

- LUCIANO YENOUNMOU
- 2SLAM
  
- GASTON BERGER LILLE



Planning de stage deuxième année						
	Semaine 1	Semaine 2	Semaine 3	Semaine 4	Semaine 5	Semaine 6
Lun	Choix des technologies et constructions des maquettes restants.	Corrections des erreurs et modifications de la base données	Modification du MCD, MLD... Constructions des maquettes restants avec figma.	Développement de l'application web en Php MVC	Développement de l'application web en Php MVC	Développement de l'application web en Php MVC
Mar	Discutions des besoins a mettre en place avec mon tuteur et redaction des conditions générales d'utilisations.		Apporter des nouvelles modifications en fonctions des nouvelles fonctionnalités et test de la base de données			
Mer	Conception du MCD, MLD... avec Looping. Codage de la base de données sous SQL	Test de la base de données				
Jeu	Test de la base de données		Test de la base de données et validation de la base de données			
Ven		Mise en point de l'avancement du projet avec mon tuteur	Mise en point de l'avancement du projet avec mon tuteur	Mise en point de l'avancement du site avec mon tuteur.	Mise en point de l'avancement du site avec mon tuteur	Mise en point de l'avancement du site avec mon tuteur et signature de l'attestation de stage

## Contexte du stage :

### 1.1. Contexte :

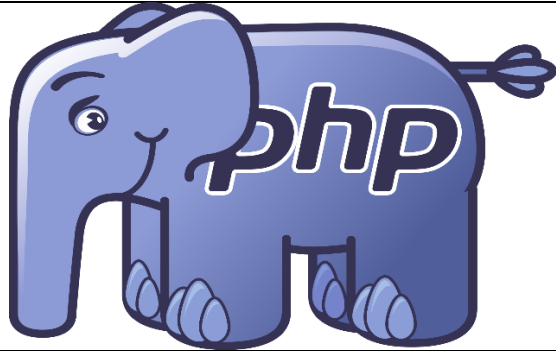
J'ai été sélectionné pour effectuer un stage de développement informatique au sein du salon de coiffure "Fe Ta Coup". La direction du salon souhaite développer une application permettant à ses clients de prendre des rendez-vous en ligne et de laisser des commentaires sur les services proposés.

Ma mission était de concevoir et développer une application web qui permettra aux clients du salon de prendre rendez-vous en ligne, de laisser des commentaires sur les services proposés, et aux coiffeurs de gérer leur emploi du temps.

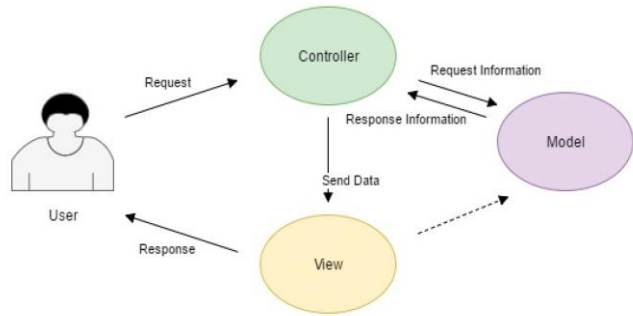
## Choix des technologies :

### 1.1.1. PHP

PHP : PHP est un langage de programmation populaire pour le développement web côté serveur. Il est utilisé pour créer des applications web dynamiques et des sites web. PHP est souvent associé avec des bases de données comme MySQL pour stocker et récupérer des données.

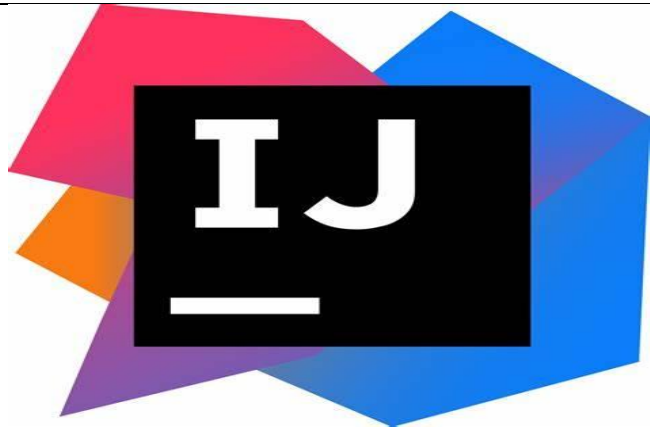


MVC : Le pattern MVC (Modèle-Vue-Contrôleur) est une structure d'architecture logicielle utilisée pour organiser les composants d'une application web. Il permet de séparer les différentes couches de l'application pour faciliter la maintenance et la mise à jour du code.



#### 1.1.2. INTELLIJ IDEA :

IntelliJ : IntelliJ est un environnement de développement intégré (IDE) pour Java et d'autres langages de programmation. Il offre des fonctionnalités avancées telles que la coloration syntaxique, l'achèvement de code, le débogage, le refactoring, etc.



#### 1.1.3. XAMPP :

XAMPP : XAMPP est une distribution Apache pour les systèmes d'exploitation Windows, Linux et Mac OS. Elle comprend également des outils tels que PHP, MySQL et Apache. XAMPP est utilisé pour mettre en place un environnement de développement local pour le développement d'applications web.



#### 1.1.4. MYSQL :

MySQL : MySQL est un système de gestion de base de données relationnelles (SGBDR) open-source très populaire. Il est souvent utilisé avec PHP pour stocker et récupérer des données dans les applications web.



#### 1.1.5. SQL :

SQL : SQL (Structured Query Language) est un langage utilisé pour communiquer avec les bases de données relationnelles. Il permet de créer, modifier et interroger des bases de données. SQL est souvent utilisé en combinaison avec des SGBDR tels que MySQL.



#### 1.1.6. LOOPING.IO :

Looping est un logiciel de modélisation de données qui permet de concevoir des modèles de données conceptuels (MCD), logiques (MLD) et physiques (MPD), ainsi que des diagrammes UML.



En utilisant PHP MVC, vous pouvez facilement organiser votre code en séparant les différentes couches de l'application. IntelliJ est un IDE qui offre des fonctionnalités avancées pour le développement de logiciels. XAMPP est utilisé pour mettre en place un environnement de développement local pour le développement d'applications web, tandis que MySQL est un SGBDR utilisé pour stocker et récupérer des données dans les applications web. Enfin, SQL est utilisé pour communiquer avec les bases de données relationnelles.

### Base de données et tests :

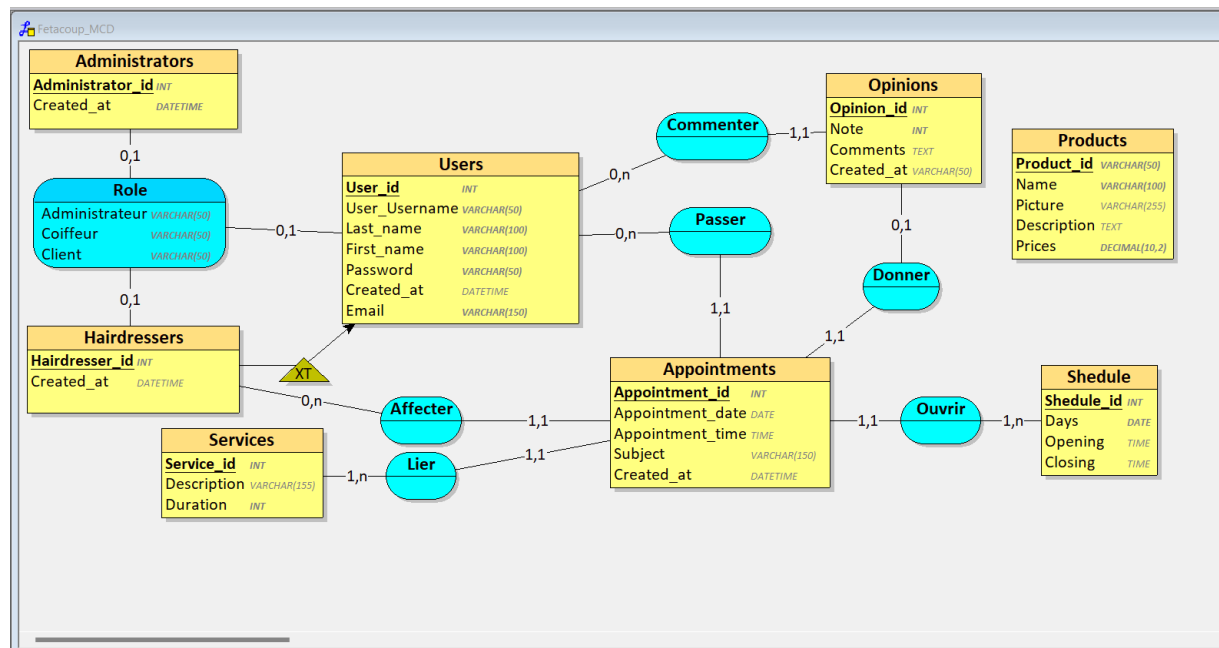
#### 1.1. L'idée que j'ai proposé à mon tuteur pour concevoir la base de données :

La base de données relationnelle contiendra plusieurs tables : la table "Users" qui contient les informations sur les clients du salon, la table "Hairdressers" qui contient les informations sur les

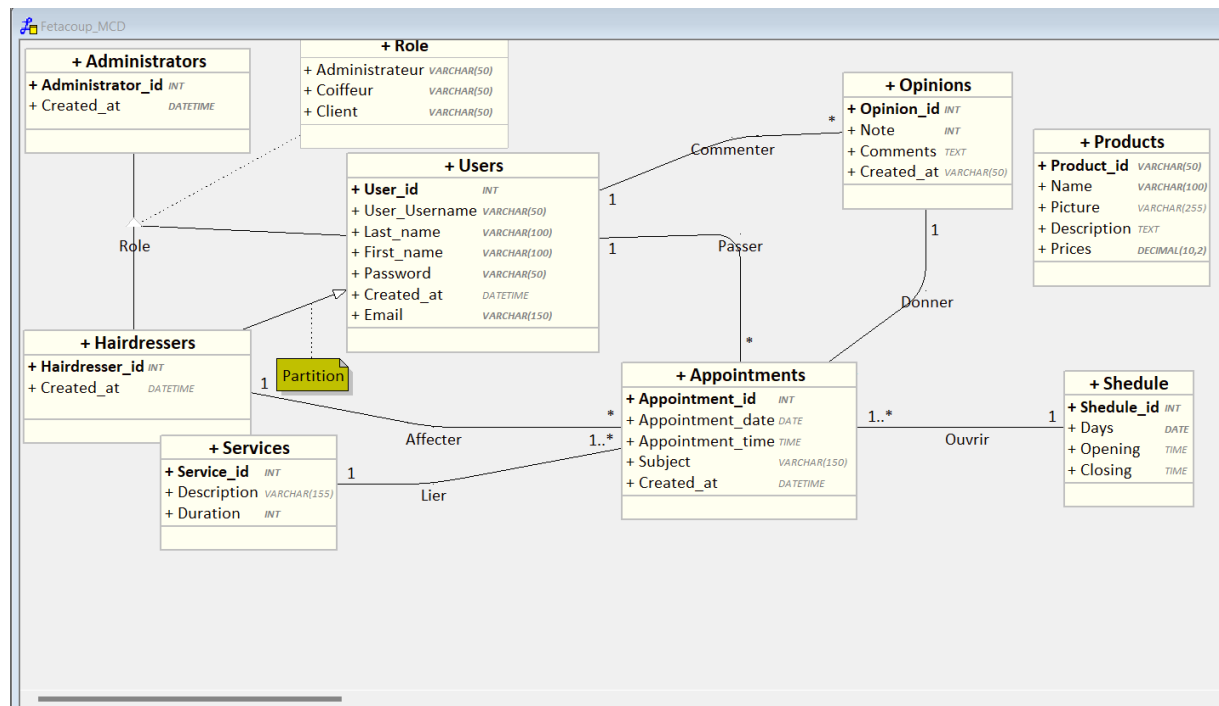
coiffeurs, la table "Services" qui décrit les services proposés par le salon, la table "Opinions" qui contient les commentaires des clients, la table "Shedule" qui indique les horaires d'ouverture du salon, la table "Appointments" qui contient les informations sur les rendez-vous, la table "Role" qui définit les rôles des différents utilisateurs, et la table "Products" qui décrit les produits capillaires vendus par le salon.

## 1.2. Modèle de données de conceptuels (MCD) :

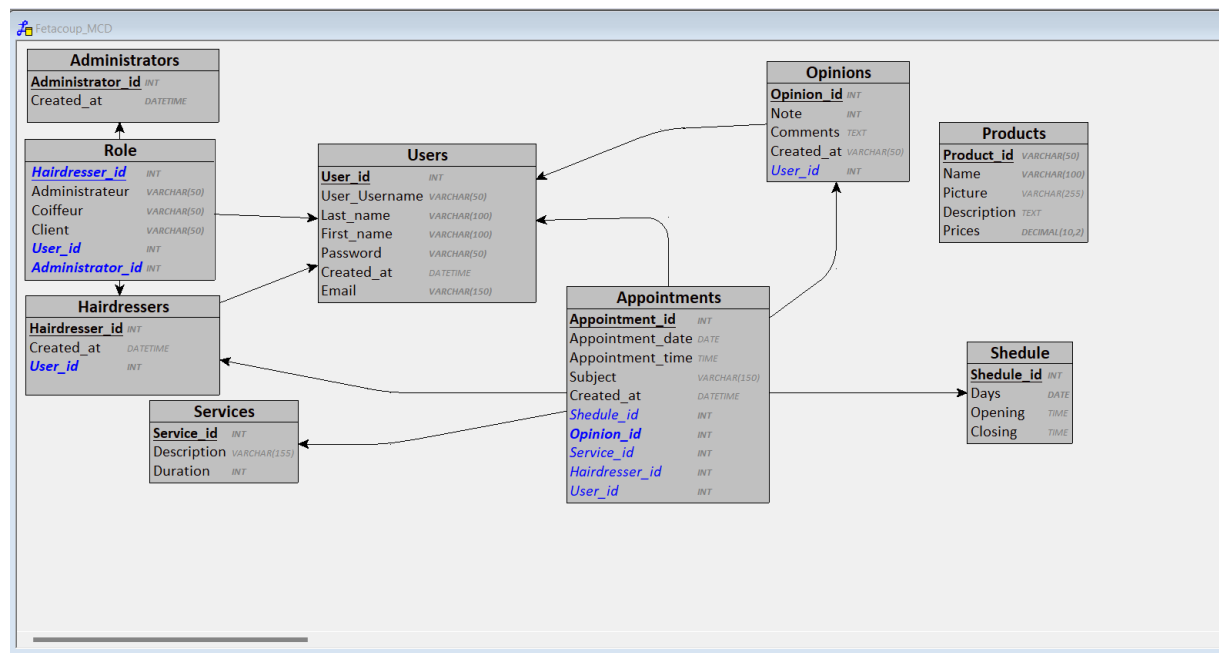
Version début :



UML :



MLD :



### MLD Textuel:

```

Administrators = (Administrator_id INT, Created_at DATETIME);
Users = (User_id INT, User_Username VARCHAR(50), Last_name
VARCHAR(100), First_name VARCHAR(100), Password VARCHAR(50),
Created_at DATETIME, Email VARCHAR(150));
Hairdressers = (Hairdresser_id INT, Created_at DATETIME,
#User_id);
Services = (Service_id INT, Description VARCHAR(155), Duration
INT);
Opinions = (Opinion_id INT, Note INT, Comments TEXT,
Created_at VARCHAR(50), #User_id);
Shedule = (Shedule_id INT, Days DATE, Opening TIME, Closing
TIME);
Products = (Product_id VARCHAR(50), Name VARCHAR(100), Picture
VARCHAR(255), Description TEXT, Prices DECIMAL(10,2));
Appointments = (Appointment id INT, Appointment_date DATE,
Appointment_time TIME, Subject VARCHAR(150), Created_at
DATETIME, #Shedule_id, #Opinion_id, #Service_id,
#Hairdresser_id, #User_id);
Role = (#Hairdresser_id, Administrateur VARCHAR(50), Coiffeur
VARCHAR(50), Client VARCHAR(50), #User id, #Administrator id);
  
```

### SCRIPT SQL:

```

CREATE TABLE Administrators(
    Administrator_id INT NULL AUTO_INCREMENT,
    Created_at TIMESTAMP NOT NULL,
    PRIMARY KEY(Administrator_id)
);

CREATE TABLE Users(
  
```

```

    User_id INT NULL AUTO_INCREMENT,
    User_username VARCHAR(50) NOT NULL,
    Last_name VARCHAR(100) NOT NULL,
    First_name VARCHAR(100) NOT NULL,
    Password VARCHAR(50) NOT NULL,
    Created_at TIMESTAMP,
    Email VARCHAR(150) NOT NULL,
    PRIMARY KEY(User_id)
);

CREATE TABLE Hairdressers(
    Hairdresser_id INT NULL AUTO_INCREMENT,
    User_id INT NOT NULL,
    Created_at TIMESTAMP,
    PRIMARY KEY(Hairdresser_id),
    UNIQUE(User_id),
    FOREIGN KEY(User_id) REFERENCES Users(User_id)
);

CREATE TABLE Services(
    Service_id INT NOT NULL AUTO_INCREMENT,
    Description VARCHAR(155),
    Duration INT NOT NULL,
    PRIMARY KEY(Service_id)
);

CREATE TABLE Opinions(
    Opinion_id INT NOT NULL AUTO_INCREMENT,
    User_id INT NOT NULL,
    Note INT NOT NULL,
    Comments TEXT,
    Created_at TIMESTAMP,
    PRIMARY KEY(Opinion_id),
    FOREIGN KEY(User_id) REFERENCES Users(User_id)
);

CREATE TABLE Shedule(
    Shedule_id INT NOT NULL AUTO_INCREMENT,
    Days DATE NOT NULL,
    Opening TIME,
    Closing TIME,
    PRIMARY KEY(Shedule_id)
);

CREATE TABLE Appointments(
    Appointment_id INT NOT NULL AUTO_INCREMENT,
    Shedule_id INT NOT NULL,
    Opinion_id INT NOT NULL,
    Service_id INT NOT NULL,
    Days_id INT NOT NULL,
    Hairdresser_id INT NOT NULL,

```

```

    User_id INT NOT NULL,
    Appointment_date DATE,
    Appointment_time TIME,
    Subject VARCHAR(150),
    Created_at TIMESTAMP,
    PRIMARY KEY(Appointment_id),
    UNIQUE(Opinion_id),
    FOREIGN KEY(Shedule_id) REFERENCES Shedule(Shedule_id),
    FOREIGN KEY(Opinion_id) REFERENCES Opinions(Opinion_id),
    FOREIGN KEY(Service_id) REFERENCES Services(Service_id),
    FOREIGN KEY(Days_id) REFERENCES Shedule(Shedule_id),
    FOREIGN KEY(Hairdresser_id) REFERENCES
Hairdressers(Hairdresser_id),
    FOREIGN KEY(User_id) REFERENCES Users(User_id)
);

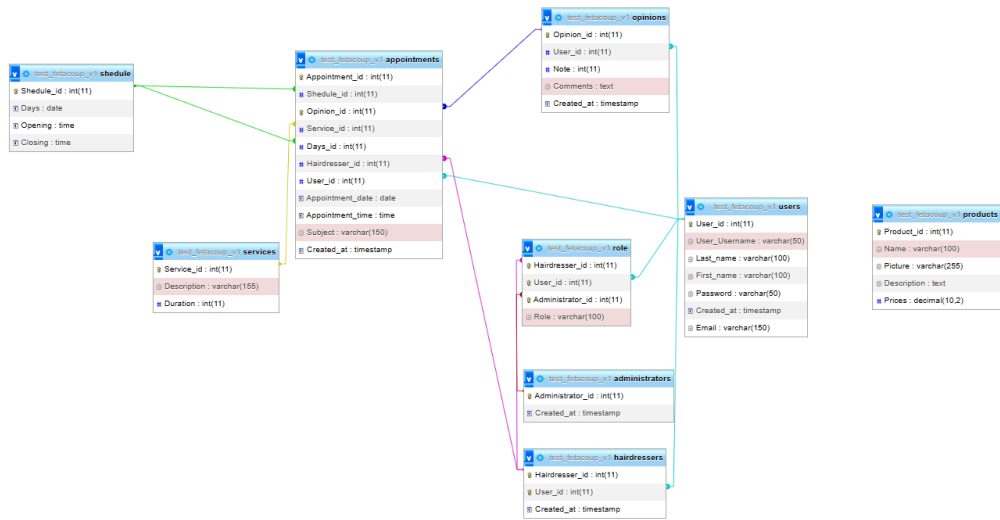
CREATE TABLE Role(
    Hairdresser_id INT NULL,
    User_id INT NULL,
    Administrator_id INT NULL,
    Role VARCHAR(100) NOT NULL,
    PRIMARY KEY(Hairdresser_id, User_id, Administrator_id),
    UNIQUE(Hairdresser_id),
    UNIQUE(User_id),
    UNIQUE(Administrator_id),
    FOREIGN KEY(Hairdresser_id) REFERENCES
Hairdressers(Hairdresser_id),
    FOREIGN KEY(User_id) REFERENCES Users(User_id),
    FOREIGN KEY(Administrator_id) REFERENCES
Administrators(Administrator_id)
);

CREATE TABLE Products(
    Product_id INT NOT NULL AUTO_INCREMENT,
    Name VARCHAR(100) NOT NULL,
    Picture VARCHAR(255),
    Description TEXT,
    Prices DECIMAL(10,2) NOT NULL,
    PRIMARY KEY(Product_id)
);

```

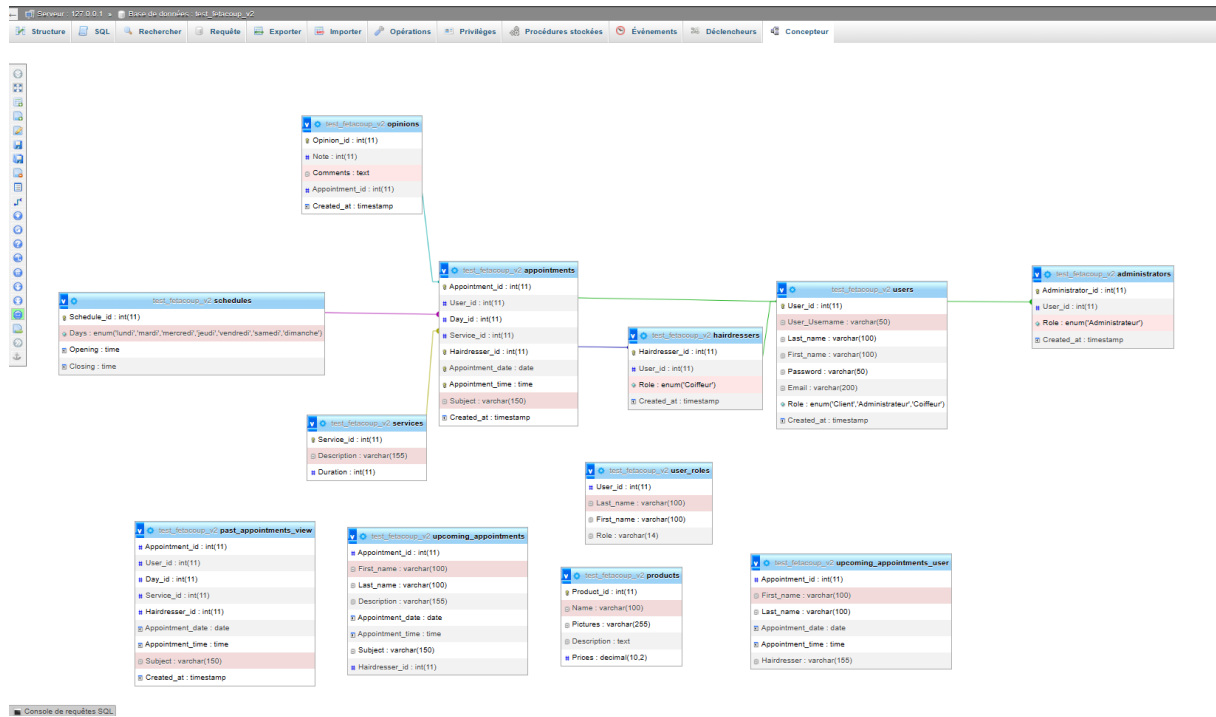
**Conceptor MySQL:**





liste de requêtes SQL

## Version 2:



## SCRIPT SQL:

```

-- phpMyAdmin SQL Dump
-- version 5.2.0
-- https://www.phpmyadmin.net/
--
-- Hôte : 127.0.0.1
-- Généré le : sam. 01 avr. 2023 à 23:49
  
```

```

-- Version du serveur : 10.4.27-MariaDB
-- Version de PHP : 8.2.0

START TRANSACTION;
SET time_zone = "+00:00";

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT
*/;
/*!40101 SET
@OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION
*/;
/*!40101 SET NAMES utf8mb4 */;

--
-- Base de données : `test_fetacoup_v2`
--

DELIMITER $$
--
-- Procédures
--
CREATE DEFINER=`root`@`localhost` PROCEDURE `add_new_user` (IN
`p_username` VARCHAR(50), IN `p_last_name` VARCHAR(100), IN
`p_first_name` VARCHAR(100), IN `p_password` VARCHAR(50), IN
`p_phone_number` VARCHAR(20), IN `p_email` VARCHAR(200), IN
`p_city` VARCHAR(100), IN `p_zip_code` VARCHAR(10), IN
`p_address` VARCHAR(200), IN `p_role`
ENUM('Client','Administrateur','Coiffeur')) BEGIN
    DECLARE v_user_id INT;
    DECLARE v_hairdresser_id INT;
    DECLARE v_administrator_id INT;

    IF p_role = 'Coiffeur' THEN
        INSERT INTO users(User_Username, Last_name,
First_name, Password, Phone_Number, Email, City, Zip_Code,
Address, Role)
            VALUES(p_username, p_last_name, p_first_name,
p_password, p_phone_number, p_email, p_city, p_zip_code,
p_address, p_role);
        SET v_user_id = LAST_INSERT_ID();
        INSERT INTO hairdressers(Hairdresser_id, User_id,
Role)
            VALUES(v_user_id, v_user_id, p_role);
        SET v_hairdresser_id = LAST_INSERT_ID();
        SELECT CONCAT(p_first_name, ' ', p_last_name, ' a été
ajouté(e) à la base de données avec succès.') AS message;
    ELSEIF p_role = 'Administrateur' THEN
        INSERT INTO users(User_Username, Last_name,
First_name, Password, Phone_Number, Email, City, Zip_Code,

```

```

Address, Role)
    VALUES (p_username, p_last_name, p_first_name,
p_password, p_phone_number, p_email, p_city, p_zip_code,
p_address, p_role);
    SET v_user_id = LAST_INSERT_ID();
    INSERT INTO administrators (Administrator_id, User_id,
Role)
    VALUES (v_user_id, v_user_id, p_role);
    SET v_administrator_id = LAST_INSERT_ID();
    SELECT CONCAT(p_first_name, ' ', p_last_name, ' a été
ajouté(e) à la base de données avec succès.') AS message;
    ELSE
        SELECT 'Le rôle fourni n''est pas valide. Les rôles
valides sont ''Coiffeur'' et ''Administrateur''.' AS
error_message;
    END IF;
END$$

CREATE DEFINER=`root`@`localhost` PROCEDURE
`complete_appointment` (IN `appointment_id` INT, IN `comments`
TEXT) BEGIN
    UPDATE appointments SET Completed = 1 WHERE Appointment_id
= appointment_id;
    INSERT INTO opinions (Appointment_id, Comments) VALUES
(appointment_id, comments);
END$$

CREATE DEFINER=`root`@`localhost` PROCEDURE
`create_appointment_user` (IN `p_user_id` INT, IN `p_day_id`
INT, IN `p_service_id` INT, IN `p_hairdresser_id` INT, IN
`p_appointment_date` DATE, IN `p_appointment_time` TIME, IN
`p_subject` VARCHAR(150)) BEGIN
    DECLARE appointmentCount INT;
    SELECT COUNT(*) INTO appointmentCount
    FROM appointments
    WHERE Hairdresser_id = p_hairdresser_id
    AND Day_id = p_day_id
    AND Appointment_date = p_appointment_date
    AND Appointment_time = p_appointment_time;
    IF appointmentCount = 0 THEN
        INSERT INTO appointments (User_id, Day_id, Service_id,
Hairdresser_id, Appointment_date, Appointment_time, Subject)
        VALUES (p_user_id, p_day_id, p_service_id,
p_hairdresser_id, p_appointment_date, p_appointment_time,
p_subject);
    ELSE
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Cette
plage horaire n''est pas disponible pour le rendez-vous';
    END IF;
END$$

```

```

CREATE DEFINER=`root`@`localhost` PROCEDURE `update_schedule`
(IN `hairstresser_id` INT, IN `appointment_date` DATE, IN
`start_time` TIME, IN `end_time` TIME) BEGIN
    DECLARE day_id INT;
    SELECT Schedule_id INTO day_id FROM Schedules WHERE Days =
DAYNAME(appointment_date);

    IF (SELECT COUNT(*) FROM appointments WHERE Hairstresser_id
= hairstresser_id AND Day_id = day_id AND (
        (Appointment_time <= start_time AND
ADDTIME(Appointment_time, SEC_TO_TIME(Duration * 60)) >
start_time)
        OR (Appointment_time >= start_time AND
Appointment_time < end_time)
    )) > 0 THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Le
coiffeur est déjà pris à cette plage horaire';
    ELSE
        INSERT INTO appointments (User_id, Day_id, Service_id,
Hairstresser_id, Appointment_date, Appointment_time, Subject)
VALUES (user_id, day_id, service_id, hairstresser_id,
appointment_date, start_time, subject);
    END IF;
END$$

--
-- Fonctions
--

CREATE DEFINER=`root`@`localhost` FUNCTION `total_revenue`
(`start_date` DATE, `end_date` DATE) RETURNS DECIMAL(10,2)
BEGIN
    DECLARE revenue DECIMAL(10,2);
    SELECT SUM(s.Prices) INTO revenue
    FROM appointments a
    JOIN services s ON a.Service_id = s.Service_id
    WHERE Appointment_date BETWEEN start_date AND end_date;
    RETURN revenue;
END$$

DELIMITER ;

-- -----
--
-- Structure de la table `administrators`
--

CREATE TABLE `administrators` (
  `Administrator_id` int(11) NOT NULL,
  `User_id` int(11) NOT NULL,
  `Role` enum('Administrateur') NOT NULL,

```

```

    `Created_at` timestamp NOT NULL DEFAULT current_timestamp()
) ;

--
-- Déchargement des données de la table `administrators`
--

INSERT INTO `administrators` (`Administrator_id`, `User_id`,
`Role`, `Created_at`) VALUES
(1, 3, 'Administrateur', '2023-02-13 11:25:08');

-- -----

--
-- Structure de la table `appointments`
--

CREATE TABLE `appointments` (
    `Appointment_id` int(11) NOT NULL,
    `User_id` int(11) NOT NULL,
    `Day_id` int(11) NOT NULL,
    `Service_id` int(11) NOT NULL,
    `Hairdresser_id` int(11) NOT NULL,
    `Appointment_date` date NOT NULL,
    `Appointment_time` time NOT NULL,
    `Subject` varchar(150) NOT NULL,
    `Created_at` timestamp NOT NULL DEFAULT current_timestamp()
) ;

--
-- Déchargement des données de la table `appointments`
--

INSERT INTO `appointments` (`Appointment_id`, `User_id`,
`Day_id`, `Service_id`, `Hairdresser_id`, `Appointment_date`,
`Appointment_time`, `Subject`, `Created_at`) VALUES
(67, 1, 1, 6, 2, '2023-02-20', '15:00:00', 'Coupe de cheveux',
'2023-02-13 11:39:40'),
(68, 1, 1, 1, 3, '2023-02-21', '10:00:00', 'Coupe de cheveux',
'2023-02-13 11:39:40'),
(69, 1, 1, 2, 3, '2023-02-21', '11:00:00', 'Coupe de cheveux',
'2023-02-13 11:39:40'),
(70, 1, 2, 2, 2, '2023-02-22', '13:30:00', 'Coloration',
'2023-02-13 11:39:40'),
(71, 1, 1, 2, 3, '2023-03-04', '11:30:00', 'degrader', '2023-
03-02 10:17:12');

-- -----

--
-- Structure de la table `hairdressers`

```

```
--
CREATE TABLE `hairstressers` (
  `Hairstresser_id` int(11) NOT NULL,
  `User_id` int(11) NOT NULL,
  `Role` enum('Coiffeur') NOT NULL,
  `Created_at` timestamp NOT NULL DEFAULT current_timestamp()
) ;

--
-- Déchargement des données de la table `hairstressers`
--

INSERT INTO `hairstressers` (`Hairstresser_id`, `User_id`,
`Role`, `Created_at`) VALUES
(1, 4, 'Coiffeur', '2023-02-13 11:23:29'),
(2, 5, 'Coiffeur', '2023-02-13 11:23:29'),
(3, 6, 'Coiffeur', '2023-02-13 11:23:29'),
(4, 7, 'Coiffeur', '2023-02-13 11:23:29'),
(5, 8, 'Coiffeur', '2023-02-13 11:23:29'),
(6, 9, 'Coiffeur', '2023-02-13 11:23:29'),
(7, 10, 'Coiffeur', '2023-02-13 11:23:29');

-----

--
-- Structure de la table `opinions`
--

CREATE TABLE `opinions` (
  `Opinion_id` int(11) NOT NULL,
  `Note` int(11) NOT NULL,
  `Comments` text NOT NULL,
  `Appointment_id` int(11) NOT NULL,
  `Created_at` timestamp NOT NULL DEFAULT current_timestamp()
) ;

--
-- Déchargement des données de la table `opinions`
--

INSERT INTO `opinions` (`Opinion_id`, `Note`, `Comments`,
`Appointment_id`, `Created_at`) VALUES
(3, 4, 'Excellent service, je recommande vivement', 67, '2023-
02-13 11:44:47'),
(4, 5, 'Très bon service, je recommande', 68, '2023-02-13
11:44:47'),
(5, 3, 'Bon service mais pas exceptionnel', 69, '2023-02-13
11:44:47'),
(6, 3, 'Bon service mais pas exceptionnel', 70, '2023-02-13
11:44:47');
```

```

-- -----
--
-- Doublure de structure pour la vue `past_appointments_view`
-- (Voir ci-dessous la vue réelle)
--
CREATE TABLE `past_appointments_view` (
  `Appointment_id` int(11)
, `User_id` int(11)
, `Day_id` int(11)
, `Service_id` int(11)
, `Hairdresser_id` int(11)
, `Appointment_date` date
, `Appointment_time` time
, `Subject` varchar(150)
, `Created_at` timestamp
);

-- -----

--
-- Structure de la table `products`
--

CREATE TABLE `products` (
  `Product_id` int(11) NOT NULL,
  `Name` varchar(100) NOT NULL,
  `Pictures` varchar(255) NOT NULL,
  `Description` text NOT NULL,
  `Prices` decimal(10,2) NOT NULL
) ;

--
-- Déchargement des données de la table `products`
--

INSERT INTO `products` (`Product_id`, `Name`, `Pictures`,
`Description`, `Prices`) VALUES
(1, 'Gel', 'gel.jpg', 'Gel coiffant pour une tenue longue
durée', '8.99'),
(2, 'Gel', 'gel.jpg', 'Gel de douche pour une tenue longue
durée', '8.99'),
(3, 'Gel', 'gel.jpg', 'Gel coiffant pour une tenue longue
durée', '8.99'),
(4, 'Gel', 'gel.jpg', 'Gel coiffant pour une tenue longue
durée', '8.99'),
(5, 'Gel', 'gel.jpg', 'Gel coiffant pour une tenue longue
durée', '8.99'),
(6, 'Gel', 'gel.jpg', 'Gel coiffant pour une tenue longue
durée', '8.99'),

```

```

(7, 'Shampooing', 'shampoo.jpg', 'Shampooing nourrissant pour
cheveux secs', '10.50');

-- -----

--
-- Structure de la table `schedules`
--

CREATE TABLE `schedules` (
  `Schedule_id` int(11) NOT NULL,
  `Days`
enum('lundi','mardi','mercredi','jeudi','vendredi','samedi','d
imanche') NOT NULL,
  `Opening` time NOT NULL DEFAULT '09:00:00',
  `Closing` time NOT NULL DEFAULT '20:00:00'
) ;

--
-- Déchargement des données de la table `schedules`
--

INSERT INTO `schedules` (`Schedule_id`, `Days`, `Opening`,
`Closing`) VALUES
(1, 'lundi', '09:00:00', '19:00:00'),
(2, 'mardi', '09:00:00', '19:00:00'),
(3, 'mercredi', '09:00:00', '19:00:00'),
(4, 'jeudi', '09:00:00', '19:00:00'),
(5, 'vendredi', '09:00:00', '19:00:00'),
(6, 'samedi', '09:00:00', '19:00:00'),
(7, 'dimanche', '09:00:00', '19:00:00');

-- -----

--
-- Structure de la table `services`
--

CREATE TABLE `services` (
  `Service_id` int(11) NOT NULL,
  `Description` varchar(155) NOT NULL,
  `Duration` int(11) NOT NULL
) ;

--
-- Déchargement des données de la table `services`
--

INSERT INTO `services` (`Service_id`, `Description`,
`Duration`) VALUES
(1, 'Coupe Homme', 30),

```



```

(2, 'Coupe Femme', 60),
(3, 'Couleur', 90),
(4, 'Brushing', 45),
(5, 'Chignon', 120),
(6, 'Mèches', 90);

-- -----

--
-- Doublure de structure pour la vue `upcoming_appointments`
-- (Voir ci-dessous la vue réelle)
--
CREATE TABLE `upcoming_appointments` (
  `Appointment_id` int(11)
, `First_name` varchar(100)
, `Last_name` varchar(100)
, `Description` varchar(155)
, `Appointment_date` date
, `Appointment_time` time
, `Subject` varchar(150)
, `Hairdresser_id` int(11)
);

-- -----

--
-- Doublure de structure pour la vue
-- `upcoming_appointments_user`
-- (Voir ci-dessous la vue réelle)
--
CREATE TABLE `upcoming_appointments_user` (
  `Appointment_id` int(11)
, `First_name` varchar(100)
, `Last_name` varchar(100)
, `Appointment_date` date
, `Appointment_time` time
, `Hairdresser` varchar(155)
);

-- -----

--
-- Structure de la table `users`
--
CREATE TABLE `users` (
  `User_id` int(11) NOT NULL,
  `User_Username` varchar(50) NOT NULL,
  `Last_name` varchar(100) NOT NULL,
  `First_name` varchar(100) NOT NULL,
  `Password` varchar(50) NOT NULL,

```

```

    `Email` varchar(200) NOT NULL,
    `Role` enum('Client','Administrateur','Coiffeur') NOT NULL,
    `Created_at` timestamp NOT NULL DEFAULT current_timestamp()
) ;

--
-- Déchargement des données de la table `users`
--

INSERT INTO `users` (`User_id`, `User_Username`, `Last_name`,
`First_name`, `Password`, `Email`, `Role`, `Created_at`)
VALUES
(1, 'johndoe', 'Doe', 'John', '123456', 'johndoe@example.com',
'Client', '2023-02-13 11:20:45'),
(2, 'janedoe', 'Doe', 'Jane', '654321', 'janedoe@example.com',
'Client', '2023-02-13 11:20:45'),
(3, 'admin', 'Admin', 'Super', 'admin123',
'admin@example.com', 'Administrateur', '2023-02-13 11:20:45'),
(4, 'hairdresser1', 'Pierre', 'Jean', 'abcdef',
'jeanpierre@example.com', 'Coiffeur', '2023-02-13 11:20:45'),
(5, 'hairdresser2', 'Martin', 'Lucie', 'ghijkl',
'luciemartin@example.com', 'Coiffeur', '2023-02-13 11:20:45'),
(6, 'hairdresser3', 'Dupont', 'Pierre', 'mnopqr',
'pierredupont@example.com', 'Coiffeur', '2023-02-13
11:20:45'),
(7, 'hairdresser4', 'Durand', 'Sophie', 'stuvwx',
'sophiedurand@example.com', 'Coiffeur', '2023-02-13
11:20:45'),
(8, 'hairdresser5', 'Lefebvre', 'Maxime', 'yzabcd',
'maximelefebvre@example.com', 'Coiffeur', '2023-02-13
11:20:45'),
(9, 'hairdresser6', 'Rousseau', 'Lucas', 'efghij',
'lucasrousseau@example.com', 'Coiffeur', '2023-02-13
11:20:45'),
(10, 'hairdresser7', 'Leblanc', 'Marie', 'klmnop',
'marieleblanc@example.com', 'Coiffeur', '2023-02-13
11:20:45');

-- -----
--
--
-- Doublure de structure pour la vue `user_roles`
-- (Voir ci-dessous la vue réelle)
--

CREATE TABLE `user_roles` (
`User_id` int(11)
, `Last_name` varchar(100)
, `First_name` varchar(100)
, `Role` varchar(14)
);

```

```

-- -----
--
-- Structure de la vue `past_appointments_view`
--
DROP TABLE IF EXISTS `past_appointments_view`;

CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL
SECURITY DEFINER VIEW `past_appointments_view` AS SELECT
`appointments`.`Appointment_id` AS `Appointment_id`,
`appointments`.`User_id` AS `User_id`, `appointments`.`Day_id`
AS `Day_id`, `appointments`.`Service_id` AS `Service_id`,
`appointments`.`Hairdresser_id` AS `Hairdresser_id`,
`appointments`.`Appointment_date` AS `Appointment_date`,
`appointments`.`Appointment_time` AS `Appointment_time`,
`appointments`.`Subject` AS `Subject`,
`appointments`.`Created_at` AS `Created_at` FROM
`appointments` WHERE `appointments`.`User_id` =
`appointments`.`User_id` AND `appointments`.`Appointment_date`
< curdate() ;

-- -----
--
-- Structure de la vue `upcoming_appointments`
--
DROP TABLE IF EXISTS `upcoming_appointments`;

CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL
SECURITY DEFINER VIEW `upcoming_appointments` AS SELECT
`a`.`Appointment_id` AS `Appointment_id`, `u`.`First_name` AS
`First_name`, `u`.`Last_name` AS `Last_name`,
`s`.`Description` AS `Description`, `a`.`Appointment_date` AS
`Appointment_date`, `a`.`Appointment_time` AS
`Appointment_time`, `a`.`Subject` AS `Subject`,
`a`.`Hairdresser_id` AS `Hairdresser_id` FROM ((`appointments`
`a` join `users` `u` on(`a`.`User_id` = `u`.`User_id`)) join
`services` `s` on(`a`.`Service_id` = `s`.`Service_id`)) WHERE
`a`.`Appointment_date` >= curdate() ORDER BY
`a`.`Appointment_date` ASC, `a`.`Appointment_time` ASC ;

-- -----
--
-- Structure de la vue `upcoming_appointments_user`
--
DROP TABLE IF EXISTS `upcoming_appointments_user`;

CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL
SECURITY DEFINER VIEW `upcoming_appointments_user` AS SELECT
`a`.`Appointment_id` AS `Appointment_id`, `u`.`First_name` AS

```

```

`First_name`, `u`.`Last_name` AS `Last_name`,
`a`.`Appointment_date` AS `Appointment_date`,
`a`.`Appointment_time` AS `Appointment_time`,
`s`.`Description` AS `Hairdresser` FROM (((`appointments` `a`
join `users` `u` on(`a`.`User_id` = `u`.`User_id`)) join
`s` `s` on(`a`.`Service_id` = `s`.`Service_id`)) join
`hairdressers` `h` on(`a`.`Hairdresser_id` =
`h`.`Hairdresser_id`)) WHERE `a`.`Appointment_date` >=
curdate() ORDER BY `a`.`Appointment_date` ASC,
`a`.`Appointment_time` ASC ;

-- -----

--
-- Structure de la vue `user_roles`
--
DROP TABLE IF EXISTS `user_roles`;

CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL
SECURITY DEFINER VIEW `user_roles` AS SELECT `u`.`User_id` AS
`User_id`, `u`.`Last_name` AS `Last_name`, `u`.`First_name` AS
`First_name`, `u`.`Role` AS `Role` FROM `users` AS `u` union
select `h`.`Hairdresser_id` AS
`Hairdresser_id`, `u`.`Last_name` AS
`Last_name`, `u`.`First_name` AS `First_name`, `u`.`Role` AS
`Role` from (`hairdressers` `h` join `users` `u`
on(`h`.`User_id` = `u`.`User_id`)) union select
`a`.`Administrator_id` AS `Administrator_id`, `u`.`Last_name`
AS `Last_name`, `u`.`First_name` AS `First_name`, `u`.`Role` AS
`Role` from (`administrators` `a` join `users` `u`
on(`a`.`User_id` = `u`.`User_id`)) ;

--
-- Index pour les tables déchargées
--

--
-- Index pour la table `administrators`
--
ALTER TABLE `administrators`
  ADD PRIMARY KEY (`Administrator_id`),
  ADD KEY `User_id` (`User_id`);

--
-- Index pour la table `appointments`
--
ALTER TABLE `appointments`
  ADD PRIMARY KEY (`Appointment_id`),
  ADD UNIQUE KEY `Appointment_date`
(`Appointment_date`, `Appointment_time`, `Hairdresser_id`),
  ADD KEY `User_id` (`User_id`),

```

```

    ADD KEY `Hairdresser_id` (`Hairdresser_id`),
    ADD KEY `Service_id` (`Service_id`),
    ADD KEY `Day_id` (`Day_id`),
    ADD KEY `idx_rendezvous_sujet` (`Subject`);

--
-- Index pour la table `hairdressers`
--
ALTER TABLE `hairdressers`
    ADD PRIMARY KEY (`Hairdresser_id`),
    ADD KEY `User_id` (`User_id`);

--
-- Index pour la table `opinions`
--
ALTER TABLE `opinions`
    ADD PRIMARY KEY (`Opinion_id`),
    ADD KEY `Appointment_id` (`Appointment_id`);

--
-- Index pour la table `products`
--
ALTER TABLE `products`
    ADD PRIMARY KEY (`Product_id`);

--
-- Index pour la table `schedules`
--
ALTER TABLE `schedules`
    ADD PRIMARY KEY (`Schedule_id`);

--
-- Index pour la table `services`
--
ALTER TABLE `services`
    ADD PRIMARY KEY (`Service_id`);

--
-- Index pour la table `users`
--
ALTER TABLE `users`
    ADD PRIMARY KEY (`User_id`),
    ADD KEY `idx_utilisateurs_nom_prenom`
    (`Last_name`, `First_name`);

--
-- AUTO_INCREMENT pour les tables déchargées
--

--
-- AUTO_INCREMENT pour la table `appointments`

```

```

--
ALTER TABLE `appointments`
  MODIFY `Appointment_id` int(11) NOT NULL AUTO_INCREMENT;

--
-- AUTO_INCREMENT pour la table `opinions`
--
ALTER TABLE `opinions`
  MODIFY `Opinion_id` int(11) NOT NULL AUTO_INCREMENT;

--
-- AUTO_INCREMENT pour la table `products`
--
ALTER TABLE `products`
  MODIFY `Product_id` int(11) NOT NULL AUTO_INCREMENT;

--
-- AUTO_INCREMENT pour la table `schedules`
--
ALTER TABLE `schedules`
  MODIFY `Schedule_id` int(11) NOT NULL AUTO_INCREMENT;

--
-- AUTO_INCREMENT pour la table `services`
--
ALTER TABLE `services`
  MODIFY `Service_id` int(11) NOT NULL AUTO_INCREMENT;

--
-- AUTO_INCREMENT pour la table `users`
--
ALTER TABLE `users`
  MODIFY `User_id` int(11) NOT NULL AUTO_INCREMENT;

--
-- Contraintes pour les tables déchargées
--

--
-- Contraintes pour la table `administrators`
--
ALTER TABLE `administrators`
  ADD CONSTRAINT `administrators_ibfk_1` FOREIGN KEY
  (`User_id`) REFERENCES `users` (`User_id`);

--
-- Contraintes pour la table `appointments`
--
ALTER TABLE `appointments`
  ADD CONSTRAINT `appointments_ibfk_1` FOREIGN KEY (`User_id`)
REFERENCES `users` (`User_id`) ON DELETE CASCADE,

```

```

    ADD CONSTRAINT `appointments_ibfk_2` FOREIGN KEY
(`Hairdresser_id`) REFERENCES `hairdressers`
(`Hairdresser_id`) ON DELETE CASCADE,
    ADD CONSTRAINT `appointments_ibfk_3` FOREIGN KEY
(`Service_id`) REFERENCES `services` (`Service_id`) ON DELETE
CASCADE,
    ADD CONSTRAINT `appointments_ibfk_4` FOREIGN KEY (`Day_id`)
REFERENCES `schedules` (`Schedule_id`) ON DELETE CASCADE;

--
-- Contraintes pour la table `hairdressers`
--
ALTER TABLE `hairdressers`
    ADD CONSTRAINT `hairdressers_ibfk_1` FOREIGN KEY (`User_id`)
REFERENCES `users` (`User_id`);

--
-- Contraintes pour la table `opinions`
--
ALTER TABLE `opinions`
    ADD CONSTRAINT `opinions_ibfk_1` FOREIGN KEY
(`Appointment_id`) REFERENCES `appointments`
(`Appointment_id`) ON DELETE CASCADE;
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT
*/;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS
*/;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION
*/;

```

## Tests :

✓ Affichage des lignes 0 - 0 (total de 1, traitement en 0,0004 seconde(s).)

`SELECT * FROM `administrators`;`

☐ Profilage [\[ Éditer en ligne \]](#) [\[ Éditer \]](#) [\[ Expliquer SQL \]](#) [\[ Créer le code source PHP \]](#) [\[ Actualiser \]](#)

☐ Tout afficher | Nombre de lignes : 25 ▼ | Filtrer les lignes:

[Options supplémentaires](#)

	Administrator_id	User_id	Role	Created_at
<input type="checkbox"/> <a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	1	3	Administrateur	2023-02-13 12:25:08

✓ Affichage des lignes 0 - 4 (total de 5, traitement en 0.0004 seconde(s).)

SELECT \* FROM `appointements` WHERE 1;

Profilage [ Éditer en ligne ][ Éditer ][ Expliquer SQL ][ Créer le code source PHP ][ Actualiser ]

☐ Tout afficher

Nombre de lignes : 25

Filter les lignes: Chercher dans cette table

Trier par clé : Aucun(e)

Options supplémentaires

	Appointment_id	User_id	Day_id	Service_id	Hairdresser_id	Appointment_date	Appointment_time	Subject	Created_at
<input type="checkbox"/> Éditer Copier Supprimer	67	1	1	6	2	2023-02-20	15:00:00	Coupe de cheveux	2023-02-13 12:39:40
<input type="checkbox"/> Éditer Copier Supprimer	68	1	1	1	3	2023-02-21	10:00:00	Coupe de cheveux	2023-02-13 12:39:40
<input type="checkbox"/> Éditer Copier Supprimer	69	1	1	2	3	2023-02-21	11:00:00	Coupe de cheveux	2023-02-13 12:39:40
<input type="checkbox"/> Éditer Copier Supprimer	70	1	2	2	2	2023-02-22	13:30:00	Coloration	2023-02-13 12:39:40
<input type="checkbox"/> Éditer Copier Supprimer	71	1	1	2	3	2023-03-04	11:30:00	dégrader	2023-03-02 11:17:12

SELECT \* FROM `hairdressers` WHERE 1;

Profilage [ Éditer en ligne ][ Éditer ][ Expliquer SQL ][ Créer le code source PHP ][ Actualiser ]

☐ Tout afficher

Nombre de lignes : 25

Filter les lignes: Chercher dans cette table

Trier par clé : Aucun(e)

Options supplémentaires

	Hairdresser_id	User_id	Role	Created_at
<input type="checkbox"/> Éditer Copier Supprimer	1	4	Coiffeur	2023-02-13 12:23:29
<input type="checkbox"/> Éditer Copier Supprimer	2	5	Coiffeur	2023-02-13 12:23:29
<input type="checkbox"/> Éditer Copier Supprimer	3	6	Coiffeur	2023-02-13 12:23:29
<input type="checkbox"/> Éditer Copier Supprimer	4	7	Coiffeur	2023-02-13 12:23:29
<input type="checkbox"/> Éditer Copier Supprimer	5	8	Coiffeur	2023-02-13 12:23:29
<input type="checkbox"/> Éditer Copier Supprimer	6	9	Coiffeur	2023-02-13 12:23:29
<input type="checkbox"/> Éditer Copier Supprimer	7	10	Coiffeur	2023-02-13 12:23:29

SELECT \* FROM `opinions`

Profilage [ Éditer en ligne ][ Éditer ][ Expliquer SQL ][ Créer le code source PHP ][ Actualiser ]

☐ Tout afficher

Nombre de lignes : 25

Filter les lignes: Chercher dans cette table

Trier par clé : Aucun(e)

Options supplémentaires

	Opinion_id	Note	Comments	Appointment_id	Created_at
<input type="checkbox"/> Éditer Copier Supprimer	3	4	Excellent service, je recommande vivement	67	2023-02-13 12:44:47
<input type="checkbox"/> Éditer Copier Supprimer	4	5	Très bon service, je recommande	68	2023-02-13 12:44:47
<input type="checkbox"/> Éditer Copier Supprimer	5	3	Bon service mais pas exceptionnel	69	2023-02-13 12:44:47
<input type="checkbox"/> Éditer Copier Supprimer	6	3	Bon service mais pas exceptionnel	70	2023-02-13 12:44:47

SELECT \* FROM `products`

Profilage [ Éditer en ligne ][ Éditer ][ Expliquer SQL ][ Créer le code source PHP ][ Actualiser ]

☐ Tout afficher

Nombre de lignes : 25

Filter les lignes: Chercher dans cette table

Trier par clé : Aucun(e)

Options supplémentaires

	Product_id	Name	Pictures	Description	Prices
<input type="checkbox"/> Éditer Copier Supprimer	1	Gel	gel.jpg	Gel coiffant pour une tenue longue durée	8.99
<input type="checkbox"/> Éditer Copier Supprimer	2	Gel	gel.jpg	Gel de douche pour une tenue longue durée	8.99
<input type="checkbox"/> Éditer Copier Supprimer	3	Gel	gel.jpg	Gel coiffant pour une tenue longue durée	8.99
<input type="checkbox"/> Éditer Copier Supprimer	4	Gel	gel.jpg	Gel coiffant pour une tenue longue durée	8.99
<input type="checkbox"/> Éditer Copier Supprimer	5	Gel	gel.jpg	Gel coiffant pour une tenue longue durée	8.99
<input type="checkbox"/> Éditer Copier Supprimer	6	Gel	gel.jpg	Gel coiffant pour une tenue longue durée	8.99
<input type="checkbox"/> Éditer Copier Supprimer	7	Shampooing	shampoo.jpg	Shampooing nourrissant pour cheveux secs	10.50

SELECT \* FROM `schedules`

Profilage [ Éditer en ligne ][ Éditer ][ Expliquer SQL ][ Créer le code source PHP ][ Actualiser ]

☐ Tout afficher

Nombre de lignes : 25

Filter les lignes: Chercher dans cette table

Trier par clé : Aucun(e)

Options supplémentaires

	Schedule_id	Days	Opening	Closing
<input type="checkbox"/> Éditer Copier Supprimer	1	lundi	09:00:00	19:00:00
<input type="checkbox"/> Éditer Copier Supprimer	2	mardi	09:00:00	19:00:00
<input type="checkbox"/> Éditer Copier Supprimer	3	mercredi	09:00:00	19:00:00
<input type="checkbox"/> Éditer Copier Supprimer	4	jeudi	09:00:00	19:00:00
<input type="checkbox"/> Éditer Copier Supprimer	5	vendredi	09:00:00	19:00:00
<input type="checkbox"/> Éditer Copier Supprimer	6	samedi	09:00:00	19:00:00
<input type="checkbox"/> Éditer Copier Supprimer	7	dimanche	09:00:00	19:00:00



`SELECT * FROM `services``

☐ Profilage [\[ Éditer en ligne \]](#) [\[ Éditer \]](#) [\[ Expliquer SQL \]](#) [\[ Créer le code source PHP \]](#) [\[ Actualiser \]](#)

☐ Tout afficher | Nombre de lignes : 25 | Filtrer les lignes: Chercher dans cette table | Trier par clé : Aucun(e)

Options supplémentaires

		Service_id	Description	Duration
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	1	Coupe Homme	30
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	2	Coupe Femme	60
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	3	Couleur	90
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	4	Brushing	45
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	5	Chignon	120
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	6	Mèches	90

`SELECT * FROM `users``

☐ Profilage [\[ Éditer en ligne \]](#) [\[ Éditer \]](#) [\[ Expliquer SQL \]](#) [\[ Créer le code source PHP \]](#) [\[ Actualiser \]](#)

☐ Tout afficher | Nombre de lignes : 25 | Filtrer les lignes: Chercher dans cette table | Trier par clé : Aucun(e)

Options supplémentaires

		User_id	User_Username	Last_name	First_name	Password	Email	Role	Created_at
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	1	john doe	Doe	John	123456	john doe@example.com	Client	2023-02-13 12:20:45
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	2	janedoe	Doe	Jane	654321	janedoe@example.com	Client	2023-02-13 12:20:45
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	3	admin	Admin	Super	admin123	admin@example.com	Administrateur	2023-02-13 12:20:45
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	4	hairstresser1	Pierre	Jean	abodef	jeanpierre@example.com	Coiffeur	2023-02-13 12:20:45
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	5	hairstresser2	Martin	Luicie	ghijkl	luiciemartin@example.com	Coiffeur	2023-02-13 12:20:45
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	6	hairstresser3	Dupont	Pierre	mnopqr	pierredupont@example.com	Coiffeur	2023-02-13 12:20:45
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	7	hairstresser4	Durand	Sophie	stuvwxyz	sophiedurand@example.com	Coiffeur	2023-02-13 12:20:45
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	8	hairstresser5	Lefebvre	Maxime	yzabod	maximelefebvre@example.com	Coiffeur	2023-02-13 12:20:45
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	9	hairstresser6	Rousseau	Lucas	efghij	lucasrousseau@example.com	Coiffeur	2023-02-13 12:20:45
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	10	hairstresser7	Leblanc	Marie	klmnop	manieleblanc@example.com	Coiffeur	2023-02-13 12:20:45

Vues des rendez-vous déjà passer :

`SELECT * FROM `past_appointments_view``

☐ Profilage [\[ Éditer en ligne \]](#) [\[ Éditer \]](#) [\[ Expliquer SQL \]](#) [\[ Créer le code source PHP \]](#) [\[ Actualiser \]](#)

☐ Tout afficher | Nombre de lignes : 25 | Filtrer les lignes: Chercher dans cette table

Options supplémentaires

		Appointment_id	User_id	Day_id	Service_id	Hairdresser_id	Appointment_date	Appointment_time	Subject	Created_at
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	67	1	1	6	2	2023-02-20	15:00:00	Coupe de cheveux	2023-02-13 12:39:40
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	68	1	1	1	3	2023-02-21	10:00:00	Coupe de cheveux	2023-02-13 12:39:40
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	69	1	1	2	3	2023-02-21	11:00:00	Coupe de cheveux	2023-02-13 12:39:40
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	70	1	2	2	2	2023-02-22	13:30:00	Coloration	2023-02-13 12:39:40
<input type="checkbox"/>	<a href="#">Éditer</a> <a href="#">Copier</a> <a href="#">Supprimer</a>	71	1	1	2	3	2023-03-04	11:30:00	degrader	2023-03-02 11:17:12

Vues des futurs rendez-vous :

✓ MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0.0005 seconde(s).)

`SELECT * FROM `upcoming_appointments``

☐ Profilage [\[ Éditer en ligne \]](#) [\[ Éditer \]](#) [\[ Expliquer SQL \]](#) [\[ Créer le code source PHP \]](#) [\[ Actualiser \]](#)

Appointment_id	First_name	Last_name	Description	Appointment_date	Appointment_time	Subject	Hairdresser_id
----------------	------------	-----------	-------------	------------------	------------------	---------	----------------

Opérations sur les résultats de la requête

Il y'a pas de futurs rendez-vous.

J'ai choisi la version 2 parce que c'est celle qui fonctionne le mieux et parce que durant les tests dans la version 1, il y'avait un bug table Rôle qui me faisait perdre du temps, admin\_id ne pouvait être nul ne pouvait être nul alors que je l'ai initialisé à nul.

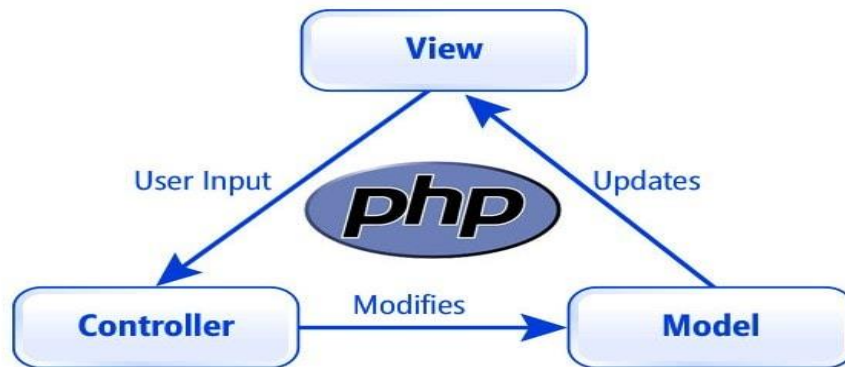
## Formation PHP MVC :

### 1.1. Comprendre le PHP MVC :

#### C'est quoi au juste ?

MVC (Modèle-Vue-Contrôleur) est une architecture de conception de logiciels qui est souvent utilisée dans le développement de sites Web en PHP.

Concrètement, l'architecture MVC sépare la logique de présentation (Vue) de la logique métier (Modèle) et de la logique de contrôle (Contrôleur). Cela permet de mieux organiser le code et de faciliter la maintenance et l'évolutivité de l'application.



Le Modèle est responsable de la manipulation des données, telles que la récupération des données à partir d'une base de données et la mise à jour de ces données. La Vue est responsable de l'affichage des données à l'utilisateur final. Enfin, le Contrôleur fait le lien entre le Modèle et la Vue, en traitant les entrées de l'utilisateur et en décidant comment répondre à ces entrées.

#### Pourquoi l'utiliser ?

En utilisant l'architecture MVC, les développeurs peuvent facilement modifier et ajouter des fonctionnalités sans avoir à toucher à tout le code, car chaque partie est indépendante et peut être modifiée sans affecter les autres parties. De plus, la séparation des préoccupations facilite également les tests unitaires et l'identification rapide des erreurs ou des bogues.

### 1.2. Comment je me suis formé :

- Le modèle mvc je l'ai commencé à l'apprendre en cours sur un site de restauration :

# Architecture MVC en PHP - Contrôleur principal

- Partie 1 : généralités
- Partie 2 : contrôleur
- Partie 3 : vue
- Partie 4 : modèle
- Partie 5 : contrôleur principal

## Fonctionnement du contrôleur principal dans le patron de conception MVC

### Rappel du contexte

R3st0.fr est un site web de critique de restaurant. À l'image des sites de ce type il a pour vocation le recensement des avis des consommateurs et la diffusion de ces avis aux visiteurs.

Ce site web est développé en PHP en suivant le patron de conception "modèle vue contrôleur" (pattern MVC). L'architecture retenue pour ce projet permet d'appréhender la programmation web d'une manière structurée.

L'objectif de ce document est d'analyser le fonctionnement du contrôleur principal dans l'architecture MVC puis d'intégrer de nouveaux modules fournis.

### Ressources à utiliser

- Dossier "base de données" : fichier [base.sql](#) contenant les tables et les données de la [base utilisée](#) par l'application web étudiée.
- Dossier site : arborescence et fichiers de l'application web.

Après avoir créé la base de données (encodage utf8mb4) et importé le fichier [base.sql](#), créer un nouveau projet PHP appelé Sl6MVCTP1 dans [Netbeans](#). Paramétrer le projet pour que celui-ci soit [téléchargé sur](#) le serveur lors de l'exécution.

Dans le gestionnaire de fichiers, supprimer le fichier [index.php](#) créé automatiquement, et remplacer le contenu du dossier "Source Files" par le contenu du dossier "site" fourni.

L'arborescence devrait être celle-ci :

Source Files

### Synthèse

La variable `$lesActions` de la fonction `contrôleurPrincipal()` contient l'ensemble des contrôleurs accessibles sur le site. Un contrôleur est l'élément central d'une fonctionnalité. Lorsqu'une nouvelle fonctionnalité est développée sur l'application, il faut l'intégrer au contrôleur principal en ajoutant une nouvelle valeur dans la variable `$lesActions`.

Chaque clé dans la variable `$lesActions` est associée et correspond à un script contrôleur.



Lorsqu'on ajoute une nouvelle fonctionnalité dans l'application, il faut créer le script contrôleur dans le dossier du même nom, puis déclarer cette nouvelle fonctionnalité dans la fonction `contrôleurPrincipal()` en ajoutant l'action correspondante (clé) et le fichier associé.

L'instruction ci-dessous permet d'ajouter une nouvelle action associée à un contrôleur dans la variable `$lesActions`.  
`$lesActions['uneAction'] = "scriptContrôleur.php";`

Accéder à une fonctionnalité sur l'application web  
Le fichier `index.php` est la porte d'entrée pour accéder aux différentes fonctionnalités. C'est lui qui réceptionne l'action envoyée en méthode GET puis charge le contrôleur associé. Dès lors, la navigation sur l'application web ne fait apparaître que le fichier `index.php` dans l'URL. Chaque fonctionnalité est demandée par l'intermédiaire de la variable action transmise.

Lorsqu'on souhaite accéder à une fonctionnalité par l'intermédiaire d'un lien, l'URL pointée doit faire référence à l'action souhaitée.



Page 4/11

### Accès au contrôleur de connexion



Le nom du fichier contrôleur est masqué à l'utilisateur qui ne voit que l'action.

Dans l'exemple ci-dessus, le nom du fichier `index.php` n'est pas visible, on peut soit l'indiquer, soit faire référence à l'emploiement " " qui désigne le fichier `index.php` dans la configuration du serveur web [Apache](#).

Les contrôleurs ne sont jamais directement appelés donc l'URL, on doit toujours passer par le contrôleur principal, et donc par `index.php`. Il en est de même avec les formulaires, comme celui de connexion.



Extrait de la vue affichant le formulaire de connexion

On remarque que les données du formulaire sont transmises en méthode POST à `index.php`. Lors de l'inclusion du fichier contrôleur (`connexion.php`), ces données seront aussi accessibles dans le contrôleur.

Pour intégrer une nouvelle fonctionnalité au site web, il faut donc :

- créer un nouveau contrôleur dans le dossier approprié,
- relier la vue et le modèle associé si besoin,
- ajouter une action dans la variable `$lesActions` associant le nom de l'action au nom du script contrôleur,
- donner l'accès à ce contrôleur par l'intermédiaire d'un lien ou d'un formulaire transmettant l'action en méthode GET.

### B - Intégration de contrôleurs pré-existants

#### Question 2 - CGU

Documents à utiliser  
- fichiers fournis en ressources  
- Annexe 4

Page 5/11

Les conditions générales d'utilisations ont déjà été écrites. La vue et le contrôleur associés sont disponibles. De même, le menu général propose un lien vers [pages 2](#).

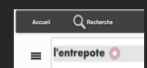
Lorsque l'on clique sur ce lien les CGU ne sont pas affichées, le contrôleur par défaut est chargé à la place.

- 2.1. Repérer dans le menu général l'action correspondant au contrôleur ayant en charge l'affichage des CGU
- 2.2. Placer les fichiers fournis en ressource dans les dossiers appropriés.
- 2.3. Rédiger l'instruction à ajouter à la fonction `contrôleurPrincipal()` pour ajouter la nouvelle action dans la variable `$lesActions`.
- 2.4. Ajouter la nouvelle action à la fonction puis tester le bon fonctionnement du lien CGU dans le menu général.

### Question 3 : aimer un restaurant

Documents à utiliser  
- fichiers fournis en ressources  
- Annexe 5

Lorsqu'un utilisateur connecté consulte la fiche descriptive d'un restaurant, il a la possibilité d'ajouter celui-ci dans la liste des restaurants qu'il aime. Cette action se fait en cliquant sur l'étoile située à droite du nom du restaurant. Tant que l'utilisateur n'a pas aimé ce restaurant, l'étoile est grisée.



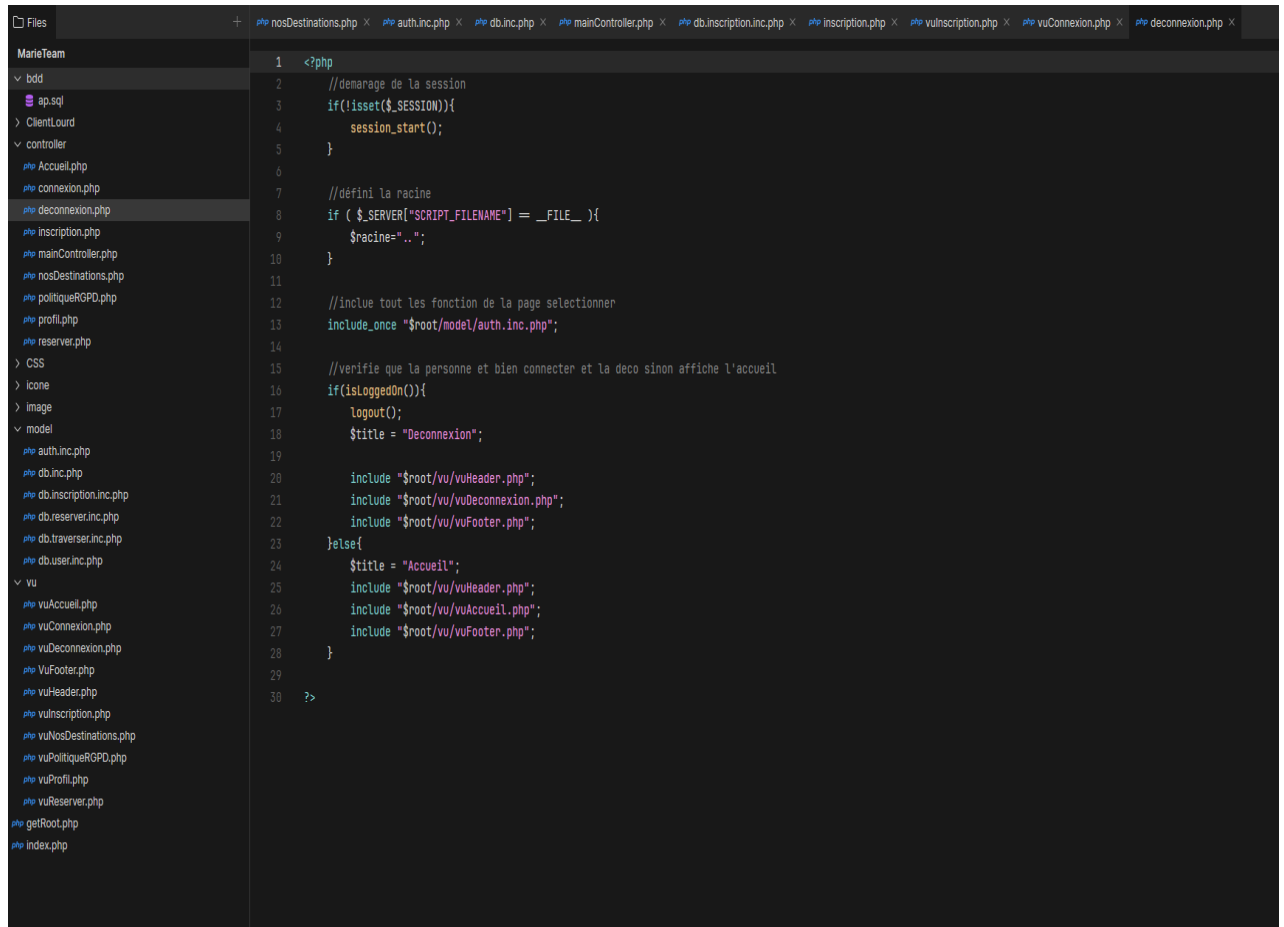
Lien sous forme d'une étoile pour aimer un restaurant

Lorsqu'on clique sur ce lien, le traitement n'est pas effectué, rien n'est enregistré dans la table aimer. Le contrôleur par défaut est chargé à la place.

- 3.1. Quelle vue permet d'afficher la fiche descriptive d'un restaurant ?
- 3.2. Repérer dans le code de la vue le lien correspondant à l'étoile.
- 3.3. Quels sont les paramètres envoyés en méthode GET lorsque l'on clique sur le lien ? Préciser le nom et la valeur de chaque paramètre.
- 3.4. À partir de vos connaissances et du script contrôleur fourni en ressource, déterminer dans quels scripts sont utilisés chaque des deux variables transmises par le lien en méthode GET (celles trouvées à la question précédente).
- 3.5. Placer le fichier fourni en ressource dans le dossier approprié.
- 3.6. Rédiger l'instruction à ajouter à la fonction `contrôleurPrincipal()` pour ajouter la nouvelle action dans la variable `$lesActions`.
- 3.7. Ajouter la nouvelle action à la fonction puis tester le bon fonctionnement du contrôleur en cliquant sur l'étoile. (Pour tester, il faut être authentifié sur le site)

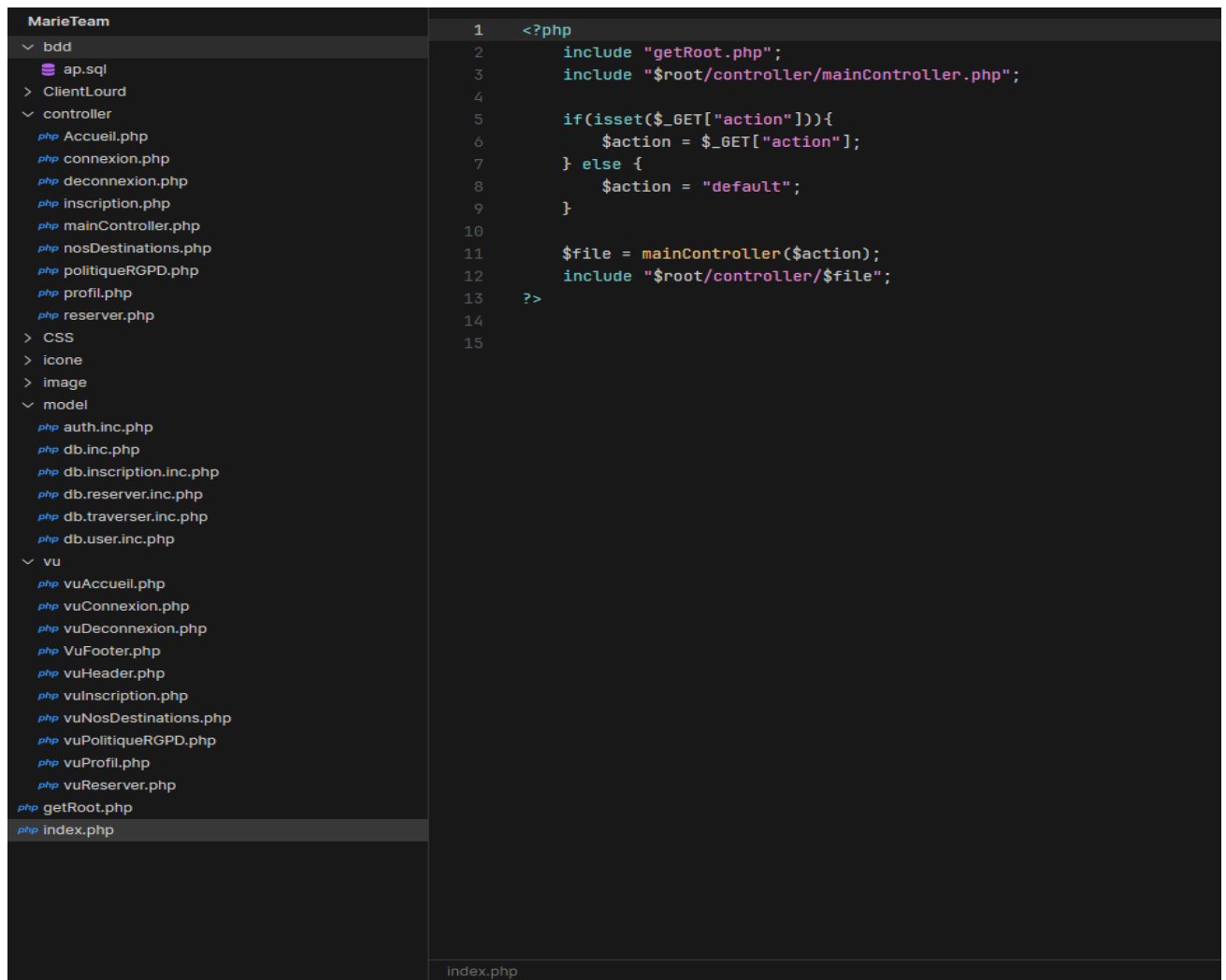
Page 6/11

- Mais aussi notre projet d'atelier professionnel :



The image shows a code editor interface with a file explorer on the left and a code editor on the right. The file explorer shows a project structure for 'MarieTeam' with folders like 'bdd', 'ClientLourd', 'controller', 'CSS', 'icone', 'image', 'model', and 'vu'. The 'controller' folder is expanded, showing files like 'Accueil.php', 'connexion.php', 'deconnexion.php', 'inscription.php', 'mainController.php', 'nosDestinations.php', 'politiqueRGPD.php', 'profil.php', and 'reserver.php'. The 'deconnexion.php' file is selected. The code editor shows the following PHP code:

```
1 <?php
2 //démarrage de la session
3 if(!isset($_SESSION)){
4     session_start();
5 }
6
7 //défini la racine
8 if ( $_SERVER["SCRIPT_FILENAME"] == __FILE__ ){
9     $racine="..";
10 }
11
12 //inclue tout les fonction de la page selectionner
13 include_once "$root/model/auth.inc.php";
14
15 //verifie que la personne et bien connecter et la deco sinon affiche l'accueil
16 if(!isLoggedIn()){
17     logout();
18     $title = "Deconnexion";
19
20     include "$root/vu/vuHeader.php";
21     include "$root/vu/vuDeconnexion.php";
22     include "$root/vu/vuFooter.php";
23 }else{
24     $title = "Accueil";
25     include "$root/vu/vuHeader.php";
26     include "$root/vu/vuAccueil.php";
27     include "$root/vu/vuFooter.php";
28 }
29
30 ?>
```



- Mais durant le stage j'ai aussi suivi une formation gratuite sur OpenClassRooms :

# Adoptez une architecture MVC en PHP

⌚ 8 heures 📶 Moyenne

Licence

Mis à jour le 25/05/2022



OpenClassrooms de OpenClassrooms... Il y a 6j

jour Luciano,

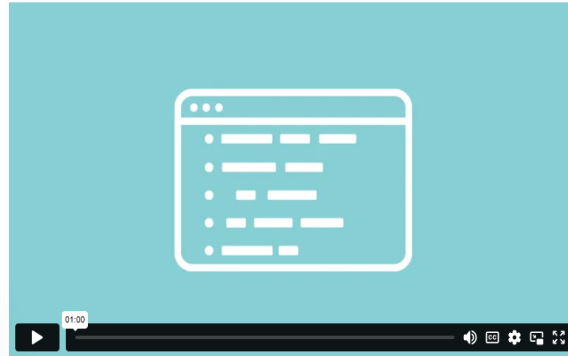
quel emploi, stage rémunéré ou non ? Augmentation salariale, promotion ou mobilité professionnelle ? Situation d'entreprise ou freelance ?

avez-vous bénéficié d'une ou plusieurs des évolutions positives au cours des 12 derniers mois de votre vie professionnelle grâce à OpenClassrooms ?

QUI

NON

Je ne souhaite pas répondre



Créé par



OpenClassrooms, Leading  
E-Learning Platform



# Vous connaissez les bases de la programmation en PHP ? Et vous voulez aller plus loin ?

Comment font les professionnels ? Quelle structure de code adoptent-ils ?

Ils utilisent des concepts de programmation plus avancés, comme l'architecture **Modèle-Vue-Contrôleur**, la programmation orientée objet (POO) et bien d'autres choses... Ce sont des techniques que nous allons découvrir pas à pas dans ce cours, sur la base d'un projet concret.

## Découvrez du code professionnel



Très souvent, j'entends des débutants me dire :

- "Où doit-on aller après avoir suivi ton cours sur PHP ?"
- "Comment est-ce que je fais pour avoir une structure propre pour mon code, comme les pros ?"

Bonne nouvelle, ce cours va vous apporter la réponse à cette grande question. Pas *toute* la réponse bien sûr, car on n'a jamais fini d'apprendre... Mais au moins un bon début de réponse.

Durant tout ce cours, on va suivre ensemble un projet fil rouge : c'est votre première semaine dans une agence de développement, où vous venez d'être accepté en stage.

Félicitations !

Votre chef de projet vous confie d'emblée une mission. Il va falloir reprendre le blog de l'Association de Volley-Ball de Nuelly, l'AVBN. À l'époque, il avait été fait par un développeur amateur, mais maintenant que le club vient de monter de division, il leur faut un outil un peu plus robuste.

# Vous ne savez pas par quoi commencer ? Pas d'inquiétude, je vais vous accompagner tout au long de ce projet pour accomplir votre mission : **professionnaliser le code de ce blog avec l'architecture MVC**.

## Professionnalisez votre code



Qu'est-ce qui fait qu'un code est "professionnel" ? 🤔

### < Isolez le modèle et la vue >

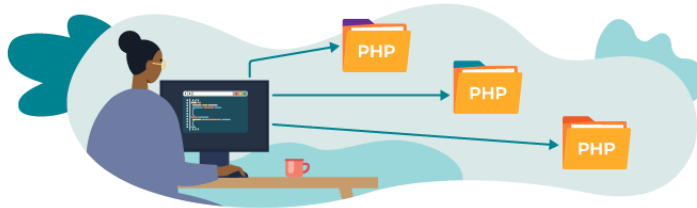
#### ► 1. Découvrez du code professionnel

2. Découvrez les limites d'un code de débutant
3. Isolez l'affichage du traitement PHP
4. Isolez l'accès aux données
5. Soignez la cosmétique du code

📋 Quiz : Isoler le modèle et la vue



## Isolez l'affichage du traitement PHP



La première bonne pratique que nous devons prendre consiste à éviter de mélanger l'affichage du reste.

? Ça veut dire qu'on va séparer le code HTML du code PHP ?

Oui, en gros. Mais vous allez voir que ce n'est pas aussi simple que ça. 😬

### Faites une première séparation

Commençons par séparer le code en deux sections :

1. **On récupère les données.** C'est là qu'on fait notamment la requête SQL. Vous voyez aussi qu'on crée une variable structurée, appelée `$posts`, qui contient les "données brutes" utiles à l'affichage de chaque billet de blog. Je vous recommande d'utiliser **uniquement des types PHP simples** pour ces variables : tableaux, chaînes de caractères, nombres entiers...
2. **On affiche les données** en les formatant. Notez qu'on utilise uniquement ces fameuses "données brutes" en provenance de la première section. Étant donné qu'on dispose de variables de types simples, il est assez facile de les formater comme on le souhaite (avec les fonctions `nl2br()` et `htmlspecialchars()`, par exemple).

C'est relativement simple à faire :

```
1 <?php
2
3 // We connect to the database
```

Notre code est déjà beaucoup mieux, mais nous pouvons aller un cran plus loin. Nous allons séparer complètement l'accès aux données (tout le traitement SQL) dans un fichier spécifique.

Nous allons avoir 3 fichiers :

- `src/model.php` : se connecte à la base de données et récupère les billets.
- `templates/homepage.php` : affiche la page. Ce fichier ne va pas changer du tout.
- `index.php` : fait le lien entre le modèle et l'affichage (oui, juste ça !).

i On y reviendra, mais sachez que ces 3 fichiers forment la base d'une structure MVC (Modèle - Vue - Contrôleur) :

- Le **modèle** traite les données ( `src/model.php` ).
- La **vue** affiche les informations ( `templates/homepage.php` ).
- Le **contrôleur** fait le lien entre les deux ( `index.php` ).

`src/model.php`

Notre nouveau fichier `src/model.php` contient une fonction `getPosts()` qui renvoie la liste des billets :

```
1 <?php
2
3 function getPosts() {
4     // We connect to the database.
5     try {
6         $database = new PDO('mysql:host=localhost;dbname=blog;charset=utf8', 'blog', 'password');
7     } catch (Exception $e) {
8         die('Erreur : ' . $e->getMessage());
9     }
10
11     // We retrieve the 5 last blog posts.
12     $statement = $database->query(
13         "SELECT id, titre, contenu, DATE_FORMAT(date_creation, '%d/%m/%Y à %HhMmss') AS
14         date_creation_fr FROM billets ORDER BY date_creation DESC LIMIT 0, 5"
15     );
16     $posts = [];
17     while (($row = $statement->fetch())) {
18         $post = [
19             'title' => $row['titre'],
20             'french_creation_date' => $row['date_creation_fr'],
21             'content' => $row['contenu'],
22         ];
23     }
24 }
```

<

Isolez le modèle et la vue

>

✓

1. Découvrez du code professionnel

✓

2. Découvrez les limites d'un code de débutant

▶

3. Isolez l'affichage du traitement PHP

4. Isolez l'accès aux données

5. Soignez la cosmétique du code

📋

Quiz : Isoler le modèle et la vue

🐦

📘

✉

Il y a 6j

u  
ile,  
nelle ?  
?

sieurs  
irs des

rooms

répondre

5. Soignez la cosmétique du code

📋 Quiz : Isoler le modèle et la vue



## Découvrez comment fonctionne une architecture MVC



Dans les chapitres précédents, nous avons petit à petit construit (sans le savoir) les bases d'une architecture MVC.

Nous avons en fait reproduit le même raisonnement que de nombreux développeurs avant nous. En fait, il y a des problèmes en programmation qui reviennent tellement souvent qu'on a créé toute une série de bonnes pratiques que l'on a réunies sous le nom de *design patterns*. Vous les retrouverez aussi, en français, sous le nom de patrons de conception.

Un des plus célèbres *design patterns* s'appelle MVC, qui signifie **Modèle - Vue - Contrôleur**. C'est celui que nous allons découvrir maintenant.

Le pattern MVC permet de bien organiser son code source. Il va vous aider à savoir quels fichiers créer, mais surtout à définir leur rôle. Le but de MVC est justement de séparer la logique du code en trois parties que l'on retrouve dans des fichiers distincts.

- **Modèle** : cette partie gère ce qu'on appelle la **logique métier** de votre site. Elle comprend notamment la gestion des données qui sont stockées, mais aussi tout le code qui prend des décisions autour de ces données. Son objectif est de fournir une interface d'action la plus simple possible au contrôleur. On y trouve donc entre autres des algorithmes complexes et des requêtes SQL.
- **Vue** : cette partie se concentre sur l'**affichage**. Elle ne fait presque aucun calcul et se contente de récupérer des variables pour savoir ce qu'elle doit afficher. On y trouve essentiellement du code HTML mais aussi quelques boucles et conditions PHP très simples, pour afficher par exemple une liste de messages.
- **Contrôleur** : cette partie gère les **échanges** avec l'utilisateur. C'est en quelque sorte l'intermédiaire entre l'utilisateur, le modèle et la vue. Le contrôleur va recevoir des requêtes

### Factorisez votre code dans une architecture MVC

1. Découvrez comment fonctionne une architecture MVC
  2. Affichez des commentaires
  3. Créez un template de page
  4. Créez un routeur
  5. Organisez en dossiers
  6. Ajoutez des commentaires
  7. Gérez les erreurs
- 📖 Quiz : Factoriser votre code dans une architecture MVC



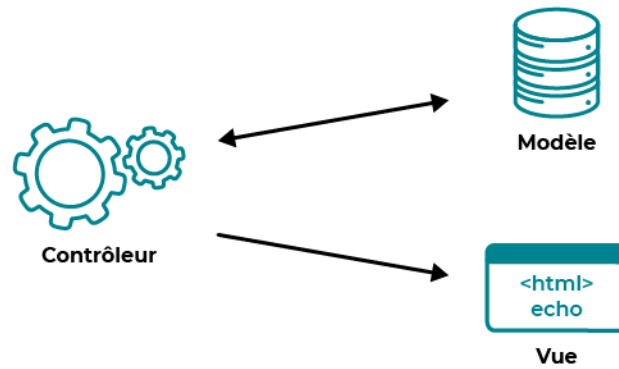


La figure suivante schématise le rôle de chacun de ces éléments.



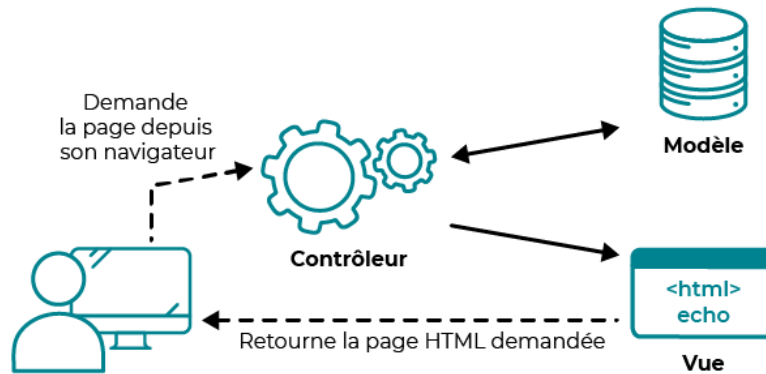
L'architecture MVC

Il est important de bien comprendre comment ces éléments s'agencent et communiquent entre eux. Regardez bien la figure suivante.



Échange d'informations entre les éléments

Concrètement, le visiteur demandera la page au contrôleur et c'est la vue qui lui sera retournée, comme schématisé sur la figure suivante. Bien entendu, tout cela est transparent pour lui, il ne voit pas tout ce qui se passe sur le serveur. C'est un schéma plus complexe que ce à quoi vous avez été habitués, bien évidemment : c'est pourtant sur ce type d'architecture que repose un grand nombre de sites professionnels !



Le client et l'architecture MVC

Voilà la théorie. Vous avez pu expérimenter la pratique dans les chapitres précédents où, pour notre exemple très simple du blog, nous avons 3 fichiers :

- `index.php` : le contrôleur (chef d'orchestre) ;
- `templates/homepage.php` : la vue (page HTML...) ;
- `src/model.php` : le modèle (requêtes SQL...).



C'est quand même pas mal plus compliqué, vous êtes sûr que ça vaut le coup ?

Pour l'instant, vous avez quand même vu que ça vous permettait de travailler avec d'autres professionnels, et c'est déjà un énorme gain ! Cela dit, en travaillant tout seul ou avec d'autres développeurs, le gain ne semble pas très flagrant. Mais attendez que le projet se complexifie un

Mes cours

Cours suivis

Cours recommandés

Certificats de cours

Derniers cours publiés

Mon parcours **NOUVEAU**

OpenClassrooms de OpenClassrooms... Il y a 6j

Bonjour Luciano,

Nouvel emploi, stage rémunéré ou alternance ? Augmentation salariale

**Cours suivis**

Cours	Inscription	Progression	Certificats
Adoptez une architecture MVC en PHP	02/04/2023	28 %	
Débutez avec React	18/05/2022	79 %	
Débutez avec Angular	18/05/2022	32 %	
Créez votre site web avec HTML5 et CSS3	21/09/2021	100 %	Obtenir votre certificat
Apprenez à programmer avec JavaScript	03/12/2021	100 %	Obtenir votre certificat

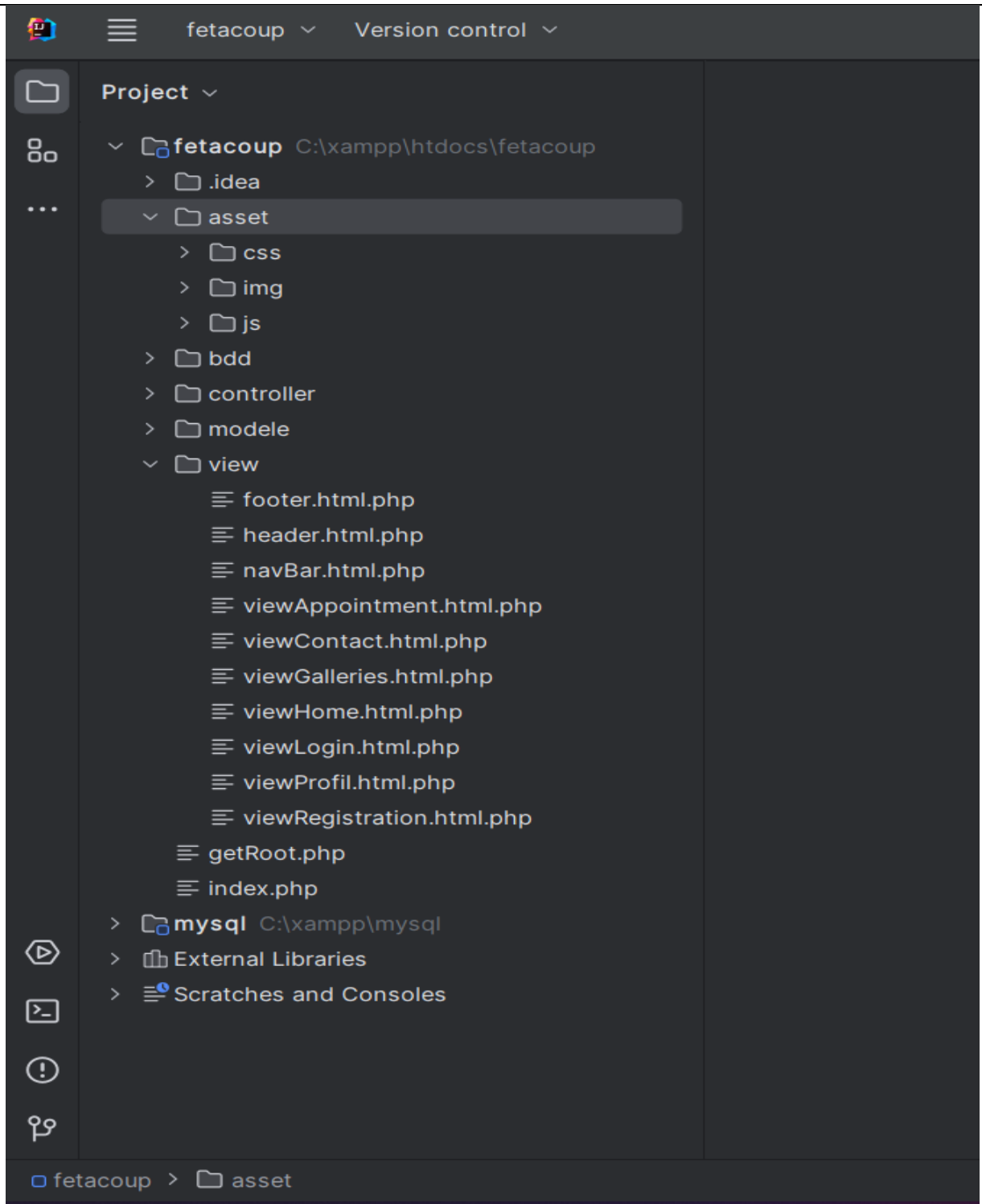
### 1.3. Ce que j'ai appris durant cette formation :

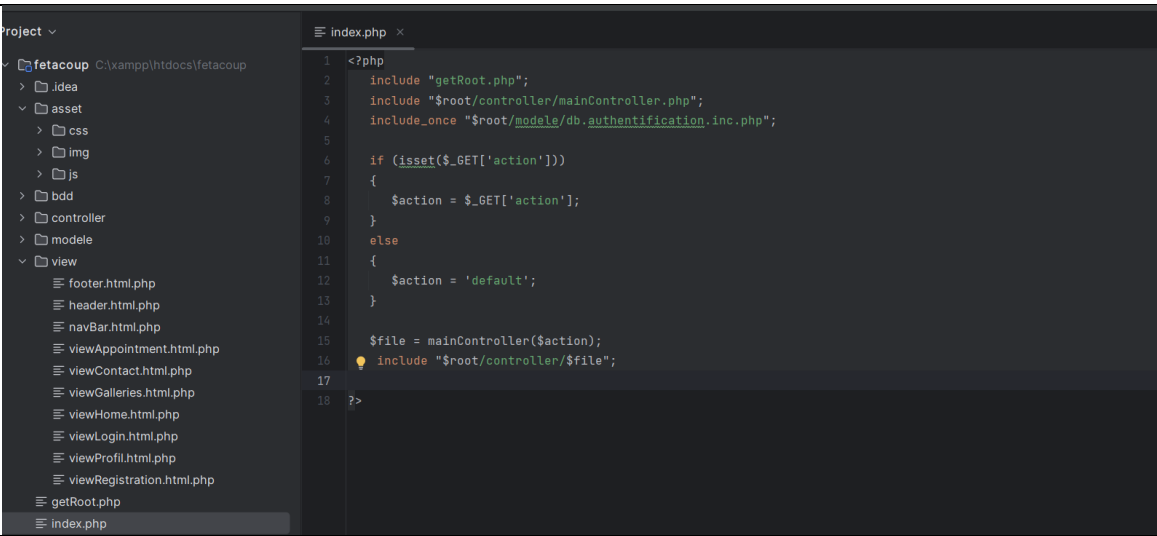
- J'ai acquis une compréhension approfondie du fonctionnement de cette architecture, ainsi que de l'importance d'organiser correctement les différents composants d'un projet dans cette structure.

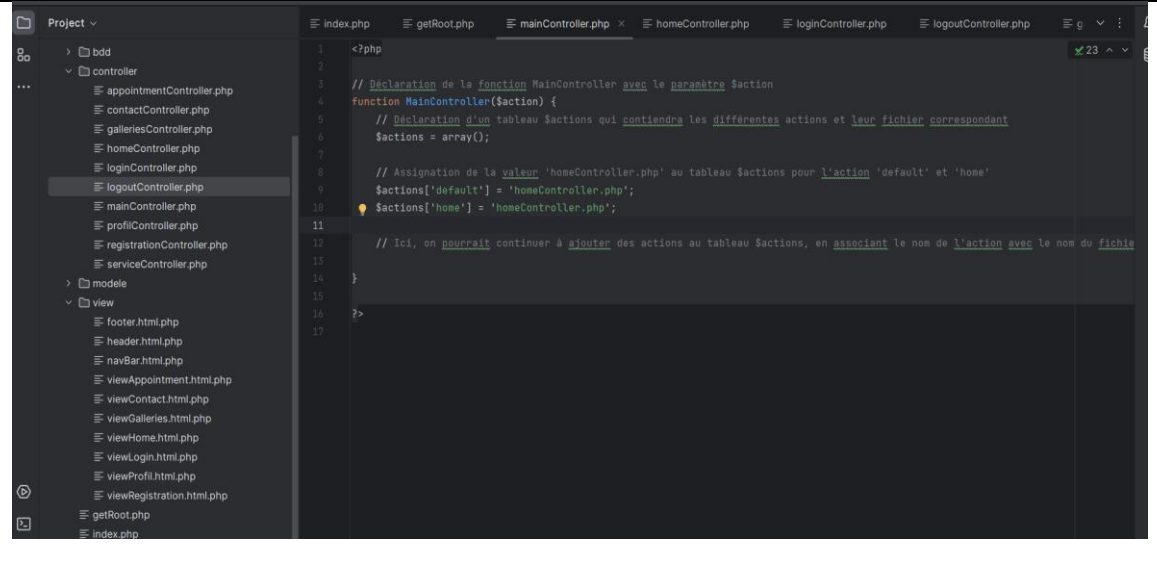
## Développement et mise en pratique des acquis de la formation :

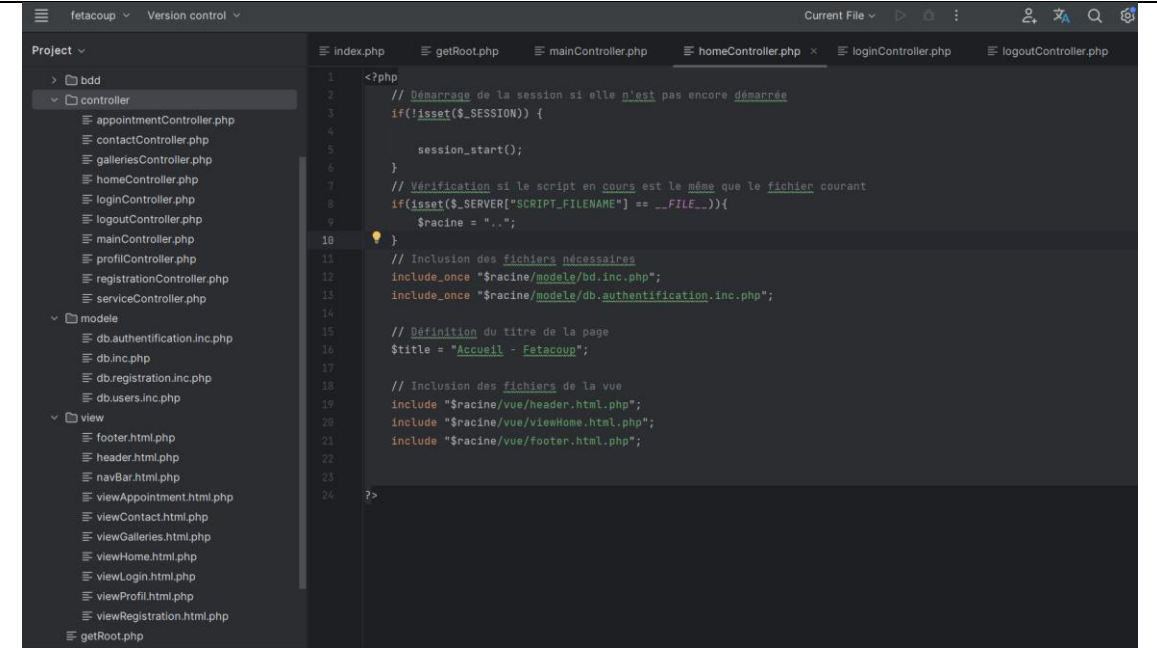
### 1.1. Configuration du projet:

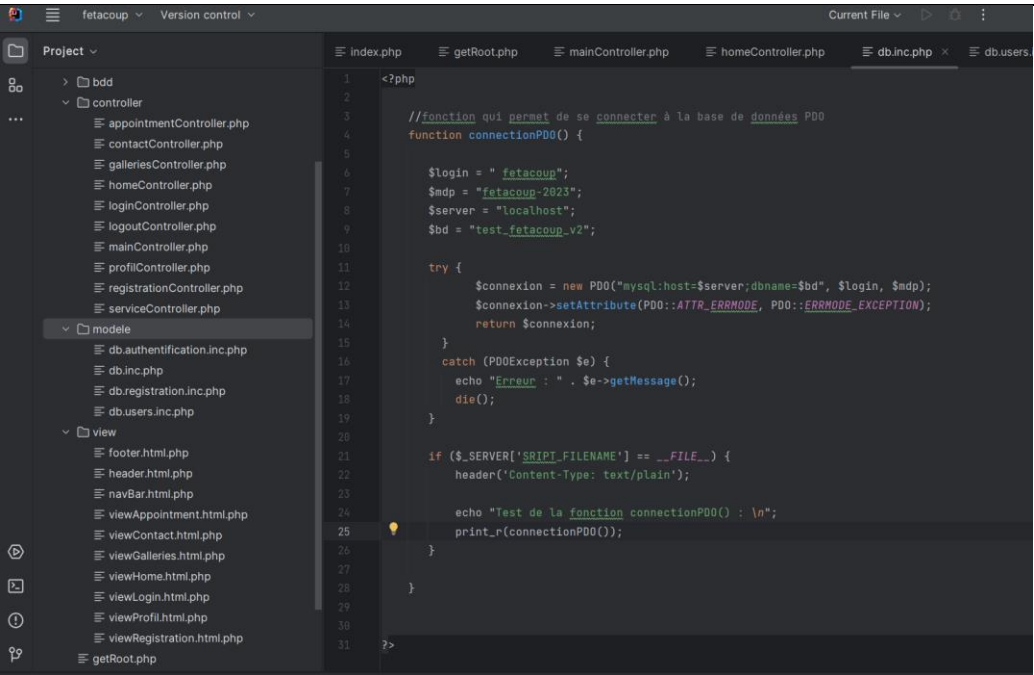
Voici l'arborescence du projet



<p>Le fichier index.php est le point d'entrée de l'application. Il joue un rôle important dans le processus de routage des requêtes entrantes vers les bonnes ressources de l'application.</p>	
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------

<p>Le MainController.php est le chef d'orchestre qui orchestre la communication entre les modèles et les vues. Il récupère les données à partir des modèles (par exemple, en appelant des méthodes de classe pour récupérer les données stockées dans une base de données) et les transmet aux vues appropriées (par exemple ci-dessus ).</p>	
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------

<p>Ce fichier permet l’affichage fichier html viewHome.html.php</p>	
---------------------------------------------------------------------	--------------------------------------------------------------------------------------

Db.inc.php permet de se connecter à la base grâce à la fonction connectionPDO()	

Conclusion et attestation de stage :

Au terme de ce stage, j'ai eu l'opportunité de travailler sur un projet passionnant pour l'entreprise. Les objectifs qui m'ont été fixés étaient clairs et précis, et j'ai eu la chance de travailler avec des technologies modernes telles que la méthode pattern mvc en php.

Les différentes formations que j'ai suivies ont été très bénéfiques pour moi et m'ont permis d'acquérir de nouvelles compétences en développement web tout en me permettant de mieux comprendre les technologies utilisées dans le projet.

Au cours de ce projet, j'ai rencontré des difficultés techniques, mais j'ai réussi à les surmonter grâce à mon travail d'équipe et à mon engagement à trouver des solutions.

Logo de l'organisme d'accueil

## ATTESTATION DE STAGE

à remettre à la ou au stagiaire à l'issue du stage

## ORGANISME D'ACCUEIL

Nom ou dénomination sociale : Fé ta Coup'  
Adresse : 147 Rue de Lannoy, 59100 Roubaix  
☎ : 03 20 69 49 39 / 03 20 58 43 10

certifie que

## LA OU LE STAGIAIRE

Nom : Yencoumou Prénom : Luciana  
Né(e) le : 24/02/2003 Sexe : F ☐ M ☒  
Adresse : 212 Rue Pierre de Roubaix, 59100 Roubaix  
☎ : 06 09 95 85 46 Mail : Luciana.yencoumou@hotmaile.com

## ÉTUDIANT(E) EN BTS Services Informatiques aux organisations

Option ☐ SISR ☒ SLAM

AU SEIN DE (nom de l'établissement d'enseignement supérieur ou de l'organisme de formation) :

Lycée Gaston Berger

a effectué un stage prévu dans le cadre de ses études

## DURÉE DU STAGE

Dates de début et de fin du stage : Du 09/01/2023 au 17/02/2023.Représentant une durée totale de 6 nombre de semaines / de mois  
(rayer la mention inutile).

La durée totale du stage est appréciée en tenant compte de la présence effective de la ou du stagiaire dans l'organisme, sous réserve des droits à congés et autorisations d'absence prévus à l'article L.124-13 du code de l'éducation (art. L.124-18 du code de l'éducation). Chaque période au moins égale à 7 heures de présence consécutives ou non est considérée comme équivalente à un jour de stage et chaque période au moins égale à 22 jours de présence consécutifs ou non est considérée comme équivalente à un mois.

## MONTANT DE LA GRATIFICATION VERSÉE À LA OU AU STAGIAIRE

La ou le stagiaire a perçu une gratification de stage pour un montant total de 0 euros.

L'attestation de stage est indispensable pour pouvoir, sous réserve du versement d'une cotisation, faire prendre en compte le stage dans les droits à retraite. La législation sur les retraites (loi n°2014-40 du 20 janvier 2014) ouvre aux étudiants dont le stage a été gratifié la possibilité de faire valider celui-ci dans la limite de deux trimestres, sous réserve du versement d'une cotisation. La demande est à faire par l'étudiant(e) dans les deux années suivant la fin du stage et sur présentation obligatoire de l'attestation de stage mentionnant la durée totale du stage et le montant total de la gratification perçue. Les informations précises sur la cotisation à verser et sur la procédure à suivre sont à demander auprès de la Sécurité sociale (code de la Sécurité sociale art. L.351-17 - code de l'éducation art. D.124-8)

Fait à Roubaix le 13/02/2023.

Nom, fonction et signature de la personne représentant de l'organisme d'accueil

**FÉ TA COUP'**  
147 rue de Lannoy 59100 ROUBAIX  
Tél. 03 20 69 49 39  
SIRET : 887 889 637 00010 APE : 9602A