[Date]

SIBOLD Charly

Etudiant en 2eme année de bts sio option slam.

Site Web Ligue1

Sibold Charly.

Sommaire

[1. Un Site web. 2](#_Toc153266760)

[2.Conception du Site. 3](#_Toc153266761)

[2.1.Model View Controller. 3](#_Toc153266762)

[2.2. Router. 3](#_Toc153266763)

[3.Conception des Formulaire. 3](#_Toc153266764)

[3.1. Conception de L’inscription. 3](#_Toc153266765)

[3.1.1. Creation du compte. 3](#_Toc153266766)

[3.2. Conception de la connexion. 3](#_Toc153266767)

[3.2.1.Verification du compte. 3](#_Toc153266768)

[3.3. Conception des Commentaire. 3](#_Toc153266769)

[3.3.1. Ajout du commentaire 3](#_Toc153266770)

[3.3.2. Sécurité injection html css. 3](#_Toc153266771)

# Un Site web.

Un site web, c’est quoi ? Il s’agit d’un ensemble de page web et de ressource reliée par des hyperliens[[1]](#footnote-1), défini et accessible par une adresse web. Il peut appartenir a une entreprise, à une communauté ou bien même à un utilisateur.

Chaque site web est constituée de code HTML , PHP , CSS , javascript et de MySQL.



Au fil des années, les site web sont devenu un point très important donnant un nouveau type de travail , la programmation donc un programmeur . Ce programmeur peut être payer pour créer un site web pour une entreprise avec leur exigence mais , il peut bien créer des site pour lui-même .

Il existe de façon de créer un site web : la conception manuelle et la conception par worbpress.

Le premier site web qui a était créer est World Wide Web de Tim Berners-lee en 1991 .

# 2.Conception du Site.

Pendant la deuxième année de BTS SIO , nous entamons un projet de conception de site web qui sera centre sur une ligue1 avec de club de foot. Les utilisateurs qui seront sur le site pourrons avoir des news de leur club favori quand ils auront créer un compte mais ils peuvent tout a fait commentaire en lisant un article de club proposé par le site

Lors de la création d’un site , il faut respecter certains point mais aussi être bien organisateur car il est tout a fait possible d’avoir énormément de page différent qui nécessite des Controller.

## 2.1.Model View Controller.

Dans netbeans , quand un nouveau projet est créer de type php, un fichier index.php est créer . Quand nous allons créer de nouvelle pages web , de base il se retrouverons dans le même endroit que index.php mais pour un rangement plus propre , on créer un dossier View qui concentrera tout les fichier qui sera le visuel du site .

Une image contenant texte, capture d’écran, Police, conception

Description générée automatiquement

## 2.2. Router.

Le router.php a pour but de permettre de la création de chemin dans le fichier index.php. Si le chemin du fichier est mal saisi ou que le fichier n’existe pas, le routeur enverra un message lorsque que la page en question sera cliqué

|  |
| --- |
| /\*\*  \* Description of Router  \*  \* @author charlysbl26  \*/  class Router {  private array $routes;  public function \_\_construct() {  $this->routes=[];  }      // Ajoute une route au routeur  public function addRoute($url, $controllerFile) {  $this->routes[$url] = $controllerFile;  }  // Traite la demande actuelle  public function execute($url) {  if (array\_key\_exists($url, $this->routes)) {  // Si l'URL correspond à une route, incluez le fichier du contrôleur  $controllerFile = $this->routes[$url];  if (file\_exists($controllerFile)) {  include\_once($controllerFile);  } else {  // Gérer les erreurs si le fichier du contrôleur n'existe pas  echo "Erreur : Contrôleur non trouvé";  }  } else {  // Gérer les erreurs 404 si l'URL n'est pas trouvée  echo "Page non trouvée (Erreur 404)";  }  }  } |

Grace au routeur , il permettra une bonne connexion entre les différent fichier dans l’index.php.

## 2.3. Connexion a la base de données.

Le fichier db\_connect s’occupe de créer la connexion entre le site web et la base de données. Il faut saisir le nom de la base de donnée , l’user ainsi que le mot de passe . Il est tout à fait possible de rajouter d’autre informations pour précisé plus en détaille la base de données que nous voulons connecter.

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">  <title></title>  </head>  <body>  <?php  $host = 'pgsql:host=localhost;dbname=Ligue1';  #$port = '5432';  #$database = 'ligue\_football';  $user = 'postgres';  $password = 'p@ssw0rd';  try {  $pdo = new PDO($host, $user, $password);  $pdo->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);  } catch (PDOException $e) {  die("Erreur de conenxion a la base de données: " . $e->getMessage());  }  ?>  </body>  </html> |

De ce fait, tout les données comme l’inscription du client, la connexion du client et les commentaire pourront être ajouter et récupéré depuis la base de données.

## 2.4. Page Index.

Une fois le routeur configuré, il est maintenant possible de créer une instance ainsi d’ajouter les différentes routes de chaque fichier qu’on associe au site. Lorsque le site est en route, nous verrons l’url que nous avons saisi dans les addRoute au lieu d’avoir le chemin du fichier directement.

|  |
| --- |
| <!DOCTYPE html>  <html lang="fr">  <head>  <meta charset="uff-8">  <link href="./Css/apparance\_site.css" rel="stylesheet">  <title>Accueil</title>  </head>  <body>    <?php  session\_start();  //creer une instance du routeur  include\_once './router.php';  include\_once './menu.php';  $router = new Router();    //Ajouter des routes.  $router->addRoute('/', 'index.php');  $router->addRoute('/Inscription', './Controller/C\_Inscription.php');  $router->addRoute('/liste\_des\_club', './Controller/C\_Club.php');  $router->addRoute('/user\_success', './View/user\_sucess.php' );  $router->addRoute('/GestionMdp', './View/gestionMdp.php');  $router->addRoute('/Connexion', './Controller/C\_Connexion.php');  $router->addRoute('/Article', './Controller/C\_Article.php');  $router->addRoute('/Deconnexion', './View/Deconnexion.php');      $url = $\_SERVER['REQUEST\_URI'];    $router->execute($url);  ?>  </body>  </html> |

# 3.Conception des Formulaire.

Les formulaires sont l’une des essences même des sites internet. Il permet plus possibilité en fonction du besoin de l’auteur du site comme l’ajoute d’un nouveau utilisateur sur son site , lorsqu’il se connecte ou bien même laissé la possibilité aux utilisateurs de donnée leur avis .

## 3.1. Conception de L’inscription.

Le formulaire d’inscription est l’une des formulaires les plus utiliser par les sites, permettant d’ajouter de nouveau utilisateur sur le site. Il demande beaucoup d’informations qui peuvent varier en fonction du type du site.

3.1.1. Création du compte.

Page inscription et son controller.

|  |
| --- |
| <!DOCTYPE html>  <!--  Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license  Click nbfs://nbhost/SystemFileSystem/Templates/Scripting/EmptyPHPWebPage.php to edit this template  -->  <html>  <head>  <meta charset="UTF-8">  <link href="../Css/formulaire\_incription.css" rel="stylesheet">  <title>Inscription</title>  </head>  <body>  <div class="formulaire">  <form method="post" action="./Controller/C\_inscription.php" id="Formulaire">  <p>inscription</p>  <?php  // Vérifiez s'il y a un message d'erreur dans la session  if (isset($\_SESSION['error\_message'])) {  $errorMessage = $\_SESSION['error\_message'];  // Affichez l'alerte d'erreur  echo "<div class='alert'>$errorMessage</div>";  // Effacez le message d'erreur de la session pour éviter de l'afficher à nouveau  unset($\_SESSION['error\_message']);  $Snom = isset($\_SESSION['Snom']) ? $\_SESSION['Snom'] : '';  $Sprenom = isset($\_SESSION['Sprenom']) ? $\_SESSION['Sprenom'] : '';  $Semail = isset($\_SESSION['Semail']) ? $\_SESSION['Semail'] : '';  $Smdp = isset($\_SESSION['Smdp']) ? $\_SESSION['Smdp'] : '';  $Ssexe = isset($\_SESSION['Ssexe']) ? $\_SESSION['Ssexe'] : '';  $emailExists = isset($\_SESSION['SemailExists']) ? $\_SESSION['SemailExists'] : '';  } else {  $Snom = $Sprenom = $Semail = $Smdp = $Ssexe = $emailExists = '';  }  ?>  <div id="alert-box" style="display:none;" class="alert">  <span class="closebtn" onclick="closeAlert()">&times;</span>  Le mot de passe doit comporter au moins une majuscule, une minuscule, un chiffre, un caractère spécial, et faire au moins 8 caractères.  </div>  <div>  <section>  <label for="nom">Nom : </label>  <input class="basic-slide" type="text" name="nom" id="nom" value="<?php echo $Snom; ?>" /><br><br>  </section>  </div>  <div>  <label for="prenom">Prénom : </label>  <input class="basic-slide" type="text" name="prenom" id="prenom" value="<?php echo $Sprenom; ?>" /><br><br>  </div>  <div>  <label for="adresse-email">Email : </label>  <input class="basic-slide" type="text" name="adresse-email" id="adresse-email" value="<?php echo $Semail; ?>"/><br><br>  </div>  <div>  <label for="mdp">mot de passe : </label>  <input type="password" name="mdp" id="mdp" value="<?php echo $Smdp; ?>" /><input type="checkbox" id="showPassword"><br><br>  <text>le mot de passe doit contenir 8 caractère minimun avec une majuscule , une minuscule , un nombre et un caractère spéciaux.</text><br><br>  <div id="strengthBar"></div>  <div id="strengthText">Faible</div>  </div><br>  <div>  <div>  <legend>Sexe</legend>  <label for="Homme">Homme</label>  <input type="radio" id="Homme" name="sexe" value="Homme" <?php echo ($Ssexe === 'Homme') ? 'checked' : ''; ?> />  </div>  <div>  <label for="Femme">Femme</label>  <input type="radio" id="Femme" name="sexe" value="Femme" <?php echo ($Ssexe === 'Femme') ? 'checked' : ''; ?> /><br><br>  </div>  </div>  <div>  <label for="Club\_favori">Votre club favori : </label>  <select name="Club\_favori" id="Club\_favori">  <?php  require\_once('Model/db\_connect.php');  $query = "SELECT id\_club, nom\_club FROM club";  $result = $pdo->query($query);  while ($row = $result->fetch(PDO::FETCH\_ASSOC)) {  $nomClub = $row['nom\_club'];  $idClub = $row['id\_club'];  echo "<option value=\"" . $idClub . "\">" . $nomClub . "</option>";  }  ?>  </select><br><br>  </div>  <div>  <label for="photo">Photo de profil :</label>  <input type="file" id="photo" name="photo" accept="image/\*"><br><br>  </div>  <div>  <p>Choisissez les clubs auxquels vous souhaitez avoir des new :</p>  <input type="checkbox" id="paris\_fc" name="clubs[]" value="Paris\_fc">  <label for="paris\_fc">Paris FC</label><br>  <input type="checkbox" id="valence" name="clubs[]" value="valence">  <label for="valence">valence</label><br>  <input type="checkbox" id="lyonnais" name="clubs[]" value="lyonnais">  <label for="lyonnais">lyonnais</label><br>  <input type="checkbox" id="marseille" name="clubs[]" value="marseille">  <label for="marseille">Olympique de marseille</label><br>  <input type="checkbox" id="monaco" name="clubs[]" value="monaco">  <label for="monaco">AS Monaco</label><br><br>  </div>  <div>  <div>Teachable Machine Image Model</div>  <button type="button" onclick="init()">Start</button>  <div id="webcam-container"></div>  <div id="label-container"></div>  <script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@latest/dist/tf.min.js"></script>  <script src="https://cdn.jsdelivr.net/npm/@teachablemachine/image@latest/dist/teachablemachine-image.min.js"></script>  <script type="text/javascript">  // More API functions here:  // https://github.com/googlecreativelab/teachablemachine-community/tree/master/libraries/image  // the link to your model provided by Teachable Machine export panel  const URL = "../model/";  let model, webcam, labelContainer, maxPredictions;  // Load the image model and setup the webcam  async function init() {  const modelURL = URL + "model.json";  const metadataURL = URL + "metadata.json";  // load the model and metadata  // Refer to tmImage.loadFromFiles() in the API to support files from a file picker  // or files from your local hard drive  // Note: the pose library adds "tmImage" object to your window (window.tmImage)  model = await tmImage.load(modelURL, metadataURL);  maxPredictions = model.getTotalClasses();  // Convenience function to setup a webcam  const flip = true; // whether to flip the webcam  webcam = new tmImage.Webcam(200, 200, flip); // width, height, flip  await webcam.setup(); // request access to the webcam  await webcam.play();  window.requestAnimationFrame(loop);  // append elements to the DOM  document.getElementById("webcam-container").appendChild(webcam.canvas);  labelContainer = document.getElementById("label-container");  for (let i = 0; i < maxPredictions; i++) { // and class labels  labelContainer.appendChild(document.createElement("div"));  }  }  async function loop() {  webcam.update(); // update the webcam frame  await predict();  window.requestAnimationFrame(loop);  }  // run the webcam image through the image model  async function predict() {  // predict can take in an image, video or canvas html element  const prediction = await model.predict(webcam.canvas);  for (let i = 0; i < maxPredictions; i++) {  const classPrediction =  prediction[i].className + ": " + prediction[i].probability.toFixed(2);  labelContainer.childNodes[i].innerHTML = classPrediction;  }  }  </script><br><br>  </div>      <div>  <input type="submit" value="S'inscrire" name="submit">  </div>  </form>  </div>      <script>  const passwordInput = document.getElementById("mdp");  const showPasswordCheckbox = document.getElementById("showPassword");  showPasswordCheckbox.addEventListener("change", function () {  if (showPasswordCheckbox.checked) {  passwordInput.type = "text"; // Affiche le mot de passe  } else {  passwordInput.type = "password"; // Masque le mot de passe  }  });  passwordInput.addEventListener("input", function () {  const password = passwordInput.value;  const length = password.length;  const hasSpecialChar = /[!@#$%^&\*()\_+\-=\[\]{};':"\\|,.<>\/?]+/.test(password);  const hasDigit = /[0-9]+/.test(password);  const hasUppercase = /[A-Z]+/.test(password);  const hasLowercase = /[a-z]+/.test(password);  let strength = 0;  if (length >= 8)  strength++;  if (hasSpecialChar)  strength++;  if (hasDigit)  strength++;  if (hasUppercase)  strength++;  if (hasLowercase)  strength++;  const colors = ["white", "red", "yellow", "green", "darkgreen"];  strengthBar.style.width = Math.min((strength \* 33.33), 99.99) + "%"; // 33.33% for each strength level  strengthBar.style.backgroundColor = colors[strength];  const strengthLabels = ["Faible", "peu faible", "Moyen", "Fort", "Très\_fort"];  strengthText.innerText = strengthLabels[strength];  });  function closeAlert() {  document.getElementById("alert-box").style.display = "none";  }  document.getElementById("Formulaire").addEventListener("submit", function (event) {  var nom = document.getElementById("nom").value;  var prenom = document.getElementById("prenom").value;  var password = document.getElementById("mdp").value;  var errorMessage = "";  if (nom.trim() === "") {  errorMessage += "Le champ nom est obligatoire. ";  } else if (prenom.trim() === "") {  errorMessage = "Le champ prénom est obligatoire.";  } else if (email.trim() === "" || !email.includes("@")) {  errorMessage = "L'adresse email n'est pas valide.";  } else if (  nom.trim() === "" ||  prenom.trim() === "" ||  password.length < 8 ||  !/[A-Z]/.test(password) || // Au moins une majuscule  !/[a-z]/.test(password) || // Au moins une minuscule  !/[0-9]/.test(password) || // Au moins un chiffre  !/[^a-zA-Z0-9]/.test(password) // Au moins un caractère spécial  ) {  event.preventDefault(); // Empêche l'envoi du formulaire  document.getElementById("alert-box").style.display = "block";  }  });  </script>  </body>  </html> |
| Controller  <?php  //echo "Controleur inscription " . $\_SERVER['SCRIPT\_NAME'];  $page = $\_SERVER['SCRIPT\_NAME'];  if ($page == "/index.php") {  require\_once ('./Model/db\_connect.php');  include("./View/Inscription.php");  } else { //page Controller/c\_inscription  require\_once ('../Model/db\_connect.php');  //Tableau contenant les informatique de l'utilisateur.  //var\_dump($\_POST);  if (isset($\_POST['submit']) /\* && $\_POST['nom'] != null && $\_POST['prenom'] != null && $\_POST['adresse-email'] != null && $\_POST['mdp'] != null && $\_POST['sexe'] != null \*/) {  echo "premier if";  $nom = $\_POST['nom'];  $prenom = $\_POST['prenom'];  $email = $\_POST['adresse-email'];  $mdp = $\_POST['mdp'];  $sexe = $\_POST['sexe'];  $clubFavori = $\_POST['Club\_favori'];  $photo = $\_FILES['photo'];  #$clubNews=$\_POST['checkbox'];  $hashingMethod = $\_POST['hashingMethod'];  $hashed\_password = '';  $errorMessage = "";  if (empty($nom)) {  $errorMessage .= "Le champ nom est obligatoire. <br>";  } else {  $\_SESSION['Snom'] = $nom;  }  if (empty($prenom)) {  $errorMessage .= "Le champ prénom est obligatoire. <br>";  } else {  $\_SESSION['Sprenom'] = $prenom;  }  if (empty($email) || !filter\_var($email, FILTER\_VALIDATE\_EMAIL)) {  $errorMessage .= "L'adresse email n'est pas valide.<br> ";  } else {  $\_SESSION['Semail'] = $email;  }  $checkEmailSql = "SELECT COUNT(\*) FROM utilisateur WHERE mail\_uti = :mail";  $checkEmailQuery = $pdo->prepare($checkEmailSql);  $checkEmailQuery->bindParam(':mail', $email, PDO::PARAM\_STR);  $checkEmailQuery->execute();  $emailExists = $checkEmailQuery->fetchColumn();  if ($emailExists) {  $errorMessage .= "Cet email est déjà enregistré. Veuillez utiliser un autre email.<br>";  } else {  $\_SESSION['SemailExists'] = $emailExists;  }  if (  strlen($mdp) < 8 ||  !preg\_match('/[A-Z]/', $mdp) ||  !preg\_match('/[a-z]/', $mdp) ||  !preg\_match('/[0-9]/', $mdp) ||  !preg\_match('/[^a-zA-Z0-9]/', $mdp)  ) {  $errorMessage .= "Le mot de passe doit respecter les critères requis.<br> ";  } else {  $hashed\_password = password\_hash($mdp, PASSWORD\_DEFAULT);  $\_SESSION['Smdp'] = $mdp;  }  if (empty($sexe)) {  $errorMessage .= "Le choix du sexe est obligatoire. <br>";  } else {  $\_SESSION['Ssexe'] = $sexe;  }  if (!empty($errorMessage)) {  // Rediriger l'utilisateur vers le formulaire avec les erreurs spécifiques  $\_SESSION['error\_message'] = $errorMessage;  header("Location: /Inscription");  exit;  }  if (!empty($\_FILES['photo']['name'])) {  $photo = $\_FILES['photo'];  $target\_directory = "../img";  $target\_file = $target\_directory . basename($photo['name']);  move\_uploaded\_file($photo['tmp\_name'], $target\_file);  } else {  $target\_file = null; // ou une valeur par défaut  }  $sql = "INSERT INTO utilisateur (id\_club ,nom\_uti , prenom\_uti , sexe\_uti , password\_uti ,date\_inscription, image\_uti, mail\_uti) VALUES ( :id\_club, :nom , :prenom , :sexe , :pwd, NOW() ,:image, :mail)";  $requete = $pdo->prepare($sql);  $requete->bindParam(':id\_club', $clubFavori, PDO::PARAM\_STR);  $requete->bindParam(':nom', $nom, PDO::PARAM\_STR);  $requete->bindParam(':prenom', $prenom, PDO::PARAM\_STR);  $requete->bindParam(':sexe', $sexe, PDO::PARAM\_STR);  $requete->bindParam(':pwd', $hashed\_password, PDO::PARAM\_STR);  $requete->bindParam(':image', $target\_file, PDO::PARAM\_STR);  $requete->bindParam(':mail', $email, PDO::PARAM\_STR);  try {  $requete->execute();  header("Location: /user\_success");  } catch (PDOException $e) {  die("Erreur de la requete sql " . $e->getMessage());  }  $pdo = null;  } else {  include("./View/Inscription.php");  }  }  ?> |

Ces codes sont séparés en deux fichiers. L’un est la vue du formulaire tandis que l’autre est le contrôleur. Ces deux fichiers sont associés par le link ref qui précise ou le formulaire est associer permettant ainsi de vérifier si toutes les données ont été saisi dans le formulaire avant de l’enregistrer dans la base de données. Il passe par une série de vérifications avant l’envoie, si une erreur est détectée alors un message d’erreur sera envoyé tandis que la page se rechargera avec le message d’erreur.

## 3.2. Conception de la connexion.

3.2.1.Verification du compte.

Etant donnée que maintenant la base de données récupère des informations d’utilisateurs , il sera possible de permettre au utilisateurs de se connecter avec leur propre identifiants. C’est pourquoi le formulaire de connexion est créer , il va vérifier les informations saisi si ils sont dans la base de données ou pas.

|  |
| --- |
| <!DOCTYPE html>  <!--  Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license  Click nbfs://nbhost/SystemFileSystem/Templates/Scripting/EmptyPHPWebPage.php to edit this template  -->  <html>  <head>  <meta charset="UTF-8">  <link href="../Css/apparance\_connexion.css" rel="stylesheet">  <title>Connexion</title>  </head>  <body>  <form method="post" action="/Connexion" id="Connexion">      <div class="Connexion">  <p>Connexion</p>  <?php  // Vérifiez s'il y a un message d'erreur dans la session  if (isset($\_SESSION['error\_message'])) {  $errorMessage = $\_SESSION['error\_message'];  // Affichez l'alerte d'erreur  echo "<div class='alert'>$errorMessage</div>";  // Effacez le message d'erreur de la session pour éviter de l'afficher à nouveau  unset($\_SESSION['error\_message']);  $Snom = isset($\_SESSION['Snom']) ? $\_SESSION['Snom'] : '';  $Smdp = isset($\_SESSION['Smdp']) ? $\_SESSION['Smdp'] : '';  } else {  $Snom = $Smdp = '';  }  ?>  <section>  <label for="nom">Nom : </label>  <input class="basic-slide" type="text" name="nom" id="nom" value="" /><br><br>  </section>  <div>  <label for="mdp">mot de passe : </label>  <input type="password" name="mdp" id="mdp" value="" /><input type="checkbox" id="showPassword"><br><br>  </div>  <div>  <input type="submit" value="Connexion" name="submit">  </div>  </div>  </form>  <script>  <script>  const passwordInput = document.getElementById("mdp");  const showPasswordCheckbox = document.getElementById("showPassword");  showPasswordCheckbox.addEventListener("change", function () {  if (showPasswordCheckbox.checked) {  passwordInput.type = "text"; // Affiche le mot de passe  } else {  passwordInput.type = "password"; // Masque le mot de passe  }  });  </script>  </body>  </html> |
| Controller  <?php  $page = $\_SERVER['SCRIPT\_NAME'];  //echo "<script>alert('" . $page . "');</script>";  /\* if ($page == "/index.php") {  require\_once ('./Model/db\_connect.php');  include("./View/Connexion.php");  } else { //page Controller/C\_Connexion.php  require\_once ('../Model/db\_connect.php');  \*/  require\_once ('./Model/db\_connect.php');  if (isset($\_POST['submit'])) {  echo "<script>alert('" . $page . "');</script>";  echo "formulaire soumis";  $nom = $\_POST['nom'];  $mdp = $\_POST['mdp'];    echo $nom." ".$mdp;  $errorMessage = "";  $query = "SELECT nom\_uti, password\_uti , image\_uti FROM utilisateur WHERE nom\_uti = ?";  $stmt = $pdo->prepare($query);  $stmt->execute([$nom]);  $user = $stmt->fetch(PDO::FETCH\_ASSOC);  if (!$user) {  $errorMessage .= "Le login n'existe pas";  } else {  // Vérifier le mot de passe haché  $hashedPassword = $user['password\_uti'];  if (!$user || !password\_verify($mdp, $user['password\_uti'])) {  // Identifiants incorrects  $errorMessage .= "Le nom d'utilisateur ou le mot de passe est incorrect";  } else {  // Authentification réussie  $\_SESSION['authenticated'] = true;  $\_SESSION['Snom'] = $nom;  }  }  if (!empty($errorMessage)) {  // Rediriger l'utilisateur vers le formulaire avec les erreurs spécifiques  $\_SESSION['error\_message'] = $errorMessage;  var\_dump($errorMessage);  header("Location: /Connexion");  //exit;  } else {  // Authentification réussie, rediriger vers la page appropriée  header("Location:/user\_success");  //exit;  }  } else {  include("./View/Connexion.php");  } |

Une fois la vérification terminé est que les informations existe dans la base de données , l’utilisateur est renvoyer sur la page d’accueil et il sera possible de voir que le bouton connexion a disparu laissant place au nom de l’utilisateur plus une icone si l’utilisateur a mis une image. Et un bouton de déconnexion est ajouter pour permettre a l’utilisateur de ce déconnecter.

## 3.3. Conception des Commentaire.

A la l’inverse des formulaires d’inscription et de connexions. La page du commentaire a pour but de récupéré les différents commentaires des utilisateurs. Que ce soit des personnes connecter au site ou non

3.3.1. Ajout du commentaire

|  |
| --- |
| <!DOCTYPE html>  <!--  Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license  Click nbfs://nbhost/SystemFileSystem/Templates/Scripting/EmptyPHPWebPage.php to edit this template  -->  <html>  <head>  <meta charset="UTF-8">  <link href="../Css/apparance\_article.css" rel="stylesheet">  <title></title>  </head>  <body>  <div class="article">  <form method="post" action="./Controller/C\_Article.php" id="Article">  <div>  <?php  // Vérifiez s'il y a un message d'erreur dans la session  if (isset($\_SESSION['error\_message'])) {  $errorMessage = $\_SESSION['error\_message'];  // Affichez l'alerte d'erreur  echo "<div class='alert'>$errorMessage</div>";  // Effacez le message d'erreur de la session pour éviter de l'afficher à nouveau  unset($\_SESSION['error\_message']);  $Snom = isset($\_SESSION['SnomCom']) ? $\_SESSION['SnomCom'] : '';  $Sprenom = isset($\_SESSION['Scommentaire']) ? $\_SESSION['Scommentaire'] : '';  } else {  $SnomCom = $Scommentaire = '';  }  ?>  <p name="articleNews" id="articleNews">  <?php  require\_once ('Model/db\_connect.php');  $query = "SELECT id\_news, titre\_news , article\_news from news";  $result = $pdo->query($query);  while ($row = $result->fetch(PDO::FETCH\_ASSOC)) {  $articleId = $row['id\_news'];  $titreNews = $row['titre\_news'];  $article = $row['article\_news'];  echo '<p>';  echo $titreNews;  echo '</p><br><br>';  echo '<p>';  echo $article;  echo '</p>';  // Champ caché avec l'ID de l'article  echo '<input type="hidden" name="articleNews" value="' . $articleId . '" />';  }    ?>  </p><br><br>  </div>  <div>  <label for="nomCom">Nom du commentateur : </label>  <input class="basic-slide" type="text" name="nomCom" id="nomCom" value="" /><br><br>  </div>  <div>  <label for="commentaire">Commentaire : </label>  <textarea name="commentaire" id="commentaire" rows="4"></textarea><br><br>  </div>  <div>  <input type="submit" value="Commenter" name="submit">  </div>  </form>  <p id="commentaireP">  <?php  // Récupérer les commentaires de la base de données pour un article spécifique  $article\_id = $articleId;  $query = "SELECT nom\_com , text\_com from commentaire comments WHERE id\_news = $article\_id";  $result = $pdo->query($query);    // Afficher les commentaires  while ($row = $result->fetch(PDO::FETCH\_ASSOC)) {  echo "<p><strong>{$row['nom\_com']} :</strong> {$row['text\_com']}</p>";  }  ?>  </p>  </div>  </body>  </html> |
| Controller  <?php  session\_start();  $page = $\_SERVER['SCRIPT\_NAME'];  if ($page == "/index.php") {  require\_once ('Model/db\_connect.php');  include("./View/Article.php");  } else { //page Controller/C\_Article.php  require\_once ('../Model/db\_connect.php');  if (isset($\_POST['submit'])) {  $articleNews = $\_POST['articleNews'];  $nomCom = $\_POST['nomCom'];  $commentaire = $\_POST['commentaire'];  $errorMessage = "";  if (empty($nomCom)) {  $errorMessage .= "Le champ nom du commentaire est obligatoire. <br>";  } else {  $\_SESSION['SnomCom'] = $nomCom;  }  if (empty($commentaire)) {  $errorMessage .= "Le champ commentaire est obligatoire. <br>";  } else {  $\_SESSION['Scommentaire'] = $commentaire;  }  if (!empty($errorMessage)) {  // Rediriger l'utilisateur vers le formulaire avec les erreurs spécifiques  $\_SESSION['error\_message'] = $errorMessage;  header("Location: /Article");  exit;  }  $sql = "INSERT INTO commentaire (id\_news , nom\_com , text\_com) VALUES (:id\_news ,:nomCom , :commentaire)";  $requete = $pdo->prepare($sql);  $commentaire = strip\_tags($commentaire);  $requete->bindParam(':id\_news', $articleNews, PDO::PARAM\_INT);  $requete->bindParam(':nomCom', $nomCom, PDO::PARAM\_STR);  $requete->bindParam(':commentaire', $commentaire, PDO::PARAM\_STR);  try {  $requete->execute();  header("Location: /Article");  } catch (PDOException $e) {  die("Erreur de la requete sql " . $e->getMessage());  }  $pdo = null;  } else {  include("./View/Article.php");  }  }  ?> |

Le programme ci-dessus va récupéré le commentaire de l’utilisateur avec le nom ainsi que le commentaire pour l’insérer dans la base donnée dans la table commentaire. Cependant , il est possible que quelqu’un envoie une commande javascript ou html depuis le commentaire. Ce qui correspond un attaque a donnée Stockée , ce qui signifie que tout les utilisateurs seront touchée par cet attaque qui sera enregistré dans la base de donnée. Il est possible d’empeché son fonctionnement avec des commande comme par exemple le « strip\_tags » qui supprime tout les balise php et html dans le « $commentaire ».

# 4. Procédures stockée

Les procédures stockée sont des commande sql que nous créons pour différent possibilité et aussi pour facilité la recherche des tables dans la base de données. Ça permet d’éviter d’écrire tout la commande et donc permet une vérification plus rapide.

|  |
| --- |
| --@auteur Sibold Charly DROP FUNCTION match\_en\_cours(); CREATE OR REPLACE FUNCTION match\_en\_cours() RETURNS TABLE ( nom\_equipe\_domicile varchar, nom\_equipe\_visiteur varchar, date\_match DATE, score\_domicile INTEGER, score\_visiteur INTEGER ) AS $$ BEGIN RETURN QUERY SELECT e1.nom\_club AS nom\_equipe\_domicile, e2.nom\_club AS nom\_equipe\_visiteur, r.date\_match, r.score\_domicile, r.score\_visiteur FROM rencontre r INNER JOIN club e1 ON [r.id](http://r.id)\_equipe\_domicile = e1.id\_club INNER JOIN club e2 ON [r.id](http://r.id)\_equipe\_visiteur = e2.id\_club WHERE r.status\_rencontre = '1'; END; $$ LANGUAGE plpgsql;  SELECT \* FROM match\_en\_cours(); |

Cette procédure stockée a pour but de renseigner l’utilisateur de tout les match en cours. Il vérifie si le statut est 1 qui signifie que le match est en cours. Sur le site ligue football , ça donne ce résultat.

Une image contenant texte, capture d’écran, plein air

Description générée automatiquement

Pour l’instant il y a qu’un seul match en cours à la date cité.

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">  <link href="../Css/apparance\_match\_en\_cours.css" rel="stylesheet">  <title>Match en cours</title>  </head>  <body>  <div class="match\_en\_cours">  <form method="post" action="" id="Match\_en\_cours">  <h2>Match en cours</h2>  <table>  <tr>  <th>Équipe Domicile</th>  <th>Équipe Visiteur</th>  <th>Date du Match</th>  <th>Score Domicile</th>  <th>Score Visiteur</th>  </tr>  <?php  $host = 'pgsql:host=localhost;dbname=Ligue1';  $user = 'postgres';  $password = 'p@ssw0rd';  $pdo = null;  try {  $pdo = new PDO($host, $user, $password);  $pdo->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);  } catch (PDOException $e) {  die("Erreur de connexion à la base de données: " . $e->getMessage());  }    $query = "SELECT \* FROM match\_en\_cours()";  $result = $pdo->query($query);    while ($row = $result->fetch(PDO::FETCH\_ASSOC)) {  echo '<tr>';  echo '<td>' . $row['nom\_equipe\_domicile'] . '</td>';  echo '<td>' . $row['nom\_equipe\_visiteur'] . '</td>';  echo '<td>' . $row['date\_match'] . '</td>';  echo '<td>' . $row['score\_domicile'] . '</td>';  echo '<td>' . $row['score\_visiteur'] . '</td>';  echo '</tr>';  }  ?>  </table>  </form>  </div>  </body>  </html> |

Et donc , on peut notamment aussi utiliser les procédure stockée directement dans le code php tant qu’il soit connecter à la base de donnée en question sinon rien ne sera afficher.

# 5.Conclusion.

Effectuer ce TP nous a permis de découvrir de nouvelles commandes, les nombre possibilité que nous pouvons confectionner lors de la création d’un site. Et bien sûr les possibilité à faire dans une base de données. Ce TP nous en avant différent commande PHP, SQL que nous pouvons utiliser voir créer afin de donnée site fonctionnel.

1. <https://fr.wikipedia.org/wiki/Hyperlien> [↑](#footnote-ref-1)