

EQUIPE YASSINE

PROJET : Caisse enregistreuse automatique

CLIENT : BDE

2022/2023

BELLAGRAA Yassine
Medhi DOVIFAAZ
CHEN Patrick
FATHI Marie
LILE Vithia



Sommaire

Partie I : Base de données

- L'architecture générale : le modèle entité-association
- Le script de la base de données
- Les tests principaux

Partie II : Programmation

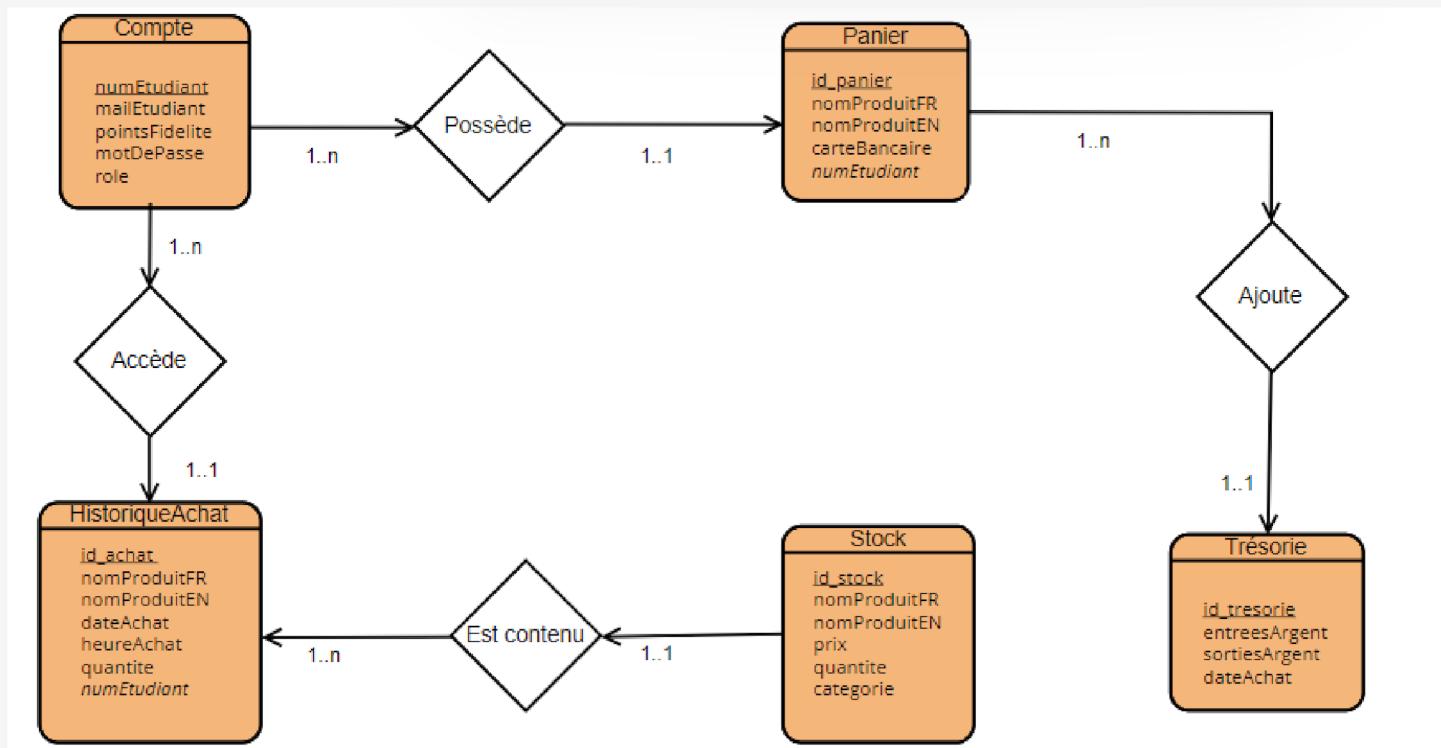
- L'organisation des classes : architecture client-serveur et MVC
- Les explications des principales fonctions
- Les tests
- Le bilan

Annexes

- Le code des principales fonctions

I : Base de données

L'architecture générale : le modèle entité-association



I : Base de données

Le script de la base de données

```
CREATE DATABASE caisseEnregistreuseAutomatique;

USE caisseEnregistreuseAutomatique;

CREATE TABLE Compte
(
    numEtudiant INT PRIMARY KEY,
    mailEtudiant VARCHAR(50),
    pointsFidelite INT,
    motDePasse VARCHAR(200),
    role VARCHAR(6)
);

CREATE TABLE HistoriqueAchat
(
    id_achat SERIAL PRIMARY KEY,
    nomProduitFR VARCHAR(30),
    nomProduitEN VARCHAR(30),
    dateAchat DATE,
    heureAchat TIME,
    quantite INT
);

CREATE TABLE Tresorie
(
    id_tresorie SERIAL PRIMARY KEY,
    entreesArgent NUMERIC(4,2),
    sortiesArgent NUMERIC(4,2),
    dateAchat DATE
);

CREATE TABLE Panier
(
    id_panier SERIAL PRIMARY KEY,
    nomProduitFR VARCHAR(30),
    nomProduitEN VARCHAR(30),
    numEtudiant INT REFERENCES Compte,
    carteBancaire BOOLEAN
);

CREATE TABLE IF NOT EXISTS Stock(
    idStock SERIAL PRIMARY KEY,
    NomProduitFR VARCHAR(255) NOT NULL,
    NomProduitEN VARCHAR(255) NOT NULL,
    prix DOUBLE PRECISION NOT NULL,
    quantite INTEGER NOT NULL,
    categorie VARCHAR (255)
);
```

I : Base de données

Les tests principaux

Insérer les achats dans la table Panier

INSERT INTO panier

```
(NomproduitFR,NomproduitEN,NumEtudiant,prix,quantite,carte_bancaire)
values('cristaline','cristaline',12101111,0.5,2,'true'),
('malabar','malabar',12101111,0.1,10,'true'),
('kit kat','kit kat',12101111,0.70,2,'true'),
('kinder bueno white','kinder bueno white',12101111,0.80,1,'true');
```

SELECT * from Panier;

id_panier	nomproduitfr	nomproduiten	numetudiant	prix	quantite	carte_bancaire
1	cristaline	cristaline	12101111	0.5	2	t
2	malabar	malabar	12101111	0.1	10	t
3	kit kat	kit kat	12101111	0.7	2	t
4	kinder bueno white	kinder bueno white	12101111	0.8	1	t

I : Base de données

Les tests principaux

Insérer les produits dans la table Stock

INSERT INTO Stock

```
(NomProduitFR, NomProduitEN ,prix ,quantite ,categorie)
values('cristaline','cristaline',0.5,1,'boisson'),
('capri sun','capri sun',0.5,1,'boisson'),
('oasis tropical','oasis tropical',0.8,1,'boisson'),
('oasis pomme','oasis pomme',0.8,1,'boisson'),
('lipton ice tea','lipton ice tea',0.8,1,'boisson'),
('malabar','malabar',0.1,1,'snacks'),
('lays nature','lays nature',0.6,1,'snacks'),
('lays barbecue','lays barbecue',0.7,1,'snacks'),
('mars','mars',0.7,1,'snacks'),
('kit kat','kit kat',0.7,1,'snacks'),
('kinder bueno','kinder bueno',0.8,1,'snacks'),
('kinder bueno white','kinder bueno white',0.8,1,'snacks'),
('fraise','fraise',0.8,1,'eau+sirops'),
('peche','peche',0.8,1,'eau+sirops'),
('cassis','cassis',0.8,1,'eau+sirops'),
('pomme verte','pomme verte',0.8,1,'eau+sirops'),
('fruits de la passion','fruits de la passion',0.8,1,'eau+sirops'),
('coca-cola','coca-cola',0.8,1,'soda'),
('coca-cola cherry','coca-cola cherry',0.8,1,'soda'),
('fanta citron','fanta citron',0.8,1,'soda');
```

SELECT * from Stock;

idstock	nomproduitfr	nomproduiten	prix	quantite	categorie
1	cristaline	cristaline	0.5	1	boisson
2	capri sun	capri sun	0.5	1	boisson
3	oasis tropical	oasis tropical	0.8	1	boisson
4	oasis pomme	oasis pomme	0.8	1	boisson
5	lipton ice tea	lipton ice tea	0.8	1	boisson
6	malabar	malabar	0.1	1	snacks
7	lays nature	lays nature	0.6	1	snacks
8	lays barbecue	lays barbecue	0.7	1	snacks
9	mars	mars	0.7	1	snacks
10	kit kat	kit kat	0.7	1	snacks
11	kinder bueno	kinder bueno	0.8	1	snacks
12	kinder bueno white	kinder bueno white	0.8	1	snacks
13	fraise	fraise	0.8	1	eau+sirops
14	peche	peche	0.8	1	eau+sirops
15	cassis	cassis	0.8	1	eau+sirops
16	pomme verte	pomme verte	0.8	1	eau+sirops
17	fruits de la passion	fruits de la passion	0.8	1	eau+sirops
18	coca-cola	coca-cola	0.8	1	soda
19	coca-cola cherry	coca-cola cherry	0.8	1	soda
20	fanta citron	fanta citron	0.8	1	soda

I : Base de données

Les tests principaux

Insérer les achats dans la table HistoriqueAchat

INSERT INTO HistoriqueAchat

```
(nomproduitFR,nomproduitEN,numEtudiant,prix,dates,heure,quantite)
values('cristaline','cristaline',12101111,0.5,now(),now(),2),
('malabar','malabar',12101111,0.1,now(),now(),10),
('kit kat','kit kat',12101111,0.7,now(),now(),2),
('kinder bueno white','kinder bueno white',12101111,0.8,now(),now(),1);
```

SELECT* from HistoriqueAchat;

id_achat	nomproduitfr	nomproduiten	numetudiant	prix	dates	heure	quantite
1	cristaline	cristaline	12101111	0.5	20/01/2023	12:52:10	2
2	malabar	malabar	12101111	0.1	20/01/2023	12:52:10	10
3	kit kat	kit kat	12101111	0.7	20/01/2023	12:52:10	2
4	kinder bueno white	kinder bueno white	12101111	0.8	20/01/2023	12:52:10	1

Insérer les clients dans la table Compte

INSERT INTO Compte

```
(NumEtudiant,adrMail,password,pointFideliter,role)
values(12101111,'a@gmail.com','mdp',0,'true'),
(12102222,'b@gmail.com','mdp',0,'true'),
(12103333,'c@gmail.com','mdp',0,'true'),
(12104444,'d@gmail.com','mdp',0,'true'),
(12105555,'e@gmail.com','mdp',0,'true');
```

SELECT * from Compte;

SELECT* from Compte;

numetudiant	adrmail	password	pointfideliter	role
12101111	a@gmail.com	mdp	0	t
12102222	b@gmail.com	mdp	0	t
12103333	c@gmail.com	mdp	0	t
12104444	d@gmail.com	mdp	0	t
12105555	e@gmail.com	mdp	0	t

I : Base de données

Les tests principaux

Insérer les échanges d'argent dans la table Tresorie

```
INSERT INTO tresorie  
(entrees,sortie,dates)  
values(10,0,now()),  
(50,0,'12-10-2024'),  
(0,45,now()),  
(24,0,now());
```

```
SELECT * from tresorie;
```

id_tresorie	entrees	sortie	dates
1	10	0	20/01/2023
2	50	0	12/10/2024
3	0	45	20/01/2023
4	24	0	20/01/2023

4 rows)

II- Programmation

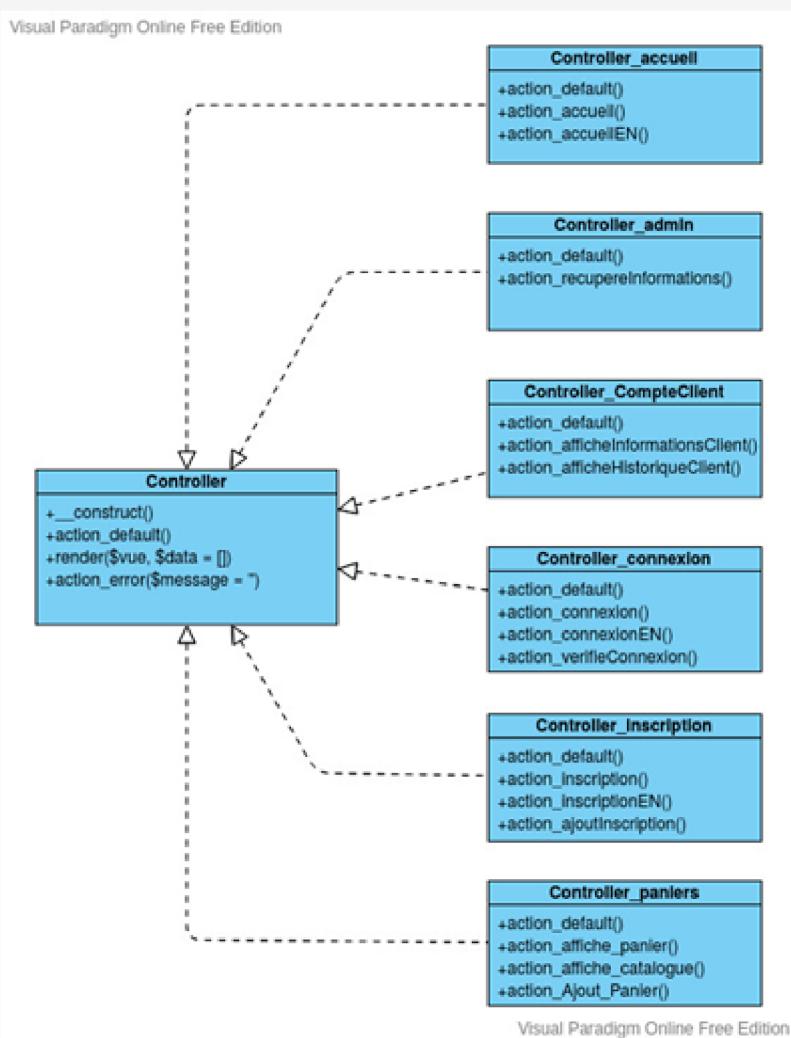
L'organisation des classes : architecture client-serveur et MVC

Nous avons vu du côté client, les vues qui font office d'interface utilisateur pour le client.

Liste des vues : view_inscription, view_inscriptionEN, view_connexion, view_client, view_catalogue_client, view_catalogue_admin, view_admin, view_accueil, view_accueil_anglais

Puis du côté serveur, nous avons les contrôleurs qui font le lien entre le Modèle et les Vues.

Liste des contrôleurs : Controller_panier, Controller_inscription, Controller_connexion, Controller_client, Controller_admin, Controller_accueil, Controller



II- Programmation

L'organisation des classes : architecture client-serveur et MVC

Nous avons vu du côté client, les vues qui font office d'interface utilisateur pour le client.

Liste des vues : view_inscription, view_inscriptionEN, view_connexion, view_client, view_catalogue_client, view_catalogue_admin, view_admin, view_accueil, view_accueil_anglais

Puis du côté serveur, nous avons les contrôleurs qui font le lien entre le Modèle et les Vues.

Liste des contrôleurs : Controller_panier, Controller_inscription, Controller_connexion, Controller_client, Controller_admin, Controller_accueil, Controller

II- Programmation

Les explications des principales fonctions

Les fonctions du contrôleur

Controller_admin

```
public function action_recupereInformations()
```

```
# récupère l'ensemble des tables Stock et Tresorie pour les transmettre à la vue view_admin.
```

Controller_connexion

```
public function action_verifieConnexion()
```

```
# vérifie si le client existe dans la table Compte si c'est le cas il est redirigé vers la vue_accueil sinon vers la vue_connexion.
```

Controller_inscription

```
public function action_ajoutInscription()
```

```
# enregistre un utilisateur dans la table Compte si et seulement si celui-ci n'existe pas déjà dans celle-ci puis redirige le client vers la page d'accueil.
```

Les fonctions du modèle

```
public function getPassword($login)
```

```
# récupère le mot de passe crypté associé au numéro étudiant $login.
```

```
public function pasDanslaBDD($login)
```

```
# vérifie si le numéro étudiant $login n'existe pas dans la table Compte, si c'est le cas on retourne true sinon false.
```

```
public function enregistreUtilisateur($login, $mdp, $mail)
```

```
# enregistre un utilisateur dans la base de données par rapport aux paramètres en entrée $login, $mdp et $mail.
```

```
public function affiche_stock()
```

```
# récupère toutes les données de la table Stock.
```

```
public function affiche_tresorie()
```

```
# récupère toutes les données de la table Tresorie.
```

II- Programmation

Les Tests

Afficher les entrées et sorties d'argent de la table Tresorie

```
-- TRESORERIE -->

```

Inventaire	Trésorerie	Paiement
Trésorerie		
	Sortie: <input type="text"/>	
	Entrer	
Entrées	Sorties	
20	0	
0	7	

Vérifier lors de l'ajout d'un article à la table Stock depuis le site si l'ajout a été pris en compte en listant l'inventaire et en listant le catalogue

```
<foreach($tab as $ligne):?>
<div class="produit">
    <img src=<?= $ligne["img"] ?>>
    <div class="">
        <h1><?= $ligne["NomProduitFR"] ?></h1>
        <div class="prix"><?= $ligne['prix'] ?></div>
        <button class="Ajouter au panier"> Ajouter au panier </button>
    </div>
</div>
<?php endforeach; ?>
```

Inventaire	Trésorerie	Paiement
Ajouter un produit		
Nom de l'article FR <input type="text"/>	Nom de l'article EN <input type="text"/>	
Prix <input type="text"/>	quantité <input type="text"/>	Categorie <input type="text" value="boisson"/>
		Ajouter
 cristalline 0.5 Ajouter au panier	 capri sun 0.5 Ajouter au panier	 oasis tropical 0.8 Ajouter au panier

II- Programmation

Les Tests

Afficher l'historique d'achat d'un client

Historique d'achats		Compte	
Article	Quantité	Date	Heures
oasis tropical	2	2023-01-10	02:39:32
cristalline	1	2023-01-02	12:30:31
Malabar	5	2023-01-02	12:39:32
lays nature	1	2023-01-02	16:00:00

```
<div id="historique_achats" class="tabcontent">
<table>
  <thead>
    <tr><th> Article </th><th> Quantité </th><th> Date </th><th> Heures </th></tr>
  </thead>
  <tbody>
    <?php require_once("../Controllers/controller_client.php")?>
    <?php foreach($tab as $ligne):?>
      <tr><td><?= $ligne["nomProduitFR"]?></td><td><?= $ligne["quantite"]?></td><td><?= $ligne["dateAchat"]?>
        <?php endforeach ?>
    </tbody>
  </table>
</div>
```

II- Programmation

Bilan technique

Ce qui fonctionne

L'inscription client et sa traduction en anglais

La connexion client et sa traduction en anglais

L'ajout d'un nouveau article dans l'inventaire et le catalogue

L'affichage du stock et du catalogue

L'affiche de l'historique client

Ce qui ne fonctionne pas

Le panier : il manque le code pour garder en mémoire le panier

L'inventaire : il n'est pas encore possible de augmenter ou diminuer la quantité d'un article, de supprimer un article en cliquant sur le bouton supprimer.

La calculatrice : le panier n'étant pas encore implémenté, il n'est pas possible d'afficher la somme totale du panier

De plus, nous n'avons pas encore mis en place le système de points de fidélité.

Le MVC : certaines pages ont été faites sans suivre le modèle MVC, il suffit juste de réussir à les intégrer dans l'architecture MVC.

Annexes

```
public function action_recupereInformations(){
    $m = Model::getModel();
    $stock = $m->affiche_stock();
    $tresorie = $m->affiche_tresorie();
    $data = ["stock" => $stock, "tresorie" => $tresorie];
    $this->render("admin", $data);
}
```

```
function action_verifieConnexion(){
    $m = Model::getModel();
    if (isset($_POST["numEtudiant"]) and isset($_POST["motDePasse"])){
        if (! $m->pasDanslaBDD($_POST["numEtudiant"])){
            $mdpCrypte = $m->getPassword($_POST['numEtudiant']);
            if (password_verify($_POST["motDePasse"], $mdpCrypte)){
                $_SESSION["connecte"] = true;
                $_SESSION["numEtudiant"] = $_POST["numEtudiant"];
                $_SESSION["email"] = $m->getEmail($_POST["numEtudiant"]);
                $_SESSION["role"] = $m->getRole($_POST['numEtudiant']);
                $this->render('accueil', []);
            } else {
                $this->render('connexion', []);
            }
        } else {
            if ($_COOKIE['lang'] == 'fr'){
                $data = ["message"=> "L'utilisateur n'existe pas."];
                $this->render('connexion', $data);
            }
            else if ($_COOKIE['lang'] == 'en'){
                $data = ["message"=> "User doesn't exists."];
                $this->render('connexionEN', $data);
            }
        }
    }
}
```

```
public function action_ajoutInscription(){
    $m = Model::getModel();
    if (isset($_POST["numEtudiant"]) and isset($_POST["motDePasse"]) and isset($_POST["mailEtudiant"])){
        if ($m->pasDanslaBDD($_POST['numEtudiant'])){
            $m->enregistreUtilisateur($_POST["numEtudiant"], $_POST["motDePasse"], $_POST["mailEtudiant"]);
            header('Location: Utils/accueil.php');
            if ($_COOKIE['lang'] == 'fr'){
                $data = ["message"=> "L'utilisateur ". $_POST["numEtudiant"] ." a été ajouté."];
                $this->render('inscription', $data);
            }

            else if ($_COOKIE['lang'] == 'en'){
                $data = ["message"=> "User ". $_POST["numEtudiant"] ." have been added."];
                $this->render('inscriptionEN', $data);
            }
        }
        if ($_COOKIE['lang'] == 'fr'){
            $data = ["message"=> "L'utilisateur existe déjà."];
            $this->render('inscription', $data);
        }
        else if ($_COOKIE['lang'] == 'en'){
            $data = ["message"=> "User already exists."];
            $this->render('inscriptionEN', $data);
        }
    }
    if ($_COOKIE['lang'] == 'fr'){
        $data = ["message"=> "Vous avez saisi une valeur incorrecte."];
        $this->render('inscription', $data);
    }
    else if ($_COOKIE['lang'] == 'en'){
        $data = ["message"=> "You entered an incorrect value."];
        $this->render('inscriptionEN', $data);
    }
}
```

Annexes

```
public function getPassword($login){  
    $req = $this->bd->prepare('SELECT numEtudiant, motdepasse FROM Compte WHERE numEtudiant = :id ' );  
    $req->bindValue(':id', $login);  
    $req->execute();  
    $ligne = $req->fetch(PDO::FETCH_ASSOC);  
    if (!empty($ligne)) { return $ligne['motdepasse']; }  
    return false;  
}
```

```
public function pasDanslaBDD($login) {  
    $req = $this->bd->prepare('SELECT numEtudiant FROM Compte WHERE numetudiant = :id');  
    $req->bindValue(':id', $login);  
    $req->execute();  
    $identifiant = $req->fetchAll(PDO::FETCH_ASSOC);  
    if ( empty($identifiant[0])){  
        return true;  
    }  
    return false;  
}
```

```
public function enregistreUtilisateur($login,$mdp,$mail){  
    $role = 'client';  
    $pts = 0;  
    $req = $this->bd->prepare('INSERT INTO compte(numetudiant, mailetudiant,pointsfidelite,motdepasse,role)  
values (:numetudiant,:mailetudiant,:pointsfidelite, :motdepasse,:role)');  
    $req->bindValue(':numetudiant', $login);  
    $req->bindValue(':mailetudiant', $mail);  
    $req->bindValue(':pointsfidelite', $pts);  
    $req->bindValue(':motdepasse', password_hash($mdp, PASSWORD_DEFAULT) );  
    $req->bindValue(':role', $role);  
    $req->execute();  
    return $req->rowCount();  
}
```

```
public function affiche_stock(){  
    $req = $this->bd->prepare('SELECT * FROM stock');  
    $req->execute();  
    $stock = $req->fetchAll(PDO::FETCH_ASSOC);  
    return $stock ;  
}  
  
public function affiche_tresorie(){  
    $req = $this->bd->prepare('SELECT * FROM tresorie');  
    $req->execute();  
    $tresorie = $req->fetchAll(PDO::FETCH_ASSOC);  
    return $tresorie ;  
}
```