



# DOSSIER PROFESSIONNEL (DP)

<i>Nom de naissance</i>	► ABDOUN
<i>Nom d'usage</i>	► ABDOUN
<i>Prénom</i>	► Walid
<i>Adresse</i>	► 708 boulevard République François Mitterand 59240 Dunkerque

## Titre professionnel visé

Développeur Web Web Mobile

### MODALITE D'ACCES :

- Parcours de formation
- Validation des Acquis de l'Expérience (VAE)

# DOSSIER PROFESSIONNEL (DP)

## Présentation du dossier

Le dossier professionnel (DP) constitue un élément du système de validation du titre professionnel.  
**Ce titre est délivré par le Ministère chargé de l'emploi.**

Le DP appartient au candidat. Il le conserve, l'actualise durant son parcours et le présente **obligatoirement à chaque session d'examen.**

Pour rédiger le DP, le candidat peut être aidé par un formateur ou par un accompagnateur VAE.

Il est consulté par le jury au moment de la session d'examen.

### Pour prendre sa décision, le jury dispose :

1. des résultats de la mise en situation professionnelle complétés, éventuellement, du questionnaire professionnel ou de l'entretien professionnel ou de l'entretien technique ou du questionnement à partir de productions.
2. du **Dossier Professionnel** (DP) dans lequel le candidat a consigné les preuves de sa pratique professionnelle.
3. des résultats des évaluations passées en cours de formation lorsque le candidat évalué est issu d'un parcours de formation
4. de l'entretien final (dans le cadre de la session titre).

*[Arrêté du 22 décembre 2015, relatif aux conditions de délivrance des titres professionnels du ministère chargé de l'Emploi]*

### Ce dossier comporte :

- ▶ pour chaque activité-type du titre visé, un à trois exemples de pratique professionnelle ;
- ▶ un tableau à renseigner si le candidat souhaite porter à la connaissance du jury la détention d'un titre, d'un diplôme, d'un certificat de qualification professionnelle (CQP) ou des attestations de formation ;
- ▶ une déclaration sur l'honneur à compléter et à signer ;
- ▶ des documents illustrant la pratique professionnelle du candidat (facultatif)
- ▶ des annexes, si nécessaire.

*Pour compléter ce dossier, le candidat dispose d'un site web en accès libre sur le site.*



<http://travail-emploi.gouv.fr/titres-professionnels>

# DOSSIER PROFESSIONNEL (DP)

## Sommaire

### Exemples de pratique professionnelle

Développer la partie front-end d'une application web en intégrant les recommandations de sécurité.	p. 5
▶ Maquettage .....	p. p. 5
▶ JAVASCRIPT .....	p. p. 7
▶ HTML/CSS/BOOTSTRAP .....	p. p. 10
Développer la partie back-end d'une application web en intégrant les recommandations de sécurité.	p. 13
▶ Projet Walidify .....	p. p. 13
▶ MCD .....	p. p. 28
▶ MVC/CRUD .....	p. p. 50
▶ Connexion/Inscription .....	p. p. 75
▶ Commentaire .....	p. p. 86
<b>Titres, diplômes, CQP, attestations de formation (facultatif)</b>	p. 93
<b>Déclaration sur l'honneur</b>	p. 94
<b>Documents illustrant la pratique professionnelle (facultatif)</b>	p. 95
<b>Annexes (Si le RC le prévoit)</b>	p. 96

# **EXEMPLES DE PRATIQUE PROFESSIONNELLE**

# DOSSIER PROFESSIONNEL (DP)

## Activité-type 1

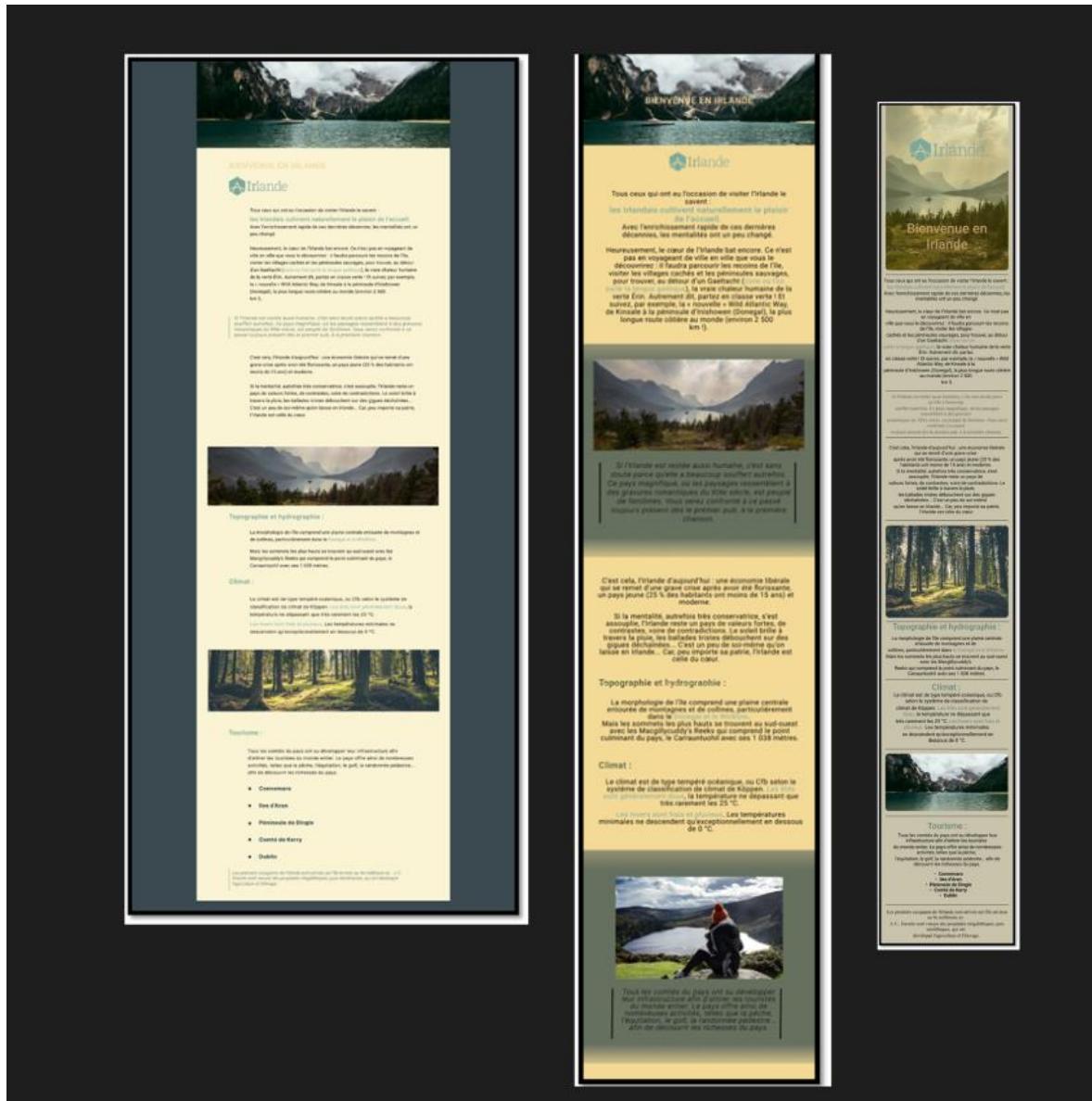
Développer la partie front-end d'une application web en intégrant les recommandations de sécurité.

Exemple n° 1 ► Maquettage

### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

J 'ai maquetté une page sur le thème de l'Irlande » dans 3 formats :

PC/Tablette/Téléphone



# DOSSIER PROFESSIONNEL (DP)

## 2. Précisez les moyens utilisés :

J'ai utilisé FIGMA pour la maquette afin d'agencer l'emplacement des éléments selon le support.

## 3. Avec qui avez-vous travaillé ?

J'ai effectué ce travail seul.

## 4. Contexte

Nom de l'entreprise, organisme ou association ►

AFCI Saint Pol Sur Mer

Chantier, atelier, service ► Centre  
de formation

Période d'exercice ► Du : 10/10/2022 au : 12/10/2022

## 5. Informations complémentaires (*facultatif*)

# DOSSIER PROFESSIONNEL (DP)

## Activité-type 1

Développer la partie front-end d'une application web en intégrant les recommandations de sécurité.

Exemple n° 2 ► Javascript

J'ai développé un quiz en HTML et Javascript

Bienvenue sur ce quiz !

Il sera composé de 9 questions...

OK

```
1  <!DOCTYPE html>
2  <html lang="fr">
3  <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>Quiz</title>
8      <link rel="preconnect" href="https://fonts.googleapis.com">
9      <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
10     <link href="https://fonts.googleapis.com/css2?family=Grandstander:wght@100&display=swap" rel="stylesheet">
11     <link rel="stylesheet" href="style.css">
12 </head>
13 <body>
14
15     <main class="quiz">
16         <div id="welcomescreen" class="welcomescreen">
17
18             <h1 class="quiz_maintitle">Bienvenue sur ce quiz !</h1>
19
20             <p class="quiz_description">Il sera composé de
21                 <span class="nbquestions">X</span> questions...</p>
22             <button id="welcomebtn" class="quiz_btn">OK</button>
23
24         </div>
25
26         <div id="questionscreen" class="questionscreen">
27
28             </div>
29
30         <div id="resultscreen" class="resultscreen">
31
32             <h2 class="subtitle">Résultats</h2>
33
34             <p class="quiz_description">
35                 <span id="nbcorrects">2</span> réponses sur <span class="nbquestions">X</span> correctes</p>
36             </div>
37         </main>
38     <script src="app.js"></script>
39 </body>
40 </html>
```

# DOSSIER PROFESSIONNEL (DP)

```
1 let screenQuestion = document.getElementById("screenQuestion");
2 let screenQuestionList = document.getElementById("screenQuestionList");
3 let screenResult = document.getElementById("screenResult");
4
5 function quiz(){
6     this.questions = [];
7     this.answers = {};
8     this.isInQuestion = false;
9     this.currentIndexQuestion = 0;
10
11     this.addQuestion = function(question){
12         this.questions.push(question);
13     }
14
15     this.showCurrentQuestion = function(){
16
17         if(this.currentIndexQuestion < this.questions.length){
18             this.questions[this.currentIndexQuestion].getQuestion();
19             this.isInQuestion = this.questions[this.currentIndexQuestion];
20
21             this.addQuestion(this.questions[this.currentIndexQuestion]);
22
23             if(allMCorrects = document.querySelector("#allCorrect")){
24                 allMCorrects.classList.add("allCorrect");
25                 document.body.classList.add("allCorrect");
26                 screenResult.style.display = "block";
27             }
28         }
29     }
30
31     this.getAnswer = function(question, answers){
32
33         this.title = question.title;
34         this.answers = answers;
35         this.answerCorrect = answers[0];
36
37         this.getQuestionText = function(indiceQuestion, nbQuestions) {
38
39             let questionText = document.createElement("p");
40             let questionTitle = document.createElement("h3");
41             questionTitle.textContent = this.title;
42             questionText.textContent = question + indiceQuestion + "/" + nbQuestions;
43             console.log(questionText);
44             screenQuestion.appendChild(questionText);
45
46             screenQuestion.appendChild(questionTitle);
47
48             let questionList = document.createElement("ul");
49             questionList.setAttribute("list-style-type", "none");
50             questionList.classList.add("list");
51             screenQuestion.appendChild(questionList);
52
53             let questionAnswers = document.createElement("ul");
54             questionAnswers.classList.add("questionAnswers");
55
56             this.answers.forEach((answer, index) => {
57
58                 let element = document.createElement("li");
59                 element.setAttribute("list-style-type", "none");
60                 element.textContent = answer;
61                 element.id = index + 1;
62                 element.addEventListener("click", this.checkAnswer);
63
64                 questionAnswers.appendChild(element);
65             });
66
67             screenQuestion.appendChild(questionAnswers);
68         }
69
70         if(functions.quiz.clickOnAnswer != null)
71             this.answers.forEach(answer => {
72                 answer.addEventListener("click", this.clickOnAnswer);
73             });
74
75         screenQuestion.appendChild(questionText);
76     }
77
78     // Fonction pour ajouter une réponse grâce au push
79     this.addAnswer = function(answer){
80
81         if(this.answers[0] == answer){
82             // si pas d'answers dans l'array = on ajoute
83             this.answers.push(answer);
84         }
85     }
86
87     // Fonction qui déclenche l'événement qui se passe lors du click sur la réponse
88     this.clickOnAnswer = (event) => {
89
90         console.log(event.target);
91         let eventTarget = event.target;
92         console.log(eventTarget);
93
94         if(eventTarget.classList.contains("list-item")){
95             let answerSelected = eventTarget.id;
96             answerSelected.classList.add("answerSelected");
97             eventTarget.classList.add("list-item");
98             eventTarget.classList.add("list-item-selected");
99             eventTarget.classList.add("list-item-active");
100
101             if(eventTarget.textContent != this.answerCorrect){
102                 eventTarget.classList.add("list-item-wrong");
103
104                 let rightAnswer = document.querySelector(`ul li#${this.answerCorrect}`);
105                 rightAnswer.classList.add("list-item-correct");
106             }
107
108             console.log(eventTarget);
109             if(eventTarget.textContent == ""){
110                 quiz.showCurrentQuestion();
111             }
112             else{
113                 return false;
114             }
115         }
116     }
117
118     let quiz = new Quiz();
119
120     let question = new Question("Combien d'au moins arrivent à manger avec leur queue ?");
121     ["l'âne", "in soi", "l'estruthorique", "une vingtaine d'espèces de managnes", "Tout ceux qui en ont une"], 4);
122     quiz.addQuestion(question);
123
124     let question = new Question("Qui est ce qui est taillé et qui attend ?");
125     ["l'heure", "l'heure claque", "le cacaïd", "le cacaïd précédent"], 3);
126     quiz.addQuestion(question);
127
128     let question = new Question("Qu'est ce qui est taillé et qui attend ?");
129     ["l'heure", "l'heure claque", "le cacaïd", "le cacaïd précédent"], 3);
130     quiz.addQuestion(question);
131
132     let question = new Question("Qui allume Johnny ?");
133     ["l'heure", "l'heure claque", "le cacaïd", "le cacaïd précédent"], 3);
134     quiz.addQuestion(question);
135
136     let question = new Question("Pourquoi il n'y a que 2 places dans le Ferrari ?");
137     ["Il force tout au fond du sac au perte des voitures", "Tous p'd s'et une twingo", "Béb parce que ça coûte cher"], 4);
138     quiz.addQuestion(question);
139
140     let question = new Question("Qui a les crampes ?");
141     ["l'heure", "l'heure claque", "le cacaïd", "le cacaïd précédent"], 3);
142     quiz.addQuestion(question);
143
144     let question = new Question("T'as dit quoi ???");
145     ["l'heure", "l'heure claque", "le cacaïd", "le cacaïd précédent"], 3);
146     quiz.addQuestion(question);
147
148     let question = new Question("T'as dit quoi ???");
149     ["l'heure", "l'heure claque", "le cacaïd", "le cacaïd précédent"], 3);
150     quiz.addQuestion(question);
151
152     console.log(quiz);
153
154     // La fonction pour retour à l'accès lorsque l'on appuie sur le bouton grâce à l'attribut de style
155     elMBQuestion.addEventListener("click",function(e){elMBQuestion.style.display = "none"});
156     // La fonction "vidéo" lâche dans
157     function seeVideoQuestion(){
158
159         screenVideo.classList.add("video");
160         screenVideo.style.display = "block";
161         screenVideo.classList.add("video");
162         screenVideo.style.display = "block";
163
164         quiz.showCurrentQuestion();
165     }
166
167     // La fonction pour retour à l'accès lorsque l'on appuie sur le bouton grâce à l'attribut de style
168     let webVideo = document.createElement("div");
169     webVideo.classList.add("video");
170     webVideo.style.display = "block";
171
172     webVideo.addEventListener("click", seeFirstQuestion);
173
174     quiz.showCurrentQuestion();
175
176     }
177
178     // La fonction pour retour à l'accès lorsque l'on appuie sur le bouton grâce à l'attribut de style
179     let webVideo2 = document.createElement("div");
180     webVideo2.classList.add("video");
181     webVideo2.style.display = "block";
182
183     webVideo2.addEventListener("click", seeFirstQuestion);
184
185     quiz.showCurrentQuestion();
186
187 }
```

# DOSSIER PROFESSIONNEL (DP)

## 3. Avec qui avez-vous travaillé ?

J'ai effectué ce travail seul.

## 4. Contexte

Nom de l'entreprise, organisme ou association

- ▶ *AFCI Saint Pol Sur Mer*

Chantier, atelier, service

- ▶ *Centre de formation*

Période d'exercice      ▶      Du :      *13/02/2023*      au :      *18/02/2023*

## 5. Informations complémentaires (*facultatif*)

# DOSSIER PROFESSIONNEL (DP)

## Activité-type 1

Développer la partie front-end d'une application web en intégrant les recommandations de sécurité.

*Exemple n° 3 ▶ Projet WALIDIFY*

### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

J'ai développé une page d'accueil en prévision de mon projet de site.



CSS :

```
1  v .content{
2      position: relative;
3      z-index: 1;
4      width: 100%;
5      display: flex;
6      flex-direction: column;
7      align-items: center;
8      justify-content: center;
9  }
10
11 v .video-background {
12     position: fixed;
13     top: 0;
14     right: 0;
15     bottom: 0;
16     left: 0;
17     overflow: hidden;
18     z-index: -1;
19 }
20
21 v .video-background video {
22     position: absolute;
23     top: 0;
24     left: 0;
25     width: 100%;
26     height: 100%;
27     object-fit: cover;
28     transition: 1s opacity;
29 }
```

# DOSSIER PROFESSIONNEL (DP)

## HTML :

```
1 <!DOCTYPE html>
2 <html lang="fr">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta http-equiv="X-UA-Compatible" content="IE=edge">
7   <meta name="viewport" content="width=device-width, initial-scale=1.0">
8   <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/OwlCarousel2/2.3.4/assets/owl.carousel.min.css" integrity="sha512-tS3S5qG0BlnQR0yJXvNjeENjMzHfQDfZJLWZPQVZlqfDwYQF0dKfCnEJUkzrOOGZuXmJLcRz0lHvKEuJ" data-bbox="125 165 915 185">
9   <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/OwlCarousel2/2.3.4/assets/owl.theme.default.min.css" integrity="sha512-sMXtMN1zRz0lHvKfCnEJUkzrOOGZuXmJLcRz0lHvKEuJ" data-bbox="125 185 915 205">
10  <link rel="stylesheet" href="https://bootswatch.com/5/lux/bootstrap.min.css">
11  <link rel="stylesheet" href="views/css/template.css">
12
13 <title>Validify</title>
14 </head>
15
16 <body>
17   <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
18     <div class="container-fluid">
19       <a class="navbar-brand" href="#"><?= URL ?>accueil
20       <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarColor02" aria-expanded="false" aria-label="Toggle navigation">
21         <span class="navbar-toggler-icon"></span>
22       </button>
23       <div class="collapse navbar-collapse justify-content-center" id="navbarColor02">
24         <ul class="navbar-nav">
25           <li class="nav-item">
26             <a class="nav-link" href="#"><?= URL ?>artistes
27           </li>
28           <li class="nav-item">
29             <a class="nav-link" href="#"><?= URL ?>albums
30           </li>
31           <li class="nav-item">
32             <a class="nav-link" href="#"><?= URL ?>playlists
33           </li>
34           <li class="nav-item">
35             <a class="nav-link" href="#"><?= URL ?>news
36           </li>
37           <li class="nav-item">
38             <a href="#"><?= URL ?>connexion
39           <?php if (!isset($_SESSION['role'])) : ?>
40             <li class="nav-item">
41               <a class="nav-link" href="#"><?= URL ?>Connexion/Inscription
42             </li>
43           </?php endif; ?>
44           <?php if (isset($_SESSION['role'])) : ?>
45             <li class="nav-item">
46               <a href="#"><?= URL ?>deconnexion
47             </li>
48           </?php endif; ?>
49         </ul>
50       </div>
51     </div>
52   </nav>
53
54   <div class="container">
55
56     <h1 class="rounded border-dark p-2 m-2 text-center texte-dark bg-danger"><?- $titre ?></h1>
57     <?=$content ?>
58   </div>
59
60   <script src="https://cdn.jsdelivr.net/npm/popper.js@core@2.11.6/dist/umd/popper.min.js" integrity="sha384-oIqD9Mte9ATkkIep9tjCxs/Z9FnFFXlDAYTuJMaBAsjFuCZ5ekh5" data-bbox="145 645 915 655">
61   <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.min.js" integrity="sha384-wQ936A66000zxjtar0SK1ohRASSY2XoFH4zFuZLkoJigXTHh4" data-bbox="145 655 915 665">
62   </body>
63
64 </html>
```

## 2. Précisez les moyens utilisés :

HTML/CSS/BOOTSTRAP.

# DOSSIER PROFESSIONNEL (DP)

## 3. Avec qui avez-vous travaillé ?

J'ai travaillé seul.

## 4. Contexte

Nom de l'entreprise, organisme ou association ► *AFCI Saint Pol Sur Mer*

Chantier, atelier, service ► *Centre de formation*

Période d'exercice ► Du : *20/02/2023* au : *24/02/2023*

## 5. Informations complémentaires (*facultatif*)

# DOSSIER PROFESSIONNEL (DP)

## Activité-type 2

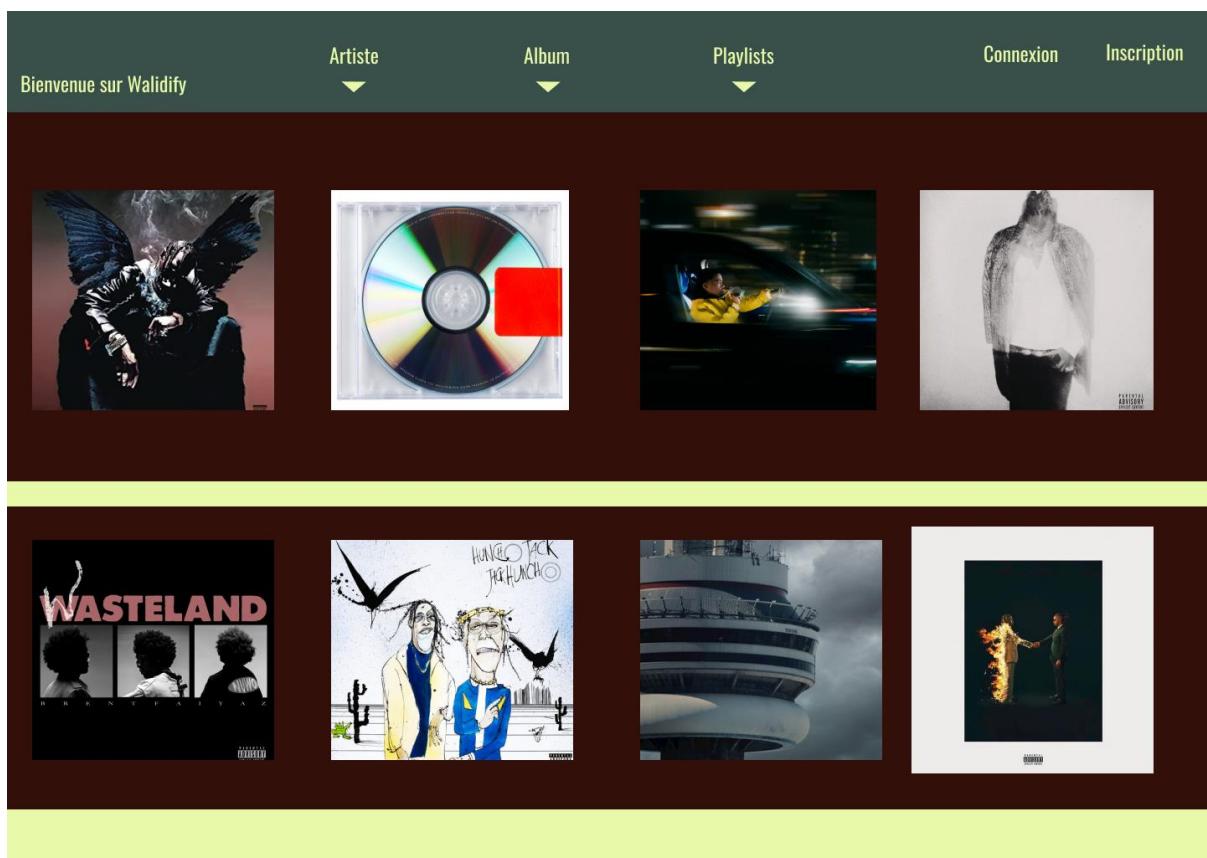
Développer la partie back-end d'une application web en intégrant les recommandations de sécurité.

*Exemple n° 1 ► Projet WALIDIFY*

### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

En premier lieu, j'ai commencé à maquetté l'interface du site.

La V1 de la maquette a pour objectif d'afficher chaque page artistes/albums/playlists de cette manière.



# DOSSIER PROFESSIONNEL (DP)

Affichage d'un album :

The screenshot shows a dark-themed user interface for a music platform. At the top, there is a navigation bar with links for "Artiste", "Album", "Playlists", "Connexion", and "Inscription". Below this, the text "Bienvenue sur Walidity" is displayed. The main content area contains the following information about the album:

- Artiste : Travis Scott
- Album : Birds In The Trap Sing McKnight
- Tracks : 14
- Durée : 53 min 46s
- Statut : Classic

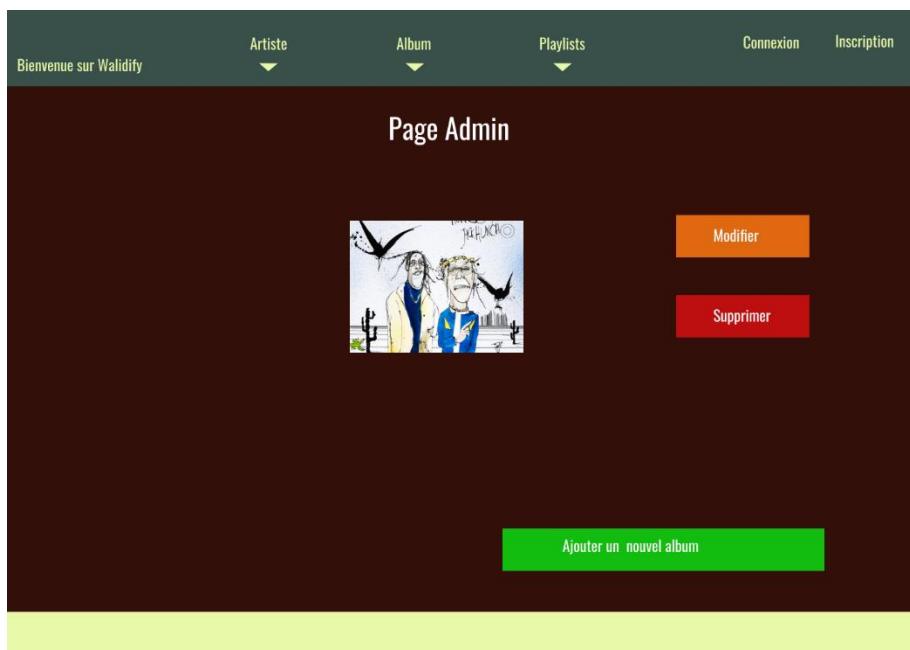
On the right side of the album title, there is a large thumbnail image of the album cover, which features a figure with large wings.

Page de connexion et d'inscription :

The screenshot shows a dark-themed login and registration form. The top navigation bar includes links for "Artiste", "Album", "Playlists", "Connexion", and "Inscription". The page title "Connexion/inscription" is centered at the top. Below the title are two input fields: one for "email" and one for "mot de passe" (password). At the bottom of the form is a single button labeled "valider" (validate).

# DOSSIER PROFESSIONNEL (DP)

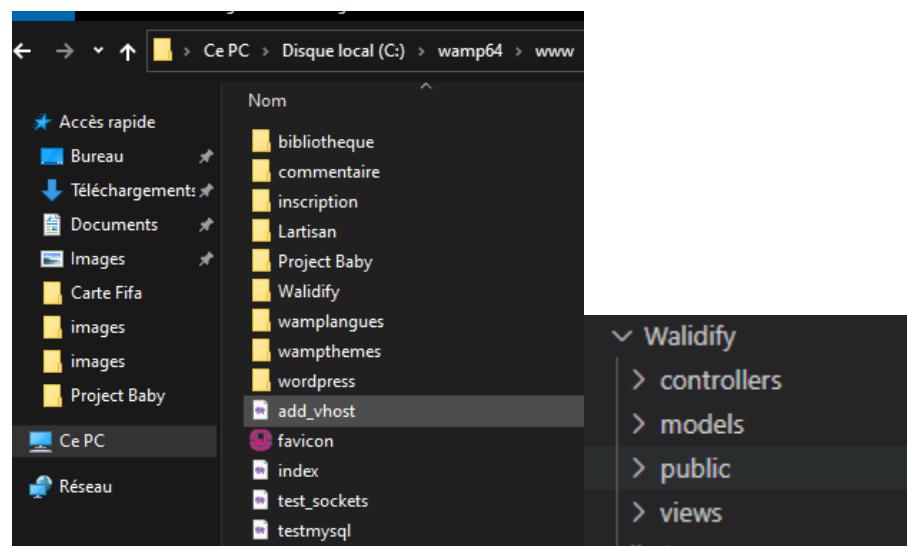
Page Admin :



En me basant sur les conceptions de ces maquettes j'ai donc édifié une base de développement pour l'aspect HTML/CSS et BOOTSTRAP.

# DOSSIER PROFESSIONNEL (DP)

Création du dossier Walidify dans le dossier WWW de wamp64 et ouverture de celui-ci dans VSC.



# DOSSIER PROFESSIONNEL (DP)

Création du site sur Visual Studio Code :

HTML :

```
Validity > views > accueilliview.php > @Html > @body > @div.message > @p
1  <?php ob_start();?
2
3  if (!empty($_SESSION['alert'])) {
4
5    ?>
6
7      <div class="alert alert-= $_SESSION['alert'][type] ? " role="alert">
8        <!-- $_SESSION['alert'][msg] -->
9      </div>
10  <?php unset($_SESSION['alert']);?
11  endif;
12  ?>
13
14
15 <!DOCTYPE html>
16 <html lang="fr">
17
18 <head>
19   <meta charset="UTF-8">
20   <meta http-equiv="X-UA-Compatible" content="IE=edge">
21   <meta name="viewport" content="width=device-width, initial-scale=1.0">
22   <link rel="stylesheet" href="views/css/accueil.css">
23   <link rel="stylesheet" href="views/css/video.css">
24   <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
25
26   <title>Accueil</title>
27 </head>
28
29 <body>
30
31   <h1 class="text-center fw-bold display-1 mb-5">Bienvenue sur <span class="text-danger">VALIDIFY</span></h1>
32
33   <div class="video-background">
34     <video autoplay loop muted src="public/images/3medias.mp4"></video>
35   </div>
36   <div class="message">
37
38     <p>
39       Bienvenue sur Validify, votre escale incontournable dans le monde passionnant du Rap et du RnB. À travers notre plateforme,
40       nous vous invitons à un voyage rythmé par les beats percutants et les mélodies captivantes qui façonnent l'essence de ces genres musicaux.
41       <br>
42       Validify n'est pas qu'un simple site, c'est une expérience immersive qui vous plonge au cœur de l'actualité du Rap et du RnB.
43       Que vous soyez un aficionado de la première heure ou un novice curieux, notre contenu riche et varié saura vous captiver.
44       <br>
45       <br>
46       Des icônes légendaires du Hip-Hop aux étoiles montantes du RnB, en passant par les dernières tendances et les analyses perspicaces,
47       Validify est votre guide pour naviguer dans l'univers fascinant de ces genres qui ont révolutionné la scène musicale mondiale.
48       <br>
49       <br>
50       Bien plus qu'un site, validify est un pont entre vous et la musique qui définit notre époque.
51       Rejoignez-nous dans cette aventure et laissez-vous porter par le courant des rythmes et des rimes qui façonnent le paysage du Rap et du RnB.
52     </p>
53   </div>
54
55
56   <script>
57     $(document).ready(function() {
58       let paragraphs = $('.message p').html().split('<br><br>');
59       $('.message p').remove();
60
61       $.each(paragraphs, function(i, paragraph) {
62         let newParagraph = $('

').html(paragraph).css({
63           opacity: 0,
64           position: 'relative',
65           left: '-100px'
66         });
67
68         $('.message').append(newParagraph);
69
70         setTimeout(function() {
71           newParagraph.animate({
72             opacity: 1,
73             left: 0
74           }, 3000);
75         }, 3000 * i);
76       });
77     });
78   </script>
79
80 </body>
81 </html>
82
83 <?php
84
85   $Content = ob_get_clean();
86   $titre = "Page d'accueil";
87
88   require "template.php";
89
90   ?>


```

# DOSSIER PROFESSIONNEL (DP)



## CSS

```
1 *{  
2   box-sizing: border-box;  
3 }  
4  
5 .message{  
6   border: 7px solid #d33c3c;  
7   border-radius: 15px;  
8   padding: 15px;  
9   margin: 15px;  
10  text-align: center;  
11  background-color: #rgba(6, 7, 7, 0.404); |  
12  color: #fff;  
13  font-style: italic;  
14  font-weight: bold;  
15 }  
16 }
```

# DOSSIER PROFESSIONNEL (DP)

CSS pour la vidéo en fond :

```
1  .content{
2      position: relative;
3      z-index: 1;
4      width: 100%;
5      display: flex;
6      flex-direction: column;
7      align-items: center;
8      justify-content: center;
9  }
10
11 .video-background {
12     position: fixed;
13     top: 0;
14     right: 0;
15     bottom: 0;
16     left: 0;
17     overflow: hidden;
18     z-index: -1;
19 }
20
21 .video-background video {
22     position: absolute;
23     top: 0;
24     left: 0;
25     width: 100%;
26     height: 100%;
27     object-fit: cover;
28     transition: 1s opacity;
29 }
```

JavaScript pour le message :

```
55
56 <script>
57     $(document).ready(function() {
58         let paragraphs = $('.message p').html().split('<br><br>');
59         $('.message p').remove();
60
61         $.each(paragraphs, function(i, paragraph) {
62             let newParagraph = $('

').html(paragraph).css({
63                 opacity: 0,
64                 position: 'relative',
65                 left: '-100px'
66             });
67
68             $('.message').append(newParagraph);
69
70             setTimeout(function() {
71                 newParagraph.animate({
72                     opacity: 1,
73                     left: 0
74                 }, 3000);
75             }, 3000 * i);
76         });
77     });
78 </script>
79


```

# DOSSIER PROFESSIONNEL (DP)

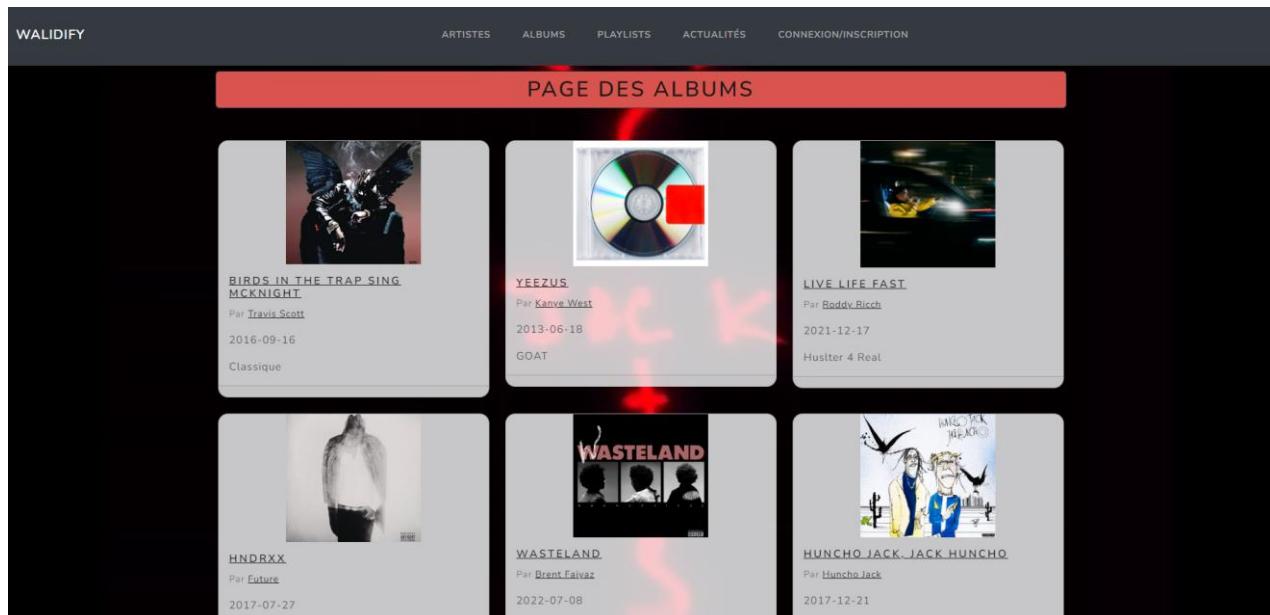
Page d'affichage des albums :

```
1  <?php
2  ob_start();
3
4  if (!empty($_SESSION['alert'])) :
5
6  ?>
7
8  <div class="alert alert-<?= $_SESSION['alert']['type'] ?>" role="alert">
9    |   <?= $_SESSION['alert']['msg'] ?>
10   </div>
11  <?php unset($_SESSION['alert']);
12  endif;
13 ?>
14
15
16  <link rel="stylesheet" href="views/css/card.css">
17  <link rel="stylesheet" href="views/css/video.css">
18
19
20
21  <div class="video-background">
22    <video autoplay loop muted src="public/images/travisvideo.mp4"></video>
23  </div>
24
25
26  <div class="content container-fluid my-5" style="background-color: #f0f0f0; padding: 10px; border-radius: 5px">
27
28  <!-- Boucle FOR afin d'afficher chaque élément souhaité de l'album grâce aux GETTER & SETTER -->
29
30  <div class="row row-cols-1 row-cols-md-2 row-cols-lg-3 g-4">
31    <?php for ($i = 0; $i < count($albums); $i++) : ?>
32    <div class="col">
33      <div class="card custom-card">
34        <div class="d-flex justify-content-center">
35          getAlbum(); ?>" style="border-radius: 10px; border: 2px solid #ccc; width: 100%; height: 100%; object-fit: cover;"/>
36        </div>
37        <div class="card-body">
38          <h5 class="card-title"><a href="<?= URL::albums/1/<?= $albums[$i]->getId(); ?>"><?= $albums[$i]->getAlbum(); ?></a></h5>
39          <p class="card-text"><small class="text-muted">Par <a href="<?= URL::artistes/1/<?= $albums[$i]->getId(); ?>"><?= $albums[$i]->getNom(); ?></a></small></p>
40          <p class="card-text"><?= $albums[$i]->getDatesSortie(); ?></p>
41          <p class="card-text"><?= $albums[$i]->getStatut(); ?></p>
42        </div>
43
44        <div class="card-footer d-flex">
45          <?php if (isset($_SESSION['role']) && $_SESSION['role'] === 'admin') : ?>
46            <a href="<?= URL::albums/<?= $albums[$i]->getId(); ?>" class="btn btn-warning">Modifier</a>
47            <form method="POST" action="<?= URL::albums/s/<?= $albums[$i]->getId(); ?>" onsubmit="return confirm('Voulez-vous vraiment supprimer cet album ?')">
48              <button class="btn btn-danger" type="submit">Supprimer</button>
49            </form>
50          <?php endif; ?>
51        </div>
52      </div>
53    </div>
54
55    <?php }; ?>
56  </div>
57
58
59  <?php if (isset($_SESSION['role']) && $_SESSION['role'] === 'admin') : ?>
60    <a href="<?= URL::albums/a" class="btn btn-success d-block" style="margin-bottom: 30px;">Ajouter</a>
61  <?php endif; ?>
62
63
64
65  <?php
66
67  $content = ob_get_clean();
68  $titre = "Page des albums";
69
70  require "template.php";
71
72 ?>
```

# DOSSIER PROFESSIONNEL (DP)

CSS :

```
Validity > views > css > # card.css > ...
1  .custom-card {
2      background-color: #rgba(248, 249, 250, 0.8);
3
4  }
5
6  .card {
7      border-radius: 15px;
8      transition: transform 0.3s ease-in-out; /* Ajoute une transition à la transformation */
9
10 }
11
12
13 .card:hover {
14     transform: translateY(-5%); /* Déplace la carte vers le haut de 15% lorsque la souris passe dessus */
15 }
16
```



# DOSSIER PROFESSIONNEL (DP)

Page d'affichage d'un album :

WALIDIFY

ARTISTES ALBUMS PLAYLISTS ACTUALITÉS CONNEXION/INSCRIPTION

HUNCHO JACK, JACK HUNCHO

Titre : Huncho Jack; Jack Huncho

Tracks : 13

Durée : 41 min

Date de sortie : 2017-12-21

HTML :

```
1 <?php
2 ob_start() ?>
3
4
5
6
7 <div class="row">
8   <div class="col-12 col-sm-6 d-flex justify-content-center">
9      connexion > connexion.view.php > ...
1  <?php ob_start();?>
2
3  <div class="login-page">
4
5      <form method="POST" action="= URL ?&gt;connexion" enctype="multipart/form-data"&gt;
6          &lt;h1&gt;Se connecter&lt;/h1&gt;
7          &lt;div class="social-media"&gt;
8              &lt;p class="fab fa-google"&gt;&lt;/p&gt;
9              &lt;p class="fab fa-youtube"&gt;&lt;/p&gt;
10             &lt;p class="fab fa-facebook"&gt;&lt;/p&gt;
11             &lt;p class="fab fa-twitter"&gt;&lt;/p&gt;
12         &lt;/div&gt;
13         &lt;p class="choose-email" &gt;Utiliser mon adresse e-mail&lt;/p&gt;
14
15
16         &lt;div class="inputs"&gt;
17             &lt;input type="email" name="mail" placeholder="Email"&gt;
18             &lt;input type="password" name="mdp" placeholder="Mot de passe"&gt;
19         &lt;/div&gt;
20         &lt;p class="inscription"&gt;Je n'ai pas de compte.&lt;a href="<?= URL ?&gt;inscription"&gt;Je m'en crée un.&lt;/a&gt;&lt;/p&gt;
21         &lt;div align="center"&gt;
22             &lt;button type="submit" name="envoi"&gt;Se connecter&lt;/button&gt;
23         &lt;/div&gt;
24     &lt;/form&gt;
25 &lt;/div&gt;
26
27
28
29
30     &lt;link rel="stylesheet" href="views/css/connexion.css"&gt;
31     &lt;script src="https://kit.fontawesome.com/d832cb9a1e.js" crossorigin="anonymous"&gt;&lt;/script&gt;
32
33 &lt;?php
34
35 $content = ob_get_clean();
36 $titre = "Page de connexion";
37
38 require "views/template.php";
39
40 ?&gt;
41
42</pre
```

# DOSSIER PROFESSIONNEL (DP)

CSS :

```
Wadify > views > css > # connexion.css > login-page
 1 √ .login-page {
 2   |   display: flex;
 3   |   justify-content: center;
 4   |   align-items: center;
 5   |   background-color: #F5F5F5;
 6   |   padding: 15px;
 7 }
 8
 9 √ .login-page form{
10   |   margin-top: 20px;
11   |   background-color: #FFF;
12   |   padding: 40px 0px;
13   |   border-radius: 10px;
14   |   min-width: 300px;
15 }
16
17
18 √ form h1{
19   |   color: #eb7371;
20   |   text-align: center;
21 }
22
23 √ form .social-media{
24   |   margin-top : -10px;
25   |   display: flex;
26   |   flex-wrap: wrap;
27   |   justify-content: center;
28 }
29
30
31 form .social-media p{
32   |   padding: 5px;
33   |   margin-left: 10px;
34   |   border: 1px solid #545454;
35   |   border-radius: 100%;
36   |   width: 25px;
37   |   text-align: center;
38   |   cursor: pointer;
39 }
40
41
42 form p.choose-email{
43   |   text-align: center;
44 }
45
46
47 form .inputs{
48   |   display: flex;
49   |   flex-direction: column;
50   |   width: 400px;
51 }
52
53 form .inputs input[type="email"], input[type="password"]){
54   |   padding: 15px;
55   |   border-radius: 5px;
56   |   border: none;
57   |   background-color: #F2F2F2;
58   |   margin-bottom: 15px;
59   |   outline: none;
60 }
61
62 form p.inscription{
63   |   font-size: 14px;
64   |   margin-bottom: 20px;
65   |   color: #878787;
66 }
67
68 form p.inscription span{
69   |   color: #eb7371;
70 }
71
72 form button{
73   |   padding: 15px 25px;
74   |   border: none;
75   |   border-radius: 5px;
76   |   font-size: 15px;
77   |   color: #FFF;
78   |   background-color: #eb7371;
79   |   outline: none;
80   |   cursor: pointer;
81 }
82
83 √ @media screen and (max-width: 768px) {
84
85 √   .login-page form{
86   |   padding: 20px 30px;
87   }
88
89 √   form .inputs{
90   |   width: 300px;
91   }
92
93 √   form button{
94   |   padding: 10px 15px;
95   }
96
97
98 }
99 }
```

# DOSSIER PROFESSIONNEL (DP)

Page des actualités :



HTML :

```
Walify > views > news.view.php > div.row
1  <?php ob_start() ?>
2
3  <link rel="stylesheet" href="views/css/carousel.css">
4
5
6  <div class="row">
7      <div class="col-12 m-auto">
8          <div class="owl-carousel owl-theme">
9              <div class="item mb-4">
10                 <div class="card border-0 shadow">
11                     
12                     <div class="card-body">
13                         <div class="card-title text-center">
14                             <h4>Owl Carousel</h4>
15                         </div>
16                     </div>
17                 </div>
18             </div>
19         </div>
20     </div>
21 </div>
22
23
24
25 <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
26
27
28 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/js/bootstrap.bundle.min.js" integrity="sha384-gtEjrD/SeCtmISkJKkJNUaaKMoLD0//ElJ19smozuHV6z3IehdsqICqC9d
29 <script src="https://cdnjs.cloudflare.com/ajax/libs/OwlCarousel2/2.3.4/owl.carousel.min.js" integrity="sha512-bPs7Ae6pVvhOvHOSIcyUClR7/q20AsRiovw4vAkX+zJbw3ShAeeQqPv+RZEnwn9EjL7nTfD1uZB8QaF6qFw==">
30 <script>
31     $('.owl-carousel').owlCarousel({
32         loop: true,
33         margin: 15,
34         nav: true,
35         responsive: {
36             0: {
37                 items: 1
38             },
39             600: {
40                 items: 2
41             },
42             1000: {
43                 items: 3
44             }
45         }
46     })
47 </script>
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101 <?php
102
103 $content = ob_get_clean();
104 $titre = "Page des news";
105
106 require "template.php";
107
108 ?>
```

# DOSSIER PROFESSIONNEL (DP)

## 2. Précisez les moyens utilisés :

J'ai utilisé Figma pour le maquettage.  
Par la suite, HTML, CSS, BOOTSTRAP ainsi que JavaScript

## 3. Avec qui avez-vous travaillé ?

Seul

## 4. Contexte

Nom de l'entreprise, organisme ou association ► *AFCI*

Chantier, atelier, service ► Atelier

Période d'exercice ► Du : *06/03/2023* au : *02/06/2023*

# DOSSIER PROFESSIONNEL (DP)

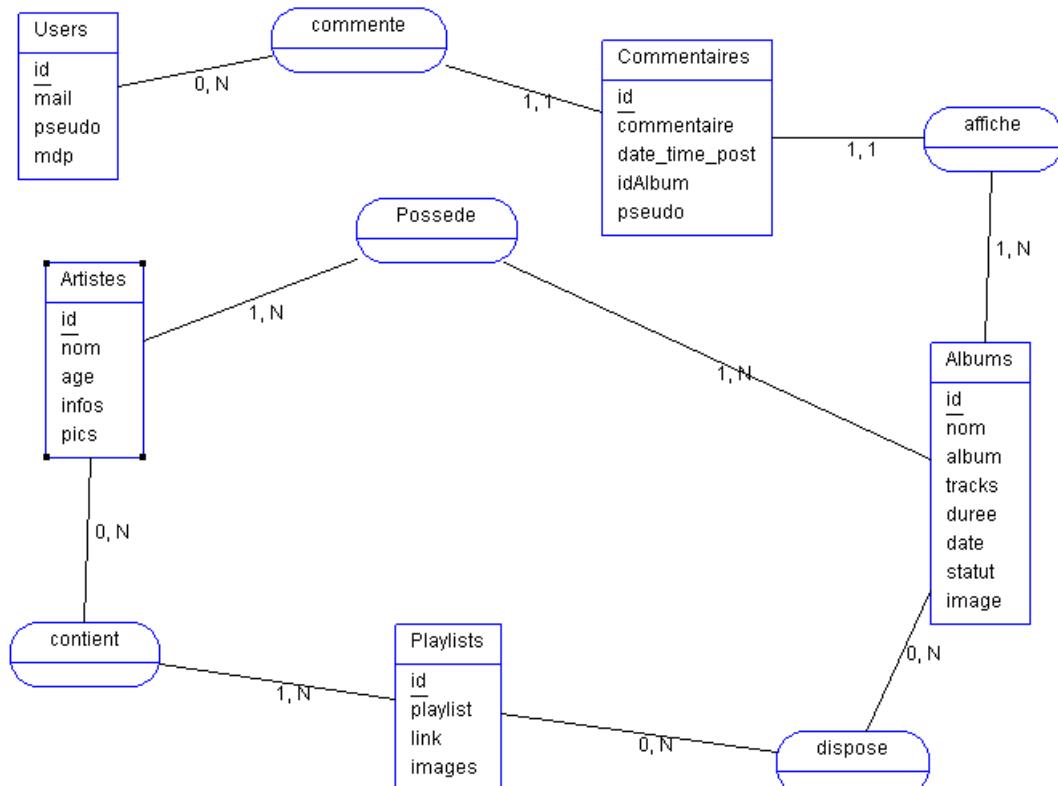
## Activité-type 2

Développer la partie back-end d'une application web en intégrant les recommandations de sécurité.

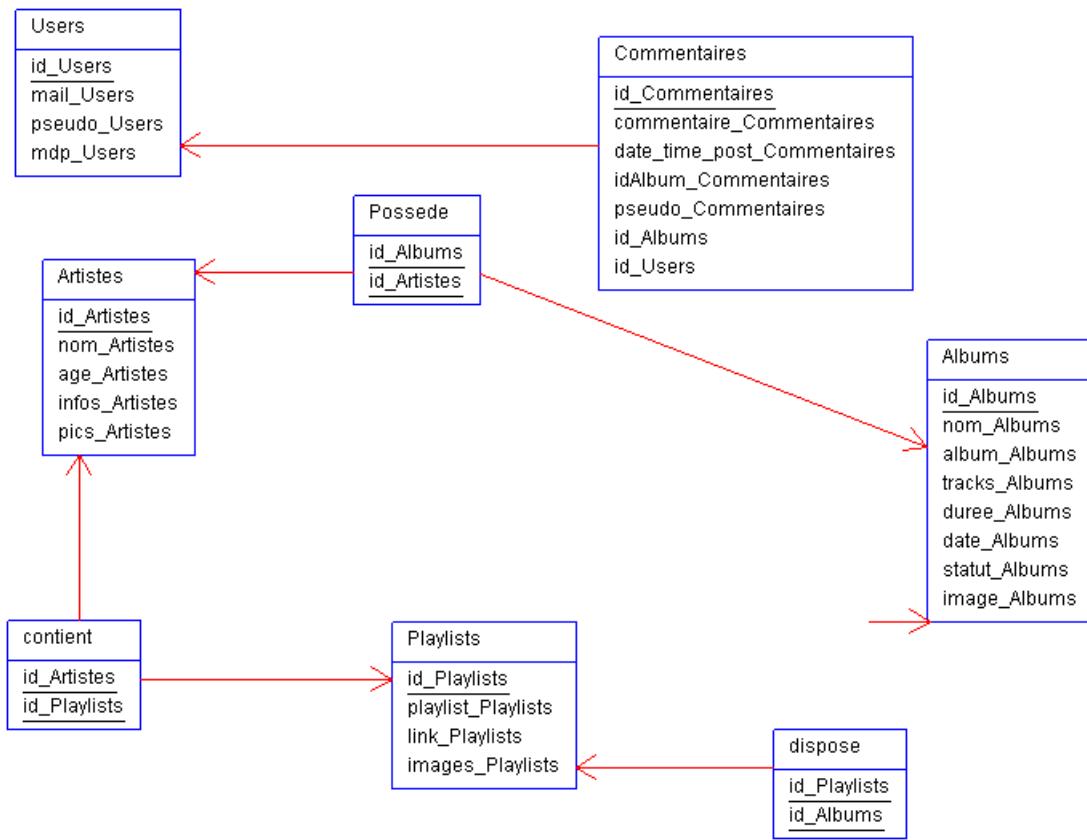
*Exemple n° 2 ▶ MCD*

### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

J'ai effectué le Modèle Concepteur de Données pour mon site de cette manière.



# DOSSIER PROFESSIONNEL (DP)



```

# Modèle créé le : Mon Jun 12 13:46:47 CEST 2023
Albums (id_Albums, nom_Albums, album_Albums, tracks_Albums, duree_Albums, date_Albums, statut_Albums, image_Albums)
Artistes (id_Artistes, nom_Artistes, age_Artistes, infos_Artistes, pics_Artistes)
Playlists (id_Playlists, playlist_Playlists, link_Playlists, images_Playlists)
Commentaires (id_Commentaires, commentaire_Commentaires, date_time_post_Commentaires, idAlbum_Commentaires, pseudo_Commentaires, #id_Albums, #id_Users)
Users (id_Users, mail_Users, pseudo_Users, mdp_Users)
Possede (id_Albums, id_Artistes)
contenu (id_Artistes, id_Playlists)
dispose (id_Playlists, id_Albums)
  
```

# DOSSIER PROFESSIONNEL (DP)

Et voici le script SQL que j'intègre dans ma base de données:

```
1  DROP TABLE IF EXISTS Albums ;
2  CREATE TABLE Albums (id_Albums INT AUTO_INCREMENT NOT NULL,
3  nom_Albums VARCHAR(25),
4  album_Albums VARCHAR(25),
5  tracks_Albums INT,
6  duree_Albums VARCHAR(20),
7  date_Albums DATE,
8  statut_Albums VARCHAR(25),
9  image_Albums TEXT,
10 PRIMARY KEY (id_Albums)) ENGINE=InnoDB;
11
12 DROP TABLE IF EXISTS Artistes ;
13 CREATE TABLE Artistes (id_Artistes INT AUTO_INCREMENT NOT NULL,
14 nom_Artistes VARCHAR(25),
15 age_Artistes INT,
16 infos_Artistes TEXT,
17 pics_Artistes TEXT,
18 PRIMARY KEY (id_Artistes)) ENGINE=InnoDB;
19
20 DROP TABLE IF EXISTS Playlists ;
21 CREATE TABLE Playlists (id_Playlists INT AUTO_INCREMENT NOT NULL,
22 playlist_Playlists VARCHAR(50),
23 link_Playlists VARCHAR(150),
24 images_Playlists VARCHAR(150),
25 PRIMARY KEY (id_Playlists)) ENGINE=InnoDB;
26
27 DROP TABLE IF EXISTS Commentaires ;
28 CREATE TABLE Commentaires (id_Commentaires INT AUTO_INCREMENT NOT NULL,
29 commentaire_Commentaires TEXT,
30 date_time_post_Commentaires DATETIME,
31 idAlbum_Commentaires INT,
32 pseudo_Commentaires VARCHAR(50),
33 id_Albums **NOT FOUND**,
34 id_Users **NOT FOUND**,
35 PRIMARY KEY (id_Commentaires)) ENGINE=InnoDB;
36
37 DROP TABLE IF EXISTS Users ;
38 CREATE TABLE Users (id_Users INT AUTO_INCREMENT NOT NULL,
39 mail_Users TEXT,
40 pseudo_Users VARCHAR,
41 mdp_Users TEXT,
42 PRIMARY KEY (id_Users)) ENGINE=InnoDB;
43
44 DROP TABLE IF EXISTS Possede ;
45 CREATE TABLE Possede (id_Albums **NOT FOUND** AUTO_INCREMENT NOT NULL,
46 id_Artistes **NOT FOUND** NOT NULL,
47 PRIMARY KEY (id_Albums,
48 | id_Artistes)) ENGINE=InnoDB;
49
50 DROP TABLE IF EXISTS contient ;
51 CREATE TABLE contient (id_Artistes **NOT FOUND** AUTO_INCREMENT NOT NULL,
52 id_Playlists **NOT FOUND** NOT NULL,
53 PRIMARY KEY (id_Artistes,
54 | id_Playlists)) ENGINE=InnoDB;
55
56 DROP TABLE IF EXISTS dispose ;
57 CREATE TABLE dispose (id_Playlists **NOT FOUND** AUTO_INCREMENT NOT NULL,
58 id_Albums **NOT FOUND** NOT NULL,
59 PRIMARY KEY (id_Playlists,
60 | id_Albums)) ENGINE=InnoDB;
61
62 ALTER TABLE Commentaires ADD CONSTRAINT FK_Commentaires_id_Albums FOREIGN KEY (id_Albums) REFERENCES Albums (id_Albums);
63
64 ALTER TABLE Commentaires ADD CONSTRAINT FK_Commentaires_id_Users FOREIGN KEY (id_Users) REFERENCES Users (id_Users);
65 ALTER TABLE Possede ADD CONSTRAINT FK_Possede_id_Albums FOREIGN KEY (id_Albums) REFERENCES Albums (id_Albums);
66 ALTER TABLE Possede ADD CONSTRAINT FK_Possede_id_Artistes FOREIGN KEY (id_Artistes) REFERENCES Artistes (id_Artistes);
67 ALTER TABLE contient ADD CONSTRAINT FK_contient_id_Artistes FOREIGN KEY (id_Artistes) REFERENCES Artistes (id_Artistes);
68 ALTER TABLE contient ADD CONSTRAINT FK_contient_id_Playlists FOREIGN KEY (id_Playlists) REFERENCES Playlists (id_Playlists);
69 ALTER TABLE dispose ADD CONSTRAINT FK_dispose_id_Playlists FOREIGN KEY (id_Playlists) REFERENCES Playlists (id_Playlists);
70 ALTER TABLE dispose ADD CONSTRAINT FK_dispose_id_Albums FOREIGN KEY (id_Albums) REFERENCES Albums (id_Albums);
71
```

# DOSSIER PROFESSIONNEL (DP)

## 2. Précisez les moyens utilisés :

J'ai utilisé AnalyseSI pour la conception du MCD, Visual Studio Code et MySQL pour la lecture et l'intégration du script SQL.

## 3. Avec qui avez-vous travaillé ?

Seul

## 4. Contexte

Nom de l'entreprise, organisme ou association ► **AFCI**

Chantier, atelier, service ► Atelier

Période d'exercice ► Du : **06/03/2023** au : **02/06/2023**

## 5. Informations complémentaires (*facultatif*)

# DOSSIER PROFESSIONNEL (DP)

## Activité-type 2

Développer la partie back-end d'une application web en intégrant les recommandations de sécurité.

*Exemple n° 3 ▶ MVC/CRUD*

### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

#### Définition de la structure MVC

##### 1. Modèle

Dans la structure MVC, le Modèle est responsable de l'interaction avec les données. En utilisant PHP et une base de données (en l'occurrence ici MySQL via wampServer), le modèle traite toutes les demandes d'information, mises à jour et modifications.

```
public function chargementPlaylist()
{
    $req = $this->getBdd()->prepare("SELECT * FROM playlists ORDER BY id ASC");
    $req->execute();
    $mesplaylist = $req->fetchAll(PDO::FETCH_ASSOC);

    foreach ($mesplaylist as $playlist) {
        $play = new Playlist(
            $playlist["id"],
            $playlist["playlist"],
            $playlist["link"],
            $playlist["images"],
        );
        $this->ajoutPlaylist($play);
    }
    $req->closeCursor();
}
```

# DOSSIER PROFESSIONNEL (DP)

## 2. Vue

La Vue est responsable de l'affichage de l'information à l'utilisateur. C'est la partie de votre application qui contient les fichiers HTML, CSS et une partie de JS.

## 3. Contrôleur

Le Contrôleur fait le lien entre le Modèle et la Vue. Il traite les requêtes de l'utilisateur, interagit avec le Modèle pour récupérer les données, puis transmet ces données à la Vue pour être affichées.

```
class PlaylistsController{

    private $playlistManager ;

    public function __construct(){

        $this->playlistManager = new PlaylistManager() ;
        $this->playlistManager->chargementPlaylist();
    }

    // Fonction afin d'afficher la liste des playlists dans la page playlists

    public function afficherPlaylist(){

        $playlists = $this->playlistManager->getPlaylists() ;
        require "views/playlist.view.php" ;
    }
}
```

# DOSSIER PROFESSIONNEL (DP)

J'ai conçu une base de données relationnelle qui stocke les informations des artistes, albums, playlists, les identifiants des utilisateurs et les commentaires. J'ai utilisé PHP pour interagir avec cette base de données, en créant des requêtes SQL pour insérer, récupérer, mettre à jour ou supprimer des données.

Voici un extrait de mon modèle d'utilisateur, qui permet de récupérer une playlist par son identifiant :

Création de la base de données Walidify et des tables dans la base de données :

The screenshot shows the phpMyAdmin interface for MySQL 3306. In the left sidebar, under 'Bases de données', there is a 'Nouvelle base de données' entry. A modal window titled 'Création d'une base de données' is open, showing the input field 'walidify' and the character set 'utf8mb4\_general\_ci'. A 'Créer' button is visible at the bottom right of the modal.

Insertion des tables dans la BDD Walidify :

Table	Action	Lignes	Type	Interclassement	Taille	Perte
admin	Parcourir Structure Rechercher Insérer Vider Supprimer	7	MyISAM	utf8mb4_general_ci	2,5 kio	-
albums	Parcourir Structure Rechercher Insérer Vider Supprimer	9	MyISAM	utf8mb4_general_ci	3,6 kio	872 o
artiste	Parcourir Structure Rechercher Insérer Vider Supprimer	8	MyISAM	utf8mb4_general_ci	2,4 kio	84 o
commentaires	Parcourir Structure Rechercher Insérer Vider Supprimer	4	MyISAM	utf8mb4_general_ci	2,1 kio	-
playlists	Parcourir Structure Rechercher Insérer Vider Supprimer	3	MyISAM	utf8mb4_general_ci	2,3 kio	144 o
users	Parcourir Structure Rechercher Insérer Vider Supprimer	17	MyISAM	utf8mb4_general_ci	3,1 kio	-
6 tables	Somme	48	MyISAM	utf8mb4_general_ci	16,0 kio	1,1 kio

# DOSSIER PROFESSIONNEL (DP)

Chaque table possède plusieurs colonnes pouvant contenir des informations, par exemple la table album :

#	Nom	Type	Interclassement	Attributs	Null	Valeur par d
1	<b>id</b>	int			Non	Aucun(e)
2	nom	varchar(150)	utf8mb4_general_ci		Non	Aucun(e)
3	album	varchar(150)	utf8mb4_general_ci		Non	Aucun(e)
4	tracks	int			Non	Aucun(e)
5	duree	varchar(150)	utf8mb4_general_ci		Non	Aucun(e)
6	date	date			Non	Aucun(e)
7	statut	varchar(150)	utf8mb4_general_ci		Non	Aucun(e)
8	image	varchar(150)	utf8mb4_general_ci		Non	Aucun(e)

L'id : “PRIMARY KEY”, autrement la clé primaire, permet d'identifier chaque enregistrement. Il est auto-incrémenté à chaque enregistrement. Peut être utilisée en tant que clé secondaire.

Nom/album/tracks/durée/date/statut : contiennent des informations à tout moment modifiables et sont relatifs à chaque artiste ou album.

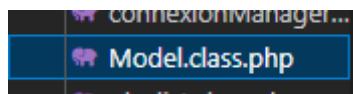
Visualisation finale de la table après l'ajout manuel des informations de l'album :

	Éditer	Copier	Supprimer	id	nom	album	tracks	duree	date	statut	image
<input type="checkbox"/>				1	Travis Scott	Birds in The Trap Sing McKnight	14	53 min	2016-09-16	Classique	birds.png
<input type="checkbox"/>				2	Kanye West	Yeezus	10	40 min	2013-06-18	GOAT	yeezus.png
<input type="checkbox"/>				3	Roddy Ricch	Live Life Fast	18	50 min	2021-12-17	Husler 4 Real	lif.png
<input type="checkbox"/>				4	Future	HNDRXX	19	1h16 min	2017-07-27	Legend	hnrrxx.png
<input type="checkbox"/>				5	Brent Faiyaz	Wasteland	14	1h04 min	2022-07-08	Sheesh	wasteland.png
<input type="checkbox"/>				6	Huncho Jack	Huncho Jack, Jack Huncho	13	41 min	2017-12-21	Freaky	huncho.png
<input type="checkbox"/>				7	Drake	Views	14	1h21 min	2016-05-06	Legend	views.png
<input type="checkbox"/>				8	Metro Boomin	Heroes & Villains	15	48 min	2022-12-02	Marvelous	heroes.png

# DOSSIER PROFESSIONNEL (DP)

## 6. Connexion à la BDD :

Création de la classe Model permettant de créer la connexion à la BDD par l'intermédiaire de PDO.  
La classe est abstraite car elle ne sera jamais instanciée directement.



```
// Class Model va permettre de gérer la connexion à la BDD
abstract class Model{

    // Définit l'attribut en static afin qu'il soit accessible par toutes les classes
    // qui héritent de la classe Model
    private static $pdo;

    private static function setBdd() {

        // Définit le chemin afin d'accéder à la BDD
        self::$pdo = new
        PDO("mysql:host=localhost;dbname=walidify;charset=utf8", "root", "");
        self::$pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_WARNING);
    }

    // Protected = accessible par les classes filles mais pas par des algorithmes
    // tiers
    protected function getBdd() {

        // Dans cette fonction, je test si j'ai déjà une instance de PDO qui existe/ si
        // mon attribut static PDO est vide ou non
        // Si c'est null, il faut que je génère une 1ère connexion
        if(self::$pdo === null){
            self::setBdd();
        }
        return self::$pdo ;
    }
}
```

Ce code définit une classe abstraite PHP appelée Model. Cette classe est destinée à gérer une connexion à une base de données en utilisant PDO (PHP Data Objects), qui est un ensemble d'interfaces de bases de données en PHP.

# DOSSIER PROFESSIONNEL (DP)

Routeur :

“index.php” doit router les demandes des utilisateurs vers les pages correspondantes. Pour cela, le fichier va regarder les urls et choisir le bon canal pour réaliser les actions.

Exemple :

```
<?php

session_start();

define("URL", str_replace("index.php", "", (isset($_SERVER['HTTPS']) ? "https" :
"http") .
"://$_SERVER[HTTP_HOST]$_SERVER[PHP_SELF]"));

try {
    if (empty($_GET['page'])) {

        require "views/accueil.view.php";
    } else {

        // Découpage de l'URL pour accéder aux routes
        $url = explode("/", filter_var($_GET['page']), FILTER_SANITIZE_URL);

        switch ($url[0]) {
            case "accueil":
                require "views/accueil.view.php";
                break;

            case "news":
                require "views/news.view.php";
                break;
        }
    }
}
```

En résumé, ce code permet de charger différentes pages du site en fonction de l'URL demandée par l'utilisateur.

# DOSSIER PROFESSIONNEL (DP)

Fichier .htaccess :

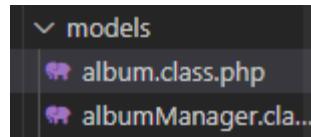
Permet de réécrire dynamiquement l'url en transformant l'affichage php pour devenir un affichage standard :

```
2
3     RewriteEngine On
4
5
6     # Evite d'aller sur le dossier ou fichier (d/f)
7     RewriteCond %{REQUEST_FILENAME} !-f
8     RewriteCond %{REQUEST_FILENAME} !-d
9
10    RewriteRule ^(.*)$ index.php?page=$1
11
```

# DOSSIER PROFESSIONNEL (DP)

Mise en place du MVC pour la page Albums :

1. La class Albums en POO situé dans Models :



```
class Album {  
    private $id;  
    private $nom;  
    private $album;  
    private $statut;  
    private $duree;  
    private $tracks;  
    private $dateSortie ;  
    private $image ;  
  
    public function __construct($id , $nom, $album , $statut , $duree, $tracks,  
$dateSortie, $image){  
        $this->id = $id;  
        $this->nom = $nom;  
        $this->album = $album;  
        $this->statut = $statut;  
        $this->duree = $duree;  
        $this->tracks = $tracks;  
        $this->dateSortie = $dateSortie;  
        $this->image = $image;  
    }  
// Getter & Setter à utiliser dans la page Albums  
  
    public function getId(){return $this->id;}  
    public function getNom(){return $this->nom;}  
    public function getAlbum(){return $this->album;}  
    public function getStatut(){return $this->statut;}  
    public function getDuree(){return $this->duree;}  
    public function getTracks(){return $this->tracks;}  
    public function getDateSortie(){return $this->dateSortie;}  
    public function getImage(){return $this->image;}  
  
    public function setNom($nom){$this->nom = $nom;}  
    public function setAlbum($album){$this->album = $album;}  
    public function setStatut($statut){$this->statut = $statut;}  
    public function setDuree($duree){$this->duree = $duree;}  
    public function setTracks($tracks){$this->tracks = $tracks;}  
    public function setDateSortie($dateSortie){$this->dateSortie = $dateSortie;}  
    public function setImage($image){$this->image = $image;}  
}
```

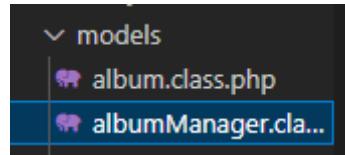
## DOSSIER PROFESSIONNEL (DP)

La class Album a huit propriétés privées : id, nom, album, statut, duree, tracks, dateSortie et image. Ces propriétés stockent les détails de l'album.

Ainsi, ce code définit essentiellement comment un album est représenté dans le cadre de cette application, y compris les données qu'il stocke et les opérations que l'on peut effectuer sur ces données.

# DOSSIER PROFESSIONNEL (DP)

## 2. Le manager d'Albums dans Models :



Il permet de gérer les albums sans passer par des attributs et fonctions "static". Par conséquent, je crée un manager d'albums.

```
require_once "Model.class.php";  
  
require_once "album.class.php";  
  
// ajout du tableau albums dans la class Manager  
  
class AlbumManager extends Model{  
  
    private $albums ;  
    // tableau d'albums  
  
    // L'objet de type $album est transféré en paramètre de fonction dans le tableau  
    $albums[] qui est dans la class album (self)  
    public function ajoutAlbum($album){  
  
        $this->albums[] = $album ;  
        return $this->albums ;  
    }  
    public function getAlbums(){  
        return $this->albums ;  
    }  
  
    // Je crée la requête pour récupérer les albums dans ma BDD  
    public function chargementAlbum(){  
        $req = $this->getBdd()->prepare("SELECT * FROM albums ORDER BY id ASC");  
        $req->execute();  
        // fetchAll permet de récupérer toutes les lignes/ FETCH_ASSOC permet d'éviter  
        // les doublons\ $mesalbums =/ attribut $albums  
        $mesalbums = $req->fetchAll(PDO::FETCH_ASSOC);  
        $req->closeCursor();  
  
        // Parcourir le tableau $mesalbums grâce au foreach et récupérer chaque élément  
        foreach($mesalbums as $album){  
  
            $al = new  
Album($album["id"],$album["nom"],$album["album"],$album["statut"],  
            $album["duree"],$album["tracks"],$album["date"],$album["image"]);  
  
            $this->ajoutAlbum($al);}}}
```

# DOSSIER PROFESSIONNEL (DP)

Ce code définit une classe PHP appelée “AlbumManager”, qui hérite de la classe Model que nous avons vue précédemment. Cette classe est destinée à gérer un tableau d’albums et à interagir avec une base de données pour récupérer les informations des albums.

Dans l’ensemble, la classe “AlbumManager” fournit des méthodes pour gérer un tableau d’albums, y compris la récupération des albums à partir d’une base de données et l’ajout de nouveaux albums à la liste.

Grâce un à un echo <pre>, nous pouvons voir que la requête afin de récupérer les albums dans le tableau albums s’est exécutée avec succès :

```
Array
(
    [0] => Array
    (
        [id] => 1
        [nom] => Travis Scott
        [album] => Birds in The Trap Sing McKnight
        [tracks] => 14
        [duree] => 53 min
        [date] => 2016-09-16
        [statut] => Classique
        [image] => birds.png
    )

    [1] => Array
    (
        [id] => 2
        [nom] => Kanye West
        [album] => Yeezus
        [tracks] => 10
        [duree] => 40 min
        [date] => 2013-06-18
        [statut] => GOAT
        [image] => yeezus.png
    )

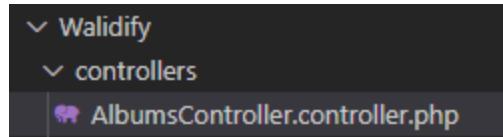
    [2] => Array
    (
        [id] => 3
        [nom] => Roddy Ricch
        [album] => Live Life Fast
        [tracks] => 18
        [duree] => 50 min
        [date] => 2021-12-17
        [statut] => Hustler 4 Real
        [image] => llf.png
    )

    [3] => Array
    (
        [id] => 4
        [nom] => Future
        [album] => HNDRXX
        [tracks] => 19
        [duree] => 1h16 min
    )
)
```

# DOSSIER PROFESSIONNEL (DP)

## 3. Le contrôleur d'albums dans Controllers :

Le contrôleur d'albums pilote la récupération de données des autres fichiers et du lancement des fonctions pour renvoyer un résultat cohérent à l'utilisateur sur la demande de routage :



```
<?php

require_once "models/albumManager.class.php" ;
require_once "CommentController.controller.php" ;

class AlbumsController{

    private $albumManager ;
    private $commentairesController;

    public function __construct() {

        $this->albumManager = new AlbumManager() ;
        $this->albumManager->chargementAlbum() ;

        $this->commentairesController = new CommentaireManager() ;

    }

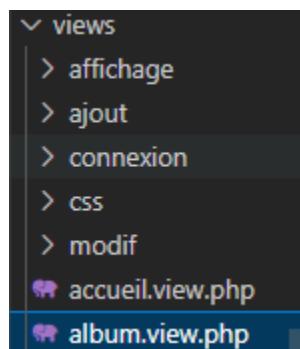
// Fonction afin d'afficher la liste des albums dans la page albums
public function afficherAlbums() {

    $albums = $this->albumManager->getAlbums() ;
    require "views/album.view.php" ;
}
```

Ce code définit une classe PHP nommée “AlbumsController”. Cette classe du contrôleur est utilisée pour gérer les interactions entre les modèles de données pour les albums et les commentaires dans un second temps, et les vues qui affichent ces données à l'utilisateur.

# DOSSIER PROFESSIONNEL (DP)

## 4. La view de la page albums :



```
<div class="content container-fluid my-5">

<!-- Boucle FOR afin d'afficher chaque élément souhaité de l'album grâce aux
GETTER &amp; SETTER --&gt;

&lt;div class="row row-cols-1 row-cols-md-2 row-cols-lg-3 g-4"&gt;
    &lt;?php for ($i = 0; $i &lt; count($albums); $i++) { ?&gt;
&lt;div class="col"&gt;
    &lt;div class="card custom-card"&gt;
        &lt;div class="d-flex justify-content-center"&gt;
&lt;img src="public/images/&lt;?= $albums[$i]-&gt;getImage(); ?&gt;" class="card-img-top
fluid w-50" alt="Image de l'album &lt;?= $albums[$i]-&gt;getAlbum(); ?&gt;"&gt;
        &lt;/div&gt;
        &lt;div class="card-body"&gt;
&lt;h5 class="card-title"&gt;&lt;a href="&lt;?= URL ?&gt;albums/l/&lt;?= $albums[$i]-&gt;getId();
?&gt;"&gt;&lt;?= $albums[$i]-&gt;getAlbum(); ?&gt;&lt;/a&gt;&lt;/h5&gt;
&lt;p class="card-text"&gt;&lt;small class="text-muted"&gt;Par &lt;a href="&lt;?= URL
?&gt;artistes/l/&lt;?= $albums[$i]-&gt;getId(); ?&gt;"&gt;&lt;?= $albums[$i]-&gt;getNom();
?&gt;&lt;/a&gt;&lt;/small&gt;&lt;/p&gt;
&lt;p class="card-text"&gt;&lt;?= $albums[$i]-&gt;getDatesortie(); ?&gt;&lt;/p&gt;
&lt;p class="card-text"&gt;&lt;?= $albums[$i]-&gt;getStatut(); ?&gt;&lt;/p&gt;
        &lt;/div&gt;
    &lt;/div&gt;
&lt;/div&gt;</pre>
```

Ce code HTML et PHP génère une grille d'albums sur la page albums. Chaque album est affiché dans une carte Bootstrap, avec une image, un lien vers les détails de l'album et de l'artiste, la date de sortie, et le statut.

# DOSSIER PROFESSIONNEL (DP)

La boucle PHP “for” parcourt chaque album dans le tableau \$albums défini auparavant dans le contrôleur :

```
$albums = $this->albumManager->getAlbums()
```

Tous les albums sont récupérés et à disposition dans la variable \$albums.

Pour chaque album, il génère un bloc div avec une carte Bootstrap.

L'ensemble de ces cartes albums est affiché en fonction de la taille de l'écran, en une, deux ou trois colonnes grâce aux classes Bootstrap spécifiques.

# DOSSIER PROFESSIONNEL (DP)

Définition du chemin dans la barre de navigation :

```
21 <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
22   <div class="container-fluid">
23     <a class="navbar-brand" href="#">accueil>Validify</a>
24     <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarColor02" aria-controls="navbarColor02" aria-expanded="false" aria-label="Toggle navigation">
25       <span class="navbar-toggler-icon"></span>
26     </button>
27     <div class="collapse navbar-collapse justify-content-center" id="navbarColor02">
28       <ul class="navbar-nav">
29         <li class="nav-item">
30           <a class="nav-link" href="#">artistes>Artistes</a>
31         </li>
32         <li class="nav-item">
33           <a class="nav-link" href="#">albums>Albums</a>
34         </li>
35         <li class="nav-item">
36           <a class="nav-link" href="#">playlists>Playlists</a>
37         </li>
38         <li class="nav-item">
39           <a class="nav-link" href="#">news>Actualités</a>
40         </li>
41         <li class="nav-item">
42           <a class="nav-link" href="#">connexion>Connexion/Inscription</a>
43         </li>
44         <li>
45           <a href="#">déconnexion>
46             <button>Se déconnecter</button>
47           </a>
48         </li>
49       </ul>

```

Indication et redirection du chemin dans le switch de l'index.php :

```
Validity > index.php > ...
9  require_once "controllers/AlbumsController.controller.php";
10 $albumController = new AlbumsController;
11
12
13 // Permet de rediriger vers la page d'accueil ou autre selon contenu de l'URL après 'page' grâce au switch
14
15
16
17 try {
18   if (empty($_GET['page'])) {
19
20     require "views/accueil.view.php";
21   } else {
22
23     // Découpage de l'URL pour accéder aux routes
24     $url = explode("/", filter_var($_GET['page']), FILTER_SANITIZE_URL);
25
26
27     switch ($url[0]) {
28       case "accueil":
29         require "views/accueil.view.php";
30         break;
31
32
33       case "albums":
34         if (empty($url[1])) {
35
36           $albumController->afficherAlbums();
37
38
39       }
40     }
41   }
42 }
```

J'instancie mon contrôleur d'albums et j'appelle la fonction "afficherAlbums()" présent dans le contrôleur d'albums.

# DOSSIER PROFESSIONNEL (DP)

Résultat sur le site :

The screenshot shows the Walidify website's homepage. At the top, there is a dark header bar with the word "WALIDIFY" on the left and navigation links for "ARTISTES", "ALBUMS", "PLAYLISTS", "ACTUALITÉS", "CONNEXION/INSCRIPTION", and "Se déconnecter" on the right. Below the header, a red banner displays the text "PAGE D'ACCUEIL". Underneath the banner, the main heading "BIENVENUE SUR WALIDIFY" is centered in large, bold, black capital letters.

Cliquez sur “albums” ->

The screenshot shows the Walidify website's "PAGE DES ALBUMS" (Albums Page). The page features a grid of six album cards, each with a thumbnail image, the album title, the artist name, the release date, and the genre. The albums listed are:

Album	Artist	Release Date	Genre
BIRDS IN THE TRAP SING MCKNIGHT	Par Travis Scott	2016-09-16	Classique
YEEZUS	Par Kanye West	2013-06-18	GOAT
LIVE LIFE FAST	Par Roddy Ricch	2021-12-17	Husiter 4 Real
HNDRXX	Par Future	2017-07-27	
WASTELAND	Par Brent Faiyaz	2022-07-08	Shanach
HUNCHO JACK, JACK HUNCHO	Par Huncho Jack	2017-12-21	Eraakku

Nous pouvons donc constater que le code est fonctionnel dans sa structure MVC. Grâce au modèle, le chargement des éléments vers la BDD s'est executé, ces données ont été récupérées par le contrôleur et transmises à la vue de la page qui les a affichées dans une card Bootstrap les unes à la suite des autres grâce à la boucle for.

# DOSSIER PROFESSIONNEL (DP)

① [localhost/Walidify/albums](http://localhost/Walidify/albums)

Par ailleurs, la réécriture de l'URL s'est bien exécutée.

Nous pouvons réécrire le code et le modifier afin de l'adapter pour l'affichage des pages Artistes et Playlist.

Playlists :

WALIDIFY

ARTISTES ALBUMS PLAYLISTS ACTUALITÉS CONNEXION/INSCRIPTION Se déconnecter

PAGE DES PLAYLISTS

PLAYLIST POUR GÉRER DE LA ZOUZ  
dfdjh

PLAYLIST RR  
XXX

PLAYLIST POUR LA MUSCU  
Walid

Artistes :

WALIDIFY

ARTISTES ALBUMS PLAYLISTS ACTUALITÉS CONNEXION/INSCRIPTION Se déconnecter

PAGE DES ARTISTES

TRAVIS SCOTT  
31  
XXX

KANYE WEST  
45  
XXX

RODDY RICCH  
24  
XXX

FUTURE  
39  
XXX

BRENT FAIYAZ  
27  
XXX

HUNCHO JACK  
31  
XXX

# DOSSIER PROFESSIONNEL (DP)

## 2. Précisez les moyens utilisés :

PHP, HTML, CSS, BOOTSTRAP

## 3. Avec qui avez-vous travaillé ?

Seul

## 4. Contexte

Nom de l'entreprise, organisme ou association ► *AFCI*

Chantier, atelier, service ► *ATELIER*

Période d'exercice ► Du : *06/03/2023* au : *02/06/2023*

## 5. Informations complémentaires (*facultatif*)

# DOSSIER PROFESSIONNEL (DP)

**Activité-type 2** Développer la partie back-end d'une application web en intégrant les recommandations de sécurité.

*Exemple n° 3 ▶ CRUD*

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

## CRUD

Voici comment j'ai intégré le CRUD à mon site afin d'ajouter, extraire, mettre à jour et supprimer des données de mon site

J'ai commencé par créer les routes menant vers les pages permettant les différentes fonctionnalités.

```
switch ($url[0]) {  
    case "accueil":  
        require "views/accueil.view.php";  
        break;  
  
    case "albums":  
        if (empty($url[1])) {  
  
            $albumController->afficherAlbums();  
  
        } else if ($url[1] === "l") {  
  
            // affichage d'un album}  
  
        else if ($url[1] === "a") {  
  
            // ajout d'un album  
  
        } else if ($url[1] === "m") {  
  
            // modification d'un album  
  
        } else if ($url[1] === "s") {  
  
            // suppression d'un album}
```

# DOSSIER PROFESSIONNEL (DP)

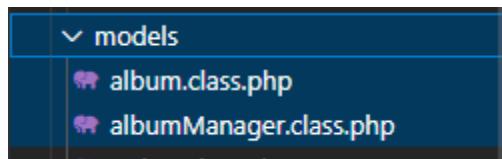
## Affichage d'un album :

Le 3ème élément de l'url doit être l'id de l'album à afficher. Je crée donc une fonction dans le manager et le contrôleur qui permet de récupérer l'id.

Je n'ai pas besoin de faire de requête en BDD car la liste des albums est présente dans l'objet et sera actualisée à chaque instant dans le tableau d'albums disponible dans l'attribut portant ce nom.

Il me suffit donc de parcourir le tableau et de rechercher l'album qui dispose du même id que celui envoyé en paramètre de la fonction. Une fois trouvé, il retourne l'album au programme.

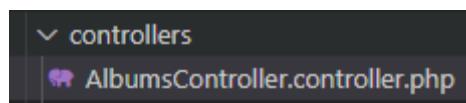
Une fois l'album récupéré au niveau du contrôleur, j'appelle la vue permettant d'afficher l'album.



```
// Fonction qui permet de récupérer un livre selon l'id
public function getAlbumById($id) {

    for($i=0; $i < count($this->albums); $i++) {
        if ($this->albums[$i]->getId() === $id) {
            return $this->albums[$i] ;
        }
    }
}
```

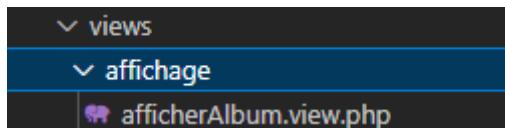
Je parcours ici mon tableau d'albums dont les informations en BDD sont déjà pré chargées.



```
// Fonction afin d'afficher un Album dans une page
public function afficherAlbum($id){
    $album = $this->albumManager->getAlbumById($id);
    require "views/affichage/afficherAlbum.view.php" ;
}
```

# DOSSIER PROFESSIONNEL (DP)

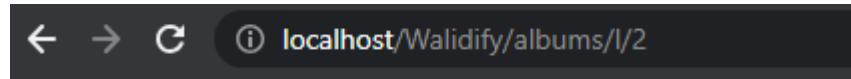
Mon contrôleur dispose donc de l'id trouvé dans sa variable \$album et appelle la vue afin d'y ajouter les informations de l'album.



```
wallyly / views / affichage / afficherAlbum.view.php > ...
1 <?php
2 ob_start()?
3
4 <div class="row">
5   <div class="col-12 col-sm-6 d-flex justify-content-center">
6     <small class="text-muted">Par <a href="= URL<br/?>artistes/l/<?= $albums[$i]->getId(); ?>"><?= $albums[$i]->getNom();  
?></a></small></p>  
<p class="card-text"><?= $albums[$i]->getDatesortie(); ?></p>
```

```
else if ($url[1] === "l") {  
  
    // affichage d'un album  
    $albumController->afficherAlbum($url[2]);  
  
}
```

Résultat dans l'URL :



Résultat à l'affichage :

A screenshot of the Validify website. The top navigation bar includes links for ARTISTES, ALBUMS, PLAYLISTS, ACTUALITÉS, CONNEXION/INSCRIPTION, and a "Se déconnecter" button. The main content area features a red header with the text "YEEZUS". Below the header is an image of a CD cover for the album "Yeezus" by Kanye West. To the right of the image, the album details are listed: Titre : Yeezus, Tracks : 10, Durée : 40 min, and Date de sortie : 2013-06-18.

# DOSSIER PROFESSIONNEL (DP)

## Ajout d'un album :

 <b>HNDRXX</b> Par <a href="#">Future</a> 2017-07-27 Legend	 <b>WASTELAND</b> Par <a href="#">Brent Faiyaz</a> 2022-07-08 Sheeesh	 <b>HUNCHO JACK. JACK HUNCHO</b> Par <a href="#">Huncho Jack</a> 2017-12-21 Freaky
 <b>VIEWS</b> Par <a href="#">Drake</a> 2016-05-06 Legend	 <b>HEROES &amp; VILLAINS</b> Par <a href="#">Metro Boomin</a> 2022-12-02 Marvelous	 <b>ALBUM3</b> Par <a href="#">Artiste3</a> 2023-05-11 statut3

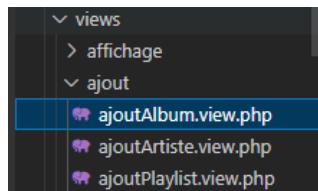
AJOUTER

```
51 |         | </div>
52 |         | </div>
53 |         | </div>
54 |         | </div>
55 |         |     <?php ?>;
56 |         |     </div>
57 |         | </div>
58 |
59 |     <a href="<?= URL ?>artistes/a" class="btn btn-success d-block" style="margin-bottom: 30px;>Ajouter</a>
60 |
61 |
```

J'ajoute l'URL qui amène vers la page d'ajout lorsque l'utilisateur clique dessus.

## **DOSSIER PROFESSIONNEL (DP)**

Création de la vue correspondante :



```
<?php ob_start() ?>

<form method="POST" action=<?= URL ?>albums/av" enctype="multipart/form-data">

    <div class="form-group">
        <label for="artiste">Artiste</label>
        <input type="text" class="form-control" id="artiste" name="artiste">
    </div>

    <div class="form-group">
        <label for="album">Titre de l'album</label>
        <input type="text" class="form-control" id="album" name="album">
    </div>

    <div class="form-group">
        <label for="tracks">Tracks</label>
        <input type="number" class="form-control" id="tracks" name="tracks">
    </div>

    <div class="form-group">
        <label for="duree">Durée</label>
        <input type="text" class="form-control" id="duree" name="duree">
    </div>

    <div class="form-group">
        <label for="dateSortie">Date de sortie</label>
        <input type="date" class="form-control" id="dateSortie" name="dateSortie">
    </div>

    <div class="form-group">
        <label for="statut">Statut</label>
        <input type="text" class="form-control" id="statut" name="statut">
    </div>

    <div class="form-group">
        <label for="image">Image</label>
        <input type="file" class="form-control" id="image" name="image">
    </div>

    <button type="submit" class="btn btn-primary">Valider</button>
</form>
<?php

$content = ob_get_clean();
$titre = "Ajout d'un album";

require "views/template.php";
```

# DOSSIER PROFESSIONNEL (DP)

Résultat dans l'affichage :

**AJOUT D'UN ALBUM**

Artiste	
Titre de l'album	
Tracks	
Durée	
Date de sortie	<input type="text" value="jj/mm/aaaa"/> <span style="float: right;">□</span>
Statut	
Image	<input type="button" value="Choisir un fichier"/> Aucun fichier choisi
<b>VALIDER</b>	

Ajout de la fonction “ajoutAlbum” dans le contrôleur qui va permettre de lancer l'affichage du formulaire d'ajout d'un album en BDD.

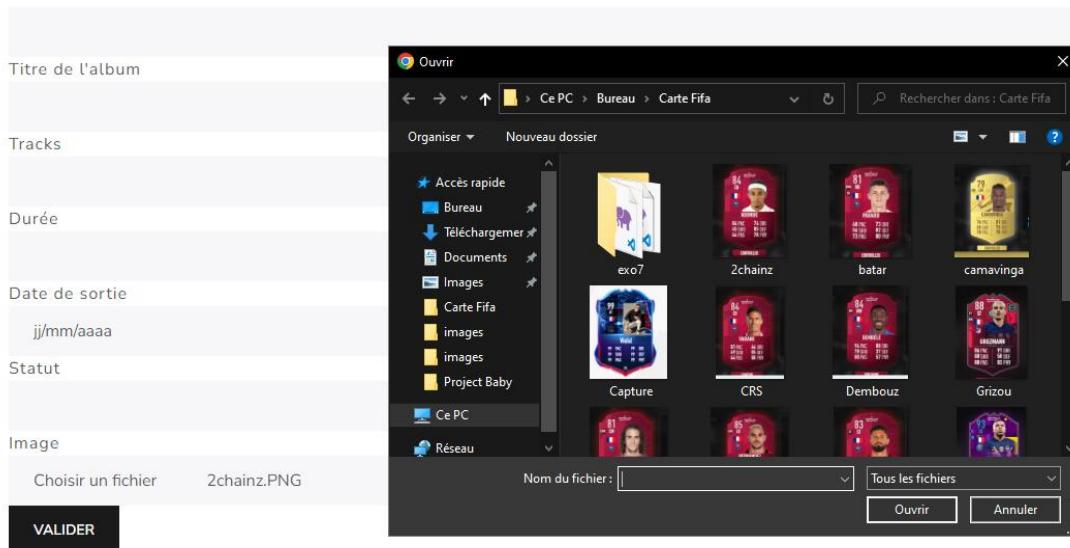
```
// Fonction afin d'ajouter un album dans la liste
public function ajoutAlbum(){
    require "views/ajout/ajoutAlbum.view.php";
}
```

```
public function ajoutAlbumValidation(){
    $file = $_FILES['image'];
    echo "<pre>";
    print_r($file);
    echo "</pre>";
}
```

# DOSSIER PROFESSIONNEL (DP)

Ajout dans index.php de la fonction d'ajout du contrôleur :

```
case "albums":  
    if (empty($url[1])) {  
  
        $albumController->afficherAlbums();  
  
    } else if ($url[1] === "l") {  
  
        // affichage d'un album  
        $albumController->afficherAlbum($url[2]);  
  
    } else if ($url[1] === "a") {  
  
        // ajout d'un album  
  
        $albumController->ajoutAlbum();  
    }  
    else if ($url[1] === "av") {  
  
        $albumController->ajoutAlbumValidation();  
    }  
}
```



```
Array  
(  
    [name] => 2chainz.PNG  
    [type] => image/png  
    [tmp_name] => C:\wamp64\tmp\phpFFCE.tmp  
    [error] => 0  
    [size] => 136995  
)
```

# DOSSIER PROFESSIONNEL (DP)

Nous pouvons constater que les informations de l'image sont bien enregistrées, nous pouvons par conséquent les manipuler en créant une fonction dans le contrôleur.

```
110
111     private function ajoutImage($file, $dir){
112         if(!isset($file['name']) || empty($file['name']))
113             throw new Exception("Vous devez indiquer une image");
114
115         if(!file_exists($dir)) mkdir($dir,0777);
116
117         $extension = strtolower(pathinfo($file['name'],PATHINFO_EXTENSION));
118         $random = rand(0,99999);
119         $target_file = $dir.$random."_".$file['name'];
120
121         if(!getimagesize($file["tmp_name"]))
122             throw new Exception("Le fichier n'est pas une image");
123         if($extension !== "jpg" && $extension !== "jpeg" && $extension !== "png" && $extension !== "gif")
124             throw new Exception("L'extension du fichier n'est pas reconnue");
125         if(file_exists($target_file))
126             throw new Exception("Le fichier existe déjà");
127         if($file['size'] > 500000)
128             throw new Exception("Le fichier est trop gros");
129         if(!move_uploaded_file($file['tmp_name'], $target_file))
130             throw new Exception("l'ajout de l'image n'a pas fonctionné");
131         else return ($random."_".$file['name']);
132     }
133 }
```

La fonction “ajoutImage()” est utilisée pour uploader une image sur le serveur. Elle vérifie plusieurs choses :

Si un fichier a bien été fourni et si le répertoire cible existe (sinon, elle le crée).

Si le fichier est bien une image et si son extension est valide (jpg, jpeg, png, gif).

Si le fichier n'existe pas déjà dans le répertoire et si sa taille n'est pas trop grande.

Finalement, elle déplace le fichier vers le répertoire cible.

Si toutes ces conditions sont respectées, elle renvoie le nom du fichier image nouvellement créé, sinon elle génère une exception expliquant le problème rencontré.

# DOSSIER PROFESSIONNEL (DP)

Dans Models je crée la fonction “ajoutAlbumBd” afin d'y intégrer les requêtes en BDD.

```
7 public function ajoutAlbumBd($nom, $album, $tracks, $duree, $dateSortie,$statut,$image){  
8  
9     // var_dump($dateSortie, $image); die();  
10  
11     $req ="INSERT INTO albums(nom, album, tracks , duree, date, statut, image)  
12         values (:nom, :album, :tracks , :duree, :date, :statut, :image)";  
13     $stmt = $this->getBdd()->prepare($req);  
14     $stmt->bindValue(":nom", $nom, PDO::PARAM_STR);  
15     $stmt->bindValue(":album", $album, PDO::PARAM_STR);  
16     $stmt->bindValue(":tracks", $tracks, PDO::PARAM_INT);  
17     $stmt->bindValue(":duree", $duree, PDO::PARAM_STR);  
18     $stmt->bindValue(":date", $dateSortie);  
19     $stmt->bindValue(":statut", $statut, PDO::PARAM_STR);  
20     $stmt->bindValue(":image", $image, PDO::PARAM_STR);  
21     $resultat = $stmt->execute();  
22     $stmt->closeCursor();  
23     if($resultat > 0){  
24  
25         $album = new Album[$this->getBdd()->lastInsertId(), $nom, $album, $tracks , $duree, $dateSortie, $statut, $image];  
26         $this->ajoutAlbum($album);  
27     }  
28 }  
29  
30 }
```

La fonction “ajoutAlbumBd()” est utilisée pour ajouter un nouvel album dans une base de données. En résumé, cette fonction ajoute un nouvel album dans une base de données et, si l'opération réussit, ajoute également cet album à la liste d'albums gérée dans le tableau par “AlbumManager”.

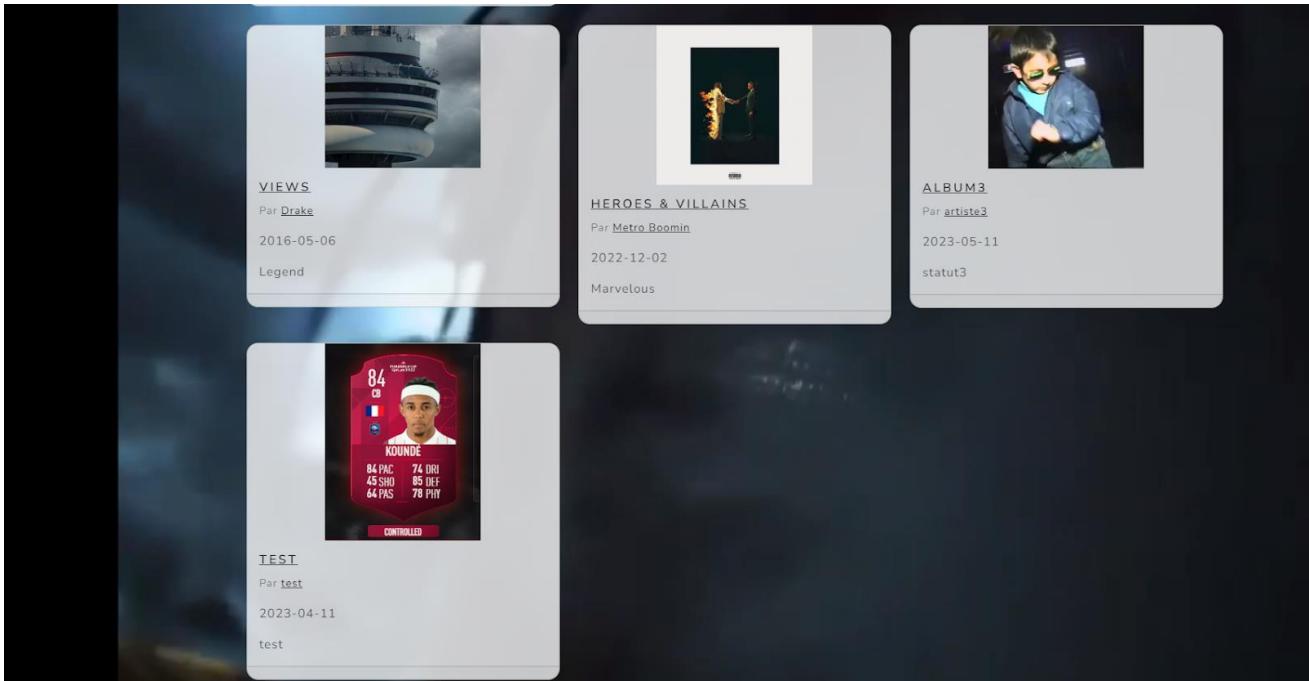
Nous pouvons tester la bonne réalisation du code d'ajout :

**AJOUT D'UN ALBUM**

Artiste	test
Titre de l'album	test
Tracks	10
Durée	54 min
Date de sortie	11/04/2023
Statut	test
Image	<input type="file" value="Choisir un fichier"/> 2chainz.PNG
<b>VALIDER</b>	

# DOSSIER PROFESSIONNEL (DP)

Résultat sur la page d'albums :



Résultat sur la page d'affichage d'un album :

## TEST

Titre : test  
Tracks : 10  
Durée : 54 min  
Date de sortie : 2023-04-11

Partagez votre expérience suite à l'écoute de test

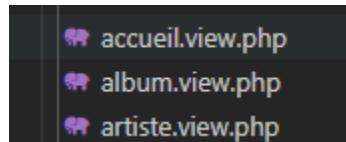
# DOSSIER PROFESSIONNEL (DP)

Résultat dans la base de données :

			id	nom	album	tracks	duree	date	statut	image	
<input type="checkbox"/>	Éditer	Copier	Supprimer	1	Travis Scott	Birds in The Trap Sing McKnight	14	53 min	2016-09-16	Classique	birds.png
<input type="checkbox"/>	Éditer	Copier	Supprimer	2	Kanye West	Yeezus	10	40 min	2013-06-18	GOAT	yeezus.png
<input type="checkbox"/>	Éditer	Copier	Supprimer	3	Roddy Ricch	Live Life Fast	18	50 min	2021-12-17	Husler 4 Real	llf.png
<input type="checkbox"/>	Éditer	Copier	Supprimer	4	Future	HNDRXX	19	1h16 min	2017-07-27	Legend	hndrxx.png
<input type="checkbox"/>	Éditer	Copier	Supprimer	5	Brent Faiyaz	Wasteland	14	1h04 min	2022-07-08	Sheesh	wasteland.png
<input type="checkbox"/>	Éditer	Copier	Supprimer	6	Huncho Jack	Huncho Jack, Jack Huncho	13	41 min	2017-12-21	Freaky	huncho.png
<input type="checkbox"/>	Éditer	Copier	Supprimer	7	Drake	Views	14	1h21 min	2016-05-06	Legend	views.png
<input type="checkbox"/>	Éditer	Copier	Supprimer	8	Metro Boomin	Heroes & Villains	15	48 min	2022-12-02	Marvelous	heroes.png
<input type="checkbox"/>	Éditer	Copier	Supprimer	37	artiste3	album3	234	54 min	2023-05-11	statut3	72398_dance.png
<input type="checkbox"/>	Éditer	Copier	Supprimer	39	test	test	10	54 min	2023-04-11	test	16641_2chainz.PNG

# DOSSIER PROFESSIONNEL (DP)

## Suppression d'un album :



```
30  <div class="row row-cols-1 row-cols-md-2 row-cols-lg-3 g-4">
31  |   <?php for ($i = 0; $i < count($albums); $i++) { ?>
32  <div class="col">
33  |   <div class="card custom-card">
34  |       <div class="d-flex justify-content-center">
35  |           getAlbum(); ?>">
36  |       </div>
37  |       <div class="card-body">
38  |           <h5 class="card-title"><a href="<?= URL ?>albums/l/<?= $albums[$i]->getId(); ?>"><?= $albums[$i]->getAlbum(); ?></a></h5>
39  |           <p class="card-text"><small class="text-muted">Par <a href="<?= URL ?>artistes/l/<?= $albums[$i]->getId(); ?>"><?= $albums[$i]->getNom(); ?></a></small> <small class="text-muted">Le <?= $albums[$i]->getDateSortie(); ?></small> <small class="text-muted">Statut : <?= $albums[$i]->getStatut(); ?></small></p>
40  |           <p class="card-text"><small class="text-muted">Le <?= $albums[$i]->getDateSortie(); ?></small> <small class="text-muted">Statut : <?= $albums[$i]->getStatut(); ?></small></p>
41  |       </div>
42
43  |       <div class="card-footer d-flex">
44  |           <a href="<?= URL ?>albums/m/<?= $albums[$i]->getId(); ?>" class="btn btn-warning">Modifier</a>
45  |           <form method="POST" action="<?= URL ?>albums/s/<?= $albums[$i]->getId(); ?>" onsubmit="return confirm('Voulez-vous vraiment supprimer cet album ?')>
46  |               <button class="btn btn-danger" type="submit">Supprimer</button>
47  |           </form>
48  |       </div>
49
50  </div>
51  </div>
```

Je transforme le bouton en mini-formulaire qui redirige vers l'URL de suppression qui prend en compte l'id de l'album concerné. J'ajoute également une confirmation JS "onSubmit" de suppression pour l'utilisateur.

# DOSSIER PROFESSIONNEL (DP)

Ajout dans index.php de la fonction de suppression du contrôleur :

```
case "albums":  
    if (empty($url[1])) {  
  
        $albumController->afficherAlbums();  
  
    } else if ($url[1] === "l") {  
  
        // affichage d'un album  
        $albumController->afficherAlbum($url[2]);  
    } else if ($url[1] === "a") {  
  
        // ajout d'un album  
  
        $albumController->ajoutAlbum();  
    }  
  
    else if ($url[1] === "s") {  
  
        // suppression d'un album  
  
        $albumController->suppressionAlbum($url[2]);  
    }else {  
        throw new Exception("La page n'existe pas.");  
    }  
break;
```

# DOSSIER PROFESSIONNEL (DP)

Ajout de la fonction dans le contrôleur :

```
public function suppressionAlbum($id) {  
  
    $nomImage = $this->albumManager->getAlbumById($id)->getImage();  
    unlink("public/images/".$nomImage);  
    $this->albumManager->suppressionAlbumBd($id);  
  
    $_SESSION['alert'] = [  
        "type" => "success",  
        "msg" => "Suppression réalisée avec succès"  
    ];  
  
    header('Location: ' . URL . "albums");  
}
```

Cette fonction supprime l'image d'un album du serveur, supprime l'album de la base de données, informe l'utilisateur que la suppression a été réussie et le redirige vers la page des albums.

Création de la fonction “suppressionAlbumBd” dans Models :

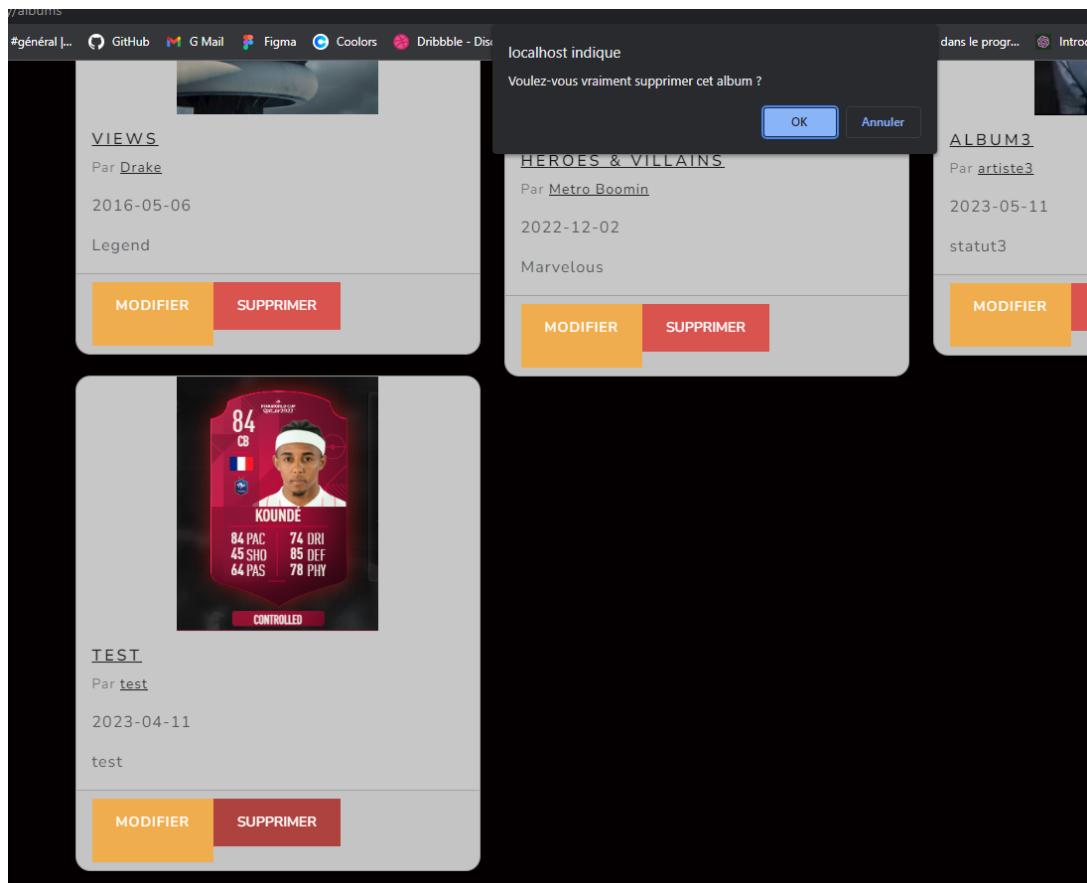
```
public function suppressionAlbumBd($id) {  
  
    $req = "  
Delete from albums where id = :idAlbum  
";  
    $stmt = $this->getBdd()->prepare($req);  
    $stmt->bindValue(":idAlbum", $id, PDO::PARAM_INT);  
    $resultat = $stmt->execute();  
    $stmt->closeCursor();  
    if($resultat>0){  
        $album = $this->getAlbumById($id);  
        unset($album);  
    }    }
```

La méthode “suppressionAlbumBd(\$id)” supprime un album et prends en paramètre de fonction l'id correspondant à l'album sélectionné :

En résumé, cette méthode prépare et exécute une requête SQL pour supprimer un album de la base de données en utilisant l'ID fourni. Si l'opération est réussie, l'album est également supprimé du code PHP.

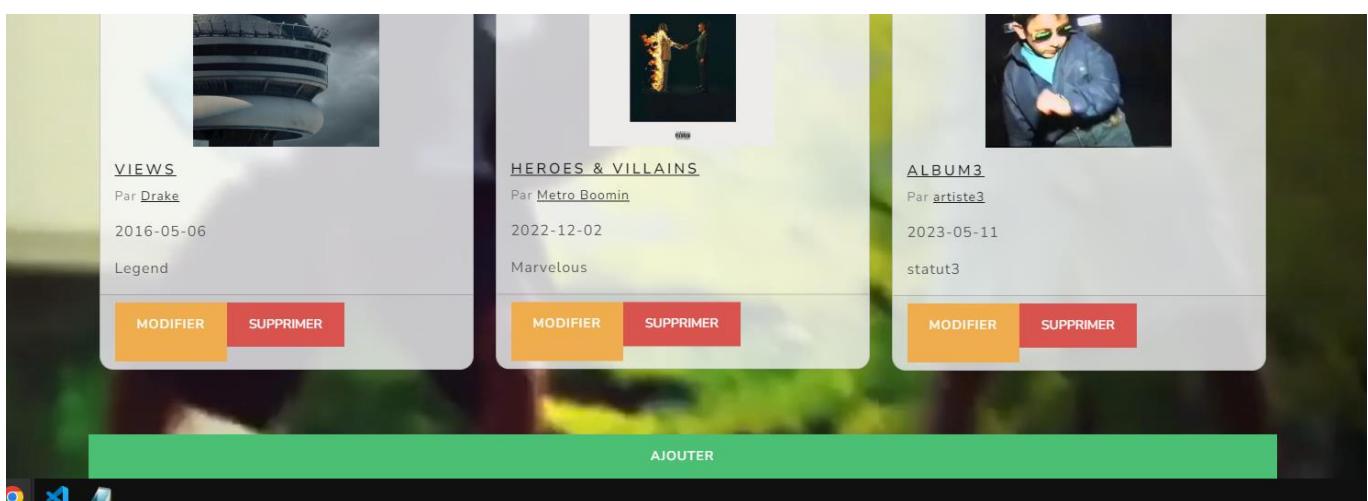
# DOSSIER PROFESSIONNEL (DP)

Suppression d'un album sur le site :



Cliquez sur supprimer du test que je viens de créer.

Je supprime donc l'album et voici le résultat sur l'affichage :



# DOSSIER PROFESSIONNEL (DP)

Résultat sur la base de données :

<input type="checkbox"/>	Éditer	Copier	Supprimer	1	Travis Scott	Birds In The Trap Sing McKnight	14	53 min	2016-09-16	Classique	birds.png
<input type="checkbox"/>	Éditer	Copier	Supprimer	2	Kanye West	Yeezus	10	40 min	2013-06-18	GOAT	yeezus.png
<input type="checkbox"/>	Éditer	Copier	Supprimer	3	Roddy Ricch	Live Life Fast	18	50 min	2021-12-17	Husler 4 Real	llf.png
<input type="checkbox"/>	Éditer	Copier	Supprimer	4	Future	HNDRXX	19	1h16 min	2017-07-27	Legend	hndrxx.png
<input type="checkbox"/>	Éditer	Copier	Supprimer	5	Brent Faiyaz	Wasteland	14	1h04 min	2022-07-08	Sheeesh	wasteland.png
<input type="checkbox"/>	Éditer	Copier	Supprimer	6	Huncho Jack	Huncho Jack, Jack Huncho	13	41 min	2017-12-21	Freaky	huncho.png
<input type="checkbox"/>	Éditer	Copier	Supprimer	7	Drake	Views	14	1h21 min	2016-05-06	Legend	views.png
<input type="checkbox"/>	Éditer	Copier	Supprimer	8	Metro Boomin	Heroes & Villains	15	48 min	2022-12-02	Marvelous	heroes.png
<input type="checkbox"/>	Éditer	Copier	Supprimer	37	artiste3	album3	234	54 min	2023-05-11	statut3	72398_dance.png

Tout cocher   Avec la sélection : Éditer Copier Supprimer Exporter

# DOSSIER PROFESSIONNEL (DP)

## Modification d'un album :

Je lance une nouvelle route pour le bouton Modifier :

```
<div class="card-footer d-flex">
    <a href="#"><?= URL ?>albums/m/<?= $albums[$i]->getId(); ?>" class="btn btn-warning">Modifier</a>
    <form method="POST" action="#"><?= URL ?>albums/s/<?= $albums[$i]->getId(); ?>" onsubmit="return confirm('Voulez-vous vraiment supprimer cet album ?')">
        <button class="btn btn-danger" type="submit">Supprimer</button>
    </form>
```

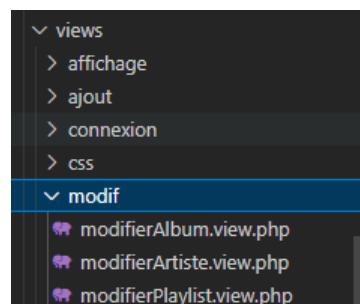
Ajout de la fonction "modificationAlbum" permettant d'afficher le formulaire de modification d'un album.

```
else if ($url[1] === "m") {
    // modification d'un album
    $albumController->modificationAlbum($url[2]);
}
```

La fonction doit récupérer un album afin de l'afficher dans le formulaire de modification.

```
public function modificationAlbum($id) {
    $album = $this->albumManager->getAlbumById($id);
    require "views/modif/modifierAlbum.view.php";
}
```

View de la page de modification d'album.



# DOSSIER PROFESSIONNEL (DP)

```
1  <?php ob_start()?
2
3
4
5  <form method="POST" action="= URL ?>albums/mv" enctype="multipart/form-data">
6
7      <div class="form-group">
8          <label for="artiste">Artiste</label>
9          <input type="text" class="form-control" id="artiste" name="artiste" value="= $album-&gt;getNom() ?">
10     </div>
11
12     <div class="form-group">
13         <label for="album">Titre de l'album</label>
14         <input type="text" class="form-control" id="album" name="album" value="= $album-&gt;getAlbum() ?">
15     </div>
16
17     <div class="form-group">
18         <label for="tracks">Tracks</label>
19         <input type="number" class="form-control" id="tracks" name="tracks" value="= $album-&gt;getTracks() ?">
20     </div>
21
22     <div class="form-group">
23         <label for="duree">Durée</label>
24         <input type="text" class="form-control" id="duree" name="duree" value="= $album-&gt;getDuree() ?">
25     </div>
26
27     <div class="form-group">
28         <label for="dateSortie">Date de sortie</label>
29         <input type="date" class="form-control" id="dateSortie" name="dateSortie" value="= $album-&gt;getDatesortie() ?">
30     </div>
31
32     <div class="form-group">
33         <label for="statut">Statut</label>
34         <input type="text" class="form-control" id="statut" name="statut" value="= $album-&gt;getStatut() ?">
35     </div>
36     <br>
37 <h3>Cover actuelle : </h3>
38 
39
40     <div class="form-group">
41         <label for="image">Changer la cover : </label>
42         <input type="file" class="form-control" id="image" name="image">
43     </div>
44
45     <input type="hidden" name="identifiant" value="= $album-&gt;getId() ?">
46     <button type="submit" class="btn btn-primary">Valider</button>
47
48 </form>
49
50
51 <?php
52
53     $content = ob_get_clean();
54     $titre = "Modification de l'album : ".$album->getId();
55
56     require "views/template.php";
57
```

La route "<?= URL ?>albums/mv" dans index.php renvoie vers la fonction suivante :

```
else if ($url[1] === "mv") {
    $albumController->modificationAlbumValidation();
}
```

# DOSSIER PROFESSIONNEL (DP)

Visuel sur le site et dans l'URL :

localhost/Walidify/albums/m/2

## MODIFICATION DE L'ALBUM : 37

Artiste

artiste3

Titre de l'album

album3

Tracks

234

Durée

54 min

Date de sortie

11/05/2023



Statut

statut3

COVER ACTUELLE :



Changer la cover :

Choisir un fichier

Aucun fichier choisi

VALIDER

Nous pouvons constater que la récupération de l'id dans l'URL et des éléments de l'album dans les values des inputs se sont bien affichées.

# DOSSIER PROFESSIONNEL (DP)

Dans le contrôleur, je peux commencer à créer la fonction “modificationAlbumValidation”.

```
public function modificationAlbumValidation() {
    $imageActuelle = $this->albumManager-
>getAlbumById($_POST['identifiant'])->
getImage();
    $file = $_FILES['image'];

    if($file['size']>0){
        unlink("public/images/".$imageActuelle);
        $repertoire = "public/images/" ;
        $nomImageToAdd =  $this->ajoutImage($file,$repertoire) ; // upload
de l'image
    } else{
        $nomImageToAdd = $imageActuelle ;
    }
    $this->albumManager-
>modificationAlbumBd($_POST['identifiant'],$_POST['artiste'],
$_POST['album'],
$_POST['tracks'],$_POST['duree'],$_POST['dateSortie'],  $_POST['statut'],$nomIma
geToAdd);

    $_SESSION['alert'] = [
        "type" => "success",
        "msg" => "Modification réalisée avec succès"
    ];

    header('Location: ' . URL . "albums");

}
```

En résumé, cette méthode gère le processus de modification d'un album, y compris le remplacement de l'image de l'album et la mise à jour des informations de l'album dans la base de données.

# DOSSIER PROFESSIONNEL (DP)

Voici la fonction “modificationAlbumBd” dans Models qui contient la requête :

```
public function modificationAlbumBd[$id, $nom, $album, $tracks , $duree, $dateSortie, $statut,$image]{  
  
    $req ="  
        update albums  
        set nom = :nom,  
            album = :album,  
            tracks = :tracks,  
            duree = :duree,  
            date = :date,  
            statut = :statut,  
            image = :image  
        where id = :id";  
  
    $stmt = $this->getBdd()->prepare($req);  
    $stmt->bindValue(":id",$id, PDO::PARAM_INT);  
    $stmt->bindValue(":nom",$nom, PDO::PARAM_STR);  
    $stmt->bindValue(":album",$album, PDO::PARAM_STR);  
    $stmt->bindValue(":tracks",$tracks, PDO::PARAM_INT);  
    $stmt->bindValue(":duree",$duree, PDO::PARAM_STR);  
    $stmt->bindValue(":date",$dateSortie);  
    $stmt->bindValue(":statut",$statut, PDO::PARAM_STR);  
    $stmt->bindValue(":image",$image, PDO::PARAM_STR);  
    $resultat = $stmt->execute();  
    $stmt->closeCursor();
```

```
if($resultat>0){  
    $this->getAlbumById($id)->setNom($nom);  
    $this->getAlbumById($id)->setAlbum($album);  
    $this->getAlbumById($id)->setTracks($tracks);  
    $this->getAlbumById($id)->setDuree($duree);  
    $this->getAlbumById($id)->setDatesortie($dateSortie);  
    $this->getAlbumById($id)->setStatut($statut);  
    $this->getAlbumById($id)->setImage($image);  
}  
}
```

# DOSSIER PROFESSIONNEL (DP)

Résultat sur le site :

## MODIFICATION DE L'ALBUM : 37

Artiste

test modif

Titre de l'album

test modif

Tracks

123

Durée

99 min

Date de sortie

27/04/2023



Statut

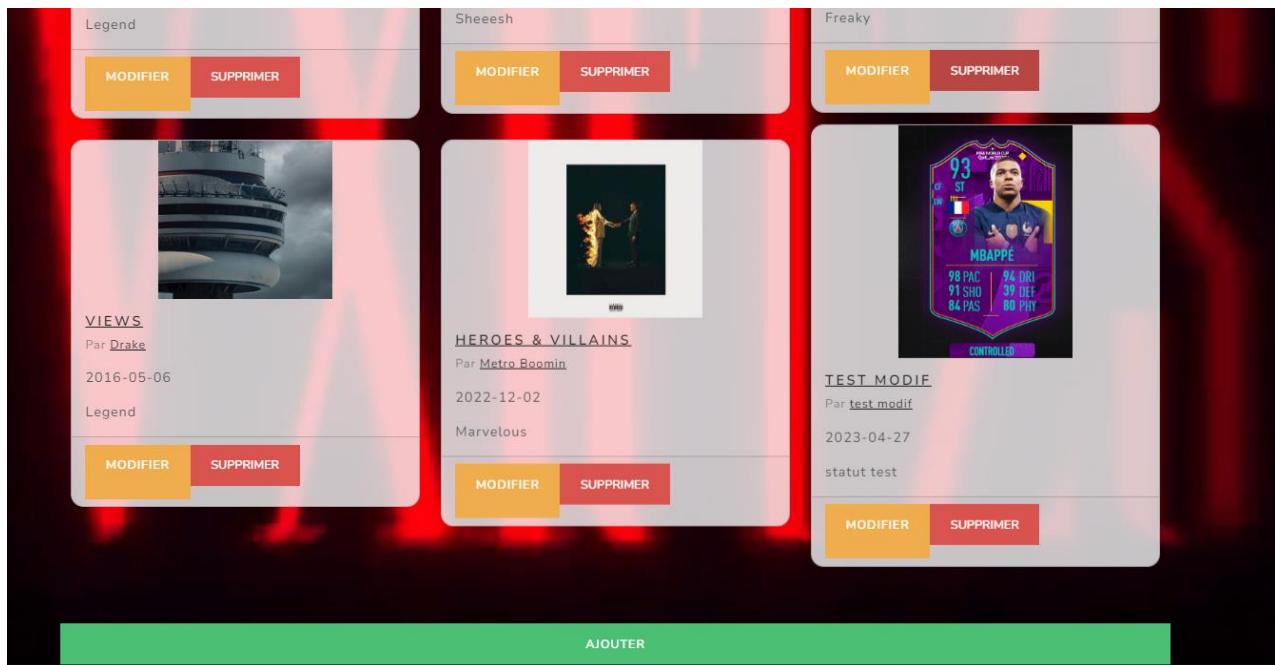
statut test

COVER ACTUELLE :



Modification réalisée avec succès

# DOSSIER PROFESSIONNEL (DP)



Voici comment j'ai développé le CRUD pour les principaux éléments qui composent le site.

## 2. Précisez les moyens utilisés :

PHP, Requêtes SQL, HTML, CSS.

## 3. Avec qui avez-vous travaillé ?

Seul

## 4. Contexte

Nom de l'entreprise, organisme ou association ► **AFCI**

Chantier, atelier, service ► **Atelier**

Période d'exercice ► Du : **06/03/2023** au : **02/06/2023**

# DOSSIER PROFESSIONNEL (DP)

## 5. Informations complémentaires (*facultatif*)

# DOSSIER PROFESSIONNEL (DP)

## Activité-type 2

Développer la partie back-end d'une application web en intégrant les recommandations de sécurité.

*Exemple n° 4 ▶ Connexion/Inscription*

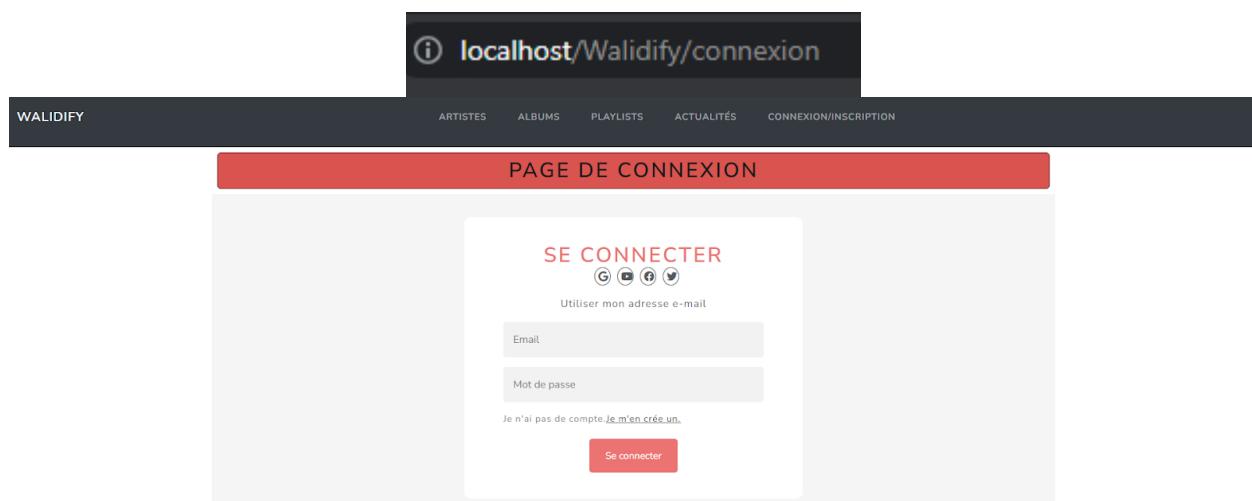
### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

D'autre part, j'ai ajouté des fonctionnalités de connexion, d'inscription et d'ajout de commentaires pour apporter de la personnalité à mon site et de la proximité avec les utilisateurs.

Voici les éléments du code qui composent ces fonctionnalités.

Connexion Inscription :

J'ai commencé par créer les pages html correspondantes et les ai stylisées avec du css :



# DOSSIER PROFESSIONNEL (DP)

```
<?php ob_start() ;?>

<div class="login-page">

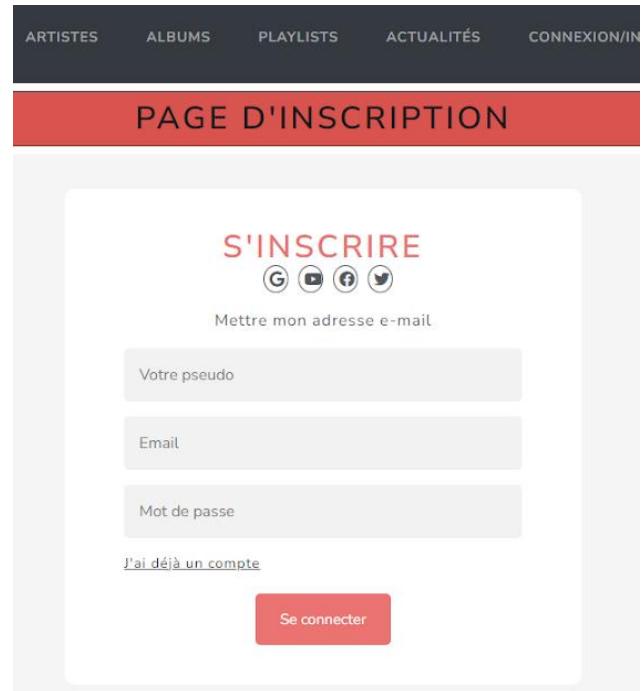
    <form method="POST" action=<?= URL ?>connexion" enctype="multipart/form-data">
        <h1>Se connecter</h1>
        <div class="social-media">
            <p class="fab fa-google"></p>
            <p class="fab fa-youtube"></p>
            <p class="fab fa-facebook"></p>
            <p class="fab fa-twitter"></p>
        </div>
        <p class="choose-email" >Utiliser mon adresse e-mail</p>

        <div class="inputs">
            <input type='email' name="mail" placeholder="Email">
            <input type="password" name="mdp" placeholder="Mot de passe">
        </div>
        <p class="inscription">Je n'ai pas de compte.<a href=<?= URL ?>inscription">Je m'en crée un.</a></p>
        <div align="center">
            <button type="submit" name="envoi">Se connecter</button>
        </div>
    </form>
</div>

<link rel="stylesheet" href="views/css/connexion.css">
<script src="https://kit.fontawesome.com/d832cb9a1e.js" crossorigin="anonymous"></script>
```

(i) localhost/Walidify/inscription

# DOSSIER PROFESSIONNEL (DP)



```
5
6
7 <div class="inscription-page">
8
9     <form method="POST" action=<?= URL ?>inscription" enctype="multipart/form-data">
10        <h1>S'inscrire</h1>
11        <div class="social-media">
12            <p class="fab fa-google"></p>
13            <p class="fab fa-youtube"></p>
14            <p class="fab fa-facebook"></p>
15            <p class="fab fa-twitter"></p>
16        </div>
17        <p class="put-email">Mettre mon adresse e-mail</p>
18
19        <div class="inputs">
20            <input type="text" name="pseudo" placeholder="Votre pseudo">
21            <input type="email" name="mail" placeholder="Email">
22            <input type="password" name="mdp" placeholder="Mot de passe">
23        </div>
24        <p class="inscription"><a href=<?= URL ?>connexion">J'ai déjà un compte</a></p>
25        <div align="center">
26            <button type="submit" name="envoi">Se connecter</button>
27        </div>
28    </form>
29 </div>
30
31     <link rel="stylesheet" href="views/css/inscription.css">
32     <script src="https://kit.fontawesome.com/d832cb9a1e.js" crossorigin="anonymous"></script>
```

# DOSSIER PROFESSIONNEL (DP)

Class et model pour les requêtes de base de données :

```
<?php

class User {
    private $mail;
    private $pseudo;
    private $mdp;
    private $id;

    public function __construct($mail, $pseudo, $mdp, $id) {
        $this->mail = $mail;
        $this->pseudo = $pseudo;
        $this->mdp = $mdp;
        $this->id = $id;
    }

    // users
    public function getId(){return $this->id;}
    public function getMail(){return $this->mail;}
    public function getPseudo(){return $this->pseudo;}
    public function getMdp(){return $this->mdp;}

    public function setMail($mail){$this->mail = $mail;}
    public function setPseudo($pseudo){$this->pseudo = $pseudo;}
    public function setMdp($mdp){$this->mdp = $mdp;}
```

```
1  <?php
2
3  require_once "Model.class.php";
4  require_once "connexion.class.php";
5
6  class UserModel extends Model
7  {
8      // Méthodes pour les utilisateurs
9      public function insertUser($pseudo,$mail,$mdp)
10     {
11         $insertUser = $this->getBdd()->prepare('INSERT INTO users(pseudo, mail, mdp)VALUES(?, ?, ?)');
12         $insertUser->execute(array($pseudo, $mail, $mdp));
13         return $this->getBdd()->lastInsertId();
14     }
15
16     public function getUser($mail, $mdp)
17     {
18         $recupUser = $this->getBdd()->prepare('SELECT * FROM users WHERE mail = ? AND mdp = ?');
19         $recupUser->execute(array($mail, $mdp));
20
21         $userData = $recupUser->fetch(PDO::FETCH_ASSOC);
22         return $userData ? new User[$userData['mail'], $userData['pseudo'], $userData['mdp'], $userData['id'],
23             null, null, null] : null;
24     }
25 }
```

# DOSSIER PROFESSIONNEL (DP)

Contrôleur de connexion :

```
4 class ConnexionController {
5     // Instance de UserModel
6     private $userModel;
7
8     public function __construct() {
9         $this->userModel = new UserModel();
10    }
11
12    public function AfficherPageConnexion() {
13        if (isset($_POST['envoi'])) {
14            if (!empty($_POST['mail']) and !empty($_POST['mdp'])) {
15                $mail = htmlspecialchars($_POST['mail']);
16                $mdp = sha1($_POST['mdp']);
17                $user = $this->userModel->getUser($mail, $mdp);
18
19                if ($user) {
20                    $_SESSION['role'] = 'user';
21                    $_SESSION['mail'] = $mail;
22                    $_SESSION['mdp'] = $mdp;
23                    $_SESSION['id'] = $user->getId();
24                    $_SESSION['pseudo'] = $user->getPseudo();
25                    $_SESSION['alert'] = [
26                        "type" => "success",
27                        "msg" => "Connexion réalisée avec succès"
28                    ];
29
30                    header('Location: ' . URL . "accueil");
31                } else {
32                    echo "Votre mot de passe ou pseudo est incorrect";
33                }
34            } else {
35                echo "Veuillez compléter tous les champs";
36            }
37        }
38        require "views/connexion/connexion.view.php";
39    }
40
41    public function AfficherPageInscription() {
42        if (isset($_POST['envoi'])) {
43            if (!empty($_POST['pseudo']) and !empty($_POST['mdp']) and !empty($_POST['mail'])) {
44                $pseudo = htmlspecialchars($_POST['pseudo']);
45                $mail = htmlspecialchars($_POST['mail']);
46                $mdp = sha1($_POST['mdp']);
47                $id = $this->userModel->insertUser($pseudo, $mail, $mdp);
48
49                $_SESSION['id'] = $id;
50                $_SESSION['role'] = 'user';
51                $_SESSION['pseudo'] = $pseudo;
52                $_SESSION['mail'] = $mail;
53                $_SESSION['mdp'] = $mdp;
54
55                $_SESSION['alert'] = [
56                    "type" => "success",
57                    "msg" => "Inscription réalisée avec succès"
58                ];
59
60                header('Location: ' . URL . "accueil");
61            }
62        }
63    }
64}
```

# DOSSIER PROFESSIONNEL (DP)

J'ajoute également une fonction de déconnexion :

```
public function deconnexion() {
    $_SESSION = array();
    session_destroy();
    header("Location: index.php");
}
```

Voici le routeur qui appelle donc les fonctions du contrôleur :

```
case "connexion":
    if (empty($url[1])) {
        $affichageController->AfficherPageConnexion();
    } else if ($url[1] === "i") {

        $affichageController->AfficherPageInscription();
    }
    else if ($url[1] === "d") {

        $affichageController->deconnexion();
    }

    break;

case "inscription":
    if (empty($url[1])) {
        $affichageController->AfficherPageInscription();

    }

    break;
```

# DOSSIER PROFESSIONNEL (DP)

Test et résultats dans la base de données :

## PAGE D'INSCRIPTION

S'INSCRIRE

(G) (Y) (F) (T)

Mettre mon adresse e-mail

walid test

test@walid

••••

[J'ai déjà un compte](#)

[Se connecter](#)

Vous êtes connecté en tant que walid test

WALIDIFY

ARTISTES ALBUMS PLAYLISTS ACTUALITÉS

SE DÉCONNECTER

## PAGE D'ACCUEIL

Inscription réalisée avec succès

# BIENVENUE SUR WALIDIFY

→ T ←	▼ id	1	mail	pseudo	mdp
<input type="checkbox"/> Éditer  Copier  Supprimer	45	test@walid	walid test	a94a8fe5ccb19ba61c4c0873d391e987982fbbd3	
<input type="checkbox"/> Éditer  Copier  Supprimer	44	walid@test	Walid	a94a8fe5ccb19ba61c4c0873d391e987982fbbd3	
<input type="checkbox"/> Éditer  Copier  Supprimer	43	oil@oil	oil	a94a8fe5ccb19ba61c4c0873d391e987982fbbd3	
<input type="checkbox"/> Éditer  Copier  Supprimer	42	re@re	re	a94a8fe5ccb19ba61c4c0873d391e987982fbbd3	

# DOSSIER PROFESSIONNEL (DP)

Tentative de connexion avec les mêmes infos :

SE CONNECTER

G D F T

Utiliser mon adresse e-mail

test@walid

\*\*\*\*

Je n'ai pas de compte. [Je m'en crée un.](#)

Se connecter

Vous êtes connecté en tant que walid test

WALIDIFY

ARTISTES ALBUMS PLAYLISTS ACTUALITES

PAGE D'ACCUEIL

Connexion réalisée avec succès

Les informations sont bien reconnues en base de données. Le programme récupère bien le pseudo dans la variable de session.

# DOSSIER PROFESSIONNEL (DP)

Par ailleurs, j'ai également fait en sorte d'ajouter une option de connexion pour les admin afin que seulement ces derniers soient en mesure d'interagir avec le CRUD.

J'ai donc réitéré le même code en faisant quelques modifications d'affichage en fonction du rôle (user ou admin).

Exemple dans la page listant les albums :

```
<div class="card-footer d-flex">
<?php if (isset($_SESSION['role'])) && $_SESSION['role'] === 'admin') : ?>
<a href="= URL ?&gt;albums/m/&lt;?= $albums[$i]-&gt;getId(); ?&gt;" class="btn btn-warning"&gt;Modifier&lt;/a&gt;
&lt;form method="POST" action="<?= URL ?&gt;albums/s/&lt;?= $albums[$i]-&gt;getId(); ?&gt;" onsubmit="return confirm('Voulez-vous vraiment supprimer cet album ?')"&gt;
    &lt;button class="btn btn-danger" type="submit"&gt;Supprimer&lt;/button&gt;
&lt;/form&gt;
&lt;?php endif; ?&gt;

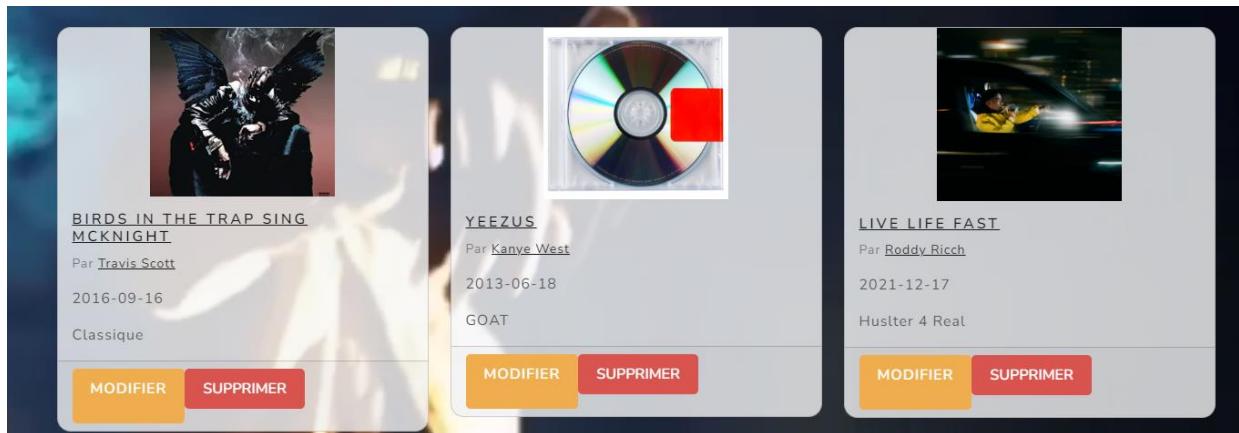
&lt;/div&gt;</pre
```

## 1. La condition

```
<?php if (isset($_SESSION['role'])) && $_SESSION['role'] === 'admin') : ?>
Vérifie si le rôle de l'utilisateur actuellement connecté est 'admin'. Si c'est le cas, les éléments suivants seront affichés.
```

# DOSSIER PROFESSIONNEL (DP)

Résultat si l'admin n'est pas connecté et s'il l'est :



## 2. Précisez les moyens utilisés :

PHP, Requêtes SQL, HTML, CSS.

## 3. Avec qui avez-vous travaillé ?

Seul

# DOSSIER PROFESSIONNEL (DP)

## 4. Contexte

Nom de l'entreprise, organisme ou association ► **AFCI**

Chantier, atelier, service ► Atelier

Période d'exercice ► Du : **03/03/2023** au : **02/06/2023**

## 5. Informations complémentaires ( *facultatif*)

# DOSSIER PROFESSIONNEL (DP)

## Activité-type 2

Développer la partie back-end d'une application web en intégrant les recommandations de sécurité.

*Exemple n° 5 ▶ Espace commentaire*

### 1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Commentaires :

Pour la section commentaires j'ai choisi de l'appliquer à chaque page d'album pour que les utilisateurs puissent partager leurs avis et analyses.

Voici comment la section se présente :

WALIDIFY

ARTISTES ALBUMS PLAYLISTS ACTUALITÉS CONNEXION/INSCRIPTION

Titre : Yeezus  
Tracks : 10  
Durée : 40 min  
Date de sortie : 2013-06-18

Partagez votre expérience suite à lécoute de Yeezus

COMMENTAIRES POSTÉS PAR NOS UTILISATEURS :

Ajouter un commentaire

Vous devez être connecté pour poster un commentaire.  
[Se connecter](#) ou [S'inscrire](#).

# DOSSIER PROFESSIONNEL (DP)

```
<section class="commentaire">

    <div class="comment">

        <p>Partagez votre expérience suite à l'écoute de <?= $album->getAlbum(); ?></p>

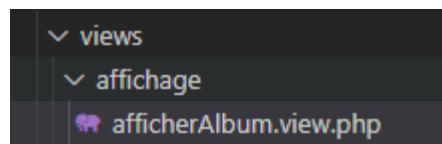
        <h2>Commentaires postés par nos utilisateurs :</h2>

        <button class="toggle-comment-form">Ajouter un commentaire</button>

        <?php if (isset($_SESSION['id']) && ($_SESSION['role'] === 'user' || $_SESSION['role'] === 'admin')) : ?>
            <form method="POST">
                <input type="text" name="pseudo" value="<?php echo $_SESSION['pseudo']; ?>" placeholder="Votre nom">
                <br />
                <textarea name="commentaire" placeholder="Votre commentaire..."></textarea>
                <br />
                <input type="submit" value="Poster mon commentaire" name="submit_commentaire">
            </form>
        <?php else : ?>
            <p>Vous devez être connecté pour poster un commentaire. <br />
            <a href="<?= URL ?>connexion">Se connecter</a> ou <a href="<?= URL ?>connexion/i">S'inscrire</a>.</p>
        <?php endif; ?>

        <?php if (isset($c_msg)) {
            echo $c_msg;
        } ?>
```

Le code html se trouve donc dans le dossier suivant :



Class et Model :

# DOSSIER PROFESSIONNEL (DP)

```
1 <?php
2 class Commentaire {
3
4     private $id;
5     private $pseudo;
6     private $commentaire;
7     private $date_time_post;
8     private $idAlbum;
9
10    public function __construct($id, $pseudo, $commentaire, $date_time_post, $idAlbum) {
11        $this->id = $id;
12        $this->pseudo = $pseudo;
13        $this->commentaire = $commentaire;
14        $this->date_time_post = $date_time_post;
15        $this->idAlbum = $idAlbum;
16    }
17
18    public function getId(){return $this->id;}
19    public function getPseudo(){return $this->pseudo;}
20    public function getCommentaire(){return $this->commentaire;}
21    public function getDate(){return $this->date_time_post;}
22    public function getIdAlbum(){return $this->idAlbum;}
23
24
25
26
27    public function setPseudo($pseudo){$this->pseudo = $pseudo;}
28    public function setCommentaire($commentaire){$this->commentaire = $commentaire;}
29
30
31 }
32
```

```
1 <?php
2
3 require_once "Model.class.php";
4 require_once "commentaire.class.php";
5
6 class CommentaireManager extends Model {
7
8     public function getCommentaires($idAlbum){
9
10         $commentaires = $this->getBdd()->prepare('SELECT * FROM commentaires WHERE idAlbum = ? ORDER BY id DESC');
11         $commentaires->execute(array($idAlbum));
12
13         $commentaireObjects = [];
14         while ($commentaireData = $commentaires->fetch()) {
15             $commentaireObjects[] = new Commentaire(
16                 $commentaireData['id'],
17                 $commentaireData['pseudo'],
18                 $commentaireData['commentaire'],
19                 $commentaireData['date_time_post'],
20                 $commentaireData['idAlbum']
21             );
22         }
23         return $commentaireObjects;
24     }
25
26     public function addCommentaire($pseudo, $commentaire, $idAlbum) {
27         $ins = $this->getBdd()->prepare('INSERT INTO commentaires (pseudo, commentaire, date_time_post, idAlbum)
28             VALUES (?, ?, NOW(), ?)');
29         $ins->execute(array($pseudo, $commentaire, $idAlbum));
30     }
31 }
```

Contrôleur :

# DOSSIER PROFESSIONNEL (DP)

```
7 class CommentaireController{
8
9     private $commentaireManager;
10
11    public function __construct() {
12        $this->commentaireManager = new CommentaireManager();
13    }
14
15    public function addCommentaire($idAlbum) { // Ajoutez $idAlbum comme paramètre
16        if(isset($_POST['submit_commentaire'])){
17            if(isset($_POST['pseudo'], $_POST['commentaire']) AND !empty($_POST['pseudo']) AND !empty($_POST['commentaire']) )
18                $pseudo = htmlspecialchars($_POST['pseudo']);
19                $commentaire = htmlspecialchars($_POST['commentaire']);
20                if(strlen($pseudo) < 25){
21                    $this->commentaireManager->addCommentaire($pseudo, $commentaire, $idAlbum);
22
23                    $c_msg = "Votre commentaire a bien été posté !";
24                }else{
25                    $c_msg = "Erreur : Le pseudo doit faire moins de 25 caractères";
26                }
27            }else{
28                $c_msg = "Erreur : Tous les champs doivent être complétés";
29            }
30        }
31    }
32
33
34
35    public function getCommentaires($idAlbum) {
36        return $this->commentaireManager->getCommentaires($idAlbum);
37    }
}
```

La fonction “addCommentaire” gère donc l’intégration et l’envoi du commentaire vers la base de données.

La fonction “getCommentaire” récupère les commentaires afin de permettre de les afficher.

J’ajoute la fonction dans mon routeur, précisément à la page d’affichage d’un album :

```
switch ($url[0]) {
    case "accueil":
        require "views/accueil.view.php";
        break;

    case "albums":
        if (empty($url[1])) {

            $albumController->afficherAlbums();

        } else if ($url[1] === "l") {

            // affichage d'un album
            $albumController->afficherAlbum($url[2]);
            $commentaireController->addCommentaire($url[2]);
            $commentaires = $commentaireController->getCommentaires($url[2]);
        }
}
```

# DOSSIER PROFESSIONNEL (DP)

J'en ai profité également pour concevoir le code afin que seuls les utilisateurs inscrits et connectés puissent poster. Sinon, je les invite à s'inscrire afin qu'ils aient cette possibilité.

Voici donc le résultat :

Partagez votre expérience suite à l'écoute de Yeezus

**COMMENTAIRES POSTÉS PAR NOS UTILISATEURS :**

[Ajouter un commentaire](#)

Mimoune Date du post : 08/06/2023  
fffff

Mimoune Date du post : 08/06/2023  
Lorem ipsum, dolor sit amet consectetur adipisicing elit. Alias excepturi consequatur sequi dolorem dolorum quasi consectetur, minus eaque! Pariatur similique officia corrupti preferendis. Consectetur nihil, odit ex pariatur deleniti sequi.

Partagez votre expérience suite à l'écoute de Yeezus

**COMMENTAIRES POSTÉS PAR I**

[Ajouter un commentaire](#)

takeoff

J'ai adoré cet album !

[Poster mon commentaire](#)

# DOSSIER PROFESSIONNEL (DP)

Partagez votre expérience suite à l'écoute de Yeezus

## COMMENTAIRES POSTÉS PAR NOS UTILISATEURS :

Ajouter un commentaire

**takeoff** Date du post : 08/06/2023

J'ai adoré cet album !

### Résultat dans la base de données :

<input type="checkbox"/>	Éditer	Copier	Supprimer	22	takeoff	J'ai adoré cet album !	2023-06-08
<input type="checkbox"/>	Éditer	Copier	Supprimer	17	Mimoune	fffff	2023-06-08
<input type="checkbox"/>	Éditer	Copier	Supprimer	16	Mimoune	Lorem ipsum, dolor sit amet consectetur...	2023-06-08
<input type="checkbox"/>	Éditer	Copier	Supprimer	15	Mimoune	Lorem ipsum, dolor sit amet consectetur...	2023-06-08

### 2. Précisez les moyens utilisés :

PHP, Requêtes SQL, HTML, CSS, JavaScript.

### 3. Avec qui avez-vous travaillé ?

Seul

# DOSSIER PROFESSIONNEL (DP)

## 4. Contexte

Nom de l'entreprise, organisme ou association ► **AFCI**

Chantier, atelier, service ► Atelier

Période d'exercice ► Du : **06/03/2023** au : **02/06/2023**

## 5. Informations complémentaires (*facultatif*)

# DOSSIER PROFESSIONNEL (DP)

## **Titres, diplômes, CQP, attestations de formation**

*(facultatif)*

Intitulé	Autorité ou organisme	Date
Baccalauréat	Lycée Louis Bascan	2017
BTS MUC	ISIFA PLUS VALUES	2020

# DOSSIER PROFESSIONNEL (DP)

## Déclaration sur l'honneur

Je soussigné(e) [prénom et nom] Walid Abdoun,

déclare sur l'honneur que les renseignements fournis dans ce dossier sont exacts et que je suis  
l'auteur(e) des réalisations jointes.

Fait à Dunkerque le 09/06/2023

pour faire valoir ce que de droit.

Signature :

## **DOSSIER PROFESSIONNEL (DP)**

## Documents illustrant la pratique professionnelle

(facultatif)

# DOSSIER PROFESSIONNEL (DP)

## ANNEXES

(*Si le RC le prévoit*)