

|  |  |  |
| --- | --- | --- |
|  |  |  |
| *Nom de naissance* |  | *BOSSU.* |
| *Nom d’usage* |  |  |
| *Prénom* |  | *Eric* |
| *Adresse* |  | *1A rue de Narvik – 42100 St Etienne* |
|  |  |

|  |  |
| --- | --- |
| **Développeur Web et Mobile** | |
|  | |
| ***Développeur Web et Mobile*** | |
|  | |
| **Modalité d’accès :** | |
|  | |
| ☐ | Parcours de formation |
| ☐ | Validation des Acquis de l’Expérience (VAE) |
|  | |

|  |  |
| --- | --- |
| **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.* | |
|  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Sommaire** | |  | |
| **Exemples de pratique professionnelle** | |  | |
| **Développer la partie front-end d’une application web ou web mobile en intégrant les recommandations de sécurité** | | | **p 07** |
|  |  Maquetter une application | | p 08 |
|  |  Réaliser une interface utilisateur web statique et adaptable | | p 08 |
|  |  Développer une interface utilisateur web dynamique | | p 09 |
|  |  | |  |
| **Développer la partie back-end d’une application web ou web mobile en intégrant les recommandations de sécurité** | | | **p 10** |
|  |  Créer une base de données | | P 10 |
|  |  Développer les composants d’accès aux données | | p 11 |
|  |  Développer la partie back-end d’une application web ou web mobile | | p 12 |
|  |  | |  |
| **Déclaration sur l’honneur** | | | **p 31** |

**Exemples de pratique**

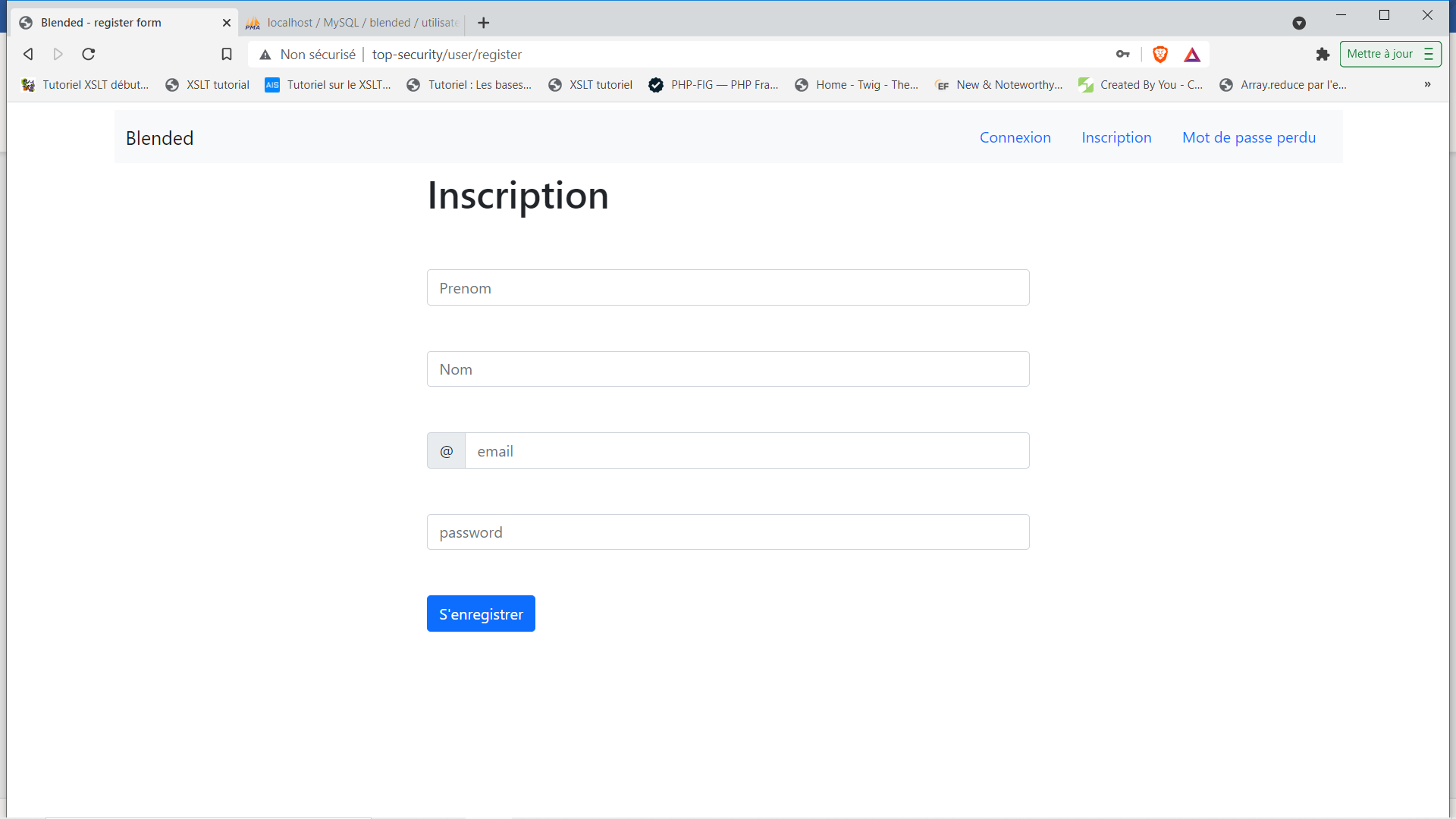
**professionnelle**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Activité-type** | **1** |  | | |
| ***Exemple n°1***  | | **Développer la partie front-end d’une application web ou web mobile en intégrant les recommandations de sécurité** | | |
|  | | | | |
|  | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | |
| **Dans le cadre des cours donnés par Fabrice GUICHARD, nous avons put mettre en place un site blended, blended représentant le nom de notre formation. Nous avons pu ainsi apprendre comment sécuriser notre code.** | | | | |
|  | | | | |
|  | | | | |
|  | | | | |
|  | | | | |
| **2. Précisez les moyens utilisés :** | | | | |
| **PHP**  **BOOTSTRAP**  **MySQL** | | | | |
|  | | | | |
|  | | | | |
|  | | | | |
|  | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | |
|  | | | | |
| Mr Fabrice GUICHARD, formateur GRETA, ainsi que Christian AUBERT, étudiant comme moi | | | | |
|  | | | | |
|  | | | | |
|  | | | | |
| **4. Contexte** | | | | |
|  | | | |  |
| Nom de l’entreprise, organisme ou association | | | | *GRETA* |
| Chantier, atelier, service | | | *Formation DWWM* | |
| Période d’exerciceDu : *02/11/20*au :*21/07/21* | | | | |
|  | | | | |
|  | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | |
|  | | | | |
|  | | | | |
|  | | | | |

|  |
| --- |
| **Maquettage de l’application :** |

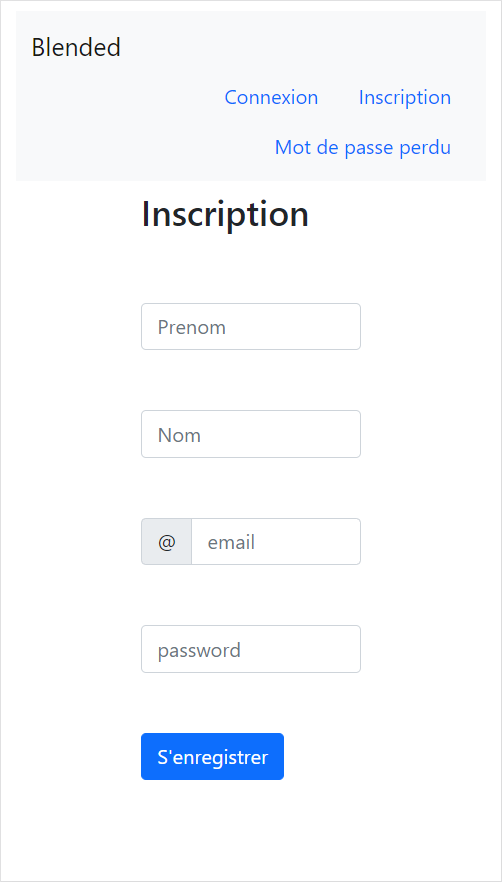


Pour le maquettage, j’ai utilisé Draw.io



Vue du site. Il dispose d’une interface utilisateur web statique et adaptable. Il est responsive. Nous avons utilisé BOOTSTRAP.

Vue Adapté à un mobile :



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Activité-type** | **2** | **Développer la partie back-end d’une application web ou web mobile en intégrant les recommandations de sécurité** | | |
|  | | | | |
|  | | | | |
| **Créer la base de données :** | | | | |
| **Toujours pour ce site blended, il à fallu créer une base de données afin de stocker celles-ci. Pour réaliser la base de données, je me suis servi d’un logiciel de modélisation UML appelé Looping. Il me permet ensuite de générer un script de création de base, pour l’importer dans MySQL par la suite.**  **Ci-dessous le plan de base de données :** | | | | |
|  | | | | |
|  | | | | |
|  | | | | |
| **Développer les composant d’accés aux données :** | | | | |
| **Pour concevoir le site, Mr GUICHARD nous à appris l’architecture MCV – Model – Vue – Control.**  **Nous avons donc utilisé cette méthode.** | | | | |
|  | | | | |
| **Développer la partie back-end d’une application web ou web mobile** | | | | |
| **Toujours en architecture MVC**  **Exemple code model :**  <?php  class Model  {  //    private $field\_list = null;      private $table\_definition = null;      private $dbh;      protected $data = [];      protected $table\_name = null;      public function updateWhere(  $aFields = null, $aWhere = null  )      {          $oQuery = new QueryPrepare( $this->table\_name, $this->table\_definition );          $sQuery = $oQuery->makeUpdateQuery( $aFields, $aWhere );          // preparation de la requete          $stmt = $this->dbh->prepare( $sQuery );          if ( $stmt === false ) {              throw new Exception("Erreur préparation requete : $sQuery");          }  //        print($sQuery);          // remplacement des parametres          $this->bindParams( $stmt, $oQuery->paramUpdate($aFields) );          $this->bindParams( $stmt, $oQuery->paramWhere($aWhere) );          // Execution de la requete          if ( $stmt->execute() === true ) {              // La requete est correctement executee              return(true);          }            return(false);      }      public function create()      {          $nReturnId = 0;          $oQuery = new QueryPrepare( $this->table\_name, $this->table\_definition );          $sQuery = $oQuery->makeInsertQuery();  //        print("Query:" . $sQuery.PHP\_EOL);          // preparation de la requete          $stmt = $this->dbh->prepare( $sQuery );          if ( $stmt === false ) {              throw new Exception("Erreur préparation requete : $sQuery");          }          // remplacement des parametres          $this->bindParams( $stmt, $oQuery->paramInsert() );          // Execution de la requete          if ( $stmt->execute() === true ) {              $nReturnId = (int)$this->dbh->lastInsertId();          }          return($nReturnId);      }      public function index( $aFields = null )      {          return( $this->indexWhere($aFields) );      }      public function indexWhere( $aFields = null, $aWhere = null )      {          $aIndex = array();          $oQuery = new QueryPrepare( $this->table\_name, $this->table\_definition );          $sQuery = $oQuery->makeIndexQuery( $aFields, $aWhere );          // preparation de la requete          $stmt = $this->dbh->prepare( $sQuery );          if ( $stmt === false ) {              throw new Exception("Erreur préparation requete : $sQuery");          }          // remplacement des parametres          $this->bindParams( $stmt, $oQuery->paramWhere($aWhere) );          // Execution de la requete          if ( $stmt->execute() === true ) {              // La requete est correctement executee              $resultat = $stmt->fetchAll(PDO::FETCH\_ASSOC);              if ($resultat !== false) {                  $aIndex = $resultat;              }          }            return($aIndex);      }      public function readWhere( $aWhere = null )      {          $oQuery = new QueryPrepare( $this->table\_name, $this->table\_definition );          $sQuery = $oQuery->makeReadQuery( $aWhere );          // preparation de la requete          $stmt = $this->dbh->prepare( $sQuery );          if ( $stmt === false ) {              throw new Exception("Erreur préparation requete : $sQuery");          }          // remplacement des parametres          $this->bindParams( $stmt, $oQuery->paramWhere($aWhere) );          // Execution de la requete          if ( $stmt->execute() === true ) {              // La requete est correctement executee              $resultat = $stmt->fetch(PDO::FETCH\_ASSOC);              if ($resultat !== false) {                  foreach ($resultat as $sChamp => $value) {                      // Conversion des entiers car PDO retourne des chaines !                      if ( $this->table\_definition[$sChamp]['pdo\_type'] == PDO::PARAM\_INT ) {                          $value = (int)$value;                      }                      $this->\_\_set( $sChamp, $value );                  }                  return(true);              }          }            return(false);      }      public function deleteWhere( $aWhere = null )      {          $oQuery = new QueryPrepare( $this->table\_name, $this->table\_definition );          $sQuery = $oQuery->makeDeleteQuery( $aWhere );          // preparation de la requete          $stmt = $this->dbh->prepare( $sQuery );          if ( $stmt === false ) {              throw new Exception("Erreur préparation requete : $sQuery");          }          // remplacement des parametres          $this->bindParams( $stmt, $oQuery->paramWhere($aWhere) );          // Execution de la requete          if ( $stmt->execute() === true ) {              return(true);          }            return(false);      }      public function bindParams( $stmt, $aParams )      {  //        print\_r($aParams);          // remplacement des parametres          foreach( $aParams as $sChamp => $aParam ) {  //            print("BindParam:".$sChamp);              $value = $aParam['value'] ?? $this->\_\_get($sChamp);              if ( ! $stmt->bindValue( $aParam['parameter'], $value, $aParam['pdo\_type'] ) ) {                  throw new Exception( sprintf( "Erreur bindValue %s %s", $aParam['parameter'], $this->\_\_get($sChamp) ) );              }          }      }      public function \_\_construct( $sTableName, $aTableDefinition)      {          if ( ! is\_string($sTableName) || empty($sTableName) ) {              throw new Exception(\_\_CLASS\_\_.": table name $sTableName incorrect", 1);          }          $this->table\_name = $sTableName;          $this->table\_definition = $aTableDefinition;          $this->setDefault();          $this->dbh = Database::connexion();      }      public function setDefault()      {          foreach ($this->table\_definition as $sNomChamp => $aChamp) {              $this->data[$sNomChamp] = $aChamp['default'];          }      }      public function \_\_get($sName)      {          if (! array\_key\_exists($sName, $this->table\_definition )) {              throw new Exception(\_\_CLASS\_\_.": undefined property $sName", 1);          }            return($this->data[$sName]);      }        public function \_\_set( $name, $value )      {          if ( ! array\_key\_exists($name, $this->table\_definition) ) {              throw new Exception(\_\_CLASS\_\_.": Le champ $name n'existe pas dans l'objet", 1);          }          if ( ! $this->validate( $name, $value ) ) {              throw new Exception(\_\_CLASS\_\_.": Erreur mise à jour champ $name avec $value. Valeur invalide", 1);          }          $this->data[$name] =  $value;      }      public function validate( $name, $value )      {  //        var\_dump($this->table\_definition);          $ValidFunction = $this->table\_definition[$name]['valid'];          $lValid = $ValidFunction($value);          return($lValid);      }      public function toArray()      {          return($this->data);      }      protected function cleanTableDefinition($aInput)      {          if ( ! is\_array($aInput) || count($aInput) == 0 ) {              throw new \Exception(\_\_CLASS\_\_.": table definition incorrecte", 1);          }          $aTableDefinition = array();          foreach ($aInput as $sChamp => $aParameters) {                if ( isset($aParameters['valid']) && is\_string($aParameters['valid']) && !empty($aParameters['valid']) ) {                  $aParameters['valid'] === $aParameters['valid'];              } else {                  $aParameters['valid'] = "Valid::alwaysTrue";              }              $aParameters['default'] = $aParameters['default'] ?? null;              if ( !isset($aParameters['pdo\_type']) ) {                  $aParameters['pdo\_type'] = PDO::PARAM\_STR;              }              if ( isset($aParameters['autoincrement']) && is\_bool($aParameters['autoincrement']) ) {                  $aParameters['autoincrement'] === $aParameters['autoincrement'];              } else {                  $aParameters['autoincrement'] = false;              }                if ( isset($aParameters['primary']) && is\_bool($aParameters['primary']) ) {                  $aParameters['primary'] === $aParameters['primary'];              } else {                  $aParameters['primary'] = false;              }              // Parametres normalisés              $aTableDefinition[$sChamp] = $aParameters;          }          return($aTableDefinition);      }  }  **Exemple Vue :**  <?php  function alert($sAlertColor, $sAlertText)  {      $sAlert = <<<'EOD'      <div class="row mt-3">      <div class="col"></div>      <div class="col-10 alert %s" role="alert">%s</div>      <div class="col"></div>    </div>  EOD;      $sReturn = sprintf( $sAlert, $sAlertColor, $sAlertText);      return($sReturn);  }  function navbar()  {      $sReturn1 = <<<'EOD'      <nav class="navbar navbar-expand-lg navbar-light bg-light my-2">      <div class="container-fluid">        <a class="navbar-brand" href="/">Blended</a>        <ul class="nav justify-content-end">  EOD;      $sMenuOptionTemplate = <<<'EOD'      <li class="nav-item %s">        <a class="nav-link" href="%s">%s</a>      </li>  EOD;      $oSession = Session::getInstance();      $sMenu = '';      if ( ! is\_null($oSession->menu) ) {          foreach ($oSession->menu as $aMenu) {              $sMenu .= sprintf(                  $sMenuOptionTemplate,                  $aMenu['active'] ? 'active': '',                  $aMenu['href'],                  $aMenu['text']                  ) . PHP\_EOL;          }      }      $sReturn2 = <<<'EOD'    </ul>  </div>  </nav>  EOD;      return($sReturn1 . $sMenu . $sReturn2);  }  function startBody()  {      $sReturn = <<<'EOD'      <body>      <div class="container-lg">  EOD;      return($sReturn);  }  function endBody()  {      $sReturn = <<<'EOD'      </div>      <!-- Option 1: Bootstrap Bundle with Popper -->      <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta2/dist/js/bootstrap.bundle.min.js" integrity="sha384-b5kHyXgcpbZJO/tY9Ul7kGkf1S0CWuKcCD38l8YkeH8z8QjE0GmW1gYU5S9FOnJ0" crossorigin="anonymous"></script>    </body>  EOD;      return($sReturn);  }  function headHtml( $sTitle = '', $aSupplement = null )  {      $sReturn1 = <<<'EOD'      <head>      <!-- Required meta tags -->      <meta charset="utf-8">      <meta name="viewport" content="width=device-width, initial-scale=1">      <!-- Bootstrap CSS -->      <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-BmbxuPwQa2lc/FVzBcNJ7UAyJxM6wuqIj61tLrc4wSX0szH/Ev+nYRRuWlolflfl" crossorigin="anonymous">  EOD;      $sTitleHtml = sprintf(          "    <title>Blended%s</title>",          empty($sTitle) ? '': ' - ' . $sTitle      );      $sSupplementHtml = '';      if ( ! is\_null($aSupplement) ) {          foreach ($aSupplement as $sLigne) {              $sSupplementHtml = $sSupplementHtml . $sLigne . "\n";          }      }  $sReturn2 = <<<'EOD'      </head>  EOD;      return( $sReturn1 . $sTitleHtml . $sSupplementHtml . $sReturn2 );  }  function startHtml()  {      $sReturn = <<<'EOD'  <!doctype html>  <html lang="en">  EOD;      return($sReturn);  }  function endHtml()  {      $sReturn = <<<'EOD'  </html>  EOD;      return($sReturn);  }  **Exemple Controller :**  <?php  function runController( $aParams )  {      $sAction = $aParams[0] ?? 'login';      $oSession = Session::getInstance();      switch ($sAction) {          case 'active':          case 'inactive':              if (                  $oSession->autorisation('admin') &&                  isset($aParams[1]) &&                  intval($aParams[1]) != 0                  ) {                  $oUser = new UtilisateursModel();                  $nUsrId = intval($aParams[1]);                  if ( $oUser->readWhere( [ 'utilisateur\_id' => $nUsrId ] ) ) {                      $oUser->usr\_active = ($sAction=='active')? "active": "inactive";                      if ( $sAction == 'active' ) {                          $oUser->usr\_active = "active";                          if ( $oUser->usr\_type == 'guest' ) {                              $oUser->usr\_type = 'user';                          }                      } else {                          $oUser->usr\_active = "inactive";                      }                      if ( $oUser->updateWhere(['usr\_active', 'usr\_type'], [ 'utilisateur\_id' => intval($aParams[1]) ] ) ) {                          Session::alertSuccess('Mise à jour effectuée');                      } else {                          Session::alertError('Mise à jour échouée');                      }                  }              }              $sTabName = Session::getSession( "LIST\_TABNAME" );              setExitRedirect('/user/list/'.$sTabName);              break;          case 'admin':          case 'user':              if (                  $oSession->autorisation('admin') &&                  isset($aParams[1]) &&                  intval($aParams[1]) != 0                  ) {                  $oUser = new UtilisateursModel();                  $oUser->usr\_type = $sAction;                    if ( $oUser->updateWhere(['usr\_type'], [ 'utilisateur\_id' => intval($aParams[1]) ] ) ) {                      Session::alertSuccess('Type utilisateur mis à jour');                  } else {                      Session::alertError('Mise à jour échouée');                  }              }              $sTabName = Session::getSession( "LIST\_TABNAME" );              setExitRedirect('/user/list/'.$sTabName);              break;          case 'delete':              if (                  $oSession->autorisation('admin') &&                  isset($aParams[1]) &&                  intval($aParams[1]) != 0                  ) {                  $oUser = new UtilisateursModel();                  if ( $oUser->deleteWhere([ 'utilisateur\_id' => intval($aParams[1]) ]) ) {                      Session::alertSuccess( 'Suppression effectuée' );                  } else {                      Session::alertError( 'Suppression échouée' );                  }              }              $sTabName = Session::getSession( "LIST\_TABNAME" );              setExitRedirect('/user/list/'.$sTabName);              break;          case 'pwd':              if (                  $oSession->autorisation('admin') &&                  isset($aParams[1]) &&                  intval($aParams[1]) != 0                  ) {                  changePassword( intval($aParams[1]) );              }              $sTabName = Session::getSession( "LIST\_TABNAME" );              setExitRedirect('/user/list/'.$sTabName);              break;          case 'list':              $sTabName = $aParams[1] ?? 'inactive';              if ( ! in\_array($sTabName, ['admin', 'user', 'inactive']) ) {                  $sTabName == 'inactive';              }              Session::setSession( "LIST\_TABNAME", $sTabName );              $oUser = new UtilisateursModel();              if ($sTabName == 'admin') {                  $aUtilisateurs = $oUser->indexWhere( null, [                      'usr\_active' => 'active',                      'usr\_type' => 'admin'                      ] );              } elseif ($sTabName == 'user') {                  $aUtilisateurs = $oUser->indexWhere( null, [                      'usr\_active' => 'active',                      'usr\_type' => 'user'                      ] );              } else {                  $aUtilisateurs = $oUser->indexWhere( null, [ 'usr\_active' => 'inactive' ] );              }              // Charge et affiche la vue              loadView( 'users' );              setExitView( usersView($aUtilisateurs) );              break;          case 'profile':              if ( $oSession->autorisation('admin') ) {                  if ( isset($aParams[1]) && intval($aParams[1]) != 0  ) {                      $nUsrId = intval($aParams[1]);                  } else {                      $oSession = Session::getInstance();                      $nUsrId = $oSession->user\_data['utilisateur\_id'];                  }              } elseif ($oSession->autorisation('user')) {                  $oSession = Session::getInstance();                  if ( isset($\_POST['email']) && $\_POST['email'] !== $oSession->user\_data['usr\_email'] ) {                      // Un usr\_type = user ne peux modifier que son propre compte                      unset($\_POST['email']);                      unset($\_POST['password']);                      unset($\_POST['prenom']);                      unset($\_POST['nom']);                  }                  $nUsrId = $oSession->user\_data['utilisateur\_id'];              } else {                  setExitRedirect('/user/login');              }              profileAction($nUsrId);              break;          case 'register':              registerAction();              break;          case 'lostpassword':              lostpasswordAction();              break;          case 'login':              loginAction();              break;          case 'logout':              $oSession->logout();              setExitRedirect('/user/login');              break;            default:              loadView( 'http404' );              setExitView( http404View() );              break;      }  }  function lostpasswordAction()  {      if ( isset($\_POST['email']) ) {          $oUser = new UtilisateursModel();          $oUser->readWhere(['usr\_email' => $\_POST['email'] ]);          changePassword( $oUser->utilisateur\_id );          setExitRedirect('/user/login');      }      // Charge et affiche la vue      loadView( 'lostpassword' );      setExitView( lostpasswordView() );  }  function changePassword( $nUsrId )  {      $oUser = new UtilisateursModel();      $oUser->readWhere(['utilisateur\_id' => $nUsrId]);      $sNewPassword = $oUser->setRandomPassword();      if ( $oUser->updateWhere(['usr\_pwd'], ['utilisateur\_id' => $nUsrId] ) ) {          $oMessage = new MessagesModel();          $oMessage->$email\_to = $oUser->$nUsrId;          $oMessage->$sujet = 'Nouveau mot de passe';          $oMessage->$message = "Nouveau mot de passe " . $sNewPassword;          $oMessage->create();          Session::alertSuccess('Nouveau mot de passe transmis par email');      } else {          Session::alertError('Mise à jour échouée');      }  }  function profileAction($nUsrId)  {      $oUser = new UtilisateursModel();      if (          isset($\_POST['email']) &&          isset($\_POST['password']) &&          isset($\_POST['prenom']) &&          isset($\_POST['nom'])          ) {          if ( $oUser->readWhere(['usr\_email' => $\_POST['email'] ]) ) {              $sPasswordHash = $\_POST['password'];              $oUser->usr\_pwd = $sPasswordHash;              $oUser->usr\_prenom = htmlspecialchars($\_POST['prenom']);              $oUser->usr\_nom = htmlspecialchars($\_POST['nom']);              $oUser->updateWhere(                  [ 'usr\_pwd', 'usr\_prenom', 'usr\_nom' ],                  [ 'utilisateur\_id' => $oUser->utilisateur\_id ]                  );              Session::alertSuccess('Mise à jour effectuée');          } else {              Session::alertError('Mise à jour échouée');          }      }      $oUser->readWhere(['utilisateur\_id' => $nUsrId ]);      $aUtilisateur = $oUser->toArray();      // Charge et affiche la vue      loadView( 'profile' );      setExitView( profileView($aUtilisateur) );  }  function registerAction()  {      if (          isset($\_POST['email']) &&          isset($\_POST['password']) &&          isset($\_POST['prenom']) &&          isset($\_POST['nom'])          ) {          $oUser = new UtilisateursModel();          $oUser->usr\_email = htmlspecialchars($\_POST['email']);          $oUser->usr\_nom = htmlspecialchars($\_POST['nom']);          $oUser->usr\_prenom = htmlspecialchars($\_POST['prenom']);          $oUser->usr\_pwd = htmlspecialchars($\_POST['password']);          $nId = $oUser->create();            if ($nId !=0 ) {              Session::alertSuccess('Utilisateur inscrit');          } else {              Session::alertError('Inscription échouée');          }          setExitRedirect('/user/login');      }      loadView( 'register' );      setExitView( registerView() );  }  function loginAction()  {      $nAuth = Session::AUTH\_GUEST;      if ( isset($\_POST['email']) && isset($\_POST['password']) ) {          // Verification authentification          $oSession = Session::getInstance();          $nAuth = $oSession->checkAuth($\_POST['email'], $\_POST['password']);          switch ($nAuth) {              case Session::AUTH\_LOGIN:                  setExitRedirect('/');                  break;              case Session::AUTH\_INACTIVE:                  http\_response\_code(401);                  Session::alertError('Compte inactif');                  break;              case Session::AUTH\_FAILED:              default:                  http\_response\_code(401);                  Session::alertError('HTTP 401 Unauthorized - Accès restreint');                  break;          }      }      if ( $nAuth !== Session::AUTH\_LOGIN ) {          // Charge et affiche la vue          loadView( 'login' );          setExitView( loginView() );      }  } | | | | |
|  | | | | |
|  | | | | |
| **2. Précisez les moyens utilisés :** | | | | |
|  | | | | |
| Looping,  MySQL  PhpMyAdmin  VsCode | | | | |
|  | | | | |
|  | | | | |
|  | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | |
|  | | | | |
| Mr Fabrice GUICHARD, formateur GRETA, ainsi que Christian AUBERT, étudiant comme moi | | | | |
|  | | | | |
|  | | | | |
|  | | | | |
| **4. Contexte** | | | | |
|  | | | |  |
| Nom de l’entreprise, organisme ou association | | | | *GRETA* |
| Chantier, atelier, service | | | *Formation DWWM* | |
| Période d’exerciceDu : *02/11/20*au :*21/07/21* | | | | |
|  | | | | |
|  | | | | |

|  |
| --- |
| **Déclaration sur l’honneur** |
|  |
|  |

Je soussigné(e) [prénom et nom] Eric BOSSU ,

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 St Etienne le 01/06/2021

pour faire valoir ce que de droit.

Signature :