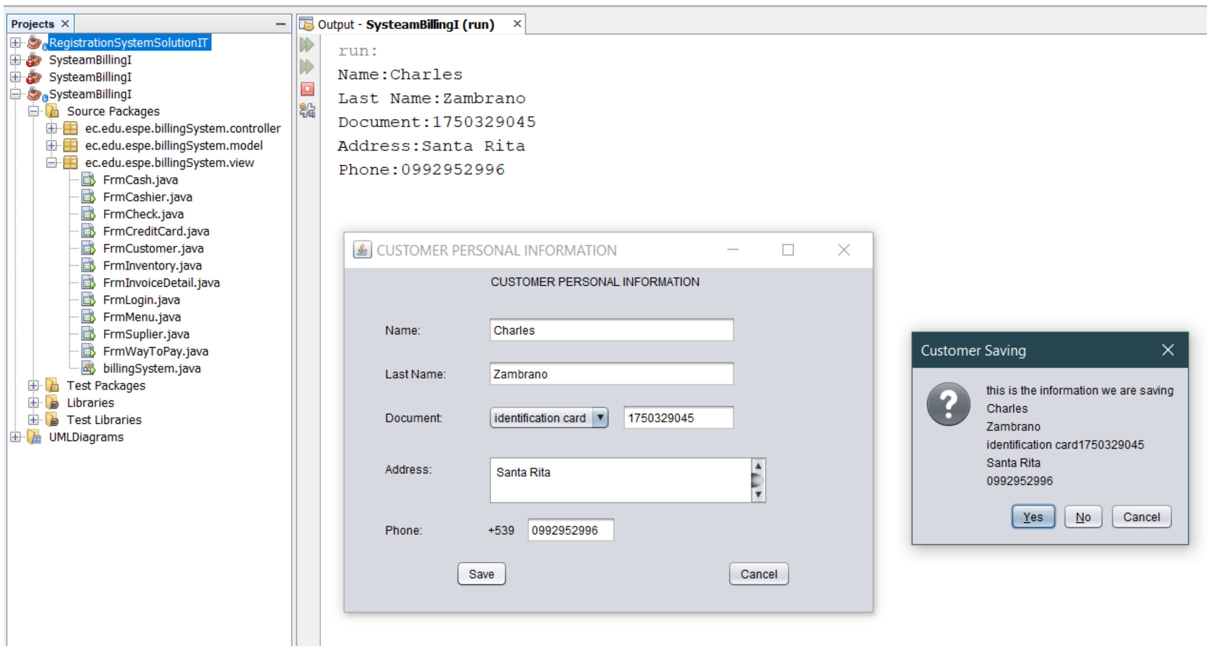
- Missing the person actor and owner actor in the main menu.





c. **Right**.

d. **Use of tables (printing) :**



```
run:
Name:Charles
Last Name:Zambrano
Document:1750329045
Address:Santa Rita
Phone:0992952996
```

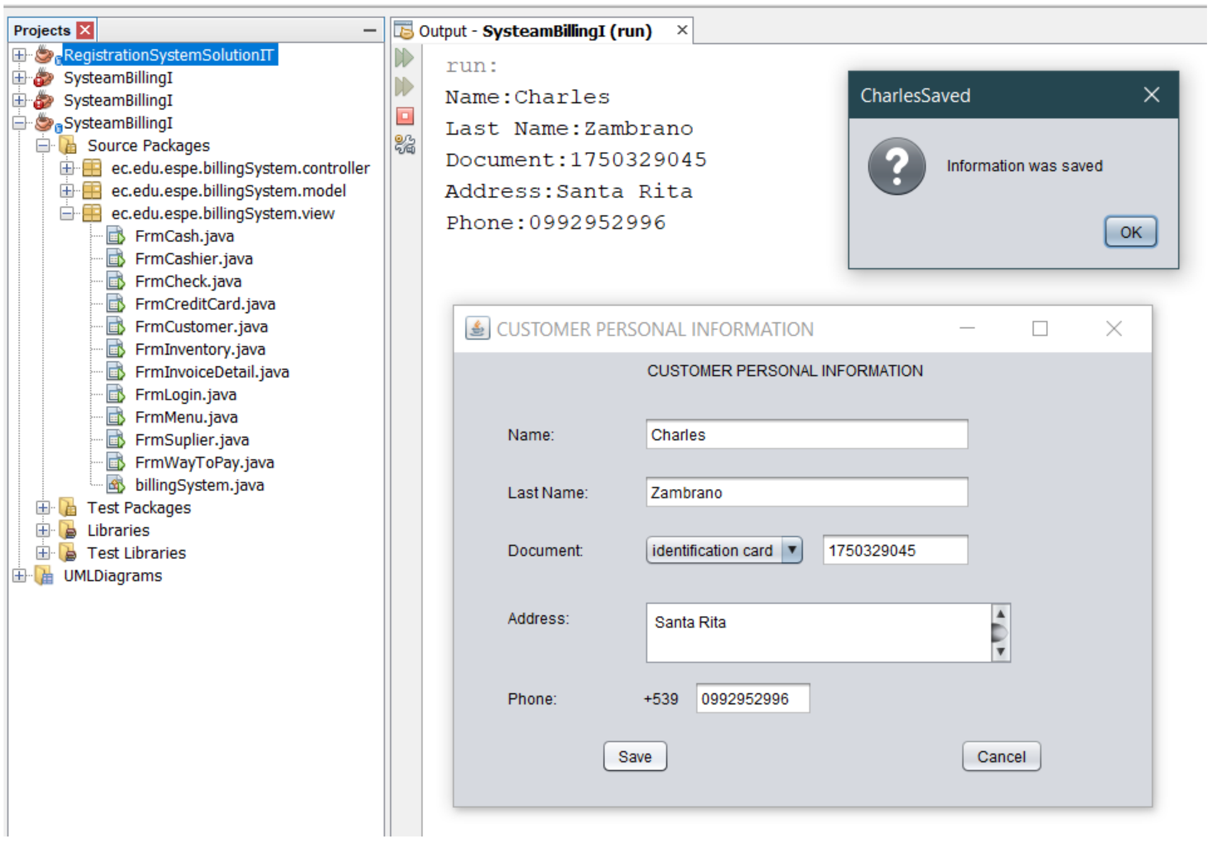
CUSTOMER PERSONAL INFORMATION

CUSTOMER PERSONAL INFORMATION

Name:	Charles
Last Name:	Zambrano
Document:	identification card 1750329045
Address:	Santa Rita
Phone:	+539 0992952996

Customer Saving

this is the information we are saving
Charles
Zambrano
identification card1750329045
Santa Rita
0992952996



```
run:
Name:Charles
Last Name:Zambrano
Document:1750329045
Address:Santa Rita
Phone:0992952996
```

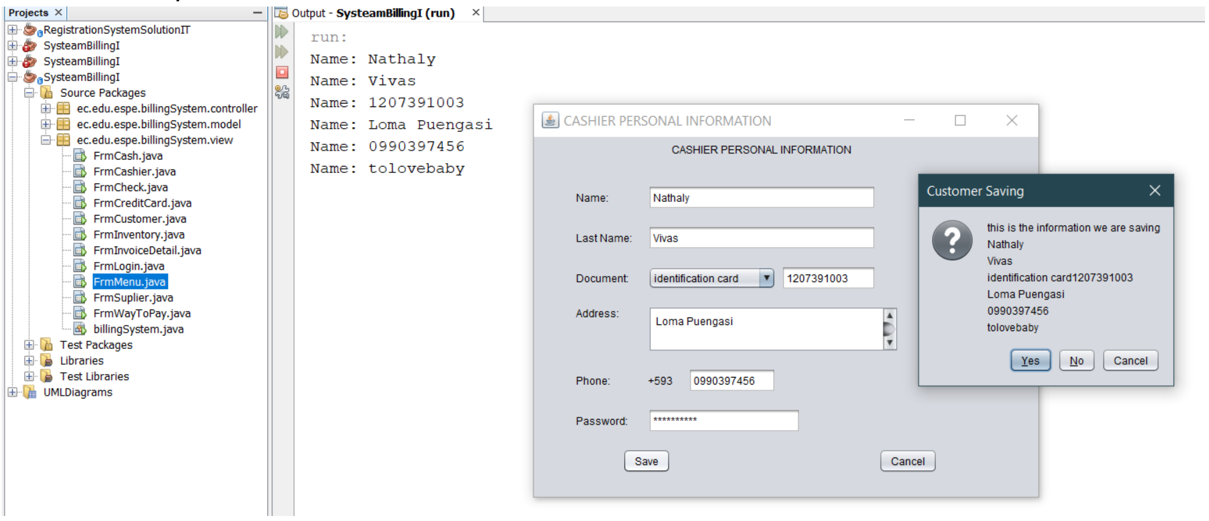
CharlesSaved

Information was saved

CUSTOMER PERSONAL INFORMATION

Name:	Charles
Last Name:	Zambrano
Document:	identification card 1750329045
Address:	Santa Rita
Phone:	+539 0992952996

- **Observation1:** And there is no return to the main menu, the program has to be runed one more time or press the button cancel that its not intuitive to the user .



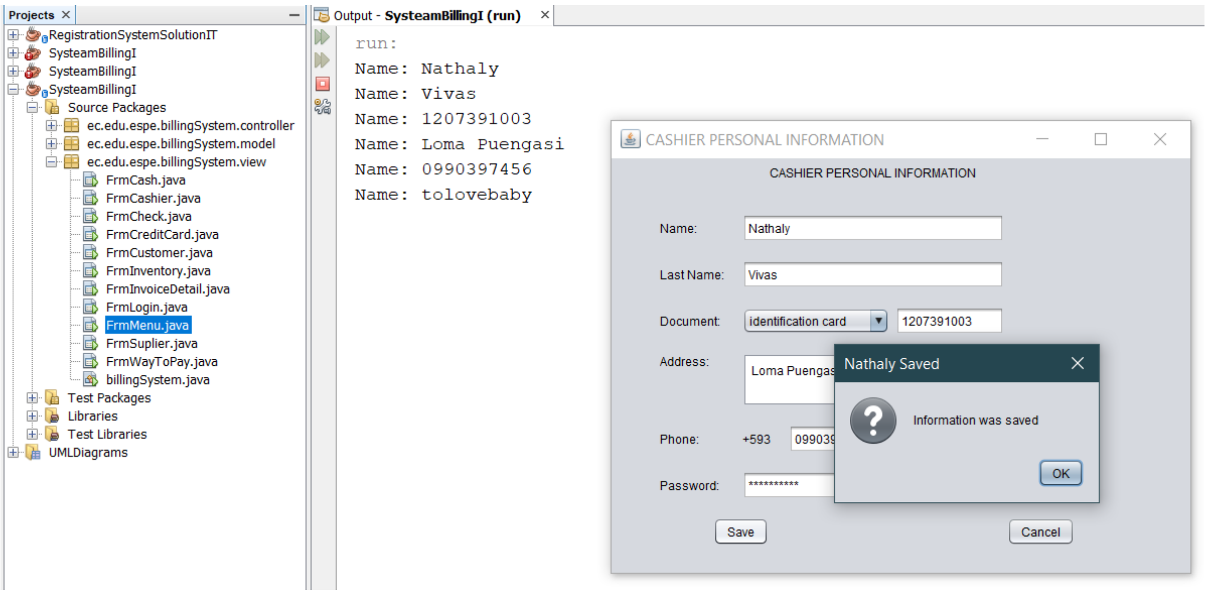
```
run:
Name: Nathaly
Name: Vivas
Name: 1207391003
Name: Loma Puengasi
Name: 0990397456
Name: tolovebaby
```

CASHIER PERSONAL INFORMATION

Name:	Nathaly
Last Name:	Vivas
Document:	identification card 1207391003
Address:	Loma Puengasi
Phone:	+593 0990397456
Password:	**********

Customer Saving

this is the information we are saving
Nathaly
Vivas
identification card1207391003
Loma Puengasi
0990397456
tolovebaby



```
run:
Name: Nathaly
Name: Vivas
Name: 1207391003
Name: Loma Puengasi
Name: 0990397456
Name: tolovebaby
```

CASHIER PERSONAL INFORMATION

CASHIER PERSONAL INFORMATION

Name: Nathaly

Last Name: Vivas

Document: identification card 1207391003

Address: Loma Puengas

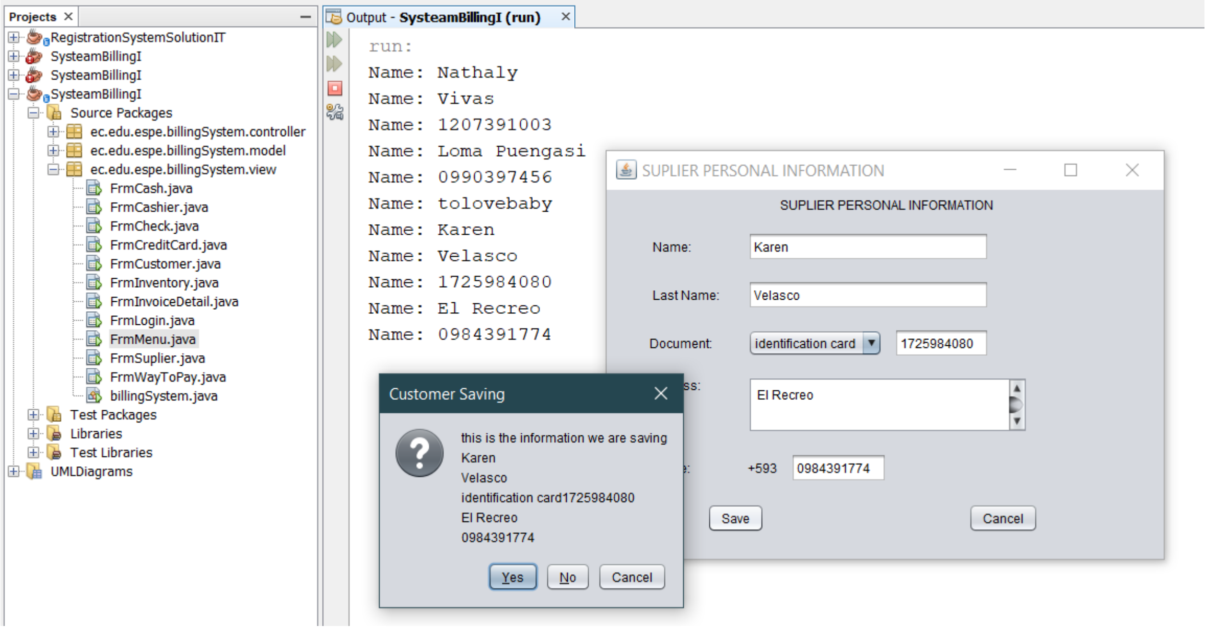
Phone: +593 099039

Password: **********

Nathaly Saved

Information was saved

- **Observation2:** The program correctly saves the cashier personal information but there aren't use of tables.



```
run:
Name: Nathaly
Name: Vivas
Name: 1207391003
Name: Loma Puengasi
Name: 0990397456
Name: tolovebaby
Name: Karen
Name: Velasco
Name: 1725984080
Name: El Recreo
Name: 0984391774
```

SUPLIER PERSONAL INFORMATION

SUPLIER PERSONAL INFORMATION

Name: Karen

Last Name: Velasco

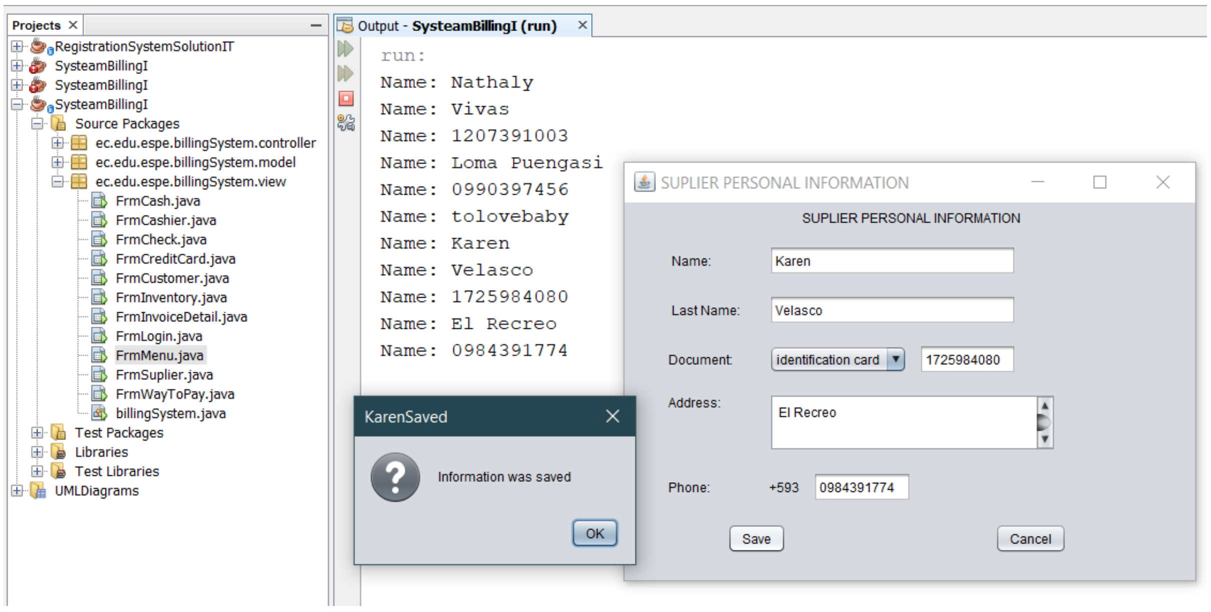
Document: identification card 1725984080

ss: El Recreo

e: +593 0984391774

Customer Saving

this is the information we are saving
Karen
Velasco
identification card1725984080
El Recreo
0984391774



```
run:
Name: Nathaly
Name: Vivas
Name: 1207391003
Name: Loma Puengasi
Name: 0990397456
Name: tolovebaby
Name: Karen
Name: Velasco
Name: 1725984080
Name: El Recreo
Name: 0984391774
```

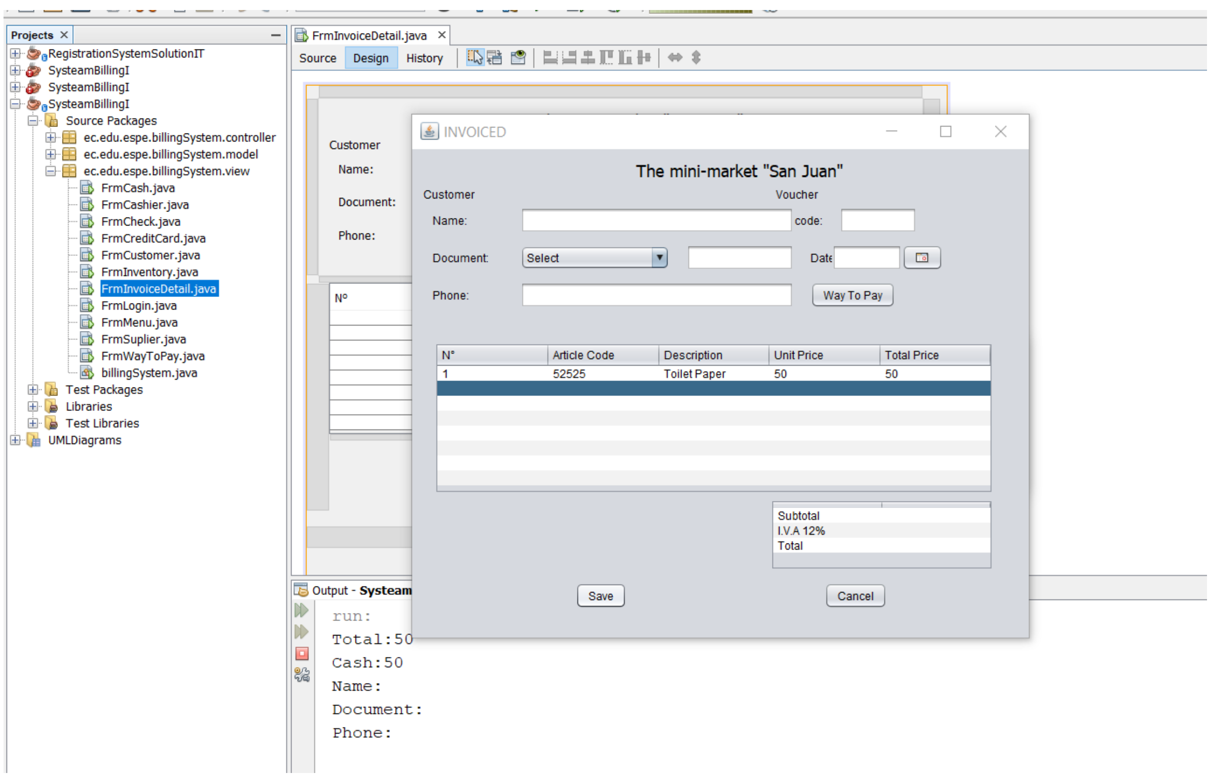
SUPLIER PERSONAL INFORMATION

Name:	Karen
Last Name:	Velasco
Document:	identification card 1725984080
Address:	El Recreo
Phone:	+593 0984391774

KarenSaved

Information was saved

- It has tables but they have to be runned from the Jframe and not from the main menu and also they don't correctly.



INVOICED

The mini-market "San Juan"

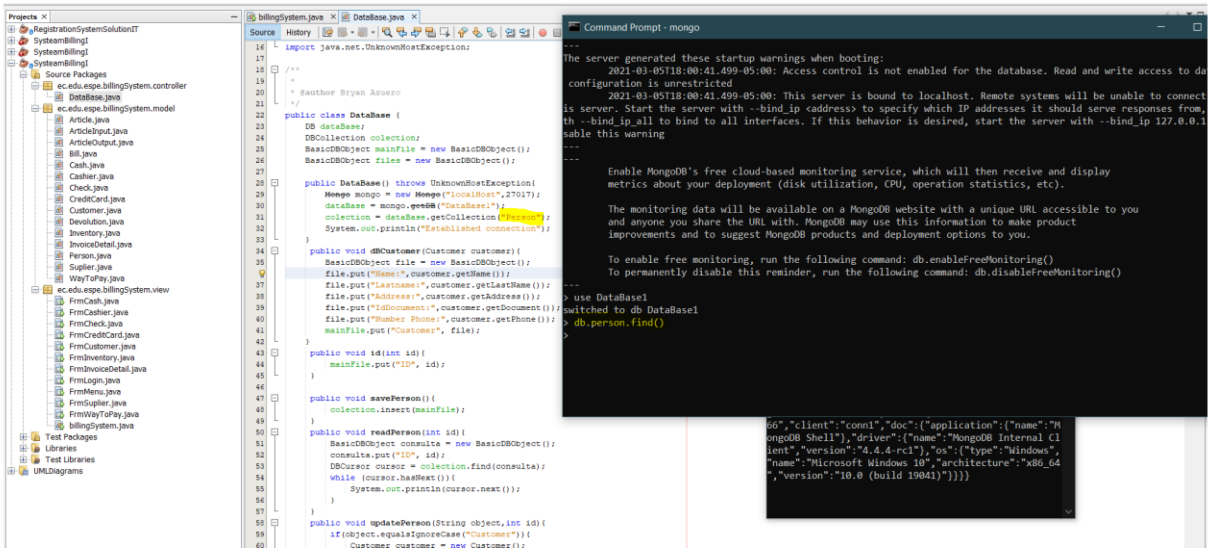
Customer		Voucher	
Name:		code:	
Document:	Select	Date	
Phone:			

N°	Article Code	Description	Unit Price	Total Price
1	52525	Toilet Paper	50	50

Subtotal
I.V.A 12%
Total

```
run:
Total:50
Cash:50
Name:
Document:
Phone:
```

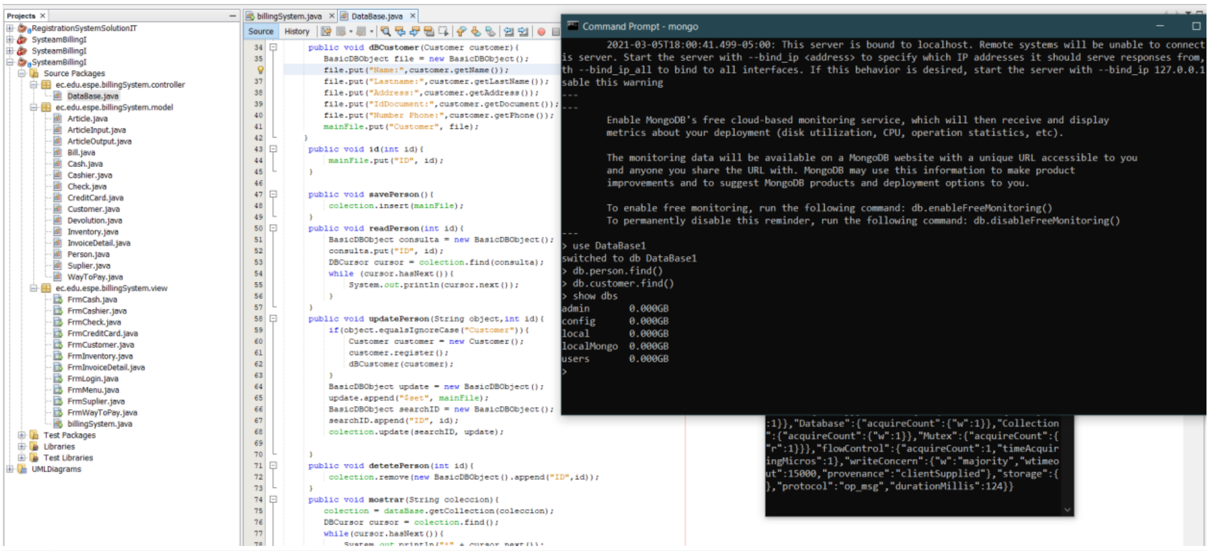
e. Functionality:



```java
import java.net.UnknownHostException;

/**
 *
 * @author Bryan Azuero
 */
public class DataBase {
    DB dataBase;
    DBCollection colection;
    BasicDBObject mainFile = new BasicDBObject();
    BasicDBObject files = new BasicDBObject();

    public DataBase() throws UnknownHostException{
        ~~Mongo~~ mongo = new ~~Mongo~~("localHost",27017);
        dataBase = mongo.~~getDB~~("DataBase1");
        colection = dataBase.getCollection("Person");
        System.out.println("Established connection");
    }
     public void dBCustomer(Customer customer){
        BasicDBObject file = new BasicDBObject();
        file.put("Name:",customer.getName());
        file.put("Lastname:",customer.getLastName());
        file.put("Address:",customer.getAddress());
        file.put("IdDocument:",customer.getDocument());
        file.put("Number Phone:",customer.getPhone());
        mainFile.put("Customer", file);
    }
     public void id(int id){
         mainFile.put("ID", id);
     }

     public void savePerson(){
         colection.insert(mainFile);
     }
     public void readPerson(int id){
         BasicDBObject consulta = new BasicDBObject();
         consulta.put("ID", id);
         DBCursor cursor = colection.find(consulta);
         while (cursor.hasNext()){
             System.out.println(cursor.next());
         }
     }
     public void updatePerson(String object,int id){
         if(object.equalsIgnoreCase("Customer")){
             Customer customer = new Customer();
```

```
---
The server generated these startup warnings when booting:
        2021-03-05T18:00:41.499-05:00: Access control is not enabled for the database. Read and write access to da
 configuration is unrestricted
        2021-03-05T18:00:41.499-05:00: This server is bound to localhost. Remote systems will be unable to connect
is server. Start the server with --bind_ip <address> to specify which IP addresses it should serve responses from,
th --bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with --bind_ip 127.0.0.1
sable this warning
---
---

        Enable MongoDB's free cloud-based monitoring service, which will then receive and display
        metrics about your deployment (disk utilization, CPU, operation statistics, etc).

        The monitoring data will be available on a MongoDB website with a unique URL accessible to you
        and anyone you share the URL with. MongoDB may use this information to make product
        improvements and to suggest MongoDB products and deployment options to you.

        To enable free monitoring, run the following command: db.enableFreeMonitoring()
        To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
> use DataBase1
switched to db DataBase1
> db.person.find()
>
```

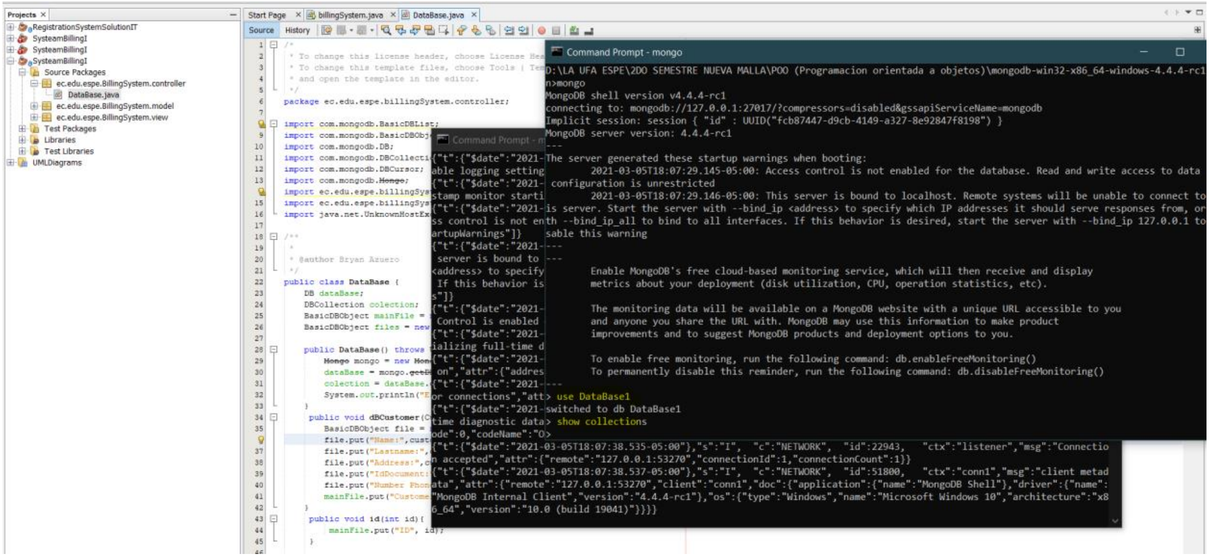
```
66","client":"conn1","doc":{"application":{"name":"M
ongoDB Shell"},"driver":{"name":"MongoDB Internal Cl
ient","version":"4.4.4-rc1"},"os":{"type":"Windows",
"name":"Microsoft Windows 10","architecture":"x86_64
","version":"10.0 (build 19041)"}}}}
```



```java
     public void dBCustomer(Customer customer){
        BasicDBObject file = new BasicDBObject();
        file.put("Name:",customer.getName());
        file.put("Lastname:",customer.getLastName());
        file.put("Address:",customer.getAddress());
        file.put("IdDocument:",customer.getDocument());
        file.put("Number Phone:",customer.getPhone());
        mainFile.put("Customer", file);
    }
     public void id(int id){
         mainFile.put("ID", id);
     }

     public void savePerson(){
         colection.insert(mainFile);
     }
     public void readPerson(int id){
         BasicDBObject consulta = new BasicDBObject();
         consulta.put("ID", id);
         DBCursor cursor = colection.find(consulta);
         while (cursor.hasNext()){
             System.out.println(cursor.next());
         }
     }
     public void updatePerson(String object,int id){
         if(object.equalsIgnoreCase("Customer")){
             Customer customer = new Customer();
             customer.register();
             dBCustomer(customer);
         }
         BasicDBObject update = new BasicDBObject();
         update.append("$set", mainFile);
         BasicDBObject searchID = new BasicDBObject();
         searchID.append("ID", id);
         colection.update(searchID, update);

     }
     public void detetePerson(int id){
         colection.remove(new BasicDBObject().append("ID",id));
     }
     public void mostrar(String coleccion){
        colection = dataBase.getCollection(coleccion);
        DBCursor cursor = colection.find();
        while(cursor.hasNext()){
```

```
        2021-03-05T18:00:41.499-05:00: This server is bound to localhost. Remote systems will be unable to connect
is server. Start the server with --bind_ip <address> to specify which IP addresses it should serve responses from,
th --bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with --bind_ip 127.0.0.1
sable this warning
---
---

        Enable MongoDB's free cloud-based monitoring service, which will then receive and display
        metrics about your deployment (disk utilization, CPU, operation statistics, etc).

        The monitoring data will be available on a MongoDB website with a unique URL accessible to you
        and anyone you share the URL with. MongoDB may use this information to make product
        improvements and to suggest MongoDB products and deployment options to you.

        To enable free monitoring, run the following command: db.enableFreeMonitoring()
        To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
> use DataBase1
switched to db DataBase1
> db.person.find()
> db.customer.find()
> show dbs
admin       0.000GB
config      0.000GB
local       0.000GB
localMongo  0.000GB
users       0.000GB
>
```

```
:1}},"Database":{"acquireCount":{"w":1}},"Collection
":{"acquireCount":{"w":1}},"Mutex":{"acquireCount":{
"r":1}}},"flowControl":{"acquireCount":1,"timeAcquir
ingMicros":1},"writeConcern":{"w":"majority","wtimeo
ut":15000,"provenance":"clientSupplied"},"storage":{
},"protocol":"op_msg","durationMillis":124}}
```



```java
/*
 * To change this license header, choose License Hea
 * To change this template files, choose Tools | Tem
 * and open the template in the editor.
 */
package ec.edu.espe.billingSystem.controller;

import com.mongodb.BasicDBList;
import com.mongodb.BasicDBObj
import com.mongodb.DB;
import com.mongodb.DBCollecti
import com.mongodb.DBCursor;
import com.mongodb.~~Mongo~~;
import ec.edu.espe.billingSys
import ec.edu.espe.billingSys
import java.net.UnknownHostEx

/**
 *
 * @author Bryan Azuero
 */
public class DataBase {
    DB dataBase;
    DBCollection colection;
    BasicDBObject mainFile =
    BasicDBObject files = new
```

```
D:\LA UFA ESPE\2DO SEMESTRE NUEVA MALLA\POO (Programacion orientada a objetos)\mongodb-win32-x86_64-windows-4.4.4-rc1
n>mongo
MongoDB shell version v4.4.4-rc1
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("fcb87447-d9cb-4149-a327-8e92847f8198") }
MongoDB server version: 4.4.4-rc1
---
The server generated these startup warnings when booting:
        2021-03-05T18:07:29.145-05:00: Access control is not enabled for the database. Read and write access to data
 configuration is unrestricted
        2021-03-05T18:07:29.146-05:00: This server is bound to localhost. Remote systems will be unable to connect to
is server. Start the server with --bind_ip <address> to specify which IP addresses it should serve responses from, or
th --bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with --bind_ip 127.0.0.1 to
sable this warning
---
---

        Enable MongoDB's free cloud-based monitoring service, which will then receive and display
        metrics about your deployment (disk utilization, CPU, operation statistics, etc).

        The monitoring data will be available on a MongoDB website with a unique URL accessible to you
        and anyone you share the URL with. MongoDB may use this information to make product
        improvements and to suggest MongoDB products and deployment options to you.

        To enable free monitoring, run the following command: db.enableFreeMonitoring()
        To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
> use DataBase1
switched to db DataBase1
> show collections
>
```

```
{"t":{"$date":"2021-03-05T18:07:38.535-05:00"},"s":"I",  "c":"NETWORK",  "id":22943,   "ctx":"listener","msg":"Connectio
n accepted","attr":{"remote":"127.0.0.1:53270","connectionId":1,"connectionCount":1}}
{"t":{"$date":"2021-03-05T18:07:38.537-05:00"},"s":"I",  "c":"NETWORK",  "id":51800,   "ctx":"conn1","msg":"client metad
ata","attr":{"remote":"127.0.0.1:53270","client":"conn1","doc":{"application":{"name":"MongoDB Shell"},"driver":{"name":
"MongoDB Internal Client","version":"4.4.4-rc1"},"os":{"type":"Windows","name":"Microsoft Windows 10","architecture":"x8
6_64","version":"10.0 (build 19041)"}}}}
```

- Does Not save nothing in the Database.

Recommendations:

- It is recommended that more caution be used when creating the methods, as there are some that are undefined.

- Dead code is not beneficial to the program.

- Practice abstraction.
  Low Coupling and high cohesion will allow a source code to be more understandable and can be better maintained.