```
{
    "_readme": [
        "This file locks the dependencies of your project to a known state",
        "Read more about it at https://getcomposer.org/doc/01-basic-
usage.md#installing-dependencies",
        "This file is @generated automatically"
    ],
    "content-hash": "85e07ea1e3620c3da1942492052d7d21",
    "packages": [
        {
            "name": "phpmailer/phpmailer",
            "version": "v6.1.5",
            "source": {
                "type": "git",
                "url": "https://github.com/PHPMailer/PHPMailer.git",
                "reference": "a8bf068f64a580302026e484ee29511f661b2ad3"
            },
            "dist": {
                "type": "zip",
                "url": "https://api.github.com/repos/PHPMailer/PHPMailer/zipball
/a8bf068f64a580302026e484ee29511f661b2ad3",
                "reference": "a8bf068f64a580302026e484ee29511f661b2ad3",
                "shasum": ""
            },
            "require": {
                "ext-ctype": "*",
                "ext-filter": "*",
                "php": ">=5.5.0"
            },
            "require-dev": {
                "doctrine/annotations": "^1.2",
                "friendsofphp/php-cs-fixer": "^2.2",
                "phpunit/phpunit": "^4.8 || ^5.7"
            },
            "suggest": {
                "ext-mbstring": "Needed to send email in multibyte encoding charset",
                "hayageek/oauth2-yahoo": "Needed for Yahoo XOAUTH2 authentication",
                "league/oauth2-google": "Needed for Google XOAUTH2 authentication",
                "psr/log": "For optional PSR-3 debug logging",
                "stevenmaguire/oauth2-microsoft": "Needed for Microsoft XOAUTH2
authentication",
                "symfony/polyfill-mbstring": "To support UTF-8 if the Mbstring PHP
extension is not enabled (^1.2)"
            },
            "type": "library",
            "autoload": {
                "psr-4": {
                    "PHPMailer\\PHPMailer\\": "src/"
                }
            },
            "notification-url": "https://packagist.org/downloads/",
            "license": [
                "LGPL-2.1-only"
            ],
```

```
            "authors": [
                {
                    "name": "Marcus Bointon",
                    "email": "phpmailer@synchromedia.co.uk"
                },
                {
                    "name": "Jim Jagielski",
                    "email": "jimjag@gmail.com"
                },
                {
                    "name": "Andy Prevost",
                    "email": "codeworxtech@users.sourceforge.net"
                },
                {
                    "name": "Brent R. Matzelle"
                }
            ],
            "description": "PHPMailer is a full-featured email creation and transfer class for PHP",
            "funding": [
                {
                    "url": "https://marcus.bointon.com/donations/",
                    "type": "custom"
                },
                {
                    "url": "https://github.com/Synchro",
                    "type": "github"
                },
                {
                    "url": "https://www.patreon.com/marcusbointon",
                    "type": "patreon"
                }
            ],
            "time": "2020-03-14T14:23:48+00:00"
        }
    ],
    "packages-dev": [],
    "aliases": [],
    "minimum-stability": "stable",
    "stability-flags": [],
    "prefer-stable": false,
    "prefer-lowest": false,
    "platform": [],
    "platform-dev": [],
    "plugin-api-version": "1.1.0"
}
```

```
{
    "require": {
        "phpmailer/phpmailer": "^6.1"
    }
}
```

```
.gitignore
.vscode
/vendor/
/database/schema.mwb.bak
/views/assets/upload/
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : register.php
 * Date         : 26.05.2020
 * Description  : Formulaire d'inscription
 * Version      : 1.0
 */

require_once dirname(__DIR__) . '/App/php/backend.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}

if (IsLogged()) {
  header('Location: ./index.php');
  exit();
}
?>

<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <!-- STYLE DU SITE -->
  <link rel="stylesheet" href="./css/style.css">
  <link rel="stylesheet" href="./css/media.css">

  <!-- SWEETALERT2 -->
  <link rel="stylesheet" href="./css/sweetalert2.css">

  <!-- GOOGLE FONTS -->
  <link href="https://fonts.googleapis.com/css?family=Poppins:600&display=swap"
rel="stylesheet">

  <!-- FONTAWESOME -->
  <script src="https://kit.fontawesome.com/a81368914c.js"></script>

  <link rel="shortcut icon" href="./assets/img/iconeWego.svg" type="image/x-icon">

  <title>Inscription</title>
</head>

<body>
  <?php include './includes/navbar.inc.php'; ?>

  <img class="wave" src="./assets/img/wave.png" alt="wave">
  <div class="container">
    <div class="img">
```

```
      <img src="./assets/img/fill_forms.svg" alt="background">
    </div>
    <div class="content">
      <form method="post">
        <img src="./assets/img/avatar.svg" alt="avatar">
        <h2 class="title">Inscription</h2>
        <div class="input-div one">
          <div class="i">
            <i class="fas fa-user"></i>
          </div>
          <div class="div">
            <h5>Nom</h5>
            <input type="text" class="input" id="lastname" maxlength="50">
          </div>
        </div>
        <div class="input-div one">
          <div class="i">
            <i class="fas fa-user"></i>
          </div>
          <div class="div">
            <h5>Prénom</h5>
            <input type="text" class="input" id="firstname" maxlength="50">
          </div>
        </div>
        <div class="input-div one">
          <div class="i">
            <i class="fas fa-user"></i>
          </div>
          <div class="div">
            <h5>Nom d'utilisateur</h5>
            <input type="text" class="input" id="nickname" maxlength="50">
          </div>
        </div>
        <div class="input-div one">
          <div class="i">
            <i class="fa fa-envelope" aria-hidden="true"></i>
          </div>
          <div class="div">
            <h5>Email</h5>
            <input type="email" class="input" id="email" maxlength="100">
          </div>
        </div>
        <div class="input-div one">
          <div class="i">
            <i class="fas fa-phone-alt"></i>
          </div>
          <div class="div">
            <h5>Téléphone</h5>
            <input type="phone" class="input" id="phoneNumber" maxlength="50">
          </div>
        </div>
        <div class="input-div pass">
          <div class="i">
            <i class="fas fa-lock"></i>
          </div>
```

```
          <div class="div">
            <h5>Mot de passe</h5>
            <input type="password" class="input" id="password">
          </div>
        </div>
        <div class="input-div pass">
          <div class="i">
            <i class="fas fa-lock"></i>
          </div>
          <div class="div">
            <h5>Confirmation mot de passe</h5>
            <input type="password" class="input" id="verifyPassword">
          </div>
        </div>
        <input type="submit" id="btnRegister" class="btn" value="Inscription">
      </form>
    </div>
  </div>

  <!-- JAVASCRIPT INCLUDES -->
  <?php include "./includes/footer.inc.html"; ?>

  <script src="./js/register.js"></script>
</body>

</html>
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : profile.php
 * Date         : JJ.03.2020
 * Description  : Page qui affiche els donnéees du profile ainsi que son formulaire de
modification.
 * Version      : 1.0.0
 */

require_once dirname(__DIR__) . '/App/php/backend.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}
if (IsLogged() == false) {
  header('Location: ./login.php');
  exit();
}
?>
<!DOCTYPE html>
<html lang="en">

<head>
<meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <!-- STYLE DU SITE -->
  <link rel="stylesheet" href="./css/style.css">
  <link rel="stylesheet" href="./css/media.css">

  <!-- SWEETALERT2 -->
  <link rel="stylesheet" href="./css/sweetalert2.css">

  <!-- GOOGLE FONTS -->
  <link href="https://fonts.googleapis.com/css?family=Poppins:600&display=swap"
rel="stylesheet">

  <!-- FONTAWESOME -->
  <script src="https://kit.fontawesome.com/a81368914c.js"></script>

  <link rel="shortcut icon" href="./assets/img/iconeWego.svg" type="image/x-icon">
  <title>Profile</title>
</head>

<body>
  <?php include './includes/navbar.inc.php' ?>

  <img class="wave" src="./assets/img/wave.png" alt="wave">
  <div class="container">
    <div class="img">
      <img src="./assets/img/profile_data.svg">
    </div>
```

```
53
54      <div class="content" id="userData"></div>
55    </div>
56
57    <?php include './includes/footer.inc.html' ?>
58
59    <script src="./js/profile.js"></script>
60  </body>
61
62  </html>
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : manageEvents.php
 * Date         : 03.06.2020
 * Description  : Page qui permet de gérer les événements.
 * Version      : 1.0
 */

require_once dirname(__DIR__) . '/App/php/backend.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}

if (IsLogged() == false) {
  header('Location: ./login.php');
  exit();
}
?>

<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <!-- STYLE DU SITE -->
  <link rel="stylesheet" href="./css/style.css">
  <link rel="stylesheet" href="./css/media.css">

  <!-- SWEETALERT2 -->
  <link rel="stylesheet" href="./css/sweetalert2.css">

  <!-- GOOGLE FONTS -->
  <link href="https://fonts.googleapis.com/css?family=Poppins:600&display=swap"
rel="stylesheet">

  <!-- FONTAWESOME -->
  <script src="https://kit.fontawesome.com/a81368914c.js"></script>

  <title>Gérer mes événements</title>
</head>

<body>
  <?php include './includes/navbar.inc.php'; ?>

  <div id="events"></div>

  <?php include './includes/footer.inc.html'; ?>

  <script src="./js/events.js"></script>
```

```
54 </body>
55
56 </html>
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : login.php
 * Date         : 26.05.2020
 * Description  : Formulaire de connexion
 * Version      : 1.0
 */

require_once dirname(__DIR__) . '/App/php/backend.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}

if (IsLogged()) {
  header('Location: ./index.php');
  exit();
}
?>
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <!-- STYLE DU SITE -->
  <link rel="stylesheet" href="./css/style.css">
  <link rel="stylesheet" href="./css/media.css">

  <!-- SWEETALERT2 -->
  <link rel="stylesheet" href="./css/sweetalert2.css">

  <!-- GOOGLE FONTS -->
  <link href="https://fonts.googleapis.com/css?family=Poppins:600&display=swap"
rel="stylesheet">

  <!-- FONTAWESOME -->
  <script src="https://kit.fontawesome.com/a81368914c.js"></script>

  <link rel="shortcut icon" href="./assets/img/iconeWego.svg" type="image/x-icon">

  <title>Connexion</title>
</head>

<body>
  <?php include './includes/navbar.inc.php'; ?>

  <img class="wave" src="./assets/img/wave.png" alt="wave">
  <div class="container">
    <div class="img">
      <img src="./assets/img/bg.svg" alt="background">
```

```
    </div>
    <div class="content">
      <form method="post">
        <img src="./assets/img/avatar.svg" alt="avatar">
        <h2 class="title">Bienvenue</h2>
        <div class="input-div one">
          <div class="i">
            <i class="fas fa-user"></i>
          </div>
          <div class="div">
            <h5>Nom d'utilisateur ou Email</h5>
            <input type="text" class="input" id="authenticator">
          </div>
        </div>
        <div class="input-div pass">
          <div class="i">
            <i class="fas fa-lock"></i>
          </div>
          <div class="div">
            <h5>Mot de passe</h5>
            <input type="password" class="input" id="password">
          </div>
        </div>
        <input type="submit" id="btnLogUser" class="btn" value="Connexion">
      </form>
    </div>
  </div>

  <?php include './includes/footer.inc.html'; ?>

  <script type="text/javascript" src="./js/login.js"></script>
</body>

</html>
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : index.php
 * Date         : 26.05.2020
 * Description  : Page d'accueil du site.
 * Version      : 1.0
 */

require_once dirname(__DIR__) . '/App/php/backend.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}
?>

<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <!-- STYLE DU SITE -->
  <link rel="stylesheet" href="./css/style.css">
  <link rel="stylesheet" href="./css/media.css">

  <!-- SWEETALERT2 -->
  <link rel="stylesheet" href="./css/sweetalert2.css">

  <!-- GOOGLE FONTS -->
  <link href="https://fonts.googleapis.com/css?family=Poppins:600&display=swap"
rel="stylesheet">

  <!-- FONTAWESOME -->
  <script src="https://kit.fontawesome.com/a81368914c.js"></script>

  <link rel="shortcut icon" href="./assets/img/iconeWego.svg" type="image/x-icon">

  <title>Accueil</title>
</head>

<body>
  <?php include './includes/navbar.inc.php'; ?>

  <div id="events"></div>

  <?php include './includes/footer.inc.html'; ?>

  <script src="./js/events.js"></script>
</body>

</html>
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : event.php
 * Date         : 02.06.2020
 * Description  : Page qui affiche les informations d'un événement spécifique.
 * Version      : 1.0
 */

require_once dirname(__DIR__) . '/App/php/backend.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}
?>

<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <!-- STYLE DU SITE -->
  <link rel="stylesheet" href="./css/style.css">
  <link rel="stylesheet" href="./css/media.css">

  <!-- SWEETALERT2 -->
  <link rel="stylesheet" href="./css/sweetalert2.css">

  <!-- GOOGLE FONTS -->
  <link href="https://fonts.googleapis.com/css?family=Poppins:600&display=swap"
rel="stylesheet">

  <!-- FONTAWESOME -->
  <script src="https://kit.fontawesome.com/a81368914c.js"></script>

  <link rel="shortcut icon" href="./assets/img/iconeWego.svg" type="image/x-icon">

  <title></title>
</head>

<body>
  <?php include './includes/navbar.inc.php'; ?>

  <img class="wave" src="./assets/img/wave.png" alt="wave">
  <div class="container">
    <div class="img">
      <img src="./assets/img/event_data.svg" alt="background">
    </div>

    <div class="content" id="eventData"></div>
  </div>
```

```
54
55    <?php include './includes/footer.inc.html'; ?>
56
57    <script src="./js/events.js"></script>
58  </body>
59
60  </html>
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : editProfile.php
 * Date         : 05.06.2020
 * Description  : Formulaire de modification du profile
 * Version      : 1.0
 */
require_once dirname(__DIR__) . '/App/php/backend.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}

$idUser = filter_input(INPUT_GET, 'idUser', FILTER_SANITIZE_NUMBER_INT);

if (!IsLogged() || $idUser != $_SESSION['loggedIn']['idUser']) {
  header('Location: ./index.php');
  exit();
}
?>
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <!-- STYLE DU SITE -->
  <link rel="stylesheet" href="./css/style.css">
  <link rel="stylesheet" href="./css/media.css">

  <!-- SWEETALERT2 -->
  <link rel="stylesheet" href="./css/sweetalert2.css">

  <!-- GOOGLE FONTS -->
  <link href="https://fonts.googleapis.com/css?family=Poppins:600&display=swap"
rel="stylesheet">

  <!-- FONTAWESOME -->
  <script src="https://kit.fontawesome.com/a81368914c.js"></script>

  <link rel="shortcut icon" href="./assets/img/iconeWego.svg" type="image/x-icon">

  <title>Modifier mon profile</title>
</head>

<body>
  <?php include './includes/navbar.inc.php'; ?>

  <img class="wave" src="./assets/img/wave.png" alt="wave">
  <div class="container">
    <div class="img">
```

```
      <img src="./assets/img/fill_forms.svg" alt="background">
    </div>
    <div class="content">
      <form method="post">
        <img src="./assets/img/avatar.svg" alt="avatar">
        <h2 class="title">Modifier mon profile</h2>
        <div class="input-div one">
          <div class="i">
            <i class="fas fa-user"></i>
          </div>
          <div class="div">
            <h5>Nom</h5>
            <input type="text" class="input" id="lastname" maxlength="50" value="<?=
isset($_SESSION['loggedIn']['nom']) ? $_SESSION['loggedIn']['nom'] : '' ?>">
          </div>
        </div>
        <div class="input-div one">
          <div class="i">
            <i class="fas fa-user"></i>
          </div>
          <div class="div">
            <h5>Prénom</h5>
            <input type="text" class="input" id="firstname" maxlength="50" value="<?=
$_SESSION['loggedIn']['prenom'] ?>">
          </div>
        </div>
        <div class="input-div one">
          <div class="i">
            <i class="fas fa-user"></i>
          </div>
          <div class="div">
            <h5>Nom d'utilisateur</h5>
            <input type="text" class="input" id="nickname" maxlength="50" value="<?=
$_SESSION['loggedIn']['pseudo'] ?>">
          </div>
        </div>
        <div class="input-div one">
          <div class="i">
            <i class="fa fa-envelope" aria-hidden="true"></i>
          </div>
          <div class="div">
            <h5>Email</h5>
            <input type="email" class="input" id="email" maxlength="100" value="<?=
$_SESSION['loggedIn']['email'] ?>">
          </div>
        </div>
        <div class="input-div one">
          <div class="i">
            <i class="fas fa-phone-alt"></i>
          </div>
          <div class="div">
            <h5>Téléphone</h5>
            <input type="phone" class="input" id="phoneNumber" maxlength="50"
value="<?= isset($_SESSION['loggedIn']['telephone']) ? $_SESSION['loggedIn']
['telephone'] : '' ?>">
```

```
103           </div>
104         </div>
105         <div class="input-div pass">
106           <div class="i">
107             <i class="fas fa-lock"></i>
108           </div>
109           <div class="div">
110             <h5>Mot de passe</h5>
111             <input type="password" class="input" id="password">
112           </div>
113         </div>
114         <div class="input-div pass">
115           <div class="i">
116             <i class="fas fa-lock"></i>
117           </div>
118           <div class="div">
119             <h5>Confirmation mot de passe</h5>
120             <input type="password" class="input" id="verifyPassword">
121           </div>
122         </div>
123         <input type="submit" id="btnRegister" class="btn" onclick="EditProfile(event)"
    value="Modifier mon profile">
124       </form>
125     </div>
126   </div>
127
128   <!-- JAVASCRIPT INCLUDES -->
129   <?php include "./includes/footer.inc.html"; ?>
130
131   <script src="./js/profile.js"></script>
132 </body>
133
134 </html>
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : editEvent.php
 * Date         : 03.06.2020
 * Description  : Formulaire de modification d'un événement
 * Version      : 1.0
 */

require_once dirname(__DIR__) . '/App/php/backend.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}

if (!IsLogged()) {
  header('Location: ./index.php');
  exit();
}

$idEvent = filter_input(INPUT_GET, 'idEvent', FILTER_SANITIZE_NUMBER_INT);
$idUser = $_SESSION['loggedIn']['idUser'];

if (IsCreator($idEvent, $idUser) == false) {
  header('Location: ./index.php');
  exit();
}

$eventData = GetEventData($idEvent, $idUser);

// dateDebut au format Y-m-d H:i:s
$eventBeginingDate = explode(' ', $eventData['dateDebut'])[0]; // Récupère la date au
format Y-m-d
$eventBeginingTime = explode(' ', $eventData['dateDebut'])[1]; // Récupère l'heure au
foormat H:m:s
$eventBeginingTime = explode(':', $eventBeginingTime)[0] . ':' . explode(':',
$eventBeginingTime)[1]; // Passe l'heure du format H:m:s au format H:m

// dateFin au format Y-m-d H:i:s
$eventEndDate = explode(' ', $eventData['dateFin'])[0]; // Récupère la date au format
Y-m-d
$eventEndTime = explode(' ', $eventData['dateFin'])[1]; // Récupère l'heure au foormat
H:m:s
$eventEndTime = explode(':', $eventEndTime)[0] . ':' . explode(':', $eventEndTime)[1];
// Passe l'heure du format H:m:s au format H:m

?>

<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
```

```
  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <!-- STYLE DU SITE -->
  <link rel="stylesheet" href="./css/style.css">
  <link rel="stylesheet" href="./css/media.css">

  <!-- SWEETALERT2 -->
  <link rel="stylesheet" href="./css/sweetalert2.css">

  <!-- GOOGLE FONTS -->
  <link href="https://fonts.googleapis.com/css?family=Poppins:600&display=swap" rel="stylesheet">

  <!-- FONTAWESOME -->
  <script src="https://kit.fontawesome.com/a81368914c.js"></script>

  <link rel="shortcut icon" href="./assets/img/iconeWego.svg" type="image/x-icon">

  <title>Modifier mon événement</title>
</head>

<body>
  <?php include './includes/navbar.inc.php' ?>

  <img class="wave" src="./assets/img/wave.png" alt="wave">
  <div class="container">
    <div class="img">
      <img src="./assets/img/events.svg" alt="background">
    </div>

    <div class="content">
      <form method="post">
        <img src="./assets/img/createEventsLogo.svg" alt="avatar">
        <h2 class="title">Modifier mon événement</h2>
        <div class="input-div one">
          <div class="i">
            <i class="fa fa-calendar" aria-hidden="true"></i>
          </div>
          <div class="div">
            <h5>Nom de l'événement</h5>
            <input type="text" class="input" id="eventName" maxlength="50" value="<?= $eventData['nom'] ?>">
          </div>
        </div>
        <div class="input-div one">
          <div class="i">
            <i class="fas fa-pen"></i>
          </div>
          <div class="div">
            <h5>Descriptif de l'événement</h5>
            <input type="text" class="input" id="eventDescription" maxlength="255" value="<?= $eventData['descriptif'] ?>">
          </div>
        </div>
        <div class="input-div one">
```

```
101           <div class="i">
102             <i class="fas fa-map-marker-alt"></i>
103           </div>
104           <div class="div">
105             <h5>Lieu</h5>
106             <input type="text" class="input" id="place" maxlength="255" value="<?=
    $eventData['lieu'] ?>">
107           </div>
108         </div>
109         <div class="input-div one">
110           <div class="i">
111             <i class="fas fa-calendar-alt"></i>
112           </div>
113           <div class="div">
114             <h5>Date de début</h5>
115             <input class="input" id="beginningDate" type="text" onfocus="
    (this.type='date')" value="<?= $eventBeginingDate ?>" />
116           </div>
117         </div>
118         <div class="input-div one">
119           <div class="i">
120             <i class="fas fa-clock"></i>
121           </div>
122           <div class="div">
123             <h5>Heure de début</h5>
124             <input class="input" id="beginningTime" type="text" onfocus="
    (this.type='time')" value="<?= $eventBeginingTime ?>" />
125           </div>
126         </div>
127         <div class="input-div one">
128           <div class="i">
129             <i class="fas fa-calendar-alt"></i>
130           </div>
131           <div class="div">
132             <h5>Date de fin</h5>
133             <input class="input" id="endDate" type="text" onfocus="(this.type='date')"
    value="<?= $eventEndDate ?>" />
134           </div>
135         </div>
136         <div class="input-div one">
137           <div class="i">
138             <i class="fas fa-clock"></i>
139           </div>
140           <div class="div">
141             <h5>Heure de fin</h5>
142             <input class="input" id="endTime" type="text" onfocus="(this.type='time')"
    value="<?= $eventEndTime ?>" />
143           </div>
144         </div>
145         <div class="input-div one">
146           <div class="i">
147             <i class="fas fa-user"></i>
148           </div>
149           <div class="div">
150             <h5>Nombre maximal de participant</h5>
```

```
151             <input class="input" id="nbMaxGuest" type="number" min="0" value="<?=
    $eventData['nbMaxParticipant'] ?>" />
152           </div>
153         </div>
154         <div class="input-div one">
155           <div class="i">
156             <i class="fas fa-images"></i>
157           </div>
158           <div class="div">
159             <h5><label for="img">Image de l'événement</label></h5>
160             <input class="input" name="img" id="img" type="file" accept="image/*" />
161           </div>
162         </div>
163         <input type="submit" id="btnEditEvent" onclick="EditEvent(event, <?= $idEvent
    ?>)" class="btn" value="Modifier l'événement">
164       </form>
165     </div>
166
167     <?php include './includes/footer.inc.html' ?>
168     <script src="./js/events.js"></script>
169 </body>
170
171 </html>
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : createEvent.php
 * Date         : 27.05.2020
 * Description  : Formulaire de création d'événement.
 * Version      : 1.0
 */

require_once dirname(__DIR__) . '/App/php/backend.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}

if (!IsLogged()) {
  header('Location: ./index.php');
  exit();
}

$datetime = new DateTime('tomorrow');
?>

<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <!-- STYLE DU SITE -->
  <link rel="stylesheet" href="./css/style.css">
  <link rel="stylesheet" href="./css/media.css">

  <!-- SWEETALERT2 -->
  <link rel="stylesheet" href="./css/sweetalert2.css">

  <!-- GOOGLE FONTS -->
  <link href="https://fonts.googleapis.com/css?family=Poppins:600&display=swap"
rel="stylesheet">

  <!-- FONTAWESOME -->
  <script src="https://kit.fontawesome.com/a81368914c.js"></script>

  <link rel="shortcut icon" href="./assets/img/iconeWego.svg" type="image/x-icon">

  <title>Créer un événement</title>
</head>

<body>
  <?php include './includes/navbar.inc.php' ?>

  <img class="wave" src="./assets/img/wave.png" alt="wave">
```

```
  <div class="container">
    <div class="img">
      <img src="./assets/img/events.svg" alt="background">
    </div>

    <div class="content">
      <form method="post">
        <img src="./assets/img/createEventsLogo.svg" alt="avatar">
        <h2 class="title">Créer un événement</h2>
        <div class="input-div one">
          <div class="i">
            <i class="fa fa-calendar" aria-hidden="true"></i>
          </div>
          <div class="div">
            <h5>Nom de l'événement</h5>
            <input type="text" class="input" id="eventName" maxlength="50">
          </div>
        </div>
        <div class="input-div one">
          <div class="i">
            <i class="fas fa-pen"></i>
          </div>
          <div class="div">
            <h5>Descriptif de l'événement</h5>
            <input type="text" class="input" id="eventDescription" maxlength="255">
          </div>
        </div>
        <div class="input-div one">
          <div class="i">
            <i class="fas fa-map-marker-alt"></i>
          </div>
          <div class="div">
            <h5>Lieu</h5>
            <input type="text" class="input" id="place" maxlength="255">
          </div>
        </div>
        <div class="input-div one">
          <div class="i">
            <i class="fas fa-calendar-alt"></i>
          </div>
          <div class="div">
            <h5>Date de début</h5>
            <input class="input" id="beginningDate" type="text" onfocus="
(this.type='date')" value="<?= $datetime->format('Y-m-d'); ?>" />
          </div>
        </div>
        <div class="input-div one">
          <div class="i">
            <i class="fas fa-clock"></i>
          </div>
          <div class="div">
            <h5>Heure de début</h5>
            <input class="input" id="beginningTime" type="text" onfocus="
(this.type='time')" value="<?= date('H:i'); ?>" />
          </div>
```

```
107         </div>
108         <div class="input-div one">
109           <div class="i">
110             <i class="fas fa-calendar-alt"></i>
111           </div>
112           <div class="div">
113             <h5>Date de fin</h5>
114             <input class="input" id="endDate" type="text" onfocus="(this.type='date')"
    />
115           </div>
116         </div>
117         <div class="input-div one">
118           <div class="i">
119             <i class="fas fa-clock"></i>
120           </div>
121           <div class="div">
122             <h5>Heure de fin</h5>
123             <input class="input" id="endTime" type="text" onfocus="(this.type='time')"
    />
124           </div>
125         </div>
126         <div class="input-div one">
127           <div class="i">
128             <i class="fas fa-user"></i>
129           </div>
130           <div class="div">
131             <h5>Nombre maximal de participant</h5>
132             <input class="input" id="nbMaxGuest" type="number" min="0" />
133           </div>
134         </div>
135         <div class="input-div one">
136           <div class="i">
137             <i class="fas fa-images"></i>
138           </div>
139           <div class="div">
140             <h5><label for="img">Image de l'événement</label></h5>
141             <input class="input" name="img" id="img" type="file" accept="image/*" />
142           </div>
143         </div>
144         <div class="radio-group">
145           <label class="radio">
146             <input type="radio" class="radio" name="type" id="public" value="public"
    checked>
147             publique
148             <span></span>
149           </label>
150           <label class="radio">
151             <input type="radio" class="radio" name="type" id="private" value="private"
152             privé
153             <span></span>
154           </label>
155         </div>
156         <input type="submit" id="btnCreateEvent" class="btn" value="Créer un
    événement">
157       </form>
```

```
158     </div>
159
160     <?php include './includes/footer.inc.html' ?>
161     <script src="./js/events.js"></script>
162 </body>
163
164 </html>
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : about.php
 * Date         : 08.06.2020
 * Description  : Page d'information sur le site
 * Version      : 1.0
 */

require_once dirname(__DIR__) . '/App/php/backend.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}
?>

<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <!-- STYLE DU SITE -->
  <link rel="stylesheet" href="./css/style.css">
  <link rel="stylesheet" href="./css/media.css">

  <!-- SWEETALERT2 -->
  <link rel="stylesheet" href="./css/sweetalert2.css">

  <!-- GOOGLE FONTS -->
  <link href="https://fonts.googleapis.com/css?family=Poppins:600&display=swap"
rel="stylesheet">

  <!-- FONTAWESOME -->
  <script src="https://kit.fontawesome.com/a81368914c.js"></script>

  <link rel="shortcut icon" href="./assets/img/iconeWego.svg" type="image/x-icon">

  <title>But du site</title>
</head>

<body>
  <?php include './includes/navbar.inc.php'; ?>

  <img class="wave" src="./assets/img/wave.png" alt="wave">
  <div class="container">
    <div class="img">
      <img src="./assets/img/about.svg" alt="background">
    </div>
    <div class="content">
      <div class="container-about">
        <div class="div about">Ce site permet de créer des événements privés ou
```

```
publiques afin que d'autres utilisateurs puissent y participer.</div>
54          <div class="div about">Si vous souhaitez en savoir plus sur le fonctionnement
    du site, le manuel utilisateur est disponible <a href="#">ici</a>.</div>
55
56        </div>
57      </div>
58    </div>
59
60    <?php include './includes/footer.inc.html'; ?>
61  </body>
62
63  </html>
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : 404.php
 * Date         : 02.06.2020
 * Description  : Page qui affiche l'erreur 404
 * Version      : 1.0
 */

require_once dirname(__DIR__) . '/App/php/backend.php';
?>
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <!-- STYLE DU SITE -->
  <link rel="stylesheet" href="./css/style.css">
  <link rel="stylesheet" href="./css/media.css">

  <!-- SWEETALERT2 -->
  <link rel="stylesheet" href="./css/sweetalert2.css">

  <!-- GOOGLE FONTS -->
  <link href="https://fonts.googleapis.com/css?family=Poppins:600&display=swap"
rel="stylesheet">

  <!-- FONTAWESOME -->
  <script src="https://kit.fontawesome.com/a81368914c.js"></script>

  <link rel="shortcut icon" href="./assets/img/iconeWego.svg" type="image/x-icon">

  <title>Page Not Found</title>
</head>

<body>
  <?php include './includes/navbar.inc.php'; ?>

  <img src="./assets/img/404.svg" alt="404">

</body>

</html>
```

```
$(document).ready(() => {
  $("#btnRegister").click(Register);
});

/**
 * @author Hoarau Nicolas
 * @date 26.05.20
 * @brief fonction qui filtre les données du formulaire d'inscription et les envois au serveur vie un call Ajax
 * @param {*} event
 * @version 1.0
 */
function Register(event) {
  if (event)
    event.preventDefault();

  let lastname = $('#lastname').val();
  let firstname = $('#firstname').val();
  let nickname = $('#nickname').val();
  let email = $('#email').val();
  let phoneNumber = $('#phoneNumber').val();
  let password = $('#password').val();
  let verifyPassword = $('#verifyPassword').val();

  if (firstname.length == 0) {
    $('#firstname').focus();
    return;
  }

  if (nickname.length == 0) {
    $('#nickname').focus();
    return;
  }

  if (email.length == 0) {
    $('#email').focus();
    return;
  }

  if (password.length == 0) {
    $('#password').focus();
    return;
  }

  if (verifyPassword.length == 0) {
    $('#verifyPassword').focus();
    return;
  }

  $.ajax({
    type: "post",
    url: "../App/php/register.php",
    data: {
      lastname: lastname,
```

```
      firstname: firstname,
      nickname: nickname,
      email: email,
      phoneNumber: phoneNumber,
      password: password,
      verifyPassword: verifyPassword
    },
    dataType: "json",
    success: (response) => {
      switch (response.ReturnCode) {
        case 0:
          Swal.fire({
            title: "Inscription",
            text: response.Success,
            icon: "success",
            timer: 1300,
            showConfirmButton: false
          }).then(() => {
            window.location.href = './login.php'
          });
          break;
        case 1:
        case 2:
        case 3:
        case 4:
        case 5:
          Swal.fire({
            title: "Inscription",
            text: response.Error,
            icon: "error",
            timer: 1500,
            showConfirmButton: false
          });
          break;
      }
    },
    error: (error) => {
      console.error(error);
    }
  });
```

```
$(document).ready(() => {
  let pageName = window.location.pathname.substring(
    location.pathname.lastIndexOf("/") + 1
  );

  switch (pageName) {
    case "editProfile.php":
      $("#lastname").focus();
      $("#firstname").focus();
      $("#nickname").focus();
      $("#email").focus();
      $("#phoneNumber").focus();
      break;
    case 'profile.php':
      GetProfileData();
      break;
  }
});

/**
 * @author Hoarau Nicolas
 * @date 03.06.20
 * @brief Fonction qui récupère les infrormations de l'utilisateur
 */
function GetProfileData() {
  $.ajax({
    type: "post",
    url: "../App/php/getUserData.php",
    dataType: "json",
    success: (response) => {
      switch (response.ReturnCode) {
        case 0:
          ShowProfileData(response.Data);
          break;
        case 1:
          Swal.fire({
            title: "Mon profil",
            text: response.Error,
            icon: "error",
            timer: 1300,
            showConfirmButton: false
          });
          break;
      }
    },
    error: (error) => {
      console.error(error);
    }
  });
}

/**
 * @author Hoarau Nicolas
 * @date 03.06.20
```

```
 * @brief Fonction qui affiche les infrormations de l'utilisateur
 * @param {array} data
 */
function ShowProfileData(data) {
  document.title = data.pseudo;

  let html = `<div class="event-data">
  <img src="./assets/img/avatar.svg" alt="avatar">
  <h2 class="title">${data.pseudo}</h2>
  <div class="div">${data.nom == null ? '' : data.nom} ${data.prenom}</div>
  <div class="div">${data.email}</div>
  <div class="div place">${data.telephone == null ? '' : data.telephone}</div>
  <button class="btn btn-edit" onclick="window.location.href =
'./editProfile.php?idUser=${data.idUser}'">Modifier</button>
  <button class="btn btn-delete"
onclick="DeleteProfile(${data.idUser})">Supprimer</button>`;

  html += `</div>`;

  $("#userData").html(html);
}

/**
 * @author Hoarau Nicolas
 * @date 04.06.20
 * @brief Fonction qui affiche les infrormations de l'utilisateur
 * @param {int} idUser
 */
function DeleteProfile(idUser) {
  $.ajax({
    type: "post",
    url: "../App/php/isProfileDeletable.php",
    data: { idUser: idUser },
    dataType: "json",
    success: (response) => {
      switch (response.ReturnCode) {
        case 0:
          Swal.fire({
            title: "Suppression du profile",
            text: response.Success,
            icon: "question",
            showConfirmButton: true,
            confirmButtonText: 'Valider',
            showCancelButton: true,
            cancelButtonText: 'Annuler',
            reverseButtons: true
          }).then((result) => {
            if (result.value) {
              $.ajax({
                type: "post",
                url: "../App/php/deleteProfile.php",
                data: { idUser: idUser },
                dataType: "json",
                success: (data) => {
                  switch (data.ReturnCode) {
```

```
                    case 0:
                      Swal.fire({
                        title: "Suppression du profile",
                        text: data.Success,
                        icon: "success",
                        timer: 13000,
                        showConfirmButton: false,
                      }).then(() => {
                        window.location.href = '../App/php/logout.php';
                      });
                      break;
                    case 1:
                      Swal.fire({
                        title: "Suppression du profile",
                        text: data.Error,
                        icon: "error",
                        timer: 1500,
                        showConfirmButton: false,
                      });
                      break;
                  }
                },
                error: (error) => {
                  console.error(error);
                }
              });
            }
          });
          break;
        case 1:
          Swal.fire({
            title: "Suppression du profile",
            text: response.Error,
            icon: "error",
            timer: 1800,
            showConfirmButton: false
          });
          break;
      }
    },
    error: (error) => {
      console.error(error);
    }
  });
}

/**
 * @author Hoarau Nicolas
 * @date 04.06.20
 * @brief Fonction qui affiche les infrormations de l'utilisateur
 * @param {*} event
 */
function EditProfile(event) {
  if (event)
    event.preventDefault();
```

```
163
164   let lastname = $('#lastname').val();
165   let firstname = $('#firstname').val();
166   let nickname = $('#nickname').val();
167   let email = $('#email').val();
168   let phoneNumber = $('#phoneNumber').val();
169   let password = $('#password').val();
170   let verifyPassword = $('#verifyPassword').val();
171
172   if (firstname.length == 0) {
173     $('#firstname').focus();
174     return;
175   }
176
177   if (nickname.length == 0) {
178     $('#nickname').focus();
179     return;
180   }
181
182   if (email.length == 0) {
183     $('#email').focus();
184     return;
185   }
186
187   $.ajax({
188     type: "post",
189     url: "../App/php/editProfile.php",
190     data: {
191       lastname: lastname,
192       firstname: firstname,
193       nickname: nickname,
194       email: email,
195       phoneNumber: phoneNumber,
196       password: password,
197       verifyPassword: verifyPassword
198     },
199     dataType: "json",
200     success: (response) => {
201       switch (response.ReturnCode) {
202         case 0:
203           Swal.fire({
204             title: "Modification du profile",
205             text: response.Success,
206             icon: "success",
207             timer: 1300,
208             showConfirmButton: false
209           }).then(() => {
210             window.location.href = './profile.php'
211           });
212           break;
213         case 1:
214         case 2:
215         case 3:
216           Swal.fire({
217             title: "Modification du profile",
```

```
218             text: response.Error,
219             icon: "error",
220             timer: 1500,
221             showConfirmButton: false
222           });
223           break;
224       }
225     },
226     error: (error) => {
227       console.error(error);
228     }
229   });
```

```
// Conatantes login
const inputs = document.querySelectorAll(".input");

// Constantes navbar
const hamburger = document.querySelector('.hamburger');
const navlinks = document.querySelector('.nav-links');
const links = document.querySelectorAll('.nav-links li');

/**
 * @brief ajoute le focus sur l'input cliqué
 */
function addcl() {
  let parent = this.parentNode.parentNode;
  parent.classList.add("focus");
}

/**
 * @brief fonction qui retire le focus de l'input
 */
function remcl() {
  let parent = this.parentNode.parentNode;
  if (this.value == "") {
    parent.classList.remove("focus");
  }
}

// ajoute aux input les evnt focus et blur avec addcl et remcl en fontion
inputs.forEach(input => {
  input.addEventListener("focus", addcl);
  input.addEventListener("blur", remcl);
});

// ajoute l'event click au hamburger et faitl'affichage de la navbar
hamburger.addEventListener('click', () => {
  let content = document.getElementsByClassName('content')[0];

  navlinks.classList.toggle('open');

  content.style.display == 'none' ? content.style.display = 'flex' :
content.style.display = 'none';

  links.forEach(link => {
    link.classList.toggle('fade');
  })
```

```
$(document).ready(() => {
  $("#btnLogUser").click(Login);
});

/**
 * @author Hoarau Nicolas
 * @date 20.03.20
 * @brief Fonction qui rcupère les données du formulaire et les envois au serveur via
un call ajax
 * @param {*} event
 * @version 1.0
 */
function Login(event) {
  if (event)
    event.preventDefault();

  let authenticator = $('#authenticator').val();
  let password = $('#password').val();

  if (authenticator.length == 0) {
    $('#authenticator').focus();
    return;
  }

  if (password.length == 0) {
    $("#password").focus();
    return;
  }

  $.ajax({
    type: "post",
    url: "../App/php/login.php",
    data: { authenticator: authenticator, password: password },
    dataType: "json",
    success: (response) => {
      switch (response.ReturnCode) {
        case 0:
          Swal.fire({
            title: "Authentification",
            text: response.Success,
            icon: "success",
            timer: 1300,
            showConfirmButton: false
          }).then(() => {
            window.location.href = './index.php'
          });
          break;
        case 1:
          Swal.fire({
            title: "Authentification",
            text: response.Error,
            icon: "error",
            timer: 1300,
            showConfirmButton: false
```

```
54            });
55            break;
56        }
57      },
58      error: (error) => {
59        console.error(error);
60      }
61    });
```

```
const Toast = Swal.mixin({
  toast: true,
  position: 'top-end',
  showConfirmButton: false,
  timer: 1500,
  timerProgressBar: true
});

var pageName = window.location.pathname.substring(
  location.pathname.lastIndexOf("/") + 1
);

$(document).ready(() => {
  switch (pageName) {
    case "index.php":
    case "":
      GetPublicEvents();
      break;
    case "createEvent.php":
      $("#beginningDate").focus();
      $("#beginningTime").focus();
      $("#name").focus();
      break;
    case 'event.php':
      let urlParams = new URLSearchParams(window.location.search);
      let idEvent = urlParams.get('idEvent');

      GetEventData(idEvent)
      break;
    case 'manageEvents.php':
      GetManageEvents();
      break;
    case 'editEvent.php':
      $("#nbMaxGuest").focus();
      $("#eventDescription").focus();
      $("#beginningDate").focus();
      $("#beginningTime").focus();
      $("#endDate").focus();
      $("#endTime").focus();
      $("#place").focus();
      $("#eventName").focus();
      break;
  }

  $("#btnCreateEvent").click(CreateEvent);
});

// Change de couleur l'icone de l'image lorsqu'une image est sélectionné dans la
création ou la modification de l'événement
$("#img").on('change', () => {
  $("#img")[0].parentNode.parentNode.children[0].style.color = "#38d39f";
});

/**
```

```
 * @author Hoarau Nicolas
 * @date 27.05.20
 * @brief Fonction qui envoie les données pour créer un événement via un call ajax
 * @param {*} event
 * @version 1.0
 */
async function CreateEvent(event) {
  if (event)
    event.preventDefault();

  //#region Initialisation
  let formdata = new FormData();
  let eventName = $("#eventName").val();
  let place = $("#place").val();
  let eventDescription = $("#eventDescription").val();
  let nbMaxGuest = $("#nbMaxGuest").val();
  let type = $('input[name="type"]:checked').val();
  let img = document.getElementById("img").files[0];
  let beginningDate = new Date($("#beginningDate").val());
  let endDate = new Date($("#endDate").val());
  let beginningTime = $("#beginningTime").val();
  let endTime = $("#endTime").val();
  let now = new Date();

  beginningDate.setHours(beginningTime.split(':')[0], beginningTime.split(':')[1], 0,
0);
  endDate.setHours(endTime.split(':')[0], endTime.split(':')[1], 0, 0);
  now.setHours(now.getHours(), now.getMinutes(), 0, 0);
  //#endregion

  //#region Conditions
  if (eventName.length == 0) {
    $("#eventName").focus();
    return;
  } else {
    formdata.append("eventName", eventName);
  }

  if (eventDescription.length == 0) {
    $("#eventDescription").focus();
    return;
  } else {
    formdata.append("eventDescription", eventDescription);
  }

  if (place.length == 0) {
    $("#place").focus();
    return;
  } else {
    formdata.append("place", place);
  }

  if (nbMaxGuest == 0 || nbMaxGuest == "") {
    $("#nbMaxGuest").focus();
    return;
```

```
108   } else {
109     formdata.append("nbMaxGuest", nbMaxGuest);
110   }
111
112   if (beginningDate === null) {
113     $("#beginningDate").focus();
114     return;
115   } else if (beginningDate < now) {
116     Swal.fire({
117       title: "Création de l'événement",
118       text: "La date choisis est déjà passé",
119       icon: "error",
120       timer: 1500,
121       showConfirmButton: false
122     });
123     return;
124   } else {
125
126     formdata.append("beginningDate", beginningDate.toLocaleDateString());
127     formdata.append("beginningTime", beginningTime)
128   }
129
130   if (endDate === null) {
131     $("#endDate").focus();
132     return;
133   } else if (endDate < beginningDate) {
134     Swal.fire({
135       title: "Création de l'événement",
136       text: "La date choisis est avant le début de l'événement",
137       icon: "error",
138       timer: 1500,
139       showConfirmButton: false
140     });
141     return;
142   } else {
143     formdata.append("endDate", endDate.toLocaleDateString());
144     formdata.append("endTime", endTime);
145   }
146
147   if (typeof img !== 'undefined') {
148     formdata.append("img", img);
149   }
150
151   if (typeof type !== 'undefined') {
152     if (type == 'public') {
153       formdata.append("type", 0);
154     } else {
155       formdata.append("type", 1);
156       formdata.append('guestlist', await GetGuestList());
157     }
158   } else {
159     Swal.fire({
160       title: "Création de l'événement",
161       text: "Vous devez choisir si l'événement est public ou privé",
162       icon: "error",
```

```
163       timer: 1500,
164       showConfirmButton: false
165     });
166     return;
167   }
168   //#endregion
169
170   $.ajax({
171     type: "post",
172     url: "../App/php/createEvent.php",
173     // pour l'upload de fichier
174     contentType: false,
175     processData: false,
176     data: formdata,
177     dataType: "json",
178     success: (response) => {
179       switch (response.ReturnCode) {
180         case 0:
181           Swal.fire({
182             title: "Création de l'événement",
183             text: response.Success,
184             icon: "success",
185             timer: 1300,
186             showConfirmButton: false,
187           }).then(() => {
188             EventReminder();
189             window.location.href = './index.php'
190           });
191           break;
192         case 1:
193         case 2:
194         case 3:
195         case 4:
196           Swal.fire({
197             title: "Création de l'événement",
198             text: response.Error,
199             icon: "error",
200             timer: 1500,
201             showConfirmButton: false
202           });
203           break;
204       }
205     },
206     error: (error) => {
207       console.error(error);
208     }
209   });
210 }
211
212 /**
213  * @author Hoarau Nicolas
214  * @date 27.05.20
215  * @brief Fonction qui récupère les tous les utilisateur à part le créateur de
    l'événement
216  * @return le tableau de guests
```

```
 * @version 1.0
 */
function GetGuestList() {
  if (event)
    event.preventDefault();

  let guests = [];

  // Récupère les invités
  return fetch("../App/php/getGuestList.php", {
    method: 'POST',
    headers: {
      'Accept': 'application/json',
      'Content-Type': 'application/json'
    }
  }).then(response => response.json())
    .then(async data => {
      let html = `<div class="menu"><ul>`;

      $.each(data, (key, user) => {
        html += `<li>
       <label>
         <input type="checkbox" value="${user.idUser}">
         <span class="icon"></span>
         <span class="list">${user.pseudo}</span>
       </label>
     </li>`;
      });

      html += `</ul></div>`;

      // Attend la validation de la modal par l'utilisateur et récupère les invités coché
      guests = await Swal.fire({
        title: "Liste d'invités",
        html: html,
        showCancelButton: true,
        cancelButtonText: 'Annuler',
        confirmButtonText: 'Valider'
      }).then((result) => {
        let modalGuest = [];
        // Récupère les case cochées de la modal
        let pathToLi = Swal.getContent().childNodes[0].childNodes[0].children[0].children;

        // Pour chaque <li>
        for (let i = 0; i < pathToLi.length; i++) {
          const checkbox = pathToLi[i].children[0].children[0];

          if (checkbox.checked == true) {
            modalGuest.push(checkbox.value)
          }
        }
        return modalGuest;
      });
```

```
270       return guests;
271     });
272 }
273
274 /**
275  * @author Hoarau Nicolas
276  * @date 28.05.20
277  * @brief Fonction qui récupère les événement publique qui ne sont pas encore passé
278  * @version 1.0
279  */
280 function GetPublicEvents() {
281   $.ajax({
282     type: "post",
283     url: "../App/php/getPublicEvents.php",
284     dataType: "json",
285     success: (response) => {
286       ShowPublicEvents(response);
287     },
288     error: (error) => {
289       console.error(error);
290     }
291   });
292 }
293
294 /**
295  * @author Hoarau Nicolas
296  * @date 28.05.20
297  * @brief Fonction qui récupère les événement publique qui ne sont pas encore passé,
    ainsi que les événements auxquels l'utilisateur participe et est invités
298  * @version 1.0
299  */
300 function GetManageEvents() {
301   $.ajax({
302     type: "post",
303     url: "../App/php/getManageEvents.php",
304     dataType: "json",
305     success: (response) => {
306       ShowManageEvents(response);
307     },
308     error: (error) => {
309       console.error(error);
310     }
311   });
312 }
313
314 /**
315  * @author Hoarau Nicolas
316  * @date 28.05.20
317  * @brief Fonction qui affiche les événements reçus
318  * @param {array} events
319  * @version 1.0
320  */
321 function ShowPublicEvents(data) {
322   console.log(data);
323
```

```
324   let html = `<div class="card-container">`;
325
326   for (let i = 0; i < data.events.length; i++) {
327     const event = data.events[i];
328     html += `<div class="card" id="${event.idEvenement}"
    onclick="window.location.href = './event.php?idEvent=${event.idEvenement}'">
329                 <div class="card-image"><img src="${event.urlImage}"></div>
330                 <div class="card-text">
331                 <h2>${event.nom}</h2>
332                 <span class="date">Par ${event.pseudo} <br/> de ${event.dateDebut} à
    ${event.dateFin}</span>
333                 <p>${event.descriptif}</p>
334               </div></div>`;
335
336     if (i % 5 == 0 && i != 0)
337       html += `</div><div class="card-container">`;
338   }
339
340   $("#events").html(html);
341 }
342
343 /**
344  * @author Hoarau Nicolas
345  * @date 28.05.20
346  * @brief Fonction qui récupère les événement publique qui ne sont pas encore passé,
    ainsi que les événements auxquels l'utilisateur participe et est invités
347  * @version 1.0
348  */
349 function ShowManageEvents(data) {
350   let html = ``;
351
352   // Object.keys(data).length parce data.length retourne 'undefined'
353   for (let i = 0; i < Object.keys(data).length - 1; i++) {
354     const eventZone = Object.values(data)[i];
355
356     switch (i) {
357       case 0:
358         html += `<h3>Événements auxquels vous pourrez participer</h3>
359         <div class="card-container">`;
360         break;
361       case 1:
362         html += `</div><h3>Événements auxquels vous êtes invités</h3>
363         <div class="card-container">`;
364         break;
365       case 2:
366         html += `</div><h3>Événements auxquels vous participez</h3>
367         <div class="card-container">`;
368         break;
369       case 3:
370         html += `</div><h3>Événements créé</h3>
371         <div class="card-container">`;
372         break;
373     }
374
375     if (eventZone.length === 0)
```

```
376       html += `<p style="color:#999;">Vous n'avez pas d'événements dans cette
    catégorie</p></div>`;
377
378     for (let j = 0; j < eventZone.length; j++) {
379       const event = eventZone[j];
380
381       html += `<div class="card" id="${event.idEvenement}">
382       <div class="card-image" onclick="window.location.href =
    './event.php?idEvent=${event.idEvenement}'"><img src="${event.urlImage}"></div>
383       <div class="card-text">
384         <h2>${event.nom}</h2>
385         <span class="date">Par ${event.pseudo} <br/> de ${event.dateDebut} à
    ${event.dateFin}</span>
386         <p>${event.descriptif}</p>
387       </div>`;
388
389       if (data.nickname != event.pseudo) {
390         if (i === 2) {
391           html += `<button id="paricipate" class="btn"
    onclick="Participate(${event.idEvenement}, '${event.nom}', '${event.dateDebut}',
    '${event.dateFin}')">Se désinscrire</button></div>`;
392         } else {
393           html += `<button id="paricipate" class="btn"
    onclick="Participate(${event.idEvenement}, '${event.nom}', '${event.dateDebut}',
    '${event.dateFin}')">S'inscrire</button></div>`;
394         }
395       } else {
396         html += `</div>`;
397       }
398
399       if (j % 5 == 0 && j != 0)
400         html += `</div><div class="card-container">`;
401     }
402   }
403   html += `</div>`;
404   $("#events").html(html);
405 }
406
407 /**
408  * @author Hoarau Nicolas
409  * @date 28.05.20
410  * @brief Fonction qui inscrit ou désinscris l'utilisateur
411  * @param {*} btn Les informations du bouton cliqué
412  * @version 1.0
413  */
414 function Participate(idEvent, eventName, beginDate, endDate) {
415   $.ajax({
416     type: "post",
417     url: "../App/php/willParticipate.php",
418     data: {
419       "idEvent": idEvent,
420       "eventName": eventName,
421       "beginDate": beginDate,
422       "endDate": endDate
423     },
```

```
424     dataType: "json",
425     success: (response) => {
426       switch (response.ReturnCode) {
427         case 0:
428           Toast.fire({
429             icon: 'success',
430             title: response.Success
431           }).then(() => {
432             if (pageName == "index.php" || pageName == "") {
433               GetPublicEvents()
434             } else if (pageName == "manageEvents.php") {
435               GetManageEvents()
436             } else {
437               GetEventData(idEvent);
438             }
439           });
440           break;
441         case 1:
442           Toast.fire({
443             icon: 'error',
444             title: response.Error
445           });
446           break;
447       }
448     },
449     error: (error) => {
450       console.error(error);
451     }
452   });
453 }
454
455 /**
456  * @author Hoarau Nicolas
457  * @date 02.06.20
458  * @brief Fonction qui récupère les infrormations de l'id d'événement reçus
459  * @param {int} idEvent
460  */
461 function GetEventData(idEvent) {
462   $.ajax({
463     type: "post",
464     url: "../App/php/getEventData.php",
465     data: { idEvent: idEvent },
466     dataType: "json",
467     success: (response) => {
468       switch (response.ReturnCode) {
469         case 0:
470           ShowEventData(response.Data);
471           break;
472         case 1:
473           window.location.href = './index.php'
474           break;
475         case 404:
476           window.location.href = './404.php'
477           break;
478       }
```

```
    },
    error: (error) => {
      console.error(error);
    }
  });
}

/**
 * @author Hoarau Nicolas
 * @date 02.06.20
 * @brief Fonction qui affiche les données d'un événement choisis
 * @param {array} data
 */
function ShowEventData(data) {
  document.title = data.nom;

  let html = `<div class="event-data">
  <img src="${data.urlImage}">
  <h2 class="title">${data.nom}</h2>
  <div class="div place">${data.prive == 1 ? 'Privé' : 'Publique'}</div>
  <div class="div date">De ${data.dateDebut} à ${data.dateFin}<br> Par ${data.pseudo}
</div>
  <div class="div">${data.descriptif}</div>
  <div class="div place">${data.lieu}</div>
  <div class="div">${data.nbGuest} sur ${data.nbMaxParticipant} participants <a
id="showGuests" onclick="GetEventGuestList(${data.idEvenement})">Voir les
participants</a></div>`;

  if (data.isLogged === true) {
    if (data.nickname === data.pseudo) {
      if (data.prive == 1)
        html += `<div class="div"><a id="showInvited"
onclick="GetInvited(${data.idEvenement})"> Voir les invités</a> <a id="showInvited"
onclick="GetEditGuestList(${data.idEvenement})">Modifier la liste des invités</a>
</div>`;

      html += `<button class="btn btn-edit" onclick="window.location.href =
'./editEvent.php?idEvent=${data.idEvenement}'">Modifier</button>
      <button class="btn btn-delete" onclick="DeleteEvent(${data.idEvenement},
'${data.nom}', '${data.prive}')">Supprimer</button>`;
    } else {
      if (data.participating == 1) {
        html += `<button class="btn" onclick="Participate(${data.idEvenement},
'${data.nom}', '${data.dateDebut}', '${data.dateFin}')">Se désinscrire</button>`;
      } else {
        if (data.nbGuest < data.nbMaxParticipant) {
          html += `<button class="btn" onclick="Participate(${data.idEvenement},
'${data.nom}', '${data.dateDebut}', '${data.dateFin}')">S'inscrire</button>`;
        }
      }
    }
  } else {
    Toast.fire({
      icon: 'info',
      title: "Vous ne pouvez pas participer à l'événement sans être connecté"
```

```
524     });
525   }
526
527   html += `</div>`;
528
529   $("#eventData").html(html);
530 }
531
532 /**
533  * @author Hoarau Nicolas
534  * @date 02.06.20
535  * @brief Fonction qui récupère les les participants de l'événement voulus
536  * @param {int} idEvent
537  */
538 function GetEventGuestList(idEvent) {
539   $.ajax({
540     type: "post",
541     url: "../App/php/getEventGuestList.php",
542     data: { idEvent: idEvent },
543     dataType: "json",
544     success: (response) => {
545       ShowEventGuestList(response.Data);
546     },
547     error: (error) => {
548       console.error(error);
549     }
550   });
551 };
552
553 /**
554  * @author Hoarau Nicolas
555  * @date 02.06.20
556  * @brief Fonction qui affiche participants d'un événement
557  * @param {array} guestList
558  */
559 function ShowEventGuestList(guestlist) {
560   let html = `<div class="menu">
561   <ul>`;
562
563   for (let i = 0; i < guestlist.length; i++) {
564     const guest = guestlist[i];
565     html += `<li>
566     <label>
567       <span class="list">${guest.pseudo}</span>
568     </label>
569   </li>`;
570   }
571
572   html += `</ul></div>`;
573
574   Swal.fire({
575     title: "Liste des participants",
576     html: html,
577     confirmButtonText: 'Valider'
578   });
```

```
579 }
580
581 /**
582  * @author Hoarau Nicolas
583  * @date 03.06.20
584  * @brief Fonction qui envoie un mail de rappel pour les événements
585  */
586 function EventReminder() {
587   $.ajax({
588     type: "post",
589     url: "../App/php/eventReminder.php",
590     dataType: "json",
591     success: (response) => {
592       console.log('email correctly send');
593     },
594     error: (error) => {
595       console.error(error);
596     }
597   });
598 }
599
600 /**
601  * @author Hoarau Nicolas
602  * @date 03.06.20
603  * @brief Fonction qui envoie les données pour supprimer un événement
604  * @param {int} idEvent
605  */
606 function DeleteEvent(idEvent, eventName, private) {
607   console.log(private);
608
609   $.ajax({
610     type: "post",
611     url: "../App/php/isEventDeletable.php",
612     data: {
613       idEvent: idEvent,
614       eventName: eventName
615     },
616     dataType: "json",
617     success: (response) => {
618       switch (response.ReturnCode) {
619         case 0:
620           Swal.fire({
621             title: "Suppression de l'événement",
622             text: response.Success,
623             icon: "question",
624             showConfirmButton: true,
625             confirmButtonText: 'Valider',
626             showCancelButton: true,
627             cancelButtonText: 'Annuler',
628             reverseButtons: true
629           }).then((result) => {
630             if (result.value) {
631               $.ajax({
632                 type: "post",
633                 url: "../App/php/deleteEvent.php",
```

```
634                 data: {
635                   idEvent: idEvent,
636                   eventName: eventName,
637                   private: private
638                 },
639                 dataType: "json",
640                 success: (data) => {
641                   switch (data.ReturnCode) {
642                     case 0:
643                       Swal.fire({
644                         title: "Suppression de l'événement",
645                         text: response.Success,
646                         icon: "success",
647                         timer: 1300,
648                         showConfirmButton: false,
649                       }).then(() => {
650                         window.location.href = './index.php'
651                       });
652                       break;
653                     case 1:
654                       Swal.fire({
655                         title: "Suppression de l'événement",
656                         text: response.Error,
657                         icon: "error",
658                         timer: 1500,
659                         showConfirmButton: false,
660                       });
661                       break;
662                   }
663                 },
664                 error: (error) => {
665                   console.error(error);
666                 }
667               });
668             }
669           });
670           break;
671         case 1:
672           Swal.fire({
673             title: "Suppression de l'événement",
674             text: response.Error,
675             icon: "error",
676             timer: 1500,
677             showConfirmButton: false
678           });
679           break;
680       }
681     },
682     error: (error) => {
683       console.error(error);
684     }
685   });
686 }
687
688 /**
```

```
 * @author Hoarau Nicolas
 * @date 04.06.20
 * @brief Fonction qui envoie les données pour modifier un événement
 * @param {*} event
 * @param {int} idEvent
 */
function EditEvent(event, idEvent) {
  if (event)
    event.preventDefault();

  let formdata = new FormData();
  let eventName = $("#eventName").val();
  let place = $("#place").val();
  let eventDescription = $("#eventDescription").val();
  let nbMaxGuest = $("#nbMaxGuest").val();
  let img = document.getElementById("img").files[0];
  let beginningDate = new Date($("#beginningDate").val());
  let endDate = new Date($("#endDate").val());
  let beginningTime = $("#beginningTime").val();
  let endTime = $("#endTime").val();
  let now = new Date();

  beginningDate.setHours(beginningTime.split(':')[0], beginningTime.split(':')[1], 0,
0);
  endDate.setHours(endTime.split(':')[0], endTime.split(':')[1], 0, 0);
  now.setHours(now.getHours(), now.getMinutes(), 0, 0);
  formdata.append('idEvent', idEvent);
  //#endregion

  //#region Conditions
  if (eventName.length == 0) {
    $("#eventName").focus();
    return;
  } else {
    formdata.append("eventName", eventName);
  }

  if (eventDescription.length == 0) {
    $("#eventDescription").focus();
    return;
  } else {
    formdata.append("eventDescription", eventDescription);
  }

  if (place.length == 0) {
    $("#place").focus();
    return;
  } else {
    formdata.append("place", place);
  }

  if (nbMaxGuest == 0 || nbMaxGuest == "") {
    $("#nbMaxGuest").focus();
    return;
  } else {
```

```
743     formdata.append("nbMaxGuest", nbMaxGuest);
744   }
745
746   if (beginningDate === null) {
747     $("#beginningDate").focus();
748     return;
749   } else if (beginningDate < now) {
750     Swal.fire({
751       title: "Création de l'événement",
752       text: "La date choisis est déjà passé",
753       icon: "error",
754       timer: 1500,
755       showConfirmButton: false
756     });
757     return;
758   } else {
759
760     formdata.append("beginningDate", beginningDate.toLocaleDateString());
761     formdata.append("beginningTime", beginningTime)
762   }
763
764   if (endDate === null) {
765     $("#endDate").focus();
766     return;
767   } else if (endDate < beginningDate) {
768     Swal.fire({
769       title: "Création de l'événement",
770       text: "La date choisis est avant le début de l'événement",
771       icon: "error",
772       timer: 1500,
773       showConfirmButton: false
774     });
775     return;
776   } else {
777     formdata.append("endDate", endDate.toLocaleDateString());
778     formdata.append("endTime", endTime);
779   }
780
781   if (typeof img !== 'undefined') {
782     formdata.append("img", img);
783   }
784   //#endregion
785
786   $.ajax({
787     type: "post",
788     url: "../App/php/editEvent.php",
789     // pour l'upload de fichier
790     contentType: false,
791     processData: false,
792     data: formdata,
793     dataType: "json",
794     success: (response) => {
795       switch (response.ReturnCode) {
796         case 0:
797           Swal.fire({
```

```
798             title: "Création de l'événement",
799             text: response.Success,
800             icon: "success",
801             timer: 1300,
802             showConfirmButton: false,
803           }).then(() => {
804             window.location.href = './index.php'
805           });
806           break;
807         case 1:
808         case 2:
809         case 3:
810         case 4:
811           Swal.fire({
812             title: "Création de l'événement",
813             text: response.Error,
814             icon: "error",
815             timer: 1500,
816             showConfirmButton: false
817           });
818           break;
819       }
820     },
821     error: (error) => {
822       console.error(error);
823     }
824   });
825 }
826
827 /**
828  * @author Hoarau Nicolas
829  * @date 05.06.20
830  * @brief Fonction qui récupère les invités à un événement
831  * @param {int} idEvent
832  */
833 function GetInvited(idEvent) {
834   $.ajax({
835     type: "post",
836     url: "../App/php/getInvited.php",
837     data: { idEvent: idEvent },
838     dataType: "json",
839     success: (response) => {
840       switch (response.ReturnCode) {
841         case 0:
842           ShowEventInvitedList(response.Data);
843           break;
844         case 1:
845           Swal.fire({
846             title: "Liste des invités",
847             text: response.Error,
848             icon: "error",
849             timer: 1500,
850             showConfirmButton: false
851           });
852           break;
```

```
853       }
854     },
855     error: (error) => {
856       console.error(error);
857     }
858   });
859 }
860
861 /**
862  * @author Hoarau Nicolas
863  * @date 05.06.20
864  * @brief Fonction qui affiche les invités à événement
865  * @param {array} invitedList
866  */
867 function ShowEventInvitedList(invitedList) {
868   let html = `<div class="menu"><ul>`;
869
870   for (let i = 0; i < invitedList.length; i++) {
871     const invited = invitedList[i];
872     html += `<li>
873     <label>
874       <span class="list">${invited.pseudo}</span>
875     </label>
876   </li>`;
877   }
878
879   html += `</ul></div>`;
880
881   Swal.fire({
882     title: "Liste des invités",
883     html: html,
884     confirmButtonText: 'Valider'
885   });
886 }
887
888 /**
889  * @author Hoarau Nicolas
890  * @date 05.06.20
891  * @brief Fonction qui récupère les utilisateurs en disant s'il participe ou non
    l'événement donnée
892  * @param {int} idEvent
893  */
894 function GetEditGuestList(idEvent) {
895   $.ajax({
896     type: "post",
897     url: "../App/php/getEditGuestList.php",
898     data: { idEvent: idEvent },
899     dataType: "json",
900     success: (response) => {
901       switch (response.ReturnCode) {
902         case 0:
903           ShowEditGuestList(response.Data, idEvent);
904           break;
905         case 1:
906           Swal.fire({
```

```
907             title: "Liste des invités",
908             text: response.Error,
909             icon: "error",
910             timer: 1500,
911             showConfirmButton: false
912           });
913           break;
914       }
915     },
916     error: (error) => {
917       console.error(error);
918     }
919   });
920 }
921
922 /**
923  * @author Hoarau Nicolas
924  * @date 05.06.20
925  * @brief Fonction qui affche la modal pour la modification de la liste des invités e
    envoie les données de modification
926  * @param {array} data
927  * @param {int} idEvent
928  */
929 function ShowEditGuestList(data, idEvent) {
930   let html = `<div class="menu"><ul>`;
931
932   $.each(data, (key, user) => {
933     html += `<li>
934    <label>
935      <input type="checkbox" value="${user.idUser}" ${user.invited == 1 ? 'checked' :
    ''}>
936      <span class="icon"></span>
937      <span class="list">${user.pseudo}</span>
938    </label>
939  </li>`;
940   });
941
942   html += `</ul></div>`;
943
944   Swal.fire({
945     title: "Liste d'invités",
946     html: html,
947     showCancelButton: true,
948     cancelButtonText: 'Annuler',
949     confirmButtonText: 'Valider'
950   }).then((result) => {
951     if (result.value) {
952       let editGuestlist = [];
953       // Récupère les case cochées de la modal
954       let pathToLi =
    Swal.getContent().childNodes[0].childNodes[0].children[0].children;
955
956       // Pour chaque <li>
957       for (let i = 0; i < pathToLi.length; i++) {
958         const checkbox = pathToLi[i].children[0].children[0];
```

```
959
960         // regarde els différences avec la liste de base
961         if (data[i].invited != checkbox.checked) {
962           editGuestlist.push([checkbox.value, checkbox.checked])
963         }
964       }
965
966       if (editGuestlist.length > 0) {
967         $.ajax({
968           type: "post",
969           url: "../App/php/editGuestList.php",
970           data: {
971             editGuetsList: editGuestlist,
972             idEvent: idEvent
973           },
974           dataType: "json",
975           success: (response) => {
976             switch (response.ReturnCode) {
977               case 0:
978                 Swal.fire({
979                   title: "Liste d'invités",
980                   text: response.Success,
981                   icon: "success",
982                   timer: 1300,
983                   showConfirmButton: false
984                 });
985                 break;
986               case 1:
987                 Swal.fire({
988                   title: "Liste d'invités",
989                   text: response.Error,
990                   icon: "error",
991                   timer: 1300,
992                   showConfirmButton: false
993                 });
994                 break;
995             }
996           },
997           error: (error) => {
998             console.error(error);
999           }
1000        });
1001      }
1002    }
1003  });
```

```php
<nav>
  <div class="hamburger">
    <div class="line"></div>
    <div class="line"></div>
    <div class="line"></div>
  </div>
  <a href="./index.php"><img src="./assets/img/logoWego.svg" class="logo" alt="logo WEGO"></a>
  <ul class="nav-links">
    <li><a href="./index.php">Accueil</a></li>
    <?php if (IsLogged() == false) : ?>
      <li><a href="./login.php">Connexion</a></li>
      <li><a href="./register.php">Inscription</a></li>
    <?php else : ?>
      <li><a href="./profile.php">Mon profile</a></li>
      <li><a href="./manageEvents.php">Gérer mes événements</a></li>
      <li><a href="./createEvent.php">Créer un événement</a></li>
      <li><a href="../App/php/logout.php">Déconnexion</a></li>
    <?php endif; ?>
      <li><a href="./about.php">But du site</a></li>
  </ul>
</nav>
```

```
<!--    IMPORT JQUERY   -->
<script src="https://code.jquery.com/jquery.min.js"></script>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.0/jquery.min.js">
</script>

<!--    IMPORT SWEETALERTS   -->
<script src="./js/sweetalert2.min.js"></script>

<!-- IMPORT JS FOR STYLE -->
<script type="text/javascript" src="./js/main.js"></script>
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : const.inc.php
 * Date         : 27.05.2020
 * Description  : fichier qui contient les constantes du projet
 * Version      : 1.0
 */

define('UPLOAD_PATH', dirname(dirname(__DIR__)) . '/views/assets/upload/');
define('DEFAULT_IMG', './assets/img/default.svg');
```

```css
* {
  padding: 0;
  margin: 0;
  box-sizing: border-box;
}

body {
  font-family: "Poppins", sans-serif;
  overflow-y: auto;
}

/* Chrome, Safari, Edge, Opera */
input::-webkit-outer-spin-button,
input::-webkit-inner-spin-button {
  -webkit-appearance: none;
  margin: 0;
}

/* Firefox */
input[type=number] {
  -moz-appearance: textfield;
}

input[type="file"] {
  display: none;
}

/********************************** LOGIN ***********************************/
.wave {
  position: fixed;
  bottom: 0;
  left: 0;
  height: 100%;
  z-index: -1;
}

.container {
  width: 100vw;
  height: 85vh;
  display: grid;
  grid-template-columns: repeat(2, 1fr);
  grid-gap: 7rem;
  padding: 0 2rem;
}

.img {
  display: flex;
  justify-content: flex-end;
  align-items: center;
}

.content {
  display: flex;
  justify-content: flex-start;
```

```
  align-items: center;
  text-align: center;
}

.img img {
  width: 500px;
}

form {
  width: 360px;
}

.content img {
  height: 100px;
}

.content h2 {
  margin: 5px 0;
  color: #333;
  text-transform: uppercase;
  font-size: 2rem;
}

.content .input-div {
  position: relative;
  display: grid;
  grid-template-columns: 7% 93%;
  margin: 15px 0;
  padding: 5px 0;
  border-bottom: 2px solid #d9d9d9;
}

.content .input-div.one {
  margin-top: 0;
}

.i {
  color: #d9d9d9;
  display: flex;
  justify-content: center;
  align-items: center;
}

.i i {
  transition: 0.3s;
}

.input-div > div {
  position: relative;
  height: 38px;
}

.input-div > div > h5 {
  position: absolute;
  left: 10px;
```

```
  top: 50%;
  transform: translateY(-50%);
  color: #999;
  font-size: 15px;
  transition: 0.3s;
}

.input-div:before,
.input-div:after {
  content: "";
  position: absolute;
  bottom: -2px;
  width: 0%;
  height: 2px;
  background-color: #https://www.wappalyzer.com/technologies/font-scripts;
  transition: 0.4s;
}

.input-div:before {
  right: 50%;
}

.input-div:after {
  left: 50%;
}

.input-div.focus:before,
.input-div.focus:after {
  width: 50%;
}

.input-div.focus > div > h5 {
  top: -5px;
  font-size: 15px;
}

.input-div.focus > .i > i {
  color: #38d39f;
}

.input-div > div > input {
  position: absolute;
  left: 0;
  top: 0;
  width: 100%;
  height: 80%;
  border: none;
  outline: none;
  background: none;
  padding: 0.5rem 0.7rem;
  font-size: 1.2rem;
  color: #555;
  font-family: "poppins", sans-serif;
}

```

```
165 .input-div.pass {
166   margin-bottom: 4px;
167 }
168
169 a {
170   display: block;
171   text-align: right;
172   text-decoration: none;
173   color: #999;
174   font-size: 0.9rem;
175   transition: 0.3s;
176 }
177
178 a:hover {
179   color: #38d39f;
180 }
181
182 .btn {
183   display: block;
184   width: 100%;
185   height: 45px;
186   border-radius: 25px;
187   outline: none;
188   border: none;
189   background-image: linear-gradient(to right, #32be8f, #38d39f, #32be8f);
190   background-size: 200%;
191   font-size: 1.2rem;
192   color: #fff;
193   font-family: "Poppins", sans-serif;
194   text-transform: uppercase;
195   margin: 1rem 0;
196   cursor: pointer;
197   transition: 0.5s;
198 }
199
200 .btn:hover {
201   background-position: right;
202 }
203
204 ul {
205   text-decoration: none;
206   display: inline;
207 }
208
209 /********************************** NAVBAR ***********************************/
210 nav {
211   height: 8vh;
212   background: transparent;
213   width: 100%;
214 }
215
216 .nav-links {
217   display: flex;
218   list-style: none;
219   width: 50%;
```

```
220   height: 100%;
221   justify-content: space-around;
222   align-items: center;
223   margin-left: auto;
224 }
225
226 /********************************* CREATE EVENTS *********************************/
227 .radio {
228   font-size: 18px;
229   text-transform: capitalize;
230   display: inline-block;
231   vertical-align: middle;
232   position: relative;
233   padding-left: 30px;
234   cursor: pointer;
235 }
236
237 .radio + .radio {
238   margin-left: 30px;
239 }
240
241 .radio input[type="radio"] {
242   display: none;
243 }
244
245 .radio span {
246   height: 18px;
247   width: 18px;
248   border-radius: 50%;
249   border: 3px solid #38d39f;
250   display: block;
251   position: absolute;
252   left: 0;
253   top: 7px;
254 }
255
256 .radio span:after {
257   content: "";
258   height: 8px;
259   width: 8px;
260   background-color: #38d39f;
261   display: block;
262   position: absolute;
263   left: 50%;
264   top: 50%;
265   transform: translate(-50%, -50%) scale(0);
266   border-radius: 50%;
267   transition: 300ms ease-in-out 0s;
268 }
269
270 .radio input[type="radio"]:checked ~ span:after {
271   transform: translate(-50%, -50%) scale(1);
272 }
273
274 /********************************* GUEST LIST *********************************/
```

```
275 ul li {
276   list-style: none;
277   padding: 5px 0;
278   font-size: 16px;
279 }
280
281 ul li input[type="checkbox"] {
282   display: none;
283 }
284
285 ul li span.list {
286   position: relative;
287   display: inline-block;
288   overflow: hidden;
289   padding: 0 5px;
290   transition: 0.5s;
291 }
292
293 ul li input[type="checkbox"]:checked ~ span.list {
294   color: #999;
295   transition-delay: 0s;
296 }
297
298 ul li span.icon {
299   position: relative;
300   top: -3px;
301   width: 18px;
302   height: 18px;
303   box-sizing: border-box;
304   border: 1px solid #262626;
305   display: inline-block;
306   margin-right: 5px;
307   overflow: hidden;
308 }
309
310 ul li span.icon:before {
311   content: "✓";
312   position: absolute;
313   color: #38d39f;
314   top: -5px;
315   left: 2px;
316   transform: translateY(-100%);
317 }
318
319 ul li input[type="checkbox"]:checked ~ span.icon:before {
320   transform: translateY(0);
321 }
322
323 /********************************** CARD ***********************************/
324 .card-container {
325   width: 100vw;
326   height: 55vh;
327   display: flex;
328   align-items: center;
329   justify-content: center;
```

```
330   overflow: hidden;
331 }
332
333 .card {
334   display: grid;
335   grid-template-columns: 250px;
336   grid-template-rows: 150px 210px 80px;
337   grid-template-areas: "image" "text" "stats";
338
339   border-radius: 18px;
340   background: white;
341   box-shadow: 5px 5px 15px rgba(0,0,0,0.9);
342   text-align: center;
343
344   margin:30px;
345   transition: 0.5s ease;
346   cursor: pointer;
347 }
348
349 .card-image {
350   grid-area: image;
351   border-top-left-radius: 15px;
352   border-top-right-radius: 15px;
353   background-size: cover;
354 }
355
356 .card-image img {
357   width: 180px;
358   height: 180px;
359 }
360
361 .card-text {
362   grid-area: text;
363   margin: 40px 25px 25px 25px;
364 }
365 .date {
366   color: #38d39f;
367   font-size: 14px;
368 }
369
370 .card-text p {
371   color: grey;
372   font-size: 13px;
373   font-weight: 300;
374 }
375 .card-text h2 {
376   margin-top:0px;
377   font-size:18px;
378 }
379
380 .card button {
381   grid-area: stats;
382   grid-template-columns: 1fr 1fr 1fr;
383   grid-template-rows: 1fr;
384
```

```
385   border-top-left-radius: 0;
386   border-top-right-radius: 0;
387   border-bottom-left-radius: 15px;
388   border-bottom-right-radius: 15px;
389   height: 80%;
390   background: #38d39f;
391   padding: 10px;
392   cursor: pointer;
393 }
394
395 .card:hover {
396   transform: scale(1.15);
397 }
398
399 #events {
400   overflow-y: auto;
401 }
402
403
404 .event-data {
405   width: 360px;
406 }
407
408 .event-data .place{
409   color: grey;
410 }
411
412 #showGuests {
413   float: right;
414   padding-top: 3px;
415   cursor: pointer;
416 }
417
418 #showInvited {
419   text-align: center;
420   cursor: pointer;
421 }
422
423 .btn-delete {
424   background-image: linear-gradient(to right, #d33838, #cf3636, #d33838);
425 }
426
427 .btn-edit {
428   background-image: linear-gradient(to right, grey, #999, grey);
429 }
430
431 .logo {
432   float: left;
433   height: 74px;
434   padding-left: 10px;
435 }
436
437 .container-about {
438   width: 750px;
439 }
```

```
440
441 .about {
442   font-weight: lighter;
443   color: #474b52;
444 }
```

```css
/********************************** LOGIN ***********************************/
@media screen and (max-width: 1050px) {
  .container {
    grid-gap: 5rem;
  }
}

@media screen and (max-width: 1000px) {
  form {
    width: 290px;
  }

  .content {
    font-size: 2.4rem;
    margin: 8px 0;
  }

  .img img {
    width: 400px;
  }
}

@media screen and (max-width: 900px) {
  nav {
    height: 8vh;
  }

  .container {
    grid-template-columns: 1fr;
  }

  .img {
    display: none;
  }

  .wave {
    display: none;
  }

  .content {
    justify-content: center;
  }

  .one {
    margin-bottom: 10px;
  }

  .input-div {
    height: 42px;
  }

  .i {
    font-size: 1.5rem;
  }
```

```
55
56    #formRegister {
57      margin-top: 0;
58    }
59  }
60
61  /********************************** NAVBAR ***********************************/
62  @media screen and (max-width: 768px) {
63    .line {
64      width: 30px;
65      height: 3px;
66      background: #999;
67      margin: 5px;
68    }
69
70    nav {
71      position: relative;
72    }
73
74    .hamburger {
75      position: absolute;
76      cursor: pointer;
77      right: 5%;
78      top: 50%;
79      transform: translate(-5%, -50%);
80      z-index: 2;
81    }
82
83    .nav-links {
84      position: fixed;
85      background: #38d39f;
86      height: 100vh;
87      width: 100%;
88      flex-direction: column;
89      clip-path: circle(100px at 90% -15%);
90      -webkit-clip-path: circle(100px at 90% -15%);
91      transition: all 0.7s ease-out;
92      pointer-events: none;
93    }
94
95    .nav-links.open {
96      clip-path: circle(1100px at 90% -10%);
97      -webkit-clip-path: circle(1100px at 90% -10%);
98      pointer-events: all;
99    }
100
101   .nav-links li {
102     opacity: 0;
103   }
104
105   .nav-links li a {
106     font-size: 25px;
107     color: #fff;
108   }
109
```

```
110   .nav-links li:nth-child(1) {
111     transition: all 0.5s ease 0.2s;
112   }
113
114   .nav-links li:nth-child(2) {
115     transition: all 0.5s ease 0.4s;
116   }
117
118   .nav-links li:nth-child(3) {
119     transition: all 0.5s ease 0.6s;
120   }
121
122   li.fade {
123     opacity: 1;
124   }
125 }
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : config.php
 * Date         : 26.05.2020
 * Description  : Fichier qui contient les constantes de connexion à la base de données
 * Version      : 1.0
 */

// Le Type de la base de donnée
define('DB_DBTYPE', "mysql");

// Connexion local
define('DB_HOST', 'localhost');
define('DB_DBNAME', 'wego');
define('DB_USER', 'root');
define('DB_PASS', 'Super');
define('DB_PORT', 3306);

// PHPMailer
define('MAIL_USERNAME', 'nicohoarau74@gmail.com');
define('MAIL_PASSWORD', "4<Rh#6F+}kPw(kRbGHGU");
define('MAIL_HOST', "smtp.gmail.com");
define('MAIL_PORT', 587);

```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : willParticipate.php
 * Date         : 28.05.2020
 * Description  : Fichier qui met à jour si on est inscrit ou non à un événement
 * Version      : 1.0
 */

require_once dirname(__DIR__) . '/Controller/DatabaseController.php';
require_once __DIR__ . './backend.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}

$idUser = $_SESSION['loggedIn']['idUser'];
$email = $_SESSION['loggedIn']['email'];
$idEvent = filter_input(INPUT_POST, 'idEvent', FILTER_SANITIZE_NUMBER_INT);
$eventName = filter_input(INPUT_POST, 'eventName', FILTER_SANITIZE_STRING);
$beginDate = filter_input(INPUT_POST, 'beginDate');
$endDate = filter_input(INPUT_POST, 'endDate');

$mailSubject = "WEGO: Validation de partcipation";

$mailMessage = <<<EX
Vous participez à l'événement "{$eventName}" prévu de {$beginDate} à {$endDate}.
EX;

$queryWillParticipate = <<<EX
INSERT INTO inscriptions (idUser, idEvenement)
VALUES (:idUser, :idEvent);
EX;

$queryWontParticipate = <<<EX
DELETE FROM inscriptions
WHERE idUser = :idUser
AND idEvenement = :idEvent;
EX;

$isParticipating = IsParticipating($idEvent, $idUser);

$query = $isParticipating == false ? $queryWillParticipate : $queryWontParticipate;
$successMessage = $isParticipating  == false ? "L'inscription à l'événement réussi" :
"La désinscription à l'événement à réussi";

try {
  DatabaseController::beginTransaction();

  $requestParticipation = DatabaseController::prepare($query);
  $requestParticipation->bindParam(':idUser', $idUser, PDO::PARAM_INT, 11);
  $requestParticipation->bindParam(':idEvent', $idEvent, PDO::PARAM_INT, 11);
```

```php
  $requestParticipation->execute();

  if ($isParticipating == false) {
    if (SendMail($email, $mailMessage, $mailSubject) == false) {
      DatabaseController::rollBack();
      echo json_encode([
        'ReturnCode' => 1,
        'Error' => "Une erreur est survenue lors de l'envoie du mail."
      ]);
      exit();
    }
  }

  DatabaseController::commit();

  echo json_encode([
    'ReturnCode' => 0,
    'Success' => $successMessage
  ]);
  exit();
} catch (PDOException $e) {
  DatabaseController::rollBack();

  echo json_encode([
    'ReturnCode' => 1,
    'Error' => "Erreur est survenue lors de l'actualisation deéa participation"
  ]);
  exit();
}
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : register.php
 * Date         : 26.05.2020
 * Description  : Fichier qui permet filtrer les données reçus du call Ajax afin de
créer un utilisateur.
 * Version      : 1.0
 */

require_once __DIR__ . '/backend.php';

$lastname = filter_input(INPUT_POST, 'lastname', FILTER_SANITIZE_STRING);
$firstname = filter_input(INPUT_POST, 'firstname', FILTER_SANITIZE_STRING);
$nickname = filter_input(INPUT_POST, 'nickname', FILTER_SANITIZE_STRING);
$email = filter_input(INPUT_POST, 'email', FILTER_SANITIZE_EMAIL);
$phoneNumber = filter_input(INPUT_POST, 'phoneNumber', FILTER_SANITIZE_STRING);
$password = filter_input(INPUT_POST, 'password', FILTER_SANITIZE_STRING);
$verifyPassword = filter_input(INPUT_POST, 'verifyPassword', FILTER_SANITIZE_STRING);

$regex = "/^(?=\w{8,})(?=[^a-z]*[a-z])(?=[^A-Z]*[A-Z])(\w*\d)\w*/"; // Faut que le mdp
contienne au minimun 8char, une majuscule et 1chiffre

// si tous les champs obligatoire sont remplis
if (strlen($nickname) > 0 && strlen($firstname) > 0 && strlen($email) > 0 &&
strlen($password) > 0 && strlen($verifyPassword) > 0) {
  if (preg_match($regex, $password)) {
    if (IsTaken(['userEmail' => $email]) == false) {
      if (IsTaken(['userNickname' => $nickname]) == false) {
        if ($password == $verifyPassword) {
          if (Register($nickname, $firstname, $email, $password, $lastname,
$phoneNumber)) {
            echo json_encode([
              'ReturnCode' => 0,
              'Success' => 'Le compte a bien été crée.'
            ]);
            exit();
          } else {
            echo json_encode([
              'ReturnCode' => 1,
              'Error' => 'Erreur est survenue lors de la création du compte'
            ]);
            exit();
          }
        } else {
          echo json_encode([
            'ReturnCode' => 2,
            'Error' => 'Les deux mots de passe données ne sont pas les mêmes.'
          ]);
          exit();
        }
      } else {
        echo json_encode([
```

```php
          'ReturnCode' => 3,
          'Error' => 'Ce nom d\'utilisateur est déjà utilisé'
        ]);
        exit();
      }
    } else {
      echo json_encode([
        'ReturnCode' => 4,
        'Error' => 'Cette adresse mail est déjà utilisé'
      ]);
      exit();
    }
  } else {
    echo json_encode([
      'ReturnCode' => 5,
      'Error' => 'Le mot de passe doit faire au moins 8 caractères et doit contenir au moins une majuscule et un chiffre.'
    ]);
    exit();
  }
}
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : logout.php
 * Date         : 27.05.2020
 * Description  : Déconnecte l'utilisateur
 * Version      : 1.0
 */

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}

unset($_SESSION['loggedIn']);

header('Location: ../../views/index.php');
exit();
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : login.php
 * Date         : 26.05.2020
 * Description  : Fichier qui permet de filtrer les données reçus du call Ajax afin de connecter l'utilisateur
 * Version      : 1.0
 */

require_once __DIR__ . '/backend.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}

// Initialisation
$authenticator = filter_input(INPUT_POST, "authenticator", FILTER_SANITIZE_STRING);
$password = filter_input(INPUT_POST, "password", FILTER_SANITIZE_STRING);
$userLogged = null;

if (strlen($authenticator) > 0 && strlen($password) > 0) {
  if (strpos($authenticator, '@')) {
    $userLogged = Login(['userEmail' => $authenticator, 'userPwd' => $password]);
  } else {
    $userLogged = Login(['userNickname' => $authenticator, 'userPwd' => $password]);
  }

  if ($userLogged !== false) {
    $_SESSION['loggedIn'] = $userLogged;

    echo json_encode([
      'ReturnCode' => 0,
      'Success' => "Connexion réussi."
    ]);
    exit();
  }

  echo json_encode([
    'ReturnCode' => 1,
    'Error' => "Pseudo/Mot de passe invalide"
  ]);
  exit();
}
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : isProfileDeletable.php
 * Date         : 04.06.2020
 * Description  : Fichier qui vérifie si le profile est supprimable
 * Version      : 1.0
 */

require_once __DIR__ . './backend.php';

$idUser = filter_input(INPUT_POST, 'idUser', FILTER_SANITIZE_NUMBER_INT);
$usersEvents = GetUsersEvent($idUser);

if ($usersEvents == 0 || CountUsersGuest($usersEvents) == 0) {
  echo json_encode([
    'ReturnCode'  => 0,
    'Success' => 'Voulez-vous vraiment supprimer votre profile ?'
  ]);
  exit();
} else {
  echo json_encode([
    'ReturnCode' => 1,
    'Error' => "Il ne doit pas avoir de participant à vos événement pour supprimer votre compte."
  ]);
  exit();
}
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : isEventDeletable.php
 * Date         : 03.06.2020
 * Description  : Vérifie si on peut supprimer l'événement.
 * Version      : 1.0
 */

require_once __DIR__ . './backend.php';

$idEvent = filter_input(INPUT_POST, 'idEvent', FILTER_SANITIZE_NUMBER_INT);
$eventName = filter_input(INPUT_POST, 'eventName', FILTER_SANITIZE_STRING);

if (GetEventNbGuest($idEvent) == 0) {
  echo json_encode([
    'ReturnCode'  => 0,
    'Success' => 'Voulez-vous vraiment supprimer ' . $eventName . '?'
  ]);
  exit();
} else {
  echo json_encode([
    'ReturnCode' => 1,
    'Error' => "L'événement \"". $eventName ."\" ne peut être supprimé car il y a des participants"
  ]);
  exit();
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : getUserData.php
 * Date         : 03.06.2020
 * Description  : Fichier qui envoie les données de l'utilisateur
 * Version      : 1.0
 */

require_once dirname(__DIR__) . '/Controller/DatabaseController.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}

$idUser = $_SESSION['loggedIn']['idUser'];

$query = <<<EX
SELECT idUser, pseudo, prenom, nom, email, telephone
FROM user
WHERE idUser = :idUser;
EX;

try {
  $requestGetUserData = DatabaseController::prepare($query);
  $requestGetUserData->bindParam(':idUser', $idUser, PDO::PARAM_STR);
  $requestGetUserData->execute();

  $_SESSION['loggedIn'] = $requestGetUserData->fetchAll(PDO::FETCH_ASSOC)[0];

  echo json_encode([
    'ReturnCode' => 0,
    'Data' => $_SESSION['loggedIn']
  ]);
} catch (PDOException $e) {
  echo json_encode([
    'ReturnCode' => 1,
    'Error' => 'Erreur est survenue lors de la modification du compte'
  ]);
  exit();
}
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : getPublicEvents.php
 * Date         : 28.05.2020
 * Description  : Fichier qui récupère les événements publiques qui ne sont pas encore passé
 * Version      : 1.0
 */

require_once dirname(__DIR__) . '/Controller/DatabaseController.php';
require_once __DIR__ . './backend.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}

$dateMoreOneHour = date('Y-m-d H:i:s', strtotime(date('Y-m-d H:i:s') . '+1hours'));

$queryGetPublicEvents = <<<EX
SELECT e.idEvenement, e.nom, e.descriptif, e.dateDebut, e.dateFin, e.urlImage, u.pseudo
FROM evenement AS e
JOIN user AS u ON e.idOrganisateur = u.idUser
WHERE prive = 0
AND TIMESTAMP(dateDebut) >= :dateTime
ORDER BY dateDebut DESC
EX;

try {
  $requestGetPublicEvents = DatabaseController::prepare($queryGetPublicEvents);
  $requestGetPublicEvents->bindParam(':dateTime', $dateMoreOneHour);
  $requestGetPublicEvents->execute();

  $result['events'] = $requestGetPublicEvents->fetchAll(PDO::FETCH_ASSOC);

  echo json_encode($result);
} catch (PDOException $e) {
  echo json_encode([
    'ReturnCode' => 1,
    'Error' => 'Erreur est survenue lors de la récupération des événements'
  ]);
  exit();
}

```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : getEvents.php
 * Date         : 28.05.2020
 * Description  : Fichier qui récupère les événements pour la gestion des événements s
on est connecté sinon récupère les événements publiques.
 * Version      : 1.0 : Récupère les événements pour la gestion des événements si on
est connecté
 */

require_once dirname(__DIR__) . '/Controller/DatabaseController.php';
require_once __DIR__ . './backend.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}

$dateMoreOneHour = date('Y-m-d H:i:s', strtotime(date('Y-m-d H:i:s') . '+1hours'));
$idUser = $_SESSION['loggedIn']['idUser'];

$queryGetEventsParticpating = <<<EX
SELECT e.idEvenement, e.nom, e.descriptif, e.dateDebut, e.dateFin, e.urlImage, u.pseuc
FROM inscriptions AS i
JOIN evenement AS e ON i.idEvenement = e.idEvenement
JOIN user AS u ON e.idOrganisateur = u.idUser
WHERE i.idUser = :idUser
AND e.idOrganisateur <> :idUser
AND TIMESTAMP(dateDebut) >= :dateTime
EX;

$queryGetPublicEventsNotParticipating = <<<EX
SELECT e.idEvenement, e.nom, e.descriptif, e.dateDebut, e.dateFin, e.urlImage, u.pseuc
FROM evenement AS e
JOIN user AS u ON u.idUser = e.idOrganisateur
WHERE idEvenement NOT IN (
  SELECT idEvenement
  FROM inscriptions
  WHERE idUser = :idUser )
AND prive = 0
AND e.idOrganisateur <> :idUser;
EX;

$queryGetInvitedEvents = <<<EX
SELECT e.idEvenement, e.nom, e.descriptif, e.dateDebut, e.dateFin, e.urlImage, u.pseuc
FROM invites AS i
JOIN evenement AS e ON e.idEvenement = i.idEvenement
JOIN user AS u ON u.idUser = e.idOrganisateur
WHERE prive = 1
AND TIMESTAMP(dateDebut) >= :dateTime
AND e.idEvenement NOT IN (
  SELECT idEvenement
  FROM inscriptions
```

```
  WHERE idUser = :idUser )
AND i.idUser = :idUser
ORDER BY dateDebut DESC;
EX;

$queryGetEventCreated = <<<EX
SELECT e.idEvenement, e.nom, e.descriptif, e.dateDebut, e.dateFin, e.urlImage, u.pseuc
FROM evenement AS e
JOIN user AS u ON u.idUser = e.idOrganisateur
WHERE e.idOrganisateur = :idUser
AND TIMESTAMP(dateDebut) >= :dateTime
EX;

try {
  $requestGetPublicEventsNotParticipating = DatabaseController::prepare($queryGetPublicEventsNotParticipating);
  $requestGetPublicEventsNotParticipating->bindParam(':idUser', $idUser, PDO::PARAM_INT, 11);
  $requestGetPublicEventsNotParticipating->bindParam(':dateTime', $dateMoreOneHour);

  $requestGetPublicEventsNotParticipating->execute();
  $result['eventsNotParticipating'] = $requestGetPublicEventsNotParticipating->fetchAll(PDO::FETCH_ASSOC);

  $requestGetInvitedEvents = DatabaseController::prepare($queryGetInvitedEvents);
  $requestGetInvitedEvents->bindParam(':idUser', $idUser, PDO::PARAM_INT, 11);
  $requestGetInvitedEvents->bindParam(':dateTime', $dateMoreOneHour);

  $requestGetInvitedEvents->execute();
  $result['invitedEvents'] = $requestGetInvitedEvents->fetchAll(PDO::FETCH_ASSOC);

  $requestGetEventsParticpating = DatabaseController::prepare($queryGetEventsParticpating);
  $requestGetEventsParticpating->bindParam(':idUser', $idUser, PDO::PARAM_INT, 11);
  $requestGetEventsParticpating->bindParam(':dateTime', $dateMoreOneHour);

  $requestGetEventsParticpating->execute();
  $result['eventsParticipating'] = $requestGetEventsParticpating->fetchAll(PDO::FETCH_ASSOC);

  $requestGetEventCreated = DatabaseController::prepare($queryGetEventCreated);
  $requestGetEventCreated->bindParam(':idUser', $idUser, PDO::PARAM_INT, 11);
  $requestGetEventCreated->bindParam(':dateTime', $dateMoreOneHour);

  $requestGetEventCreated->execute();
  $result['eventsCreated'] = $requestGetEventCreated->fetchAll(PDO::FETCH_ASSOC);

  $result['isLogged'] = true;
  $result['nickname'] = $_SESSION['loggedIn']['pseudo'];

  echo json_encode($result);
  exit();
} catch (PDOException $e) {
  echo json_encode([
    'ReturnCode' => 1,
```

```
103     'Error' => 'Erreur est survenue lors de la récupération des événements'
104   ]);
105   exit();
106 }
107
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : getInvited.php
 * Date         : 05.06.2020
 * Description  : Fichier qui récupère les invités à un événements
 * Version      : 1.0
 */

require_once __DIR__ . '/backend.php';

$idEvent = filter_input(INPUT_POST, 'idEvent', FILTER_SANITIZE_NUMBER_INT);

$data = (GetInvited($idEvent) != null) ? GetInvited($idEvent) : false;

if ($data != false) {
  echo json_encode([
    'ReturnCode' => 0,
    'Data' => $data
  ]);
  exit();
} else {
  echo json_encode([
    'ReturnCode' => 1,
    'Error' => 'Erreur est survenue lors de la récupération des invités'
  ]);
  exit();
}
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : getGuestList.php
 * Date         : 27.05.2020
 * Description  : Récupère tous les utilisateurs de la base de données à part le créateur de l'événement
 * Version      : 1.0
 */

require_once __DIR__ . '/backend.php';
require_once dirname(__DIR__) . '/Controller/DatabaseController.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}

$nickname = $_SESSION['loggedIn']['pseudo'];

$query = <<<EX
  SELECT idUser, pseudo FROM user WHERE pseudo <> :pseudo;
  EX;

try {
  $requestGetUsers = DatabaseController::prepare($query);
  $requestGetUsers->bindParam(':pseudo', $nickname, PDO::PARAM_STR, 50);
  $requestGetUsers->execute();

  $result = $requestGetUsers->fetchAll(PDO::FETCH_ASSOC);

  echo json_encode($result);
  exit();
} catch (PDOException $e) {
  echo json_encode([
    'ReturnCode' => 1,
    'Error' => 'Erreur est survenue lors de la récupération de la liste des invités'
  ]);
  exit();
}
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : getEventGuestList.php
 * Date         : 02.06.2020
 * Description  : Fichier qui récupère participant d'un événement données.
 * Version      : 1.0
 */

require_once dirname(__DIR__) . '/Controller/DatabaseController.php';

$idEvent = filter_input(INPUT_POST, 'idEvent', FILTER_SANITIZE_NUMBER_INT);

$query = <<<EX
SELECT u.pseudo
FROM inscriptions AS i
JOIN user AS u ON i.idUser = u.idUser
WHERE idEvenement = :idEvent;
EX;

try {
  $requestGetGuestList = DatabaseController::prepare($query);

  $requestGetGuestList->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
  $requestGetGuestList->execute();
  $result = $requestGetGuestList->fetchAll(PDO::FETCH_ASSOC);

  echo json_encode([
    'Data' => $result
  ]);
  exit();
} catch (PDOException $e) {
      echo json_encode([
        'ReturnCode' => 1,
        'Error' => 'Erreur est survenue lors de la récupération de la liste des participants'
      ]);
      exit();

  exit();
}
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : getEventData.php
 * Date         : 02.06.2020
 * Description  : Fichier qui récupère els informations d'un événement donnée
 * Version      : 1.0
 */

require_once __DIR__ . './backend.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}

$idEvent = filter_input(INPUT_POST, 'idEvent', FILTER_SANITIZE_NUMBER_INT);
$idUser = isset($_SESSION['loggedIn']['idUser']) ? $_SESSION['loggedIn']['idUser'] :
null;
$nickname = isset($_SESSION['loggedIn']['idUser']) ? $_SESSION['loggedIn']['pseudo'] :
null;

if (EventExist($idEvent)) {
  if ($idUser == null) {
    $data = GetEventData($idEvent);
    $data['isLogged'] = false;

    echo json_encode([
      'ReturnCode' => 0,
      'Data' => $data
    ]);
    exit();
  } else {
    if (IsPrivate($idEvent)) {
      if (IsInvited($idEvent, $idUser) || IsCreator($idEvent, $idUser)) {
        $data = GetEventData($idEvent, $idUser);
        $data['isLogged'] = true;
        $data['nickname'] = $nickname;

        echo json_encode([
          'ReturnCode' => 0,
          'Data' => $data
        ]);
        exit();
      } else {
        echo json_encode([
          'ReturnCode' =>  1,
        ]);
      }
    } else {
      $data = GetEventData($idEvent, $idUser);
      $data['isLogged'] = true;
      $data['nickname'] = $nickname;
```

```php
      echo json_encode([
        'ReturnCode' => 0,
        'Data' => $data
      ]);
      exit();
    }
  }
} else {
  echo json_encode([
    'ReturnCode' => 404
  ]);
  exit();
}
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : getEditGuestList.php
 * Date         : 05.06.2020
 * Description  : Fichier qui récupère les utilisateurs en disant s'il participe ou non
l'événement donnée
 * Version      : 1.0
 */

 require_once dirname(__DIR__) . '/Controller/DatabaseController.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}

$idEvent = filter_input(INPUT_POST, 'idEvent', FILTER_SANITIZE_NUMBER_INT);
$nickname = $_SESSION['loggedIn']['pseudo'];
$query = <<<EX
SELECT DISTINCT u.idUser, u.pseudo,
  CASE WHEN i.idUser IS NOT NULL AND i.idEvenement = :idEvent THEN TRUE ELSE FALSE END
AS invited
FROM user AS u
LEFT JOIN invites AS i ON i.idUser = u.idUser
WHERE i.idEvenement = :idEvent
OR i.idEvenement IS NULL
OR i.idUser NOT IN (
  SELECT idUser FROM invites WHERE idEvenement = :idEvent
)
AND u.pseudo <> :nickname;
EX;

try {
  $requestGetInvitedAndNotInvited = DatabaseController::prepare($query);
  $requestGetInvitedAndNotInvited->bindParam(':nickname', $nickname, PDO::PARAM_STR);
  $requestGetInvitedAndNotInvited->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
  $requestGetInvitedAndNotInvited->execute();
  $result = $requestGetInvitedAndNotInvited->fetchAll(PDO::FETCH_ASSOC);

  echo json_encode([
    'ReturnCode' => 0,
    'Data' => $result
  ]);
  exit();
} catch (PDOException $e) {
  echo json_encode([
    'ReturnCode' => 1,
    'Error' => 'Erreur est survenue lors de la récupération des utilisateurs'
  ]);
  exit();
}
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : eventReminder.php
 * Date         : 03.06.2020
 * Description  : Fichier qui vérifie si un évnemnt est dans 24h et si c'est le cas envoie un mail de rappel
 * Version      : 1.0
 */

require_once __DIR__ . './backend.php';

$events = GetAllFutureEvent();
$now = date('Y-m-d');
$mailSubject = "WEGO: Rappel de partcipation";

foreach ($events as $event) {
  $idEvent = $event['idEvenement'];
  $eventName = $event['nom'];
  $beginingDate = $event['dateDebut'];
  $dayBeforeEvent = date('Y-m-d', strtotime($beginingDate . '-1 days'));
  $mailMessage = <<<EX
  N'oubliez pas que l'événement "{$eventName}" se déroule demain.
  EX;

  if (strtotime($now) >= strtotime($dayBeforeEvent) && strtotime($now) < strtotime($beginingDate)) {
    if (GetEventNbGuest($idEvent) > 0) {
      $eventUsersMail = GetEventUsersMail($idEvent);

      foreach ($eventUsersMail as $mail) {
        if (SendMail($mail['email'], $mailMessage, $mailSubject) == false) {
          echo json_encode([
            'ReturnCode' => 1,
            'Error' => "Une erreur est survenue lors de l'envoie du mail."
          ]);
          exit();
        }
      }
    }
  }
}
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : editProfile.php
 * Date         : 05.06.2020
 * Description  : Fichier qui permet de filtrer les données reçus du call Ajax afin de
modifier le profil.
 * Version      : 1.0
 */

require_once __DIR__ . '/backend.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}

$idUser = $_SESSION['loggedIn']['idUser'];
$oldEmail = $_SESSION['loggedIn']['email'];

$lastname = filter_input(INPUT_POST, 'lastname', FILTER_SANITIZE_STRING);
$firstname = filter_input(INPUT_POST, 'firstname', FILTER_SANITIZE_STRING);
$nickname = filter_input(INPUT_POST, 'nickname', FILTER_SANITIZE_STRING);
$email = filter_input(INPUT_POST, 'email', FILTER_SANITIZE_EMAIL);
$phoneNumber = filter_input(INPUT_POST, 'phoneNumber', FILTER_SANITIZE_STRING);
$password = filter_input(INPUT_POST, 'password', FILTER_SANITIZE_STRING);
$verifyPassword = filter_input(INPUT_POST, 'verifyPassword', FILTER_SANITIZE_STRING);

$regex = "/^(?=\w{8,})(?=[^a-z]*[a-z])(?=[^A-Z]*[A-Z])(\w*\d)\w*/"; // Faut que le mdp
contienne au minimun 8char, une majuscule et 1chiffre

// si tous les champs obligatoire sont remplis
if (IsTaken(['userEmail' => $email, 'idUser' => $idUser]) == false) {
  if (IsTaken(['userNickname' => $nickname, 'idUser' => $idUser]) == false) {
    if ($password != "") {
      if (preg_match($regex, $password)) {
        if ($password != $verifyPassword) {
          echo json_encode([
            'ReturnCode' => 4,
            'Error' => 'Les deux mots de passe données ne sont pas les mêmes.'
          ]);
          exit();
        }
      } else {
        echo json_encode([
          'ReturnCode' => 5,
          'Error' => 'Le mot de passe doit faire au moins 8 caractères et doit contenir
au moins une majuscule et un chiffre.'
        ]);
        exit();
      }
    }

    if (EditProfile( $nickname,  $firstname,  $email,  $password,  $lastname,
```

```
$phoneNumber, $idUser,  $oldEmail)) {
      echo json_encode([
        'ReturnCode' => 0,
        'Success' => 'Le compte a bien été modifié.'
      ]);
      exit();
    } else {
      echo json_encode([
        'ReturnCode' => 1,
        'Error' => 'Erreur est survenue lors de la modification du compte'
      ]);
      exit();
    }
  } else {
    echo json_encode([
      'ReturnCode' => 2,
      'Error' => 'Ce nom d\'utilisateur est déjà utilisé'
    ]);
    exit();
  }
} else {
  echo json_encode([
    'ReturnCode' => 3,
    'Error' => 'Cette adresse mail est déjà utilisé'
  ]);
  exit();
}
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : editGuestList.php
 * Date         : 05.06.2020
 * Description  : fichier qui modifie la liste des invités
 * Version      : 1.0
 */

require_once __DIR__ . '/backend.php';

$idEvent = filter_input(INPUT_POST, 'idEvent', FILTER_SANITIZE_NUMBER_INT);
$editGuestList = $_POST['editGuetsList'];

foreach ($editGuestList as $user) {
  if ($user[1] == "false") {
    if (DeleteInvitation($user[0], $idEvent) == false) {
      echo json_encode([
        'ReturnCode' => 1,
        'Error' => 'Erreur est survenue lors de la modification de la base de données'
      ]);
      exit();
    }
  } else {
    if (CreateInvite($user[0], $idEvent) == false) {
      echo json_encode([
        'ReturnCode' => 1,
        'Error' => 'Erreur est survenue lors de la modification de la base de données'
      ]);
      exit();
    }
  }
}

echo json_encode([
  'ReturnCode' => 0,
  'Success' => 'La modification de la liste des invités a bien été effectué'
]);
exit();

```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : editEvent.php
 * Date         : 04.06.2020
 * Description  : Fichier qui permet de filtrer les données reçus du call Ajax afin de
modifier un événement.
 * Version      : 1.0
 */

require_once __DIR__ . '/backend.php';
require_once dirname(dirname(__DIR__)) . '/views/includes/const.inc.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}

$finfo = finfo_open(FILEINFO_MIME_TYPE);
$idEvent = filter_input(INPUT_POST, 'idEvent', FILTER_SANITIZE_NUMBER_INT);
$eventName = filter_input(INPUT_POST, "eventName", FILTER_SANITIZE_STRING);
$eventDescription = filter_input(INPUT_POST, "eventDescription",
FILTER_SANITIZE_STRING);
$place = filter_input(INPUT_POST, "place", FILTER_SANITIZE_STRING);
$beginningDate = str_replace('/', '-', filter_input(INPUT_POST, "beginningDate"));
$beginningTime = filter_input(INPUT_POST, "beginningTime");
$endDate = str_replace('/', '-', filter_input(INPUT_POST, "endDate"));
$endTime = filter_input(INPUT_POST, "endTime");
$nbMaxGuest = filter_input(INPUT_POST, "nbMaxGuest", FILTER_SANITIZE_NUMBER_INT);
$type = filter_input(INPUT_POST, "type", FILTER_SANITIZE_NUMBER_INT);
$img = isset($_FILES['img']) ? $_FILES['img'] : null;

$beginningDate = date("Y-m-d H:i", strtotime($beginningTime,
strtotime($beginningDate)));
$endDate = date("Y-m-d H:i", strtotime($endTime, strtotime($endDate)));

if (strlen($eventName) > 0 && strlen($eventDescription) > 0 && $nbMaxGuest > 0 &&
strlen($place) > 0) {
  if (strtotime($beginningDate) > strtotime(date("Y-m-d"))) {
    if (strtotime($endDate) >= strtotime($beginningDate)) {
      if (isset($img['tmp_name'])) {
        $mimeType = finfo_file($finfo, $img['tmp_name']);
        $fileType = explode('/', $mimeType)[0];

        if ($fileType == 'image') {
          $fileExtension = pathinfo($img['name'], PATHINFO_EXTENSION);
          $filename = uniqid() . '.' . $fileExtension;

          if (move_uploaded_file($img['tmp_name'], UPLOAD_PATH . $filename)) {
            if (EditEvent($eventName, $eventDescription, $place, $beginningDate,
$endDate, $nbMaxGuest, './assets/upload/' . $filename, $idEvent)) {
              echo json_encode([
                'ReturnCode' => 0,
                'Success' => "L'événement " . $eventName . " a bien été modifié"
```

```php
            ]);
            exit();
          }
        } else {
          echo json_encode([
            'ReturnCode' => 1,
            'Error' => "Un problème est survenue lors de l'upload de l'image"
          ]);
          exit();
        }
      } else {
        echo json_encode([
          'ReturnCode' => 2,
          'Error' => "Le fichier données n'est pas une image"
        ]);
        exit();
      }
    }

    if (EditEvent($eventName, $eventDescription, $place, $beginningDate, $endDate, $nbMaxGuest, null,  $idEvent)) {
      echo json_encode([
        'ReturnCode' => 0,
        'Success' => "L'événement " . $eventName . " a bien été modifié"
      ]);
      exit();
    }
  } else {
    echo json_encode([
      'ReturnCode' => 3,
      'Error' => "L'événement ne peut pas se terminer avant d'avoir commencé"
    ]);
    exit();
  }
} else {
  echo json_encode([
    'ReturnCode' => 4,
    'Error' => "L'événement ne peut pas commencer avant aujourd'hui"
  ]);
  exit();
}
}
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : deleteProfile.php
 * Date         : 04.06.2020
 * Description  : Fichier qui vérifie si le profile est supprimable
 * Version      : 1.0
 */

require_once dirname(__DIR__) . '/Controller/DatabaseController.php';
require_once __DIR__ . './backend.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}

$idUser = filter_input(INPUT_POST, 'idUser', FILTER_SANITIZE_STRING);

$query = <<<EX
DELETE FROM user WHERE idUser = :idUser;
EX;

if ($_SESSION['loggedIn']['idUser'] != $idUser) {
  echo json_encode([
    'ReturnCode' => 2
  ]);
  exit();
}

$usersEventParticipating = GetEventsParticipating($idUser);
$usersEventInvited = GetEventsInvited($idUser);
$userEventCreated = GetUsersEvent($idUser);
$getUserInvitedToTheEvent = [];

if (count($usersEventParticipating) > 0) {
  foreach ($usersEventParticipating as $eventParticipating) {
    if (DeleteParticipation($idUser, $eventParticipating['idEvenement']) == false) {
      echo json_encode([
        'ReturnCode' => 1,
        'Error' => 'Une erreur est survenue lors de la suppression de votre profile.'
      ]);
    }
  }
}

if (count($usersEventInvited) > 0) {
  foreach ($usersEventInvited as $eventInvited) {
    if (IsInvited($eventInvited['idEvenement'], $idUser)) {
      if (DeleteInvitation($idUser, $eventInvited['idEvenement']) == false) {
        echo json_encode([
          'ReturnCode' => 1,
          'Error' => 'Une erreur est survenue lors de la suppression de votre profile.
        ]);
```

```php
      }
    }
  }
}

foreach ($userEventCreated as $event) {
  $getUserInvitedToTheEvent = GetInvited($event['idEvenement']);

  if (count($getUserInvitedToTheEvent) > 0) {
    foreach ($getUserInvitedToTheEvent as $user) {
      if (DeleteInvitation($user['idUser'], $event['idEvenement']) == false) {
        echo json_encode([
          'ReturnCode' => 1,
          'Error' => 'Une erreur est survenue lors de la suppression de votre profile.
        ]);
      }
    }
  }

  if (DeleteEvent($event['idEvenement']) == false) {
    echo json_encode([
      'ReturnCode' => 1,
      'Error' => 'Une erreur est survenue lors de la suppression de votre profile.'
    ]);
  }
}

try {
  DatabaseController::beginTransaction();

  $requestDeleteEvent = DatabaseController::prepare($query);
  $requestDeleteEvent->bindParam(':idUser', $idUser, PDO::PARAM_INT);
  $requestDeleteEvent->execute();

  DatabaseController::commit();

  echo json_encode([
    'ReturnCode' => 0,
    'Success' => "Votre compte à bien été supprimé."
  ]);
} catch (PDOException $e) {
  DatabaseController::rollBack();
  echo json_encode([
    'ReturnCode' => 1,
    'Error' => 'Une erreur est survenue lors de la suppression de votre profile.'
  ]);
}

```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : deleteEvent.php
 * Date         : 03.06.2020
 * Description  : Fichier qui supprime un événement donnée
 * Version      : 1.0
 */

require_once __DIR__ . '/backend.php';

$idEvent = filter_input(INPUT_POST, 'idEvent', FILTER_SANITIZE_NUMBER_INT);
$eventName = filter_input(INPUT_POST, 'eventName', FILTER_SANITIZE_STRING);
$private = filter_input(INPUT_POST, 'private', FILTER_SANITIZE_NUMBER_INT);

if ($private == 1) {
  $invited = GetInvited($idEvent);
  foreach ($invited as $user) {
    if (DeleteInvitation($user['idUser'], $idEvent) == false) {
      echo json_encode([
        'ReturnCode' => 1,
        'Error' => "Erreur est survenue lors de la suppression de l'événement"
      ]);
      exit();
    }
  }
}

if (DeleteEvent($idEvent)) {
  echo json_encode([
    'ReturnCode' => 0,
    'Success' => "L'événement \"" . $eventName . "\" est bien supprimé"
  ]);
  exit();
} else {
  echo json_encode([
    'ReturnCode' => 1,
    'Error' => "Erreur est survenue lors de la suppression de l'événement"
  ]);
  exit();
}
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : createEvent.php
 * Date         : 27.05.2020
 * Description  : Fichier qui permet de filtrer les données reçus du call Ajax afin de
créer un événement.
 * Version      : 1.0
 */

require_once __DIR__ . '/backend.php';
require_once dirname(dirname(__DIR__)) . '/views/includes/const.inc.php';

if (session_status() == PHP_SESSION_NONE) {
  session_start();
}

$idUser = $_SESSION['loggedIn']['idUser'];
$finfo = finfo_open(FILEINFO_MIME_TYPE);

$eventName = filter_input(INPUT_POST, "eventName", FILTER_SANITIZE_STRING);
$eventDescription = filter_input(INPUT_POST, "eventDescription",
FILTER_SANITIZE_STRING);
$place = filter_input(INPUT_POST, "place", FILTER_SANITIZE_STRING);
$beginningDate = str_replace('/', '-', filter_input(INPUT_POST, "beginningDate"));
$beginningTime = filter_input(INPUT_POST, "beginningTime");
$endDate = str_replace('/', '-', filter_input(INPUT_POST, "endDate"));
$endTime = filter_input(INPUT_POST, "endTime");
$nbMaxGuest = filter_input(INPUT_POST, "nbMaxGuest", FILTER_SANITIZE_NUMBER_INT);
$type = filter_input(INPUT_POST, "type", FILTER_SANITIZE_NUMBER_INT);
$img = isset($_FILES['img']) ? $_FILES['img'] : DEFAULT_IMG;

$guestlist = isset($_POST['guestlist']) ? explode(',', $_POST['guestlist']): null;

$beginningDate = date("Y-m-d H:i", strtotime($beginningTime,
strtotime($beginningDate)));
$endDate = date("Y-m-d H:i", strtotime($endTime, strtotime($endDate)));

if (strlen($eventName) > 0 && strlen($eventDescription) > 0 && $nbMaxGuest > 0 &&
strlen($place) > 0) {
  if (strtotime($beginningDate) > strtotime(date("Y-m-d"))) {
    if (strtotime($endDate) >= strtotime($beginningDate)) {

      if (isset($img['tmp_name'])) {
        $mimeType = finfo_file($finfo, $img['tmp_name']);
        $fileType = explode('/', $mimeType)[0];

        if ($fileType == 'image') {
          $fileExtension = pathinfo($img['name'], PATHINFO_EXTENSION);
          $filename = uniqid() . '.' . $fileExtension;

          if (move_uploaded_file($img['tmp_name'], UPLOAD_PATH . $filename)) {
            if (CreateEvent($eventName, $eventDescription, $place, $beginningDate,
```

```
$endDate, $nbMaxGuest, './assets/upload/' . $filename, $type, $idUser, $guestlist)) {
              echo json_encode([
                'ReturnCode' => 0,
                'Success' => "L'événement " . $eventName . " a bien été crée"
              ]);
              exit();
            }
          } else {
            echo json_encode([
              'ReturnCode' => 1,
              'Error' => "Un problème est survenue lors de l'upload de l'image"
            ]);
            exit();
          }
        } else {
          echo json_encode([
            'ReturnCode' => 2,
            'Error' => "Le fichier données n'est pas une image"
          ]);
          exit();
        }
      }

      if (CreateEvent($eventName, $eventDescription, $place, $beginningDate, $endDate,
$nbMaxGuest, $img, $type, $idUser, $guestlist)) {
        echo json_encode([
          'ReturnCode' => 0,
          'Success' => "L'événement " . $eventName . " a bien été crée"
        ]);
        exit();
      }
    } else {
      echo json_encode([
        'ReturnCode' => 3,
        'Error' => "L'événement ne peut pas se terminer avant d'avoir commencé"
      ]);
      exit();
    }
  } else {
    echo json_encode([
      'ReturnCode' => 4,
      'Error' => "L'événement ne peut pas commencer avant aujourd'hui"
    ]);
    exit();
  }
}
```

```php
<?php
/*
 * Auteur       : Hoarau Nicolas
 * Projet       : WE GO
 * Page         : backend.php
 * Date         : 26.05.2020
 * Description  : Fichier qui contient toutes les fonctionsdu site.
 * Version      : 1.0
 */

/* Import PHPMailer classes into the global namespace
 These must be at the top of your script, not inside a function*/

use PHPMailer\PHPMailer\PHPMailer;
use PHPMailer\PHPMailer\SMTP;
use PHPMailer\PHPMailer\Exception;

// Load Composer's autoloader
require dirname(dirname(__DIR__)) . '/vendor/autoload.php';
require_once  dirname(__DIR__) . '/Controller/DatabaseController.php';
require_once dirname(dirname(__DIR__)) . '/config/config.php';

/**
 * @date 26.05.20
 * @author Hoarau Nicolas
 * @brief Fonction qui vérifie si l'utilisateur est connecté
 * @return boolean
 * @version 1.0
 */
function IsLogged(): bool
{
  return array_key_exists('loggedIn', $_SESSION) && $_SESSION['loggedIn'];
}

/**
 * @author Hoarau Nicolas
 * @date 27.05.2020
 * @brief Fonction qui récupère le salt de l'utilisateur
 * @param array $args
 * @return string
 * @version 1.0
 */
function GetSalt(array $args): ?string
{
  // initialise à null les colonnes du tableau vide/inexistante
  $args += [
    'userEmail' => null,
    'userNickname' => null
  ];

  extract($args); // extrait les données du tableau avec comme nom de vararіable son
nom de colonne

  $clauseField = "";
```

```php
  $clauseValue = "";

  if ($userEmail !== null) {
    $clauseField = "email";
    $clauseValue = $userEmail;
  } elseif ($userNickname) {
    $clauseField = "pseudo";
    $clauseValue = $userNickname;
  }

  $query = <<<EX
    SELECT salt
    FROM user
    WHERE `{$clauseField}` = :clauseValue;
    EX;

  try {
    $requestSalt = DatabaseController::prepare($query);
    $requestSalt->bindParam(':clauseValue', $clauseValue, PDO::PARAM_STR);
    $requestSalt->execute();

    $result = $requestSalt->fetch(PDO::FETCH_ASSOC);

    return $result !== false ? $result['salt'] : null;
  } catch (PDOException $e) {
    return null;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 26.05.2020
 * @brief Fonction qui créer un nouvel utilisateur dans la base de données
 * @param string $nickname
 * @param string $lastname
 * @param string $firstname
 * @param string $email
 * @param string $phoneNumber
 * @param string $password
 * @return boolean
 * @version 1.0
 *    Inscription sans salt
 * @version 1.1
 *    Inscription avec salt
 */
function Register(string $nickname, string $firstname, string $email, string $password, string $lastname = null, string $phoneNumber = null): bool
{
  $query = "";
  $salt = hash('sha256', microtime());
  $userPassword = hash('sha256', $password . $salt);

  if ($lastname != null && $phoneNumber != null) {
    $query = <<<EX
      INSERT INTO user (pseudo, prenom, nom, email, password, telephone, salt)
```

```php
      VALUES (:nickname, :firstname, :lastname, :email, :password, :phoneNumber, :salt);
      EX;
  } elseif ($phoneNumber != null) {
    $query = <<<EX
      INSERT INTO user (pseudo, prenom, nom, email, password, telephone, salt)
      VALUES (:nickname, :firstname, :email, :password, :phoneNumber, :Salt);
      EX;
  } elseif ($lastname != null) {
    $query = <<<EX
      INSERT INTO user (pseudo, prenom, nom, email, password, salt)
      VALUES (:nickname, :firstname, :lastname, :email, :password, :salt);
      EX;
  } else {
    $query = <<<EX
      INSERT INTO user (pseudo, prenom, email, password, salt)
      VALUES (:nickname, :firstname, :email, :password, :salt);
      EX;
  }

  try {
    DatabaseController::beginTransaction();

    $requestRegister = DatabaseController::prepare($query);
    $requestRegister->bindParam(':nickname', $nickname, PDO::PARAM_STR, 50);
    $requestRegister->bindParam(':firstname', $firstname, PDO::PARAM_STR, 50);

    if ($lastname != null)
      $requestRegister->bindParam(':lastname', $lastname, PDO::PARAM_STR, 50);

    $requestRegister->bindParam(':email', $email, PDO::PARAM_STR, 100);
    $requestRegister->bindParam(':password', $userPassword, PDO::PARAM_STR, 100);

    if ($phoneNumber != null)
      $requestRegister->bindParam(':phoneNumber', $phoneNumber, PDO::PARAM_STR, 50);

    $requestRegister->bindParam(':salt', $salt, PDO::PARAM_STR, 64);

    $requestRegister->execute();

    DatabaseController::commit();
    return true;
  } catch (PDOException $e) {
    DatabaseController::rollBack();
    return false;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 26.05.2020
 * @brief Fonction qui connecte l'utilisateur
 * @param array les données de conexion de l'utilisateur
 * @return array|false
 * @version 1.0
```

```php
 *    Inscription sans salt
 * @version 1.1
 *    Inscription avec salt
 */
function Login(array $args)
{
  // initialise à null les colonnes du tableau vide/inexistante
  $args += [
    'userEmail' => null,
    'userNickname' => null,
    'userPwd' => null,
  ];

  extract($args); // extrait les données du tableau avec comme nom de varariable son nom de colonne

  $loginField = "";
  $authenticator = "";
  $salt = "";

  if ($userEmail !== null) {
    $salt = GetSalt(['userEmail' => $userEmail]);
    $authenticator = $userEmail;
    $loginField = "email";
  } elseif ($userNickname !== null) {
    $salt = GetSalt(['userNickname' => $userNickname]);
    $authenticator = $userNickname;
    $loginField = "pseudo";
  } else {
    return false;
  }

  if ($userPwd == null)
    return false;

  $pwd = hash('sha256', $userPwd . $salt);

  $query = <<<EX
    SELECT idUser, pseudo, prenom, nom, email, telephone
     FROM user
     WHERE `{$loginField}` = :wayToConnectValue
     AND password = :pwd;
    EX;

  try {
    $requestLogin = DatabaseController::prepare($query);
    $requestLogin->bindParam(':wayToConnectValue', $authenticator, PDO::PARAM_STR);
    $requestLogin->bindParam(':pwd', $pwd, PDO::PARAM_STR);
    $requestLogin->execute();

    $result = $requestLogin->fetch(PDO::FETCH_ASSOC);

    return $result !== false > 0 ?  $result : false;
  } catch (PDOException $e) {
    return false;
```

```php
  }
}

/**
 * @author Hoarau Nicolas
 * @date 26.05.2020
 * @brief Fonction qui vérifie si le paramètre donnée(email, nickname) est déjà utilisé ou non
 * @param array $args
 * @return boolean|null
 * @version 1.0
 */
function IsTaken(array $args): ?bool
{
  $args += [
    'userEmail' => null,
    'userNickname' => null,
    'idUser' => null
  ];

  extract($args);

  $authenticator = "";
  $field = "";
  if ($userEmail != null) {
    $authenticator = $userEmail;
    $field = "email";
  } elseif ($userNickname != null) {
    $authenticator = $userNickname;
    $field = "pseudo";
  }

  $queryWithoutIdUser = <<<EX
    SELECT `{$field}`
    FROM user
    WHERE `{$field}` = :authenticator;
    EX;

  $queryWithIdUser = <<<EX
    SELECT `{$field}`
    FROM user
    WHERE `{$field}` = :authenticator
    AND idUser <> :idUser;
    EX;


  $query = ($idUser != null) ? $queryWithIdUser : $queryWithoutIdUser;

  try {
    $requestIsUsed = DatabaseController::prepare($query);
    $requestIsUsed->bindParam(':authenticator', $authenticator, PDO::PARAM_STR);

    if ($idUser != null)
      $requestIsUsed->bindParam(':idUser', $idUser, PDO::PARAM_INT);

```

```php
    $requestIsUsed->execute();
    $result = $requestIsUsed->fetch(PDO::FETCH_ASSOC);

    return $result !== false ? true : false;
  } catch (PDOException $e) {
    return null;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 27.05.2020
 * @brief Fonction qui créée un événement
 * @param string $eventName
 * @param string $eventDescription
 * @param string $place
 * @param Date $beginningDate
 * @param Date $endDate
 * @param integer $nbMaxGuest
 * @param FILES $img
 * @param integer $idUser
 * @return boolean
 * @version 1.0
 */
function CreateEvent(string $eventName, string $eventDescription, string $place, $beginningDate, $endDate, int $nbMaxGuest, $img, int $type, int $idUser, array $guestlist = null): bool
{
  $creationDate = date('Y-m-d H:i');
  $beginingDate = date("Y-m-d H:i", strtotime($beginningDate));
  $endDate = date("Y-m-d H:i", strtotime($endDate));

  $query = <<<EX
    INSERT INTO evenement (nom, descriptif, dateDebut, dateFin, dateCreation, lieu, prive, nbMaxParticipant, urlImage, idOrganisateur)
    VALUES (:name, :description, :beginningDate, :endDate, :creationDate, :place, :type, :nbMaxGuest, :urlImg ,:idUser);
    EX;

  try {
    DatabaseController::beginTransaction();

    $requestCreateEvent = DatabaseController::prepare($query);
    $requestCreateEvent->bindParam(':name', $eventName, PDO::PARAM_STR, 45);
    $requestCreateEvent->bindParam(':description', $eventDescription, PDO::PARAM_STR, 255);
    $requestCreateEvent->bindParam(':beginningDate', $beginingDate);
    $requestCreateEvent->bindParam(':endDate', $endDate);
    $requestCreateEvent->bindParam(':creationDate', $creationDate);
    $requestCreateEvent->bindParam(':place', $place, PDO::PARAM_STR, 255);
    $requestCreateEvent->bindParam(':type', $type, PDO::PARAM_INT, 1);
    $requestCreateEvent->bindParam(':nbMaxGuest', $nbMaxGuest, PDO::PARAM_INT, 11);
    $requestCreateEvent->bindParam(':urlImg', $img, PDO::PARAM_STR, 255);
    $requestCreateEvent->bindParam(':idUser', $idUser, PDO::PARAM_INT, 11);

```

```php
    $requestCreateEvent->execute();

    if ($guestlist != null) {
      $lastInsertId = DatabaseController::getInstance()->lastInsertId();

      for ($i = 0; $i < count($guestlist); $i++) {
        if ($idUser != $guestlist[$i]) {
          if (CreateInvite($guestlist[$i], $lastInsertId) == false) {
            DatabaseController::rollBack();
          }
        }
      }

      DatabaseController::commit();
      return true;
    }

    DatabaseController::commit();
    return true;
  } catch (PDOException $e) {
    DatabaseController::rollBack();
    return false;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 27.05.2020
 * @brief Fonction qui créée une invitation pour un utilisateur à un événement
 * @param integer $idGuest
 * @param integer $idEvent
 * @return boolean
 * @version 1.0
 */
function CreateInvite(int $idGuest, int $idEvent): bool
{
  $query = <<<EX
    INSERT INTO invites (idUser, idEvenement)
    VALUES (:idGuest, :idEvent);
    EX;

  try {
    DatabaseController::beginTransaction();

    $requestCreateInvite = DatabaseController::prepare($query);
    $requestCreateInvite->bindParam(':idGuest', $idGuest, PDO::PARAM_INT, 11);
    $requestCreateInvite->bindParam(':idEvent', $idEvent, PDO::PARAM_INT, 11);
    $requestCreateInvite->execute();

    DatabaseController::commit();

    return true;
  } catch (PDOException $e) {
    DatabaseController::rollBack();
    return false;
```

```php
  }
}

/**
 * @author Hoarau Nicolas
 * @date 05.06.2020
 * @brief Fonction qui supprime l'invitation d'un utilisateur à un événement
 * @param integer $idGuest
 * @param integer $idEvent
 * @return boolean
 * @version 1.0
 */
function DeleteInvite(int $idGuest, int $idEvent): bool
{
  $query = <<<EX
  DELETE FROM invites WHERE idUser = :idUser AND idEvenement = :idEvent;
  EX;

  try {
    DatabaseController::beginTransaction();

    $requestDeleteInvite = DatabaseController::prepare($query);
    $requestDeleteInvite->bindParam(':idGuest', $idGuest, PDO::PARAM_INT, 11);
    $requestDeleteInvite->bindParam(':idEvent', $idEvent, PDO::PARAM_INT, 11);
    $requestDeleteInvite->execute();

    DatabaseController::commit();

    return true;
  } catch (PDOException $e) {
    DatabaseController::rollBack();
    return false;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 02.06.2020
 * @brief Fonction qui vérifie si l'événement existe
 * @param integer $idEvent
 * @return boolean
 * @version 1.0
 */
function EventExist(int $idEvent): bool
{
  $query = <<<EX
  SELECT COUNT(idEvenement) AS exist
  FROM evenement
  WHERE idEvenement = :idEvent
  EX;

  try {
    $requestExist = DatabaseController::prepare($query);
    $requestExist->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
    $requestExist->execute();
```

```php

    $result = $requestExist->fetch(PDO::FETCH_ASSOC);

    return $result['exist'] == 0 ? false : true;
  } catch (PDOException $e) {
    return false;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 02.06.2020
 * @brief Fonction qui vérifie si l'utilisateur participaê à l'événement
 * @param integer $idEvent
 * @param integer $idUser
 * @return boolean
 * @version 1.0
 */
function IsParticipating(int $idEvent, int $idUser): bool
{
  $query = <<<EX
  SELECT COUNT(*) AS participating
  FROM inscriptions
  WHERE idUser = :idUser
  AND idEvenement = :idEvent
  EX;

  try {
    $requestIsParticipating = DatabaseController::prepare($query);
    $requestIsParticipating->bindParam(':idUser', $idUser, PDO::PARAM_INT);
    $requestIsParticipating->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
    $requestIsParticipating->execute();

    $result = $requestIsParticipating->fetch(PDO::FETCH_ASSOC);

    return $result['participating'] == 0 ? false : true;
  } catch (PDOException $e) {
    return false;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 02.06.2020
 * @brief Fonction qui vérifie si l'événement est privé
 * @param integer $idEvent
 * @return boolean
 * @version 1.0
 */
function IsPrivate(int $idEvent): bool
{
  $query = <<<EX
  SELECT prive
  FROM evenement
  WHERE idEvenement = :idEvent;
```

```php
  EX;

  try {
    $requestIsPrivate = DatabaseController::prepare($query);
    $requestIsPrivate->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
    $requestIsPrivate->execute();

    $result = $requestIsPrivate->fetch(PDO::FETCH_ASSOC);

    return $result['prive'] == 0 ? false : true;
  } catch (PDOException $e) {
    return false;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 02.06.2020
 * @brief Fonction qui vérifie si l'utilisateur est invité à l'événement donnée
 * @param integer $idEvent
 * @param integer $idUser
 * @return boolean
 * @version 1.0
 */
function IsInvited(int $idEvent, int $idUser): bool
{
  $query = <<<EX
  SELECT COUNT(*) AS isInvited
  FROM invites
  WHERE idUser = :idUser
  AND idEvenement = :idEvent;
  EX;

  try {
    $requestIsInvited = DatabaseController::prepare($query);
    $requestIsInvited->bindParam(':idUser', $idUser, PDO::PARAM_INT);
    $requestIsInvited->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
    $requestIsInvited->execute();
    $result = $requestIsInvited->fetch(PDO::FETCH_ASSOC);

    return $result['isInvited'] == 1 ? true : false;
  } catch (PDOException $e) {
    return false;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 02.06.2020
 * @brief Fonction qui vérifie si l'utilisateur est le créateur de l'événement donnée
 * @param integer $idEvent
 * @param integer $idUser
 * @return boolean
 * @version 1.0
 */
```

```php
function IsCreator(int $idEvent, int $idUser): bool
{
  $query = <<<EX
  SELECT COUNT(*) AS isCreator
  FROM evenement
  WHERE idOrganisateur = :idUser
  AND idEvenement = :idEvent;
  EX;

  try {
    $requestIsCreator = DatabaseController::prepare($query);
    $requestIsCreator->bindParam(':idUser', $idUser, PDO::PARAM_INT);
    $requestIsCreator->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
    $requestIsCreator->execute();

    $result = $requestIsCreator->fetch(PDO::FETCH_ASSOC);

    return $result['isCreator'] == 1 ? true : false;
  } catch (PDOException $e) {
    return false;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 02.06.2020
 * @brief Fonction qui récupère les information de l'événement
 * @param integer $idEvent
 * @param integer $idUser
 * @return array|false
 * @version 1.0
 */
function GetEventData(int $idEvent, int $idUser = null)
{
  $queryLogged = <<<EX
  SELECT e.idEvenement, e.nom, e.descriptif, e.dateDebut, e.dateFin, e.lieu,
e.nbMaxParticipant, e.prive, e.urlImage, u.pseudo, (SELECT COUNT(*) FROM inscriptions
WHERE idUser = :idUser AND idEvenement = :idEvent) AS participating, (SELECT
COUNT(idUser) FROM inscriptions WHERE idevenement = :idEvent) AS nbGuest
  FROM evenement AS e
  JOIN user AS u ON e.idOrganisateur = u.idUser
  WHERE e.idEvenement = :idEvent;
  EX;

  $queryNotLogged = <<<EX
  SELECT e.idEvenement, e.nom, e.descriptif, e.dateDebut, e.dateFin, e.lieu,
e.nbMaxParticipant, e.prive, e.urlImage, u.pseudo, (SELECT COUNT(idUser) FROM
inscriptions WHERE idEvenement = :idEvent) AS nbGuest
  FROM evenement AS e
  JOIN user AS u ON e.idOrganisateur = u.idUser
  WHERE e.idEvenement = :idEvent;
  EX;

  $query = $idUser !== null ? $queryLogged : $queryNotLogged;

```

```php
  try {
    $requestGetEventData = DatabaseController::prepare($query);

    if ($idUser !== null)
      $requestGetEventData->bindParam(':idUser', $idUser, PDO::PARAM_INT);

    $requestGetEventData->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
    $requestGetEventData->execute();

    $result = $requestGetEventData->fetch(PDO::FETCH_ASSOC);

    return $result !== false ? $result : null;
  } catch (PDOException $e) {
    return false;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 02.06.2020
 * @brief Fonction qui envoie un mail à un destinataire précit
 * @param string $recipientMail
 * @param string $message
 * @return bool
 * @version 1.0
 */
function SendMail(string $recipientMail, string $message, string $subject): bool
{
  // Instantiation and passing `true` enables exceptions
  $mail = new PHPMailer(true);

  try {
    //Enable SMTP debugging.
    $mail->SMTPDebug = 0;
    //Set PHPMailer to use SMTP.
    $mail->isSMTP();

    //Set SMTP host name
    $mail->Host = MAIL_HOST;

    //Set this to true if SMTP host requires authentication to send email
    $mail->SMTPAuth = true;

    //Provide username and password
    $mail->Username = MAIL_USERNAME;
    $mail->Password = MAIL_PASSWORD;

    //If SMTP requires TLS encryption then set it
    $mail->SMTPSecure = "tls";

    //Set TCP port to connect to
    $mail->Port = MAIL_PORT;
    $mail->From = MAIL_USERNAME;
    $mail->FromName = "WEGO Service";

```

```php
    $mail->addAddress($recipientMail);

    $mail->isHTML(true);

    $mail->Subject = $subject;
    $mail->Body =  $message;

    if ($mail->send())
      return true;
    else
      return false;
  } catch (Exception $e) {
    return false;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 03.06.2020
 * @brief Récupère tous les événements publiques pas encore passé
 * @return array|null
 * @version 1.0
 */
function GetAllFutureEvent(): ?array
{
  $dateMoreOneHour = date('Y-m-d H:i:s', strtotime(date('Y-m-d H:i:s') . '+1hours'));

  $query = <<<EX
  SELECT idEvenement, nom, dateDebut
  FROM evenement
  WHERE TIMESTAMP(dateDebut) >= :dateTime
  ORDER BY dateDebut DESC
  EX;

  try {
    $requestGetAllFutureEvent = DatabaseController::prepare($query);
    $requestGetAllFutureEvent->bindParam(':dateTime', $dateMoreOneHour);
    $requestGetAllFutureEvent->execute();
    $result = $requestGetAllFutureEvent->fetchAll(PDO::FETCH_ASSOC);
    return $result;
  } catch (PDOException $e) {
    return null;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 03.06.2020
 * @brief Fonction qui récupère le nombre de participant à un événement
 * @param integer $idEvent
 * @return integer|null
 * @version 1.0
 */
function GetEventNbGuest(int $idEvent): ?int
{
```

```php
  $query = <<<EX
  SELECT COUNT(idUser) AS nbGuest
  FROM inscriptions
  WHERE idEvenement = :idEvent
  EX;

  try {
    $requestGetEventNbGuest = DatabaseController::prepare($query);
    $requestGetEventNbGuest->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
    $requestGetEventNbGuest->execute();

    $result = $requestGetEventNbGuest->fetch(PDO::FETCH_ASSOC);

    return $result !== false ? $result['nbGuest'] : 0;
  } catch (PDOException $e) {
    return null;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 03.06.2020
 * @brief Fonction qui récupère les addresses mail des participant à l'événement donr
 * @param integer $idEvent
 * @return array|null
 * @version 1.0
 */
function GetEventUsersMail(int $idEvent): ?array
{
  $query = <<<EX
  SELECT u.email
  FROM inscriptions AS i
  JOIN user AS u on i.idUser = u.idUser
  WHERE idEvenement = :idEvent;
  EX;

  try {
    $requestGetEventUsersMail = DatabaseController::prepare($query);
    $requestGetEventUsersMail->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
    $requestGetEventUsersMail->execute();

    $result = $requestGetEventUsersMail->fetchAll(PDO::FETCH_ASSOC);

    return $result;
  } catch (PDOException $e) {
    return null;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 04.06.2020
 * @brief Fonction qui modifie un événement donné
 * @param string $eventName
 * @param string $eventDescription
```

```php
 * @param string $place
 * @param Date $beginningDate
 * @param Date $endDate
 * @param integer $nbMaxGuest
 * @param integer $idUser
 * @param integer $idEvent
 * @return boolean
 * @version 1.0
 */
function EditEvent(string $eventName, string $eventDescription, string $place, $beginningDate, $endDate, int $nbMaxGuest, $img = null, int $idEvent): bool
{
  $query = <<<EX
  UPDATE evenement
  SET nom = :eventName, descriptif = :eventDescription, lieu = :place, dateDebut = :beginningDate, dateFin = :endDate, nbMaxParticipant = :nbMaxGuest
  EX;

  if ($img != null)
    $query .= ", urlImage = :urlImg";

  $query .= " WHERE idEvenement = :idEvent";

  try {
    DatabaseController::beginTransaction();
    $requestEditEvent = DatabaseController::prepare($query);
    $requestEditEvent->bindParam(':eventName', $eventName, PDO::PARAM_STR);
    $requestEditEvent->bindParam(':eventDescription', $eventDescription, PDO::PARAM_STR);
    $requestEditEvent->bindParam(':place', $place, PDO::PARAM_STR);
    $requestEditEvent->bindParam(':beginningDate', $beginningDate);
    $requestEditEvent->bindParam(':endDate', $endDate);
    $requestEditEvent->bindParam(':nbMaxGuest', $nbMaxGuest, PDO::PARAM_INT);

    if ($img != null)
      $requestEditEvent->bindParam(':urlImg', $img, PDO::PARAM_STR, 255);

    $requestEditEvent->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
    $requestEditEvent->execute();
    DatabaseController::commit();
    return true;
  } catch (PDOException $e) {
    DatabaseController::rollBack();
    return false;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 04.06.2020
 * @brief Fonction qui récupère l'id des événements d'un utilisateur
 * @param integer $idUser
 * @return array|int|null
 * @version 1.0
 */
```

```php
function GetUsersEvent(int $idUser)
{
  $query = <<<EX
  SELECT idEvenement FROM evenement WHERE idOrganisateur = :idUser;
  EX;

  try {
    $requestGetUsersEvents = DatabaseController::prepare($query);
    $requestGetUsersEvents->bindParam(':idUser', $idUser, PDO::PARAM_INT);
    $requestGetUsersEvents->execute();

    $result = $requestGetUsersEvents->fetchAll(PDO::FETCH_ASSOC);
    return $result == false ? 0 : $result;
  } catch (PDOException $e) {
    return null;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 05.06.2020
 * @brief Fonction qui compte le nombre de participant aux événement de l'utilisateur
 * @param array $usersEvent
 * @return integer
 * @version 1.0
 */
function CountUsersGuest(array $usersEvent): int
{
  $result = 0;

  foreach ($usersEvent as $event) {
    $result += GetEventNbGuest($event['idEvenement']);

    if ($result > 0)
      break;
  }

  return $result;
}

/**
 * @author Hoarau Nicolas
 * @date 05.06.2020
 * @brief Fonction qui récupère les événements auxquels un utilisateur participe
 * @param integer $idUser
 * @return array|null
 * @version 1.0
 */
function GetEventsParticipating(int $idUser): ?array
{
  $query = <<<EX
  SELECT idEvenement FROM inscriptions WHERE idUser = :idUser;
  EX;

  try {
```

```php
    $requestGetEventsParticipating = DatabaseController::prepare($query);
    $requestGetEventsParticipating->bindParam(':idUser', $idUser, PDO::PARAM_INT);
    $requestGetEventsParticipating->execute();

    $result = $requestGetEventsParticipating->fetchAll(PDO::FETCH_ASSOC);

    return $result;
  } catch (PDOException $e) {
    return null;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 05.06.2020
 * @brief Fonction qui supprime la participation d'un utilisateur à un événement
 * @param integer $idUser
 * @param integer $idEvent
 * @return boolean
 * @version 1.0
 */
function DeleteParticipation(int $idUser, int $idEvent): bool
{
  $query = <<<EX
  DELETE FROM inscriptions WHERE idEvenement = :idEvent AND idUser = :idUser;
  EX;

  try {
    DatabaseController::beginTransaction();

    $requestDeleteParticipation = DatabaseController::prepare($query);
    $requestDeleteParticipation->bindParam(':idUser', $idUser, PDO::PARAM_INT);
    $requestDeleteParticipation->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
    $requestDeleteParticipation->execute();

    DatabaseController::commit();

    return true;
  } catch (PDOException $e) {
    DatabaseController::rollBack();
    return false;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 05.06.2020
 * @brief Fonction qui supprime l'invitation d'un utilisateur à un événement
 * @param integer $idUser
 * @param integer $idEvent
 * @return boolean
 * @version 1.0
 */
function DeleteInvitation(int $idUser, int $idEvent): bool
{
```

```php
  $query = <<<EX
  DELETE FROM invites WHERE idEvenement = :idEvent AND idUser = :idUser;
  EX;

  try {
    DatabaseController::beginTransaction();

    $requestDeleteDeleteInvitation = DatabaseController::prepare($query);
    $requestDeleteDeleteInvitation->bindParam(':idUser', $idUser, PDO::PARAM_INT);
    $requestDeleteDeleteInvitation->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
    $requestDeleteDeleteInvitation->execute();

    DatabaseController::commit();

    return true;
  } catch (PDOException $e) {
    DatabaseController::rollBack();
    return false;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 05.06.2020
 * @brief Fonction qui récupère les événements auxquels un utilisateur est invités
 * @param integer $idUser
 * @return array|null
 * @version 1.0
 */
function GetEventsInvited(int $idUser): ?array
{
  $query = <<<EX
  SELECT idEvenement FROM invites WHERE idUser = :idUser;
  EX;

  try {
    $requestGetEventsParticipating = DatabaseController::prepare($query);
    $requestGetEventsParticipating->bindParam(':idUser', $idUser, PDO::PARAM_INT);
    $requestGetEventsParticipating->execute();

    $result = $requestGetEventsParticipating->fetchAll(PDO::FETCH_ASSOC);

    return $result;
  } catch (PDOException $e) {
    return null;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 05.06.2020
 * @brief Fonction qui modifie un utilisateur donné
 * @param string $nickname
 * @param string $firstname
 * @param string $email
```

```
 * @param string $password
 * @param string $lastname
 * @param string $phoneNumber
 * @param integer $idUser
 * @param string $oldEmail
 * @return boolean
 * @version 1.0
 */
function EditProfile(string $nickname, string $firstname, string $email, string $password = null, string $lastname = null, string $phoneNumber = null, int $idUser, string $oldEmail): bool
{
  $salt = "";
  $userPassword = null;

  $query = <<<EX
  UPDATE user
  SET pseudo = :nickname, prenom = :firstname, email = :email
  EX;

  if ($lastname != null || $lastname != "")
    $query .= ", nom = :lastname";

  if ($phoneNumber != null || $phoneNumber != "")
    $query .= ", telephone = :phoneNumber";

  if ($password != null || $password != "") {
    $salt = GetSalt(['userEmail' => $oldEmail]);
    $userPassword = hash('sha256', $password . $salt);
    $query .= ", password = :password";
  }

  $query .= " WHERE idUser = :idUser;";

  try {
    DatabaseController::beginTransaction();

    $requestEditEvent = DatabaseController::prepare($query);
    $requestEditEvent->bindParam(':nickname', $nickname, PDO::PARAM_STR);
    $requestEditEvent->bindParam(':firstname', $firstname, PDO::PARAM_STR);
    $requestEditEvent->bindParam(':email', $email, PDO::PARAM_STR);

    if ($lastname != null)
      $requestEditEvent->bindParam(':lastname', $lastname, PDO::PARAM_STR);

    if ($phoneNumber != null)
      $requestEditEvent->bindParam(':phoneNumber', $phoneNumber, PDO::PARAM_STR);

    if ($userPassword != null)
      $requestEditEvent->bindParam(':password', $userPassword, PDO::PARAM_STR);

    $requestEditEvent->bindParam(':idUser', $idUser, PDO::PARAM_INT);
    $requestEditEvent->execute();
    DatabaseController::commit();
    return true;
```

```php
  } catch (PDOException $e) {
    DatabaseController::rollBack();
    return false;
  }
}

function GetInvited(int $idEvent): ?array
{
  $query = <<<EX
  SELECT u.pseudo, i.idUser
  FROM invites AS i
  JOIN user AS u ON i.idUser = u.idUser
  WHERE i.idEvenement = :idEvent
  EX;

  try {
    $requestGetInvited = DatabaseController::prepare($query);
    $requestGetInvited->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
    $requestGetInvited->execute();
    $result = $requestGetInvited->fetchAll(PDO::FETCH_ASSOC);

    return $result;
  } catch (PDOException $e) {
    return null;
  }
}

/**
 * @author Hoarau Nicolas
 * @date 08.06.2020
 * @brief Fonction qui supprime un événement données
 * @param integer $idEvent
 * @return boolean
 */
function DeleteEvent(int $idEvent): bool
{
  $query = <<<EX
  DELETE FROM evenement WHERE idEvenement = :idEvent;
  EX;

  try {
    DatabaseController::beginTransaction();
    $requestDeleteEvent = DatabaseController::prepare($query);
    $requestDeleteEvent->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
    $requestDeleteEvent->execute();
    DatabaseController::commit();
    return true;
  } catch (PDOException $e) {
    DatabaseController::rollBack();
    return false;
  }
}
```

```php
<?php
//@source https://coderwall.com/p/rml5fa/nested-pdo-transactions

/**
 * This class extends native PDO one but allow nested transactions
 * by using the SQL statements `SAVEPOINT', 'RELEASE SAVEPOINT' AND 'ROLLBACK
SAVEPOINT'
 */
class ExtendedPdo extends PDO
{
  /**
   * @var array Database drivers that support SAVEPOINT * statements.
   */
  protected static $_supportedDrivers = array("pgsql", "mysql");

  /**
   * @var int the current transaction depth
   */
  protected $_transactionDepth = 0;


  /**
   * Test if database driver support savepoints
   *
   * @return bool
   */
  protected function hasSavepoint()
  {
    return in_array($this->getAttribute(PDO::ATTR_DRIVER_NAME),
      self::$_supportedDrivers);
  }


  /**
   * Start transaction
   *
   * @return bool|void
   */
  public function beginTransaction()
  {
    if($this->_transactionDepth == 0 || !$this->hasSavepoint()) {
      parent::beginTransaction();
    } else {
      $this->exec("SAVEPOINT LEVEL{$this->_transactionDepth}");
    }

    $this->_transactionDepth++;
  }

  /**
   * Commit current transaction
   *
   * @return bool|void
   */
```

```php
  public function commit()
  {
    $this->_transactionDepth--;

    if($this->_transactionDepth == 0 || !$this->hasSavepoint()) {
      parent::commit();
    } else {
      $this->exec("RELEASE SAVEPOINT LEVEL{$this->_transactionDepth}");
    }
  }

  /**
   * Rollback current transaction,
   *
   * @throws PDOException if there is no transaction started
   * @return bool|void
   */
  public function rollBack()
  {

    if ($this->_transactionDepth == 0) {
      throw new PDOException('Rollback error : There is no transaction started');
    }

    $this->_transactionDepth--;

    if($this->_transactionDepth == 0 || !$this->hasSavepoint()) {
      parent::rollBack();
    } else {
      $this->exec("ROLLBACK TO SAVEPOINT LEVEL{$this->_transactionDepth}");
    }
  }

}
```

```php
<?php

/**
 * @author  dominique.aigroz@edu.ge.ch
 * Modify by: Hoarau Nicolas
 */
require_once dirname(dirname(__DIR__)) . '/config/config.php';
require_once __DIR__ . '/ExtendedPdo.php';

/**
 * @brief Helper class encapsulating
 *      the PDO object
 * @author  dominique.aigroz@kadeo.net
 * @remark
 */
class DatabaseController
{
    private static $pdoInstance;

    /**
     * @brief Class Constructor - Create a new database connection if one doesn't exist
       *      Set to private so no-one can create a new instance via ' = new
KDatabase();'
        */
    private function __construct()
    {
    }

    /**
     * @brief Like the constructor, we make __clone private so nobody can clone the
instance
        */
    private function __clone()
    {
    }

    /**
     * @brief Returns DB instance or create initial connection
     * @return $objInstance;
     */
    public static function getInstance()
    {
        if (!self::$pdoInstance) {
            try {
                $dsn = DB_DBTYPE . ':host=' . DB_HOST . ';port=' . DB_PORT . ';dbname=
. DB_DBNAME . ';charset=utf8';
                self::$pdoInstance = new ExtendedPdo($dsn, DB_USER, DB_PASS);
                self::$pdoInstance->setAttribute(ExtendedPdo::ATTR_ERRMODE,
ExtendedPdo::ERRMODE_EXCEPTION);
            } catch (PDOException $e) {
                echo "KDatabase Error: " . $e->getMessage();
            }
        }
        return self::$pdoInstance;
```

```php
    }

    # end method
    /**
     * @brief Passes on any static calls to this class onto the singleton PDO instance
     * @param   $chrMethod    The method to call
     * @param   $arrArguments The method's parameters
     * @return  $mix      The method's return value
     */
    final public static function __callStatic($chrMethod, $arrArguments)
    {
        $pdo = self::getInstance();
        return call_user_func_array(array($pdo, $chrMethod), $arrArguments);
    }

    # end method
}

```