

Rapport TPI et documentation technique

wego

Travail pratique individuel (TPI)

Hoarau Nicolas

09.06.2020

Table des matières

Rapport TPI et documentation technique	
Travail pratique individuel (TPI)	
Hoarau Nicolas	
09.06.2020	
Table des matières	
Tables des versions	
Introduction	
Organisation	
Livrables	
Matériels et logiciels nécessaires	
Description de l'application	
Gestion de projet	
Méthodologie	
Backlog	
Planification	
Planning prévisionnel	
Planning effectif	
Généralités concernant l'implémentation	
Base de données	
Dictionnaire de données	
Structure	
Classes	
Librairies et outils externes	
Font Awesome	
Google Fonts	
JQuery	
Ajax	
SweetAlert2	
unDraw	
Adobe Illustrator	
Composer	
PHPMailer	
Git	
Analyse des fonctionnalités majeures	
Création d'un événement public	
Création d'un événement privé	
Inscription à un événement public	
Inscription à un événement privé	
Plan de tests et tests	
Périmètre des tests	
Environnement	
Scénarios	
Suivis journaliers des tests	
Conclusion	
Difficultés rencontrées	
Variantes de solutions et choix	
Améliorations possibles	
Bilan personnel	

Annexes

Glossaire

Sources

Résumé TPI

Énoncé

Tables des versions

Version	Date	Auteur	Changement apporté
1.0	25.05.2020	Hoarau Nicolas nicolas.hr@eduge.ch	Création du document

Introduction

Ce document a pour but d'expliquer le fonctionnement du projet ainsi que le déroulement de sa conception. Ce projet a été réalisé dans le cadre du *Travail Pratique Individuel* lors de la session de mai/juin 2020 afin de valider ma formation d'informaticien. **WE GO** est une application WEB qui permet aux utilisateurs enregistrés de créer et de s'inscrire à des événements ouverts ou privés.

Organisation

Élève	Maître d'apprentissage	Experts
Nicolas Hoarau nicolas.hr@eduge.ch	Katia Mota Stroppolo katia.motastroppolo@edu.ge.ch	Olivier Maillefer olivier.maillefer@ville-ge.ch Gilles Mouchet gilles.mouchet@ville-ge.ch

Livrables

Pour les experts et la formatrice :

- La documentation technique du projet avec le code source à l'intérieur
- La documentation utilisateur
- Le journal de bord

Pour la formatrice :

- L'accès au git

Matériels et logiciels nécessaires

- Ordinateur personnel avec Windows 10
- Un serveur WEB avec un système de gestion de base de données
 - Apache 2.4.41 (Win64)
 - PHP 7.4.0
 - MySQL 8.0.18
 - MySQL Workbench 8.0
- Visual Studio Code 1.45.1

Description de l'application

WE GO est une application WEB qui permet aux utilisateurs enregistrés de créer et de s'inscrire à des événements de type ouverts ou privés

L'application prévoit 3 types d'utilisateurs: l'anonyme, le membre et l'administrateur.

Les fonctionnalités et droits à implémenter sont :

Un utilisateur anonyme a le droit de

- Afficher une page accueil présentant
 - Le nom et une image des événements ouverts à venir(heure du chargement de la page+1 heure)
 - Si l'événement a une image, l'image s'affiche. Sinon, une image par défaut s'affiche.
 - Un descriptif court du but et du fonctionnement du site
 - Une zone lui permettant de créer un compte
 - Une zone de connexion

Contraintes et règles de gestion

- Seuls les événements de type «ouvert»et à venir sont affichés sur la page d'accueil du site

Un utilisateur authentifié a tous les droits d'un utilisateur anonyme et en plus le droit de

- Se déconnecter et revenir à la page d'accueil
- Afficher une page personnelle permettant de gérer son propre profil (lire, modifier, supprimer).

Un profil contient les informations suivantes:

- un pseudo(obligatoire, unique au sein de l'application)
- un prénom(obligatoire)
- un nom(peut être vide)
- un email(obligatoire, unique au sein de l'application)
- un mot de passe (obligatoire)
- un numéro de téléphone (peut être vide)

Contraintes et règles de gestion:

- Lors de la suppression d'un profil
 - Il est impossible de supprimer un profil si celui-ci est l'organisateur d'un événement à venir où des personnes sont inscrites (la gestion des désinscriptions groupées d'autres participants que l'organisateur lui-même est hors périmètre de cette application).
 - La suppression d'un profil entraîne la suppression des inscriptions de l'utilisateur.
 - Le mot de passe doit contenir au moins 9 caractères, dont au moins 1 chiffre et 1 caractère spécial
- Afficher une page lui permettant de créer un événement. Un événement contient les informations suivantes:
 - nom(obligatoire)
 - un court descriptif(obligatoire)
 - une Date&Time de début(obligatoire)
 - une Date&Time de fin(obligatoire)
 - une Date&Time de création(obligatoire, date système)
 - un lieu(obligatoire)
 - un type: privé ou ouvert(obligatoire)

- Un nombre maximum de participants (obligatoire)
 - une image
 - l'image n'est pas obligatoire.
 - Si l'organisateur ne fournit pas d'image, une image par défaut lui est attribuée

Contraintes et règles de gestion:

- Seul l'organisateur de l'événement peut le modifier ou le supprimer
 - Un événement ne peut pas être supprimé s'il contient des inscrits
 - la gestion des désinscriptions d'autres participants que l'organisateur lui-même est fait hors périmètre de l'application.
 - Seuls les événements de type «ouvert» sont affichés dans la page d'accueil du site
 - L'organisateur de l'événement est inscrit d'office à l'événement.
 - Les événements de type privé
 - Lors de la création d'un événement privé, l'utilisateur (l'hôte), doit remplir une liste des personnes invitées parmi les utilisateurs de l'application
 - Seuls les utilisateurs faisant partie de cette liste d'invités peuvent s'inscrire à un événement privé
- Afficher une page lui permettant de visualiser les détails d'un événement qu'il organise et de les modifier ou de supprimer l'événement affiché. Cette page affiche les détails suivants:
 - le nom
 - le descriptif
 - la Date&Time de début
 - la Date&Time de fin
 - la Date&Time de création
 - le lieu
 - le type: ouvert ou privé
 - le nombre maximum de participants
 - l'image de l'événement
 - la liste d'utilisateurs inscrits

Contraintes et règles de gestion

- Seul l'organisateur de l'événement peut le modifier ou le supprimer
 - Un événement ne peut pas être supprimé s'il contient des inscrits
 - la gestion des désinscriptions d'autres participants que l'organisateur lui-même est fait hors périmètre de l'application.
- Afficher une page lui permettant de gérer ses inscriptions à un événement.

Cette page contient trois zones d'affichage:

- Une zone contenant tous les événements ouverts à venir
- Une zone contenant tous les événements privés à venir pour lesquels l'utilisateur a été invité (cf. fait partie de la liste d'invités de l'événement privé)
- Une zone contenant la liste de tous les événements pour lesquels l'utilisateur est inscrit
 - un bouton sur chaque événement de cette zone permet à l'utilisateur de se désinscrire.
 - Suite à la désinscription, l'événement n'est plus affiché dans cette zone
- Lorsqu'un utilisateur s'inscrit à un événement, une confirmation par mail lui est envoyée au mail enregistré dans la base de données.
- Un utilisateur inscrit à un événement reçoit un mail de rappel un jour avant.

- Seul un utilisateur connecté sur le site peut s'inscrire aux événements.
- Les inscriptions ne peuvent être enregistrées que jusqu'au nombre maximum saisi par l'hôte lors de la création de l'événement.
 - Lorsque ce nombre d'inscriptions est atteint, l'événement affiche l'information «complet» et plus aucune inscription ne peut être effectuée
- Un utilisateur ne peut s'inscrire à un événement de type «privé» que s'il fait partie de la liste des invités créée par l'organisateur lors de la création de l'événement.

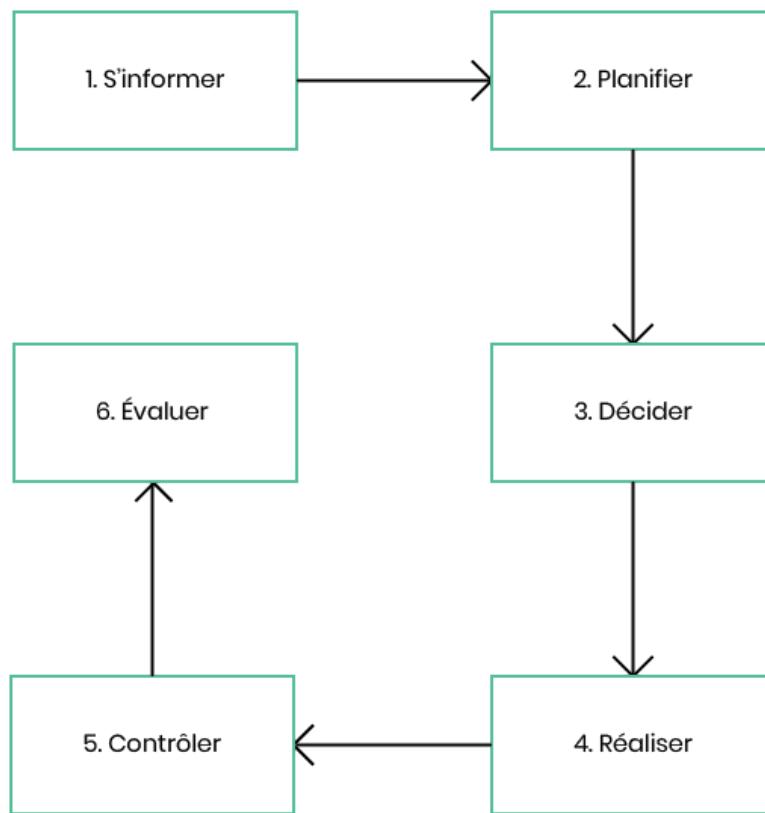
Un administrateur a, en plus des droits ci-dessus, les droits de :

- Gérer la base de données au travers de l'interface d'administration de la base donnée avec un rôle administrateur (pas de fonctionnalité d'administration à implémenter dans l'application web, le rôle BD suffit)
- Maintenir le code source de l'application (pas de fonctionnalité de développement à implémenter dans l'application web, les informations techniques permettant de modifier le code source suffisent)

Gestion de projet

Méthodologie

Afin de planifier mon TPI, j'ai choisi d'utiliser la **méthode en 6 étapes** que j'ai vue lors des cours de ma formation.



1. S'informer

C'est lors de cette étape que l'on prend connaissance de notre projet et que l'on analyse le cahier des charges afin de bien comprendre chaque fonctionnalité.

2. Planifier

Durant cette étape, un planning est créé afin d'avoir un ordre pour effectuer les tâches à faire. Pour ce projet, les tâches ont été rédigées sous le format d'*user stories*. Les *user stories* se rédige sous le point de vue de l'utilisateur final.

Pour chaque *story* du projet, la méthode MoSCoW a été utilisée afin de définir un ordre de priorité afin d'aider à identifier les tâches importantes. Les niveaux de priorités sont **Must** : Indispensable, **Should** : Critique.

Exemple de story :

Hoarau Nicolas		WEGO	09.06.2020
Nom	S.1 Inscription		
Description	En tant qu'utilisateur non inscrit, je peux m'inscrire au site.		
Test(s)	Les tests 1.1 à 1.3		
Priorité	III: Indispensable		

Toutes les *stories* ont été gardées dans un *product backlog*. Cette manière de procéder est inspirée de la méthodologie *Scrum*.

Un *diagramme de Gantt* a été créé afin de visualiser plus facilement l'avancée du projet et permet de voir la différence entre le planning prévisionnel et le planning effectif.

3. Décider

C'est lors de cette étape que les choix pour l'application sont faits. Durant cette étape, le pour et le contre sur le fait d'utiliser une technologie sont pesés.

4. Réaliser

C'est à cette étape que l'implémentation des fonctionnalités et la documentation sont faites.

5. Contrôler

Après avoir développé une nouvelle fonctionnalité, celle-ci était testée plusieurs fois et de différentes manières afin d'être sûr que dans aucun cas on puisse faire de manipulation imprévue par le système. Chaque test a été documenté dans un rapport de test. Aussi à chaque fonctionnalité ajoutée, les anciennes fonctionnalités ont aussi été testées afin d'être sûr de ne pas avoir régressé dans le projet. Une fois le projet terminé toutes les fonctionnalités de l'application ont été testées à nouveau.

6. Évaluer

C'est lors de cette dernière étape que le bilan de la journée ou celui l'élaboration du projet est effectuée. Dans cette étape, on cherche aussi ce que l'on pourrait améliorer dans l'application.

Backlog

Nom	S.1 Inscription (A15)
Description	<p>En tant qu'utilisateur anonyme, je peux m'inscrire au site. Pour m'inscrire je dois entrer les informations suivantes :</p> <ul style="list-style-type: none"> _ un pseudo (obligatoire, unique) _ un prénom (obligatoire) _ un nom (pas obligatoire) _ une adresse email (obligatoire, unique) _ un mot de passe (obligatoire) _ un numéro de téléphone (pas obligatoire)
Test(s)	Les tests 1.1 à 1.6
Priorité	¶¶: Indispensable

Nom	S.2 Connexion (A15)
Description	En tant qu'utilisateur inscrit, je peux me connecter au site.
Test(s)	Les tests 2.1 à 2.3
Priorité	¶¶: Indispensable

Nom	S.3 Crédation d'un événement (A17+A18)
Description	<p>En tant qu'utilisateur connecté, je peux créer un événement qui contient :</p> <ul style="list-style-type: none"> _ un nom (obligatoire) _ une description (obligatoire) _ un lieu (obligatoire) _ un nombre maximal de participants (obligatoire) _ une image (non obligatoire) _ une date de début (obligatoire) _ une date de fin (obligatoire) _ la date de création <p>* choisir si l'événement public ou privé (obligatoire)</p>
Test(s)	Les tests 3.1 à 3.6
Priorité	¶¶: Indispensable

Nom	S.4 Crédation de la liste des invités pour un événement privé (A18)
Description	En tant qu'utilisateur connecté, je peux créer une liste de participants pour mon événement privé.
Test(s)	Les tests 4.1 à 4.3
Priorité	¶¶: Indispensable

Nom	S.5 Inscription à un événement ouvert (A19)	WEGO	09.06.2020
Description	En tant qu'utilisateur connecté, je peux m'inscrire à un événement ouvert pour autant qu'il reste des places.		
Test(s)	Les tests 5.1 à 5.3		
Priorité	¶¶: Indispensable		

Nom	S.6 Inscription à un événement privé (A19)
Description	En tant qu'utilisateur connecté, je peux m'inscrire à un événement privé pour autant que je sois invité.
Test(s)	Les tests 6.1 à 6.3
Priorité	¶¶: Indispensable

Nom	S.7 Validation participation à un événement (A19)
Description	En tant qu'utilisateur connecté et participant à un événement ouvert ou privé, je reçois un mail de confirmation pour ma participation.
Test(s)	Les tests 7.1
Priorité	¶¶: Indispensable

Nom	S.8 Rappel participation à un événement (A19)
Description	En tant qu'utilisateur connecté et participant à un événement, je reçois un mail de rappel la veille de l'événement auquel je participe.
Test(s)	Les tests 8.1
Priorité	¶¶: Indispensable

Nom	S.9 Invitation à un événement privé (A20)
Description	En tant qu'utilisateur connecté et invité à un événement privé, je vois l'événement dans la zone contenant tous les événements privés à venir auxquels je suis invité.
Test(s)	Les tests 9.1 à 9.2
Priorité	¶¶: Indispensable

Nom	S.10 Modification d'un événement
Description	<p>En tant qu'utilisateur connecté, je peux modifier un événement que j'ai créé :</p> <ul style="list-style-type: none"> _ son nom _ sa description _ son lieu _ son nombre maximal de participants _ son image (non obligatoire) _ sa date de début * sa date de fin
Test(s)	Les tests 10.1 à 10.3
Priorité	¶: Critique

Nom	S.11 Suppression d'un événement
Description	En tant qu'utilisateur connecté, je peux supprimer les événements que j'ai créés s'il n'y a aucun participant.
Test(s)	Les tests 11.1
Priorité	¶: Critique

Nom	S.12 Affichage des événements
Description	En tant qu'utilisateur connecté, je peux voir sur la page d'accueil tous les événements qui ont été créés.
Test(s)	Le test 7.1 à 7.3
Priorité	¶: Critique

Nom	S.13 À propos de l'événement
Description	En tant qu'utilisateur connecté, je peux voir le nom, le pseudo du créateur, le nombre maximal de personne invitée, le lieu et la date de début ainsi que celle de fin de l'événement.
Test(s)	Le test 8.1
Priorité	¶: Critique

Nom	S.14 Affichage de la page profil
Description	En tant qu'utilisateur connecté, je peux avoir accès à ma page profil avec les informations de mon compte.
Test(s)	Le test 14.1
Priorité	¶: Critique

Hoarau Nicolas	WEGO	09.06.2020
Nom	S.15 Modification de la page profil (A16)	
Description	<p>En tant qu'utilisateur connecté, je peux modifier :</p> <ul style="list-style-type: none"> _ mon pseudo _ mon prénom _ mon nom _ mon adresse email _ mon mot de passe _ mon numéro de téléphone <p>de ma page profile.</p>	
Test(s)	Les tests 15.1 à 15.6	
Priorité	¶: Indispensable	

Nom	S.16 Suppression du profil (A16)
Description	En tant qu'utilisateur connecté, je peux supprimer mon profil que s'il n'y a aucun participant aux événements que j'ai créé.
Test(s)	Les tests 16.1 à 16.6
Priorité	¶: Indispensable

Nom	S.17 Gestion des événements
Description	En tant qu'utilisateur connecté, j'ai accès à une page sur laquelle je peux voir les événements ouverts à venir et les événements privés auxquels je suis invité afin de pouvoir m'y inscrire. Et je peux aussi voir les événements auxquels je participe, afin de me désinscrire.
Test(s)	Le test 17.1
Priorité	¶: Critique

Nom	S.18 Gestion des accès (A14)
Description	En tant qu'utilisateur non connecté, je peux voir que les événements publics et la barre de navigation m'affiche que les liens pour aller à l'accueil, voir le descriptif du site, me connecter et m'inscrire.
Test(s)	Le test 6.1
Priorité	¶: Indispensable

Hoarau Nicolas	WEGO	09.06.2020
Nom	S.19 Création du dépôt Git	
Description	En tant que développeur, je peux utiliser Git sur le code source du projet. Je peux également accéder au dépôt sur GitHub.	
Critère de validation	Un dépôt Git est configuré dans le dossier du projet. Le code source est disponible depuis le site https://github.com à l'adresse indiquée dans la documentation.	
Priorité	: Indispensable	

Nom	S.20 Implémentation de la base de données
Description	En tant que développeur, je peux utiliser une base de données MySQL qui respecte le schéma donné dans l'énoncé.
Critère de validation	La base de données WEGO a été créé et contient les tables user, inscription, invites, evenement et sont utilisables.
Priorité	: Indispensable

Nom	S.21 Utilisation de Composer
Description	En tant que développeur, je peux utiliser Composer afin d'installer des dépendances PHP au projet.
Critère de validation	Le projet contient un fichier <code>composer.json</code> qui permet à Composer de fonctionner.
Priorité	: Indispensable

Planification

La couleur **#3993fa** a été utilisé afin de marquer les jalons du projet.

- Le premier jalon est pour la fin de l'implémentation de la base de données.
- Le deuxième jalon est pour la fin du développement en lien avec les événements.
- Le troisième jalon est pour la fin de la phase d'implémentation.
- Le quatrième jalon est pour la fin du projet.

Planning prévisionnel

Jour	J1 25.05.2020	J2 26.05.2020	J3 27.05.2020	J4 28.05.2020	J5 29.05.2020	J6 02.06.2020	J7 03.06.2020	J8 04.06.2020	J9 05.06.2020	J10 08.06.2020	J11 09.06.2020
Lecture de l'énoncé											
Rédaction du backlog											
Rédaction des scénarios											
Planification des tâches											
S.19 Création du dépôt Git											
S.21 Utilisation de Composer											
S.20 Mise en place de la base de données											
S.18 Gestion des accès											
S1. Inscription											
S2. Connexion											
S.3 Création d'un événement											
S.4 Création de la liste des invités pour un événement privé											
S.12 Affichage des événements											

Hoarau Nicolas		WEGO										09.06.2020				
Jour		J1 25.05.2020	J2 26.05.2020	J3 27.05.2020		J4 28.05.2020		J5 29.05.2020	J6 02.06.2020	J7 03.06.2020		J8 04.06.2020	J9 05.06.2020	J10 08.06.2020		J11 09.06.2020
S.13 À propos de l'événement																
S.5 Inscription à un événement ouvert																
S.6 Inscription à un événement privé																
S.7 Validation participation à un événement																
S.8 Rappel participation à un événement																
S.11 Suppression d'un événement																
S.10 Modification d'un événement																
S.14 Affichage de la page de profil																
S.16 Suppression du profil																
S.15 Modification de la page profil																
S.17 Gestion des événements																
S.9 Invitation à un événement privé																
Tests de toute l'application et correction des bugs																
Rédaction de la documentation																
Rédaction du journal de bord																

Planning effectif

Jour	J1 25.05.2020	J2 26.05.2020	J3 27.05.2020	J4 28.05.2020	J5 29.05.2020	J6 02.06.2020	J7 03.06.2020	J8 04.06.2020	J9 05.06.2020	J10 08.06.2020	J11 09.06.2020
Lecture de l'énoncé											
Rédaction du backlog											
Planification des tâches											
Rédaction des scénarios											
S.21 Utilisation de Composer											
S.20 Mise en place de la base de données											
S.19 Création du dépôt Git											
S1. Inscription											
S.2 Connexion											
S.18 Gestion des accès											
S.3 Création d'un événement											
S.4 Création de la liste des invités pour un événement privé											
S.12 Affichage des événements											
S.5 Inscription à un événement ouvert											
S.6 Inscription à un événement privé											

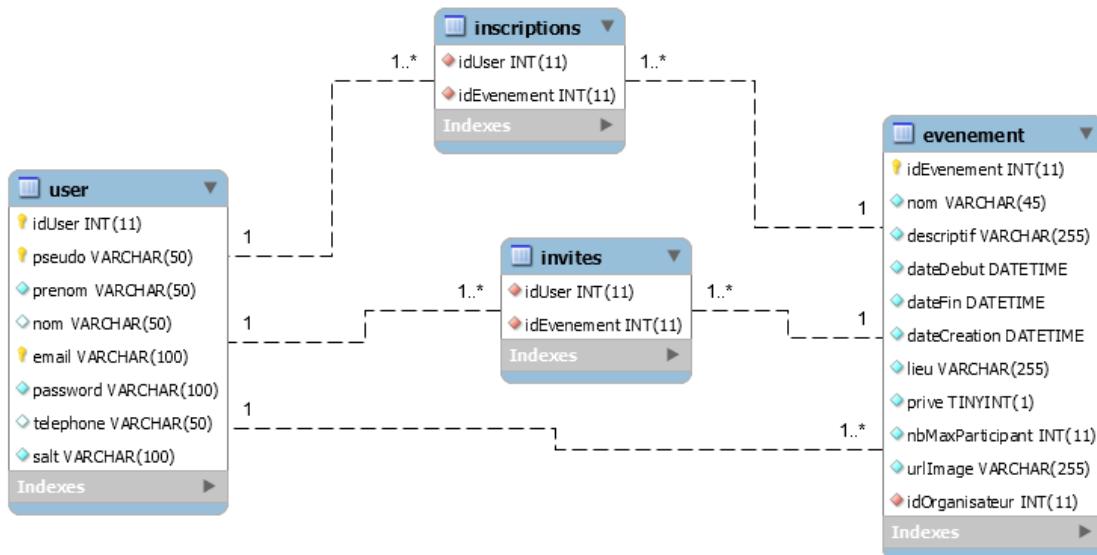
Jour	J1 25.05.2020	J2 26.05.2020	J3 27.05.2020	J4 28.05.2020	J5 29.05.2020	J6 02.06.2020	J7 03.06.2020	J8 04.06.2020	J9 05.06.2020	J10 08.06.2020	J11 09.06.2020
S.17 Gestion des événements						Green					
S.9 Invitation à un événement privé					Blue						
S.13 À propos de l'événement						Green	Green				
S.7 Validation participation à un événement						Green	Green				
S.14 Affichage de la page de profil						Green	Green				
S.8 Rappel participation à un événement						Green	Green				
S.11 Suppression d'un événement						Green	Green				
S.10 Modification d'un événement						Green	Green	Green	Blue		
S.16 Suppression du profil						Green	Green	Green	Green		
S.15 Modification de la page profil						Green	Green	Green	Green		
Tests de toute l'application et correction des bugs						Green	Green	Green	Green	Green	
Rédaction de la documentation	Green	Green									
Rédaction du journal de bord	Green	Green									

Généralités concernant l'implémentation

Base de données

WEGO utilise une base de données MySQL afin de stocker les utilisateurs ainsi que les événements, les événements auxquels ils participent ou sont invités. La version utilisée est MySQL 8.0.18. J'ai utilisé InnoDB comme moteur de stockage et *utf8_bin* comme interclassement.

Le modèle utilisé est différent de celui dans l'énoncé, car j'ai rajouté la colonne **salt** dans la table *user* afin d'avoir une meilleure sécurité pour les mots de passe.



Le modèle de la base données modifiée

Dictionnaire de données

user

Colonne	Type	Null	Défaut
idUser	int(11)	Non	
pseudo	varchar(50)	Non	
prenom	varchar(50)	Non	
nom	varchar(50)	Oui	NULL
email	varchar(100)	Non	
password	varchar(100)	Non	
telephone	varchar(50)	Oui	NULL
salt	varchar(100)	Non	

evenement

Colonne	Type	Null	Défaut
idEvenement	int(11)	Non	
nom	varchar(50)	Non	
descriptif	varchar(255)	Non	
dateDebut	datetime	Non	
dateFin	datetime	Non	
dateCreation	datetime	Non	
lieu	varchar(255)	Non	
privé	tinyint(1)	Non	
nbMaxParticipant	int(11)	Non	
urlImage	varchar(255)	Non	
idOrganisateur	int(11)	Non	

inscriptions

Colonne	Type	Null	Défaut
idUser	int(11)	Non	
idEvenement	int(11)	Non	

invites

Colonne	Type	Null	Défaut
idUser	int(11)	Non	
idEvenement	int(11)	Non	

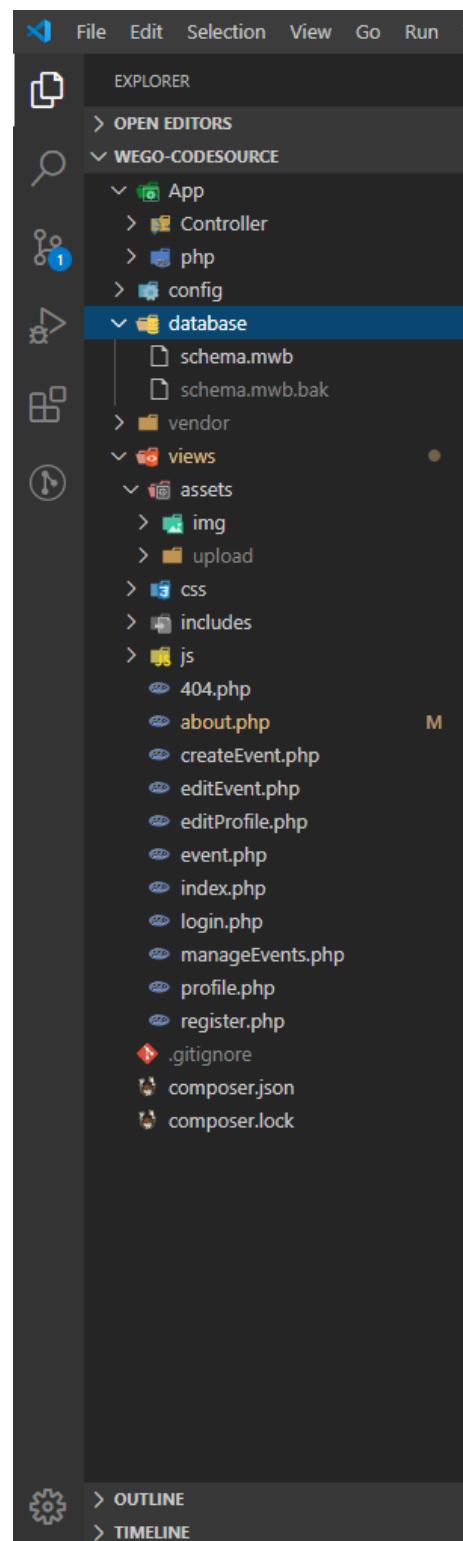
Structure

La structure utilisée pour le développement du projet est l'architecture trois tiers:

- **/App/** Dossier qui contient les dossiers qui font la logique de l'application.
 - **/App/Controller/** Dossier qui contient les classes du projet.
 - **/App/php/** Dossier qui contient tout le traitement des données de l'application
- **/config/** Dossier qui contient le fichier qui contient les constantes de connexion à la base de données et les constantes pour la connexion à l'adresse mail du mailer.
- **/database/** Dossier qui contient le schéma de la base de données.
- **/vendor/** Dossier qui contient toutes les dépendances PHP qui sont géré par Composer.
- **/view/** Dossier qui contient tout l'affichage de l'application. Les fichiers dans ce dossier sont les pages que l'utilisateur peut voir.
- **/view/assets/** Dossier qui contient toutes les images du site
- **/view/assets/img/** Dossier qui contient les images du site.
- **view/assets/upload/** Dossier qui contient les images des utilisateurs choisis pour leur événement.
- **/view/css/** Dossier qui style du style.
- **/view/includes/** Dossier qui contient les fichiers qui sont inclus toutes les pages visibles par l'utilisateur.
- **/view/js/** Dossier qui contient les fichiers JavaScript

Les dossiers en gras sont les plus importants.

Le dossier en italique contient des fichiers générés automatiquement.



Classes

WE GO\App\Controller\DatabaseController.php

Cette classe me permet de déclarer ma base de données MySQL en *Singleton*.

WE GO\public\App\Controller\ExtendedPdo.php

Cette classe me permet de faire des transactions en parallèle.

Librairies et outils externes

Font Awesome

FontAwesome est un outil d'icônes et une police d'écriture



qui est basée sur le CSS, SASS et LESS. Ceci a été créé par

Dave Grandy afin de l'utiliser avec Bootstrap. FontAwesome détient 38% des parts du marché des sites qui utilisent des scripts de polices tiers sur leur plateforme¹. Cet outil a été utilisé pour les icônes.

Google Fonts

Google Fonts est une librairie de police libre de droits. La première version a été lancée en 2010 par le groupe Google. Google Fonts détient 57% des parts du marché des sites qui utilisent des scripts de polices tiers, ce qui les met en première place devant FontAwesome¹. Cette librairie a été utilisée pour les polices qu'elle me met à disposition.

Google Fonts

JQuery

JQuery est une librairie JavaScript qui facilite l'écriture du script. La première version de la librairie date de 2006 par John Resig. Cette librairie a été créée afin d'utiliser Ajax et d'alléger le JavaScript.



Ajax

Ajax (*Asynchronous JavaScript and XML*) permet de faire des mises à jour de la page web sans que celle-ci se recharge. On peut l'utiliser grâce à la librairie JQuery. Cette technologie a été utilisée afin de ne pas avoir à rafraîchir la page WEB à chaque action effectuée.

SweetAlert2

Sweetalert2 est une librairie qui permet d'avoir un meilleur design pour les messages popup. Cette librairie est Limon Monte.

Cette librairie a été utilisée afin d'avoir des alertes avec un joli design.

sweetalert2

unDraw

UnDraw est une librairie illustration libre de droits, qui a été créée par Katerina Limpitsouni en 2017. Cette librairie est utilisée par de grands noms comme Google, Microsoft, Facebook, Harvard Business School. Cette librairie a été utilisée, car il y a une grande diversité d'illustration à disposition et de plus on peut choisir la couleur de notre illustration.

Adobe Illustrator

Adobe Illustrator est un logiciel d'édition d'image vectoriel créé le 19 mars 1987 et développé par Adobe Inc. Ce logiciel a été utilisé afin de créer un schéma de la méthode en 6 étapes.



Composer

Composer est un gestionnaire de dépendances PHP qui a été développé par Nils Adermann et Jordi Boggiano le 1er mars 2012. Il permet d'utiliser des dépendances comme *PHPMailer*, ce qui nous permet de faciliter le développement et d'utiliser facilement des librairies externes à jours.



PHPMailer

PHPMailer est une API PHP qui permet d'envoyer des mails depuis du code PHP qui a été développé par Brent R. Matzelle en 2001.



Git

Git est un logiciel de gestion de versions que j'ai utilisé pendant toute l'élaboration mon TPI afin d'avoir un historique de mon projet. Le code source de mon projet est disponible à l'adresse suivante : <https://github.com/Nicolas-hr/WEGO-CodeSource>.

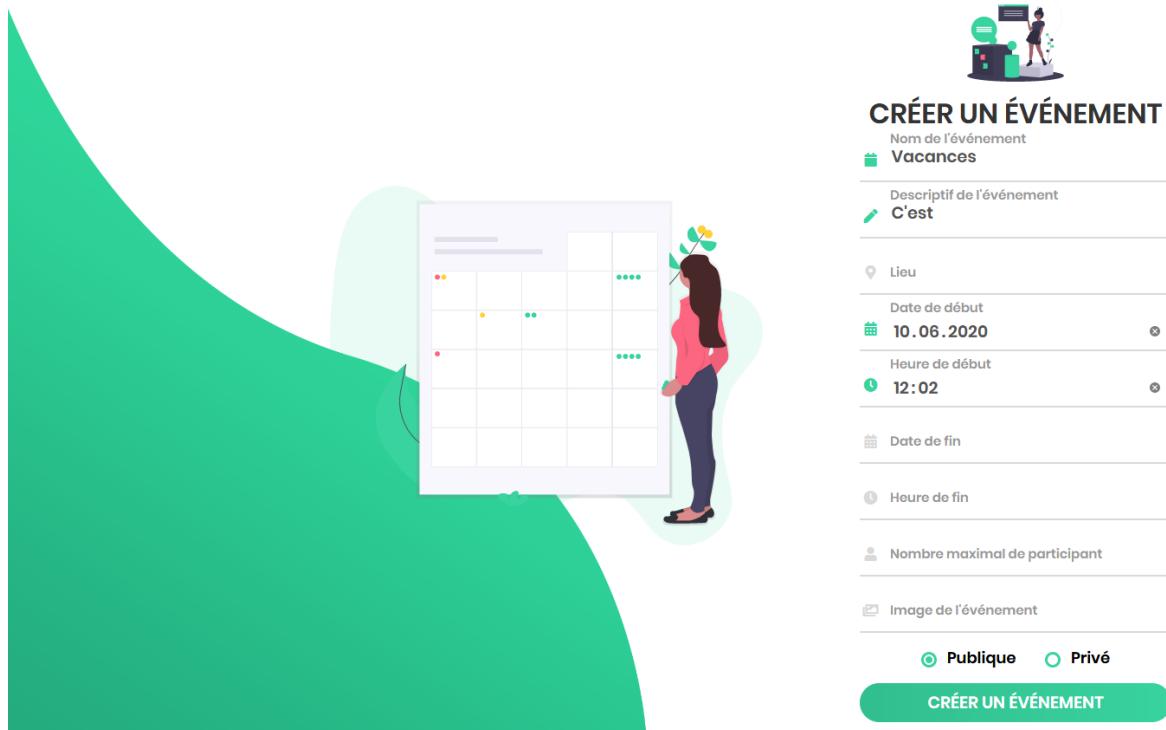


La documentation de mon projet est disponible à l'adresse suivante : <https://github.com/Nicolas-hr/WEGO-Documentation>.

Analyse des fonctionnalités majeures

Création d'un événement public

Un utilisateur authentifié peut créer un événement public qui contient un nom, un court descriptif, un lieu, une date de début, une heure de début, une date de fin, heure de fin, un nombre maximal de participants et une image qui n'est pas obligatoire. Quand l'utilisateur arrive sur la page de création de l'événement il voit ceci:



La création de l'événement fonctionne grâce à 2 pages:

- views/js/**event.js** qui récupère les données du formulaire, lorsqu'on appuis sur le bouton **CRÉER UN ÉVÉNEMENT**, et qui les traite puis les envois via un call ajax.
- App/php/**createEvent.php** qui reçoit les données du call ajax et qui les insèrent dans la base de données.

Chaque erreur possible dans le formulaire est affichée à la réponse du call ajax.

Création d'un événement privé

Un utilisateur authentifié peut créer un événement public qui contient un nom, un court descriptif, un lieu, une date de début, une heure de début, une date de fin, heure de fin, un nombre maximal de participants, une image qui n'est pas obligatoire et une liste d'invités. Quand l'utilisateur arrive sur la page pour créer son événement il atterrit sur la même page affichée précédemment. Mais la création de l'événement est un petit peu différente car il faut gérer la liste des invités. Quand l'utilisateur valide les données du formulaire, une fonction JavaScript asynchrone est appelée :

```
1 | formdata.append('guestlist', await GetGuestList());
```

Et permet d'appeler la modal qui va récupérer les utilisateurs pour les afficher ensuite :

```
// 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><ul>`;

    $.each(data, (key, user) => [
      html += `<li>
        <label>
          <input type="checkbox" value="${user.idUser}" checked="checked">
          <span>${user.icon}</span>
          <span>${user.pseudo}</span>
        </label>
      You, 7 days ago • Récupère les données d'un événement depuis la base
    </li>`);
  ]);

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

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

Et une fois la sélection validée les utilisateurs invités sont retournés dans un tableau :

```
}).then((result) => {
  let modalGuest = [];
  // Récupère les case cochées de la modal
  let pathToLi = Swal.getContent().childNodes[0].childNodes[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;
});
return guests;
});
```

Inscription à un événement public

Lorsqu'un utilisateur s'inscrit à un événement, un call ajax qui envoie les données de l'événement est effectué, lorsque la page php reçoit les données il insère utilisateur dans la table *inscription* puis il envoie un mail et si l'envoie de celui-ci échoue il annule l'insertion de l'utilisateur dans la base de données.

```
try {
    DatabaseController::beginTransaction();

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

    $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();
}
```

Inscription à un événement privé

Lorsqu'un utilisateur veut s'inscrire à un événement privé, il doit être invité à celui-ci et après il peut s'y inscrire (l'inscription se déroule comme montré précédemment).

Plan de tests et tests

Périmètre des tests

Pour WE GO, j'ai mis en place un protocole de test afin qu'un utilisateur utilisant soit Google Chrome, Mozilla Firefox ou Microsoft Edge puisse naviguer convenablement sur le site.

Environnement

Lors des tests du projet, j'ai utilisé les navigateurs :

- Mozilla Firefox 76.0.1 (64bits) sur Windows 10 Éducation, Version 1909
- Google Chrome 81.0.4044.128 (64bits) sur Windows 10 Éducation, Version 1909
- Microsoft Edge 81.0.416.72 (64bits) sur Windows 10 Éducation, Version 1909

Scénarios

Les scénarios des tests sont détaillés afin que n'importe quelle personne puisse les exécuter. Pour rédiger mes scénarios, j'ai utilisé la **syntaxe de Gherkin**.

Nom	1.1 Crédation d'un nouveau compte (toutes les données, données valides)
User Story	S.1 Inscription
Situation	Étant donné que je suis un utilisateur anonyme et que je ne possède pas de compte quand j'arrive sur la page d'accueil du site et que j'ouvre la page d'inscription et que je rentre mon pseudo Nicolas .hr et que je rentre mon prénom Nicolas et que je rentre mon nom Hoarau et que je rentre mon email nicolas .hr@eduge .ch et que je rentre mon mot de passe Super2012 et que je rentre la confirmation de mon mot de passe Super2012 et je rentre mon numéro de téléphone 123 456 78 90 et que je clique sur le bouton INSCRIPTION alors je suis redirigé vers la page Connexion .
Résultat obtenu	Un message de confirmation apparaît au centre de la page avec le texte Le compte a bien été crée. et je suis redirigé sur la page Connexion .
Statut	✓ OK

Nom	1.2 Crédation d'un nouveau compte (données obligatoires, données valides)
User Story	S.1 Inscription
Situation	Étant donné que je suis un utilisateur anonyme et que je ne possède pas de compte quand j'arrive sur la page d'accueil du site et que j'ouvre la page d'inscription et que je rentre mon pseudo Nicolas.hr et que je rentre mon prénom Nicolas et que je rentre mon email nicolas.hr@eduge.ch et que je rentre mon mot de passe Super2012 et que je rentre la confirmation de mon mot de passe Super2012 et que je clique sur le bouton INSCRIPTION alors je suis redirigé vers la page Connexion .
Résultat obtenu	Un message de confirmation apparaît au centre de la page avec le texte Le compte a bien été crée. et je suis redirigé sur la page Connexion .
Statut	✓ OK

Nom	1.3 Crédation d'un nouveau compte (mots de passe différents, données invalides)
User Story	S.1 Inscription
Situation	Étant donné que je suis un utilisateur anonyme et que je ne possède pas de compte quand j'arrive sur la page d'accueil du site et que j'ouvre la page d'inscription et que je rentre mon pseudo Nicolas.hr et que je rentre mon prénom Nicolas et que je rentre mon email nicolas.hr@eduge.ch et que je rentre mon mot de passe Super2012 et que je rentre la confirmation de mon mot de passe test et que je clique sur le bouton INSCRIPTION alors je reste sur la page Inscription
Résultat obtenu	Un message erreur apparaît au centre de la page avec le texte Les deux mots de passe donnés ne sont pas les mêmes. et je reste sur la page Inscription .
Statut	✓ OK

Nom	1.4 Crédation d'un nouveau compte (mots de passe ne correspondant pas à la règle, données invalides)
User Story	S.1 Inscription
Situation	Étant donné que je suis un utilisateur anonyme et que je ne possède pas de compte quand j'arrive sur la page d'accueil du site et que j'ouvre la page d'inscription et que je rentre mon pseudo Nicolas.hr et que je rentre mon prénom Nicolas et que je rentre mon email nicolas.hr@eduge.ch et que je rentre mon mot de passe test et que je rentre la confirmation de mon mot de passe test et que je clique sur le bouton INSCRIPTION alors je reste sur la page Inscription
Résultat obtenu	Un message erreur apparaît au centre de la page avec le texte Le mot de passe doit faire au moins 8 caractères et doit contenir au moins une majuscule et un chiffre. et je reste sur la page Inscription .
Statut	✓ OK

Nom	1.5 Crédation d'un nouveau compte (pseudo déjà utilisé, données invalides)
User Story	S.1 Inscription
Situation	Étant donné que je suis un utilisateur anonyme et que je ne possède pas de compte quand j'arrive sur la page d'accueil du site et que j'ouvre la page d'inscription et que je rentre mon pseudo Nicolas.hr alors qu'il est déjà utilisé sur le site et que je rentre mon prénom Nicolas et que je rentre mon email nicolas@gmail.com et que je rentre mon mot de passe Super2012 et que je rentre la confirmation de mon mot de passe Super2012 et que je clique sur le bouton INSCRIPTION alors je reste sur la page Inscription
Résultat obtenu	Un message erreur apparaît au centre de la page avec le texte Ce pseudo est déjà utilisé. et je reste sur la page Inscription .
Statut	✓ OK

Nom	1.6 Crédation d'un nouveau compte (email déjà utilisé, données invalides)
User Story	S.1 Inscription
Situation	Étant donné que je suis un utilisateur anonyme et que je ne possède pas de compte quand j'arrive sur la page d'accueil du site et que j'ouvre la page d'inscription et que je rentre mon pseudo Nicolas.hr et que je rentre mon prénom Nicolas et que je rentre mon email nicolas.hr@eduge.ch alors que cette adresse est déjà utilisée sur le site et que je rentre mon mot de passe Super2012 et que je rentre la confirmation de mon mot de passe Super2012 et que je clique sur le bouton INSCRIPTION alors je reste sur la page Inscription
Résultat obtenu	Un message erreur apparaît au centre de la page avec le texte Cette adresse mail est déjà utilisée. et je reste sur la page Inscription .
Statut	✓ OK

Nom	2.1 Connexion à son compte (connexion avec adresse mail, données valides)
User Story	S.2 Connexion
Situation	Étant donné que je suis un utilisateur anonyme et que je possède un compte quand j'arrive sur la page d'accueil du site et que j'ouvre la page de connexion et que je rentre mon email nicolas.hr@eduge.ch et que je rentre mon mot de passe Super2012 et que je clique sur le bouton CONNEXION alors je suis redirigé sur la page Accueil .
Résultat obtenu	Un message de confirmation apparaît au centre de la page avec le texte Connexion réussie. et je suis redirigé sur la page Accueil .
Statut	✓ OK

Nom	2.2 Connexion à son compte (connexion avec pseudo, données valides)
User Story	S.2 Connexion
Situation	Étant donné que je suis un utilisateur anonyme et que je possède un compte quand j'arrive sur la page d'accueil du site et que j'ouvre la page de connexion et que je rentre mon pseudo Nicolas.hr et que je rentre mon mot de passe Super2012 et que je clique sur le bouton CONNEXION alors je suis redirigé sur la page Accueil .
Résultat obtenu	Un message de confirmation apparaît au centre de la page avec le texte Connexion réussi. et je suis redirigé sur la page Accueil .
Statut	✓ OK

Nom	2.3 Connexion à son compte (connexion avec adresse mail ou pseudo, données invalides)
User Story	S.2 Connexion
Situation	Étant donné que je suis un utilisateur anonyme et que je possède un compte quand j'arrive sur la page d'accueil du site et que j'ouvre la page de connexion et que je rentre mon email nicolas.hr@eduge.ch ou mon pseudo Nicolas.hr et que je rentre mon mot de passe Spuer2012 au lieu de Super2012 et que je clique sur le bouton CONNEXION alors je reste sur la page Connexion .
Résultat obtenu	Un message de confirmation apparaît au centre de la page avec le texte L'identifiant de connexion ou le mot de passe est faux. et je reste sur la page Connexion .
Statut	✓ OK

Nom	3.1 Crédation d'un événement (événement public, données valides)
User Story	S.3 Crédation d'un événement
Situation	<p>Etant donné que je suis un utilisateur connecté et que l'événement que je veux crée n'existe pas encore quand j'arrive sur la page de création d'événements et que je rentre le nom de l'événement Rentrée techniciens et que je rentre un descriptif C'est le jour de la rentrée pour les technicien ES 2020/2021. et que je rentre la localisation 10, Ch. Gérard de Ternier, 1213 Petit-Lancy et que je rentre une date de début 24.08.2020 et que rentre 9:00 en heure de début et que je rentre une date de fin 24.08.2020 et que je choisis 16:15 en heure de fin et que je choisis de mettre l'événement en public et que je choisis un maximum de 12 participants et que je choisis une image qu'il y a sur mon ordinateur et que je clique sur le bouton CRÉER UN ÉVÉNEMENT alors mon événement se créé avec les informations données et je suis redirigé sur la page Accueil.</p>
Résultat obtenu	Un message de confirmation apparaît au centre de la page avec le texte L'événement "Rentrée techniciens" a bien été créé. et je suis redirigé sur la page sur la page Accueil.
Statut	✓ OK

Nom	3.2 Crédation d'un événement (événement public sans image, données valides)
User Story	S.3 Crédation d'un événement
Situation	<p>Etant donné que je suis un utilisateur connecté et que l'événement que je veux crée n'existe pas encore quand j'arrive sur la page de création d'événements et que je rentre le nom de l'événement Rentrée techniciens et que je rentre un descriptif C'est le jour de la rentrée pour les technicien ES 2020/2021. et que je rentre la localisation 10, Ch. Gérard de Ternier, 1213 Petit-Lancy et que je rentre une date de début 24.08.2020 et que rentre 9:00 en heure de début et que je rentre une date de fin 24.08.2020 et que je choisis 16:15 en heure de fin et que je choisis de mettre l'événement en public et que je choisis un maximum de 12 invités et que je ne choisis pas d'image et que je clique sur le bouton CRÉER UN ÉVÉNEMENT alors mon événement est créé avec une image par défaut et je suis redirigé sur la page Accueil.</p>
Résultat obtenu	Un message de confirmation apparaît au centre de la page avec le texte L'événement "Rentrée techniciens" a bien été créé. et je suis redirigé sur la page sur la page Accueil.
Statut	✓ OK

Hoarau Nicolas	WEGO	09.06.2020
Nom	3.3 Création d'un événement privé (données valides)	
User Story	S.3 Création d'un événement + S.4 Création de la liste des invités pour un événement privé	
Situation	<p>Etant donné que je suis un utilisateur connecté <i>et que l'événement que je veux créer n'existe pas encore</i> quand j'arrive sur la page de création d'événements <i>et que je rentre le nom de l'événement Mon anniversaire et que je rentre un descriptif C'est le groupe pour mon anniversaire. et que je rentre la localisation Parc La Grange et que je rentre une date de début 01.08.2020 et que je rentre 14:00 en heure de début et que je rentre une date de fin 02.08.2020 et que je rentre 02:00 en date de fin et que je choisis de mettre l'événement en privé et que je choisis un maximum de 3 invités et que je choisis une image et que je clique sur le bouton CRÉER UN ÉVÉNEMENT et une liste des utilisateurs apparaît et me permet de choisir les 3 invités de mon événement et je choisis mes invités Jean, Albert et John et je clique sur le bouton INVITER alors mon événement est créé avec l'image choisie et je suis redirigé sur la page Accueil.</i></p>	
Résultat obtenu	Un message de confirmation apparaît au centre de la page avec le texte L'événement "Mon anniversaire" a bien été créé. et je suis redirigé sur la page Accueil .	
Statut	✓ OK	

Nom	3.4 Création d'un événement privé sans image (données valides)
User Story	S.3 Création d'un événement + S.4
Situation	<p>Etant donné que je suis un utilisateur connecté <i>et que l'événement que je veux créer n'existe pas encore</i> quand j'arrive sur la page de création d'événements <i>et que je rentre le nom de l'événement Mon anniversaire et que je rentre un descriptif C'est le groupe pour mon anniversaire. et que je rentre la localisation Parc La Grange et que je rentre une date de début 01.08.2020 et que je rentre 14:00 en heure de début et que je rentre une date de fin 02.08.2020 et que je rentre 02:00 en date de fin et que je choisis de mettre l'événement en privé et que je choisis un maximum de 3 invités et que je ne choisis pas d'image et que je clique sur le bouton CRÉER UN ÉVÉNEMENT et une liste des utilisateurs apparaît et me permet de choisir les 3 invités de mon événement et je choisis mes invités Jean, Albert et John et je clique sur le bouton INVITER alors mon événement est créé avec une image par défaut et je suis redirigé sur la page Accueil.</i></p>
Résultat obtenu	Un message de confirmation apparaît au centre de la page avec le texte L'événement "Mon anniversaire" a bien été créé. et je suis redirigé sur la page Accueil .
Statut	✓ OK

Nom	3.5 Création d'un événement (date, données invalides)
User Story	S.3 Création d'un événement
Situation	<p>Etant donné que je suis un utilisateur connecté et que l'événement que je veux crée n'existe pas encore quand j'arrive sur la page de création d'événements et que je rentre le nom de l'événement Rentrée techniciens et que je rentre un descriptif C'est le jour de la rentrée pour les technicien ES 2020/2021. et que je rentre la localisation 10,Ch. Gérard de Ternier, 1213 Petit-Lancy et que je rentre une date de début 24.08.2020 et que rentre 9:00 en heure de début et que je rentre une date de fin 09.05.2020 et que je choisis 16:15 en heure de fin et que je choisis de mettre l'événement en public et que je choisis un maximum de 12 participants et que je ne choisis pas d'image et que je clique sur le bouton CRÉER UN ÉVÉNEMENT alors je reste sur la page de création d'événements.</p>
Résultat obtenu	Un message d'erreur apparaît au centre de la page avec le texte L'événement ne peut pas se terminer avant s'avoir commencé. et je reste sur la page Créeer un événement.
Statut	✓ OK

Nom	3.6 Création d'un événement (date, données invalides)
User Story	S.3 Création d'un événement
Situation	<p>Etant donné que je suis un utilisateur connecté et que l'événement que je veux crée n'existe pas encore quand j'arrive sur la page de création d'événements et que je rentre le nom de l'événement Rentrée techniciens et que je rentre un descriptif C'est le jour de la rentrée pour les technicien ES 2020/2021. et que je rentre la localisation 10,Ch. Gérard de Ternier, 1213 Petit-Lancy et que je rentre une date de début 25.04.2020 et que rentre 9:00 en heure de début et que je rentre une date de fin 24.08.2020 et que je choisis 16:15 en heure de fin et que je choisis de mettre l'événement en public et que je choisis un maximum de 12 participants et que je ne choisis pas d'image et que je clique sur le bouton CRÉER UN ÉVÉNEMENT alors je reste sur la page de création d'événements.</p>
Résultat obtenu	Un message d'erreur apparaît au centre de la page avec le texte L'événement ne peut pas commencer avant aujourd'hui. et je reste sur la page Créeer un événement.
Statut	✓ OK

Nom	4.1 Inscription à un événement ouvert (situation valides)
User Story	S.5 Inscription à un événement ouvert
Situation	Etant donné que je suis un utilisateur connecté <i>et que l'événement Rentrée techniciens m'intéresse et n'est pas encore passé quand j'arrive sur la page Accueil et que je vois l'événement Rentrée techniciens et que je clique sur le bouton S'INSCRIRE alors un message de validation apparaît et la page est actualisé.</i>
Résultat obtenu	Un message de confirmation apparaît au centre de la page avec le texte Vous êtes bien inscrit à l'événement "Rentrée techniciens" et je suis redirigé sur la page Accueil .
Statut	✓ OK

Nom	4.2 Inscription à un événement ouvert depuis la page Gérer mes événements (situation valides)
User Story	S.5 Inscription à un événement ouvert
Situation	Etant donné que je suis un utilisateur connecté <i>et que l'événement Rentrée techniciens m'intéresse et n'est pas encore passé quand j'arrive sur la page Gérer mes événements et que je vois l'événement Rentrée techniciens et que je clique sur le bouton S'INSCRIRE alors un message de validation apparaît et la page est actualisé.</i>
Résultat obtenu	Un message de confirmation apparaît au centre de la page avec le texte Vous êtes bien inscrit à l'événement "Rentrée techniciens" et je suis redirigé sur la page Accueil .
Statut	✓ OK

Nom	4.3 Inscription à un événement ouvert depuis la page d'information de l'événement (situation valides)
User Story	S.5 Inscription à un événement ouvert
Situation	Etant donné que je suis un utilisateur connecté <i>et que l'événement Rentrée techniciens m'intéresse et n'est pas encore passé et que je clique sur celui-ci pour avoir plus d'informations quand j'arrive sur la page d'informations de l'événement et que je clique sur le bouton S'INSCRIRE alors un message de validation apparaît et la page est actualisé.</i>
Résultat obtenu	Un message de confirmation apparaît au centre de la page avec le texte Vous êtes bien inscrit à l'événement "Rentrée techniciens" et je suis redirigé sur la page Accueil .
Statut	✓ OK

Nom	4.4 Inscription à un événement ouvert (situation invalides)
User Story	S.5 Inscription à un événement ouvert
Situation	Etant donné que je suis un utilisateur anonyme <i>et que l'événement Rentrée techniciens m'intéresse et n'est pas encore passé</i> quand j'arrive sur la page de l'événement Rentrée techniciens il n'y a pas de bouton pour m'inscrire à l'événement alors je ne peux pas m'inscrire à l'événement.
Résultat obtenu	Un message d'information apparaît en haut à droite avec le texte Vous ne pouvez pas participer à l'événement sans être connecté <i>et je reste sur la page de l'événement.</i>
Statut	✓ OK

Nom	5.1 Inscription à un événement privé (situation valides)
User Story	S.5 Inscription à un événement privé
Situation	Etant donné que je suis un utilisateur anonyme <i>et que l'événement Mon anniversaire m'intéresse et n'est pas encore passé et que je suis invité à l'événement</i> quand je vais sur la page Gérer mes événements <i>et que je vois l'événement Mon anniversaire dans la zone Événements auxquels vous êtes invités et je clique sur le bouton S'INSCRIRE alors un message de validation apparaît et la page est actualisé.</i>
Résultat obtenu	Un message d'information apparaît en haut à droite avec le texte Vous êtes bien inscrit à l'événement "Mon anniversaire" <i>et je reste sur la page de l'événement.</i>
Statut	✓ OK

Nom	6.1 Gestion des accès (situation valides)
User Story	S.18 Gestion des accès
Situation	Etant donné que je suis un utilisateur anonyme quand que je visite le site alors je peux voir les événements publics qui sont à venir <i>et je peux avoir la possibilité de me connecter et la possibilité de m'inscrire.</i>
Résultat obtenu	Je peux voir les événements publics à venir ainsi que leur description <i>et dans la barre de navigation je il y a les liens pour la page Connexion, la page Inscription et la page qui résume le but du site.</i>
Statut	✓ OK

Nom	7.1 Affichage des événements sur la page Accueil (situation valides)
User Story	S.12 Affichage des événements
Situation	Etant donné que je suis un utilisateur anonyme quand je suis sur la page Accueil alors je peux voir les événements publics qui sont à venir.
Résultat obtenu	Je suis sur la page Accueil <i>et</i> je peux voir les événements publics qui sont à venir.
Statut	✓ OK

Nom	7.2 Affichage des événements sur la page Accueil (situation valides)
User Story	S.12 Affichage des événements
Situation	Etant donné que je suis un utilisateur connecté quand je suis sur la page Accueil alors je peux voir les événements publics qui sont à venir <i>et</i> les événements auxquels je suis inscrit <i>et</i> les événements privés auxquels je suis invité.
Résultat obtenu	Je suis sur la page Accueil <i>et</i> je peux voir les événements publics qui sont à venir <i>et</i> les événements auxquels je suis inscrit <i>et</i> les événements privés auxquels je suis invité.
Statut	✓ OK

Nom	8.1 Affichage des informations de l'événement (situation valides)
User Story	S.13 À propos de l'événement
Situation	Etant donné que je suis un utilisateur anonyme <i>et</i> que je suis sur la page Accueil <i>et</i> que je suis intéressé par l'événement Rentrée techniciens quand je clique sur l'événement <i>et</i> que je suis redirigé la page de l'événement alors je peux voir les informations de l'événement.
Résultat obtenu	Je peux voir l'auteur de l'événement, sa description, son nombre maximal de personnes invitées, sa localisation, sa date de début <i>et</i> sa date de fin.
Statut	✓ OK

Nom	9.1 Validation par mail pour un événement public (situation valides)
User Story	S.7 Validation participation à un événement
Situation	Etant donné que je suis un utilisateur connecté et je vais participer à l'événement 20km de Paris quand je clique sur le bouton S'INSCRIRE alors un mail pour confirmer ma participation s'envoie.
Résultat obtenu	Je reçois un email qui confirme ma participation disant Vous participez à 1 événement "20km de Paris" prévu de 2020-10-11 10:00:00 à 2020-10-11 19:00:00. à l'événement un message de validation apparaît en haut à droite disant Vous êtes bien inscrit à l'événement "20km de Paris"
Statut	✓ OK

Nom	9.2 Validation par mail pour un événement privé (situation valides)
User Story	S.7 Validation participation à un événement
Situation	Etant donné que je suis un utilisateur connecté et je vais participer à l'événement Mon anniversaire quand je clique sur le bouton S'INSCRIRE alors un mail pour confirmer ma participation s'envoie.
Résultat obtenu	Je reçois un email qui confirme ma participation disant Vous participez à 1 événement "Mon anniversaire" prévu de 2020-08-01 14:00:00 à 2020-08-02 02:00:00. à l'événement un message de validation apparaît en haut à droite disant Vous êtes bien inscrit à l'événement "20km de Paris"
Statut	✓ OK

Nom	10.1 Mail de rappel pour la participation à un événement public (situation valides)
User Story	S.8 Rappel participation à un événement
Situation	Etant donné que je participe à l'événement 20km de Paris et je suis la veille de l'événement quand j'ouvre ma boîte mail (adresse mail : nicolas.hr@eduge.ch) alors je retrouve un mail me rappelant que l'événement est le lendemain.
Résultat obtenu	Je retrouve un mail disant N'oubliez pas que l'événement "20km de Paris" se déroule demain..

Nom	10.2 Mail de rappel pour la participation à un événement privé (situation valides)
User Story	S.8 Rappel participation à un événement
Situation	Etant donné que je suis un utilisateur connecté <i>et</i> participant à l'événement Mon anniversaire <i>et</i> que je suis la veille de l'événement quand j'ouvre ma boîte mail (adresse mail : nicolas.hr@eduge.ch) alors je retrouve un mail me rappelant que l'événement est le lendemain.
Résultat obtenu	La veille de l'événement, je reçois un mail disant N'oubliez pas que l'événement "Mon anniversaire" se déroule demain..
Statut	✓ OK

Nom	11.1 Suppression d'événement public sans participants (situation valides)
User Story	S.11 Suppression d'un événement
Situation	Etant donné que je suis un utilisateur connecté <i>et</i> ayant créée l'événement public Rentrée techniciens <i>et</i> n'ayant aucun participant <i>et</i> ne voulant plus faire cet événement quand je vais sur la page d'information de mon événement <i>et</i> que je clique sur le bouton SUPPRIMER , une fenêtre apparaît pour m'invitant à valider la suppression de mon événement <i>et</i> que je clique sur le bouton Valider alors l'événement Rentrée techniciens est supprimé .
Résultat obtenu	Une fenêtre apparaît au milieu de l'écran me demandant voulez-vous vraiment supprimer l'événement "Mon anniversaire" ? <i>et</i> je clique sur le bouton Valider <i>et</i> un message de validation apparaît au milieu de l'écran me disant L'événement "Rentrée techniciens" est bien supprimé.
Statut	✓ OK

Nom	11.2 Suppression d'événement privé sans participants (situation valides)
User Story	S.11 Suppression d'un événement
Situation	Etant donné que je suis un utilisateur connecté <i>et</i> ayant créée l'événement privé Mon anniversaire <i>et</i> n'ayant aucun participant <i>et</i> ne voulant plus faire cet événement quand je vais sur la page d'information de mon événement <i>et</i> que je clique sur le bouton SUPPRIMER , une fenêtre apparaît pour m'invitant à valider la suppression de mon événement <i>et</i> que je clique sur le bouton Valider alors l'événement Mon anniversaire est supprimé .
Résultat obtenu	Une fenêtre apparaît au milieu de l'écran me demandant voulez-vous vraiment supprimer l'événement "Mon anniversaire" ? <i>et</i> je clique sur le bouton Valider <i>et</i> un message de validation apparaît au milieu de l'écran me disant L'événement "Mon anniversaire" est bien supprimé.
Statut	✓ OK

Nom	11.3 Suppression d'événement public qui contient des participants (situation invalides)
User Story	S.11 Suppression d'un événement
Situation	Etant donné que je suis un utilisateur connecté <i>et ayant créée l'événement public Rentrée techniciens et ayant 2 participants et ne voulant plus faire cet événement quand je vais sur la page d'information de mon événement alors je clique sur le bouton SUPPRIMER.</i>
Résultat obtenu	Une fenêtre apparaît au milieu de l'écran me disant L'événement "Rentrée techniciens" ne peut être supprimé car il y a des participants.
Statut	✓ OK

Nom	11.4 Suppression d'événement privé qui contient des participants (situation invalides)
User Story	S.11 Suppression d'un événement
Situation	Etant donné que je suis un utilisateur connecté <i>et ayant créée l'événement privé Mon anniversaire et ayant 2 participants et ne voulant plus faire cet événement quand je vais sur la page d'information de mon événement alors je clique sur le bouton SUPPRIMER.</i>
Résultat obtenu	Une fenêtre apparaît au milieu de l'écran me disant L'événement "Mon anniversaire" ne peut être supprimé car il y a des participants.
Statut	✓ OK

Nom	12.1 Affichage de la page profile (situation valides)
User Story	S.14 Affichage de la page profil
Situation	Etant donné que je suis un utilisateur connecté <i>et que je vais sur la page Mon profile quand j'arrive sur la page de mon profile alors je peux voir les informations de mon profile.</i>
Résultat obtenu	Je peux voir le pseudo, le nom, le prénom, l'adresse mail, et le numéro de téléphone de mon compte.
Statut	✓ OK

Nom	13.1 Modification d'un événement public (situation valides)
User Story	S.10 Modification d'un événement
Situation	<p>Etant donné que je suis un utilisateur connecté et ayant créé l'événement public Rentrée techniciens et ayant oublié le t de techniciens et que je clique sur l'événement Rentrée techniciens et que j'arrive sur la page de l'événement et que je clique sur le bouton MODIFIER quand je suis redirigé sur la page Modifier mon événement et que je modifie le titre de l'événement en Rentrée techniciens alors je clique sur le bouton VALIDER.</p>
Résultat obtenu	Un message apparaît au milieu de l'écran disant L'événement a bien été modifié. et je suis redirigé sur la page d'information de l'événement Rentrée techniciens .
Statut	✓ OK

Nom	13.2 Modification d'un événement privé (situation valides)
User Story	S.10 Modification d'un événement
Situation	<p>Etant donné que je suis un utilisateur connecté et ayant créé l'événement public Mon anniversaire et que je veux modifier la date de l'événement car je suis occupé à ce moment-là et je clique sur l'événement Mon anniversaire quand j'arrive sur la page de l'événement et que je clique sur le bouton MODIFIER et je suis redirigé sur la page Modifier mon événement et je saisis une nouvelle date de départ de l'événement 05.08.2020 alors je clique sur le bouton VALIDER et je suis redirigé sur la page d'information de mon événement.</p>
Résultat obtenu	Un message apparaît au milieu de l'écran disant L'événement a bien été modifié. et je suis redirigé sur la page d'information de l'événement Mon anniversaire .
Statut	✓ OK

Nom	13.3 Modification d'un événement privé (situation invalides)
User Story	S.10 Modification d'un événement
Situation	<p>Etant donné que je suis un utilisateur connecté et ayant créé l'événement public Mon anniversaire et je veux modifier la date de l'événement car je suis occupé à ce moment-là et je clique sur l'événement Mon anniversaire quand j'arrive sur la page de l'événement et que je clique sur le bouton MODIFIER et je suis redirigé sur la page Modifier mon événement et je saisis une nouvelle date de départ de l'événement 05.08.2019 alors je clique sur le bouton VALIDER.</p>
Résultat obtenu	Un message d'erreur apparaît au milieu de l'écran disant L'événement ne peut pas commencer avant aujourd'hui. et je reste sur la page de modification.
Statut	✓ OK

Nom	13.4 Modification d'un événement privé (situation invalides)
User Story	S.10 Modification d'un événement
Situation	<p>Etant donné que je suis un utilisateur connecté <i>et ayant créé l'événement public Mon anniversaire</i> <i>et je veux modifier la date de l'événement car je suis occupé à ce moment-là et je clique sur l'événement Mon anniversaire quand</i> j'arrive sur la page de l'événement <i>et que je clique sur le bouton MODIFIER et je suis rédiger sur la page Modifier mon événement et je saisis une nouvelle date de départ de l'événement 05.08.2020 et que je saisis une nouvelle date de fin 04.08.2020 alors je clique sur le bouton VALIDER.</i></p>
Résultat obtenu	Un message d'erreur apparaît au milieu de l'écran disant L'événement ne peut pas se terminer avant d'avoir commencé. et je reste sur la page de modification.
Statut	✓ OK

Nom	14.1 Suppression du profile (situation valides)
User Story	S.16 Suppression du profil
Situation	<p>Etant donné que je suis un utilisateur connecté <i>et n'ayant pas de participant à mes événements et que je veux supprimer mon profil et que je clique sur Mon profile dans la barre de navigation quand j'arrive sur la page de mon profile et que je clique sur le bouton SUPPRIMER, une fenêtre apparaît pour m'invitant à valider la suppression de mon profile et que je clique sur le bouton Valider alors mon profile est supprimé.</i></p>
Résultat obtenu	Un message de validation apparaît au milieu de l'écran disant Votre compte à bien été supprimé. et je suis déconnecté et je suis redirigé vers la page Accueil .
Statut	✓ OK

Nom	14.2 Suppression du profile (situation invalides)
User Story	S.16 Suppression du profil
Situation	<p>Etant donné que je suis un utilisateur connecté <i>et ayant pas de participant à mes événements et que je veux supprimer mon profil et que je clique sur Mon profile dans la barre de navigation quand j'arrive sur la page de mon profile alors que je clique sur le bouton SUPPRIMER.</i></p>
Résultat obtenu	Un message d'erreur apparaît au milieu de l'écran disant Il ne doit pas avoir de participant à vos événements pour supprimer votre compte. et je reste sur la page de mon profile.
Statut	✓ OK

Nom	15.1 Modification adresse mail de mon profile (situation valides)
User Story	S.15 Modification du profile
Situation	Etant donné que je suis un utilisateur connecté <i>et que je veux changer d'adresse mail et que je clique sur Mon profile dans la barre de navigation et que je suis redirigé sur la page de mon profile et que je clique sur le bouton MODIFIER quand je suis redirigé sur la page Modifier mon profile et que je modifie l'adresse mail de mon compte en nicohoarau74@gmail.com alors je clique sur le bouton VALIDER.</i>
Résultat obtenu	Un message apparaît au milieu de l'écran disant Votre profile à bien été modifié. et je suis redirigé sur la page Mon profile .
Statut	✓ OK

Nom	15.2 Modification du pseudo de mon profile (situation valides)
User Story	S.15 Modification du profile
Situation	Etant donné que je suis un utilisateur connecté <i>et que je veux changer de pseudo et que je clique sur Mon profile dans la barre de navigation et que je suis redirigé sur la page de mon profile et que je clique sur le bouton MODIFIER quand je suis redirigé sur la page Modifier mon profile et que je modifie le pseudo de mon compte en Hoaraun alors je clique sur le bouton VALIDER.</i>
Résultat obtenu	Un message apparaît au milieu de l'écran disant Votre profile à bien été modifié. et je suis redirigé sur la page Mon profile .
Statut	✓ OK

Nom	15.3 Modification de l'adresse mail de mon profile (situation invalides)
User Story	S.15 Modification du profile
Situation	Etant donné que je suis un utilisateur connecté <i>et que je veux changer d'adresse mail et que je clique sur Mon profile dans la barre de navigation et que je suis redirigé sur la page de mon profile et que je clique sur le bouton MODIFIER quand je suis redirigé sur la page Modifier mon profile et que je modifie l'adresse mail de mon compte en jean@jean.com alors je clique sur le bouton VALIDER.</i>
Résultat obtenu	Un message d'erreur apparaît au milieu de l'écran disant Cette adresse mail est déjà utilisée. et je reste sur la page de modification de mon profile.
Statut	✓ OK

Nom	15.4 Modification de le pseudo de mon profile (situation invalides)
User Story	S.15 Modification du profile
Situation	Etant donné que je suis un utilisateur connecté <i>et que</i> je veux changer de pseudo <i>et que</i> je clique sur Mon profile dans la barre de navigation <i>et que</i> je suis redirigé sur la page de on profile <i>et que</i> je clique sur le bouton MODIFIER quand je suis redirigé sur la page Modifier mon profile <i>et que</i> je modifie le pseudo de mon compte en Albert alors je clique sur le bouton VALIDER .
Résultat obtenu	Un message d'erreur apparaît au milieu de l'écran disant Ce pseudo est déjà utilisé. et je reste sur la page de modification de mon profile.
Statut	✓ OK

Nom	15.5 Modification du mot de passe de mon profile (situation invalides)
User Story	S.15 Modification du profile
Situation	Etant donné que je suis un utilisateur connecté <i>et que</i> je veux changer de mot de passe <i>et que</i> je clique sur Mon profile dans la barre de navigation <i>et que</i> je suis redirigé sur la page de on profile <i>et que</i> je clique sur le bouton MODIFIER quand je suis redirigé sur la page Modifier mon profile <i>et que</i> je modifie le mot de passe de mon compte en poire alors je clique sur le bouton VALIDER .
Résultat obtenu	Un message d'erreur apparaît au milieu de l'écran disant Le mot de passe doit faire au moins 8 caractères et doit contenir au moins une majuscule et un chiffre. et je reste sur la page de modification de mon profile.
Statut	✓ OK

Nom	15.6 Modification du mot de passe différent de mon profile (situation invalides)
User Story	S.15 Modification du profile
Situation	Etant donné que je suis un utilisateur connecté <i>et que</i> je veux changer de mot de passe <i>et que</i> je clique sur Mon profile dans la barre de navigation <i>et que</i> je suis redirigé sur la page de on profile <i>et que</i> je clique sur le bouton MODIFIER quand je suis redirigé sur la page Modifier mon profile <i>et que</i> je modifie le mot de passe de mon compte en Super2020 <i>et qu'en validation de mot de passe je mets Super2012</i> alors je clique sur le bouton VALIDER .
Résultat obtenu	Un message d'erreur apparaît au milieu de l'écran disant Le mot de passe doit faire au moins 8 caractères et doit contenir au moins une majuscule et un chiffre. et je reste sur la page de modification de mon profile.
Statut	✓ OK

Suivis journaliers des tests

Numéro de test	J0 25.05.2020	J1 26.05.2020	J2 27.05.2020	J3 28.05.2020	J4 29.05.2020	J5 02.06.2020	J6 03.06.2020	J7 04.06.2020	J8 05.06.2020	J9 08.06.2020	J10 09.06.2020
1.1	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗
1.2	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗
1.3	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗
1.4	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗
1.5	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗
1.6	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗
2.1	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗
2.2	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗
2.3	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗
3.1	✗	✗	✓	✓	✓	✓	✓	✓	✓	✓	✗
3.2	✗	✗	✓	✓	✓	✓	✓	✓	✓	✓	✗
3.3	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓	✗
3.4	✗	✗	✓	✓	✓	✓	✓	✓	✓	✓	✗
3.5	✗	✗	✓	✓	✓	✓	✓	✓	✓	✓	✗
3.6	✗	✗	✓	✓	✓	✓	✓	✓	✓	✓	✗
4.1	✗	✗	✗	✗	✓	✓	✓	✓	✓	✓	✗
4.2	✗	✗	✗	✗	✓	✓	✓	✓	✓	✓	✗
4.3	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓	✗
4.4	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✗
5.1	✗	✗	✗	✗	✓	✓	✗	✓	✓	✓	✗
6.1	✗	✗	✓	✓	✓	✓	✗	✓	✓	✓	✗
7.1	✗	✗	✗	✓	✓	✓	✓	✗	✓	✓	✗
7.2	✗	✗	✗	✓	✓	✓	✓	✗	✓	✓	✗
8.1	✗	✗	✗	✗	✗	✓	✓	✗	✓	✓	✗
9.1	✗	✗	✗	✗	✗	✓	✓	✗	✓	✓	✗
9.2	✗	✗	✗	✗	✗	✓	✗	✓	✓	✓	✗
10.1	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓	✗
10.2	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓	✗
11.1	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓	✗
11.2	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✗
11.3	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓	✗
11.4	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✗
12.1	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓	✗
13.1	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓	✗
13.2	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✗
13.3	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✗
13.4	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✗
14.1	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✗
14.2	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✗
15.1	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✗
15.2	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✗
15.3	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✗
15.4	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✗
15.5	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✗
15.6	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✗

Conclusion

Difficultés rencontrées

J'ai eu quelques problèmes non bloquants lors de la réalisation du projet. Les problèmes principaux que j'ai rencontré sont les suivants :

- Lors de la réalisation du projet, j'ai eu du mal à suivre le planning que je me suis imposé car lorsque je réalisais les tâches je me suis rendu compte qu'il y avait des tâches importantes pour l'application qui était planifiée dans les derniers jours du projet.
- Lorsqu'on créait un événement privé, les données de la liste des invités s'envoyait avant que la sélection des invités ne soit faite.
 - Pour corriger ce problème j'ai essayé de mettre le call ajax en *async: false*, mais le problème était toujours présent, ensuite j'ai essayé de chercher sur internet, mais je ne trouvais pas de solution à mon problème car SweetAlert2 est une librairie peu utilisée. Ne trouvant pas de solution j'ai parlé de mon problème à Tanguy Cavagna (un camarade de classe), et après plusieurs essais, on a compris trouvé comment résoudre le problème. Il fallait utiliser *fetch()* au lieu de *\$.ajax({})* parce avec un *fetch()* on peut faire en sorte que le code attende la réponse du l'appel asynchrone.
- Lorsqu'on s'inscrit à un événement, on reçoit un mail de confirmation. J'ai eu du mal à réaliser cette fonctionnalité car je n'ai pas beaucoup utilisé mail lors de ma formation.
 - J'ai réussi à faire l'envois de mail grâce à la théorie que PHPMailer m'a fournie.

Variantes de solutions et choix

Lors de la réalisation du projet, plusieurs choix ont dû être faits concernant la manière de réaliser celui-ci. Avec du recul, je suis content des décisions que j'ai prises : elles m'ont permis de réaliser mon projet à terme, tout en prenant en compte la taille du projet et le temps à disposition. Les choix faits sont expliqués dans le journal de bord.

Améliorations possibles

Architecture technique

La structure de l'application actuelle est fonctionnelle, la partie interface est presque totalement séparée de la partie qui s'occupe du traitement du serveur. Cependant, quelques modifications pourraient être apportées si l'application venait à grandir.

Côté front-end (JavaScript), il serait nécessaire d'utiliser ESLint qui ferait en sorte d'optimiser mon code JavaScript car celui-ci contient un fichier qui se rapproche des 1000 lignes.

Côté back-end (PHP), il serait nécessaire d'ajouter à Composer la dépendances PHP_CodeSniffer afin d'optimiser le code car actuellement l'application possède un fichier de plus de 1000 lignes.

Fonctionnalités

L'application WEB actuelle correspond aux demandes du cahier des charges, mais je pense qu'il pourrait avoir autres fonctionnalités qui pourraient être ajoutées comme:

- Faire en sorte qu'on reçoive une notification lorsqu'on est invités à un événement.
- Pouvoir modifier un événement public en événement privé.
- Pouvoir avoir une photo de profile.
- Utiliser une API qui permettrait d'autocompléter la saisie d'un lieu.
- Lors de l'inscription, envoyer un token par mail pour valider l'inscription de l'utilisateur.

Bilan personnel

J'ai bien aimé travailler sur WEGO car j'adore la programmation WEB et de plus c'est un des premiers projets que j'élabore de A à Z ce qui m'a fait apprendre plein choses telles que l'envoie de mail depuis une application WEB ou même le fait de tenir un journal de bord afin de savoir les qui ont été bloquant.

Remerciements

Je tiens à remercier:

- Mme Mota qui m'a donné la théorie pour rédiger une documentation complète et qui m'a aussi suivit durant tout mon TPI.
- Tanguy Cavagna avec qui j'ai j'ai partagé lors de la réalisation de mon TPI, ce qui m'a permis de penser différemment sur la rédaction de ma documentation ou sur la rédaction de mon code.

Annexes

Glossaire

Termes métiers

Termes	Définition
Événement	Un événement est un lieu de rencontre avec les participants de celui-ci afin d'effectuer une action.
Événement privé	Un événement privé est un événement accessible que si on est invité à celui-ci.
Événement public	Un événement public est un événement où tout le monde peut y participer.
Utilisateur	Personne qui utilise WEGO et qui participe au minimum à un événement ou qui a créé au moins un événement.

Termes techniques

Termes	Définition
Back-end	Partie d'une application qui concerne tout ce qui se passe côté serveur (par exemple: base de données, authentification...).
Front-end	Partie d'une application qui concerne tout ce qui se passe côté de l'utilisateur final (par exemple: l'apparence de l'application).
Librairies (ou bibliothèque)	Une brique logicielle externe au projet qui conçut pour être utilisé par d'autres développeurs.

Sources

Lors du déroulement de mon projet, j'ai utilisé les sites WEB suivants afin de trouver de l'aide :

- La documentation officiel PHP : <https://www.php.net/>
- Le site de question/réponse StackOverflow : <https://stackoverflow.com/>
- La documentation de PHPMailer : <https://github.com/PHPMailer/PHPMailer>
- Le site de didacticiel le développement WEB : <https://www.w3schools.com/>

Résumé TPI



Hoarau Nicolas - Juin 2020

École de métiers: CFPT Informatique, Petit-Lancy GE

Entreprise formatrice : Formation plein temps (en école)

Situation de départ : Dans le cadre du *Travail Pratique Individuel* (TPI), qui est l'examen qui valide mon CFC d'informaticien, j'ai dû réaliser, sur une durée de 88 heures (11 jours) et sur la base d'un énoncé imposé, une application WEB qui permet de gérer des événements publics ou privés. Les utilisateurs doivent pouvoir créer un compte et s'authentifier afin de pouvoir participer ou de pouvoir créer des événements publics ou privés. Dans l'application retrouve d'autres fonctionnalités comme le fait de devoir créer une liste d'invités lorsqu'on créer un événement privé et de pouvoir la modifier après la création de l'événement.

Mise en œuvre : Le projet a été réalisé en JavaScript pour le front-end et en PHP et MySQL côté serveur. Le style du site a été conçus de sorte qu'il soit UI Design. Toutes les données du site sont chargées via des requêtes AJAX. Afin de réaliser ce projet, la méthodologie en 6 étapes a été utilisée tout au long du projet afin d'organiser le travail. Les différents points du cahier des charges ont été transformés en *user stories* ce qui permet d'exprimer la fonctionnalité du point de vue des utilisateurs finaux. Il y a un protocole de tests précis qui a été élaboré afin de m'assurer que l'implémentation soit conforme aux demandes du cahier des charges, le protocole contient plusieurs scénarios à respecter afin de tester l'application de différentes manières. Un journal de bord a été rédigé quotidiennement lors du déroulement du projet afin de lister les imprévus, les problèmes rencontrés et les différents choix effectués.

Résultats : Les utilisateurs finaux ont à leur disposition une application web complète et qui est simple d'utilisation. Toutes les demandes du cahier des charges ont pu être complétées. La structure du site respecte la structure demandée dans le cahier des charges. Une documentation technique ainsi qu'une documentation utilisateur ont été rédigées.

Énoncé



Travail pratique individuel (TPI)
Informaticien-ne CFC
Dossier d'inscription et description du travail

Candidat :	Entreprise formatrice :
Nom : Hoarau	Société : CFPT – Ecole d'informatique
Prénom : Nicolas	Adresse : 10, Ch. Gérard de Ternier
Classe : IDAP4B	Localité : 1213 Petit-Lancy
Tel professionnel :	Téléphone : 022 388 87 28
Tel mobile/privé : 079 481 70 28	Nom Formateur : Katia Mota Stroppolo
E-Mail : nicolas.hr@eduge.ch	Tel direct : 076 224 04 11
	E-Mail : katia.motastroppolo@edu.ge.ch

Titre du travail :

Domaine :

Développement d'applications Informatique d'entreprise Technique des systèmes

Durée du travail (comprise entre 70h et 90h) : 88h **Date de début souhaitée :** 20 avril 2020

Horaire hebdomadaire du travail : 7h30-11h40 / 12h40 -16h45

lundi _____ mardi _____ mercredi _____ jeudi _____ vendredi _____

Lieu où se déroule le TPI si différent de l'adresse de l'employeur (adresse complète) :

Salle R111 (I.DA-P4B) / R113 (I.DA-P4A)

Résumé du travail :

Création d'une application WEB permettant aux utilisateurs enregistrés de créer et de s'inscrire à des événements de type ouverts ou privés.

RAPPEL :

Il est interdit au candidat de prendre connaissance de l'énoncé du travail de TPI avant le début de celui-ci.

L'énoncé lui sera transmis par les experts, par mail, le matin du 1^{er} jour du TPI avant 7h30.

Devoir d'examen défini. L'entreprise formatrice :

Lieu : _____ **Date :** _____

Signature : _____

Les pages suivantes contiennent la description du projet. Le dossier sera ensuite validé par le collège des experts qui désignera un (et dans ce cas le chef expert participera à la présentation) ou deux d'entre eux pour le suivi du déroulement du travail. L'acceptation de celui-ci sera confirmée par leurs signatures sur la feuille d'évaluation du TPI.

Rappel : Tous les dossiers incomplets seront automatiquement refusés.

TPI - Cahier des charges

Ce document sera connu du candidat uniquement au commencement du TPI. Il est interdit d'en communiquer le contenu au candidat avant la date de TPI convenue.

1. Titre

- WE GO

2. Matériel et logiciels à disposition

- Ordinateur de l'école, 2 écrans
- Windows 10
- uWamp, WAMP
- Visual Studio Code
- MySQLWorkbench
- Suite Office

3. Prérequis

Le candidat a toutes les connaissances nécessaires afin de créer un site WEB avec

- Une base de données MySQL (Module 105 – 80h)
- Un back-end en PHP (Modules 307, 133, 151 – 120h)
- Un front-end avec HTML5, CSS3 (Module 101 – 80h)
- JavaScript, JQuery et Ajax (ateliers de développement d'application WEB - ~160h)

4. Descriptif complet du projet

4.1 Planification :

- Analyse : 16h
- Implémentation : 36h
- Test : 16h
- Documentation : 20h

Vous devez établir un planning détaillé avant la fin de la première journée.

4.2 Méthodologie

Vous devez utiliser la méthodologie en 6 étapes pour la gestion de votre projet (GANTT) et agile pour le développement (BACKLOG).

4.3 Description de l'application

L'application prévoit 3 types d'utilisateurs : l'anonyme, le membre et l'administrateur.

Les fonctionnalités et droits à implémenter sont :

Un utilisateur anonyme a le droit de

➤ Afficher une page accueil présentant

- Le nom et une image des événements ouverts à venir (heure du chargement de la page + 1 heure)
 - Si l'événement a une image, l'image s'affiche. Si non, une image par défaut s'affiche
- Un descriptif court du but et du fonctionnement du site
- Une zone lui permettant de créer un compte
- Une zone de connexion

Contraintes et règles de gestion

- Seuls les événements de type « ouvert » et à venir sont affichés sur la page d'accueil du site

Un utilisateur authentifié a tous les droits d'un utilisateur anonyme et en plus le droit de

➤ Se déconnecter et revenir à la page d'accueil

➤ Afficher une page personnelle permettant de gérer son propre profil (lire, modifier, supprimer). Un profil contient les informations suivantes :

- un pseudo (obligatoire, unique au sein de l'application)
- un prénom (obligatoire)
- un nom (peut être vide)
- un email (obligatoire, unique au sein de l'application)
- un mot de passe (obligatoire)
- un numéro de téléphone (peut être vide)

Contraintes et règles de gestion

○ Lors de la suppression d'un profil

- Il est impossible de supprimer un profil si celui-ci est l'organisateur d'un événement à venir où des personnes sont inscrites (la gestion des désinscriptions groupées d'autres participants que l'organisateur lui-même est hors périmètre de cette application).
- La suppression d'un profil entraîne la suppression des inscriptions de l'utilisateur.
- Le mot de passe doit contenir au moins 9 caractères dont au moins 1 chiffre et 1 caractère spécial

➤ Afficher une page lui permettant de créer un événement. Un événement contient les informations suivantes :

- nom (obligatoire)
- un court descriptif (obligatoire)
- une Date&Time de début (obligatoire)
- une Date&Time de fin (obligatoire)
- une Date&Time de création (obligatoire, date système)
- un lieu (obligatoire)
- un type : privé ou ouvert (obligatoire)
- un nombre maximum de participants (obligatoire)
- une image
 - l'image n'est pas obligatoire.
 - Si l'organisateur ne fournit pas d'image, une image par défaut lui est attribuée

Contraintes et règles de gestion

- Seul l'organisateur de l'événement peut le modifier ou le supprimer
 - Un événement ne peut pas être supprimé s'il contient des inscrits
 - la gestion des désinscriptions d'autres participants que l'organisateur lui-même est faite hors périmètre de l'application.
 - Seuls les événements de type « ouvert » sont affichés dans la page d'accueil du site
 - L'organisateur de l'événement est inscrit d'office à l'événement.
 - Les événements de type privé
 - Lors de la création d'un événement privé, l'utilisateur (l'hôte), doit remplir une liste des personnes invitées parmi les parmi les utilisateurs de l'application
 - Seuls les utilisateurs faisant partie de cette liste d'invités peuvent s'inscrire à un événement privé
- Afficher une page lui permettant de visualiser les détails d'un événement qu'il organise et de les modifier ou de supprimer l'événement affiché. Cette page affiche les détails suivants :
- le nom
 - le descriptif
 - la Date&Time de début
 - la Date&Time de fin
 - la Date&Time de création
 - le lieu
 - le type : ouvert ou privé
 - le nombre maximum de participants
 - l'image de l'événement
 - la liste d'utilisateurs inscrits

Contraintes et règles de gestion

- Seul l'organisateur de l'événement peut le modifier ou le supprimer
 - Un événement ne peut pas être supprimé s'il contient des inscrits
 - la gestion des désinscriptions d'autres participants que l'organisateur lui-même est faite hors périmètre de l'application.
- Afficher une page lui permettant de gérer ses inscriptions à un événement.
- Cette page contient trois zones d'affichage :
- Une zone contenant tous les événements ouverts à venir
 - Une zone contenant tous les événements privés à venir pour lesquels l'utilisateur a été invité (cf. fait partie de la liste d'invités de l'événement privé)
 - Une zone contenant la liste de tous les événements pour lesquels l'utilisateur est inscrit
 - un bouton sur chaque événement de cette zone permet à l'utilisateur de se désinscrire.
 - Suite à la désinscription, l'événement n'est plus affiché dans cette zone
 - Lorsqu'un utilisateur s'inscrit à un événement une confirmation par mail lui est envoyée au mail enregistré dans la base de données.
 - Un utilisateur inscrit à un événement reçoit un mail de rappel un jour avant.

Contraintes et règles de gestion

- Seul un utilisateur connecté sur le site peut s'inscrire aux événements.
- Les inscriptions ne peuvent être enregistrées que jusqu'au nombre maximum saisi par l'hôte lors de la création de l'événement.
 - Lorsque ce nombre d'inscription est atteint, l'événement affiche l'information « complet » et plus aucune inscription ne peut être effectuée
- Un utilisateur ne peut s'inscrire à un événement de type « privé » que s'il fait partie de la liste des invités créée par l'organisateur lors de la création de l'événement.

Un administrateur a, en plus des droits ci-dessus, les droits de :

- Gérer la base de données au travers de l'interface d'administration de la base donnée avec un rôle administrateur (pas de fonctionnalité d'administration à implémenter dans l'application web, le rôle BD suffit)
- Maintenir le code source de l'application (pas de fonctionnalité de développement à implémenter dans l'application web, les informations techniques permettant de modifier le code source suffisent)

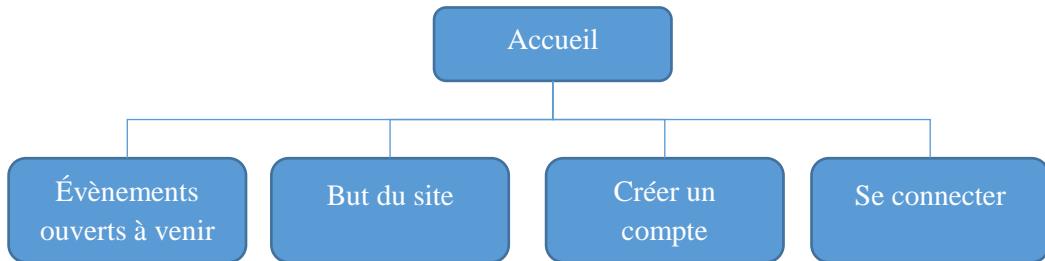
4.4 La structure du site WEB

Structure du code

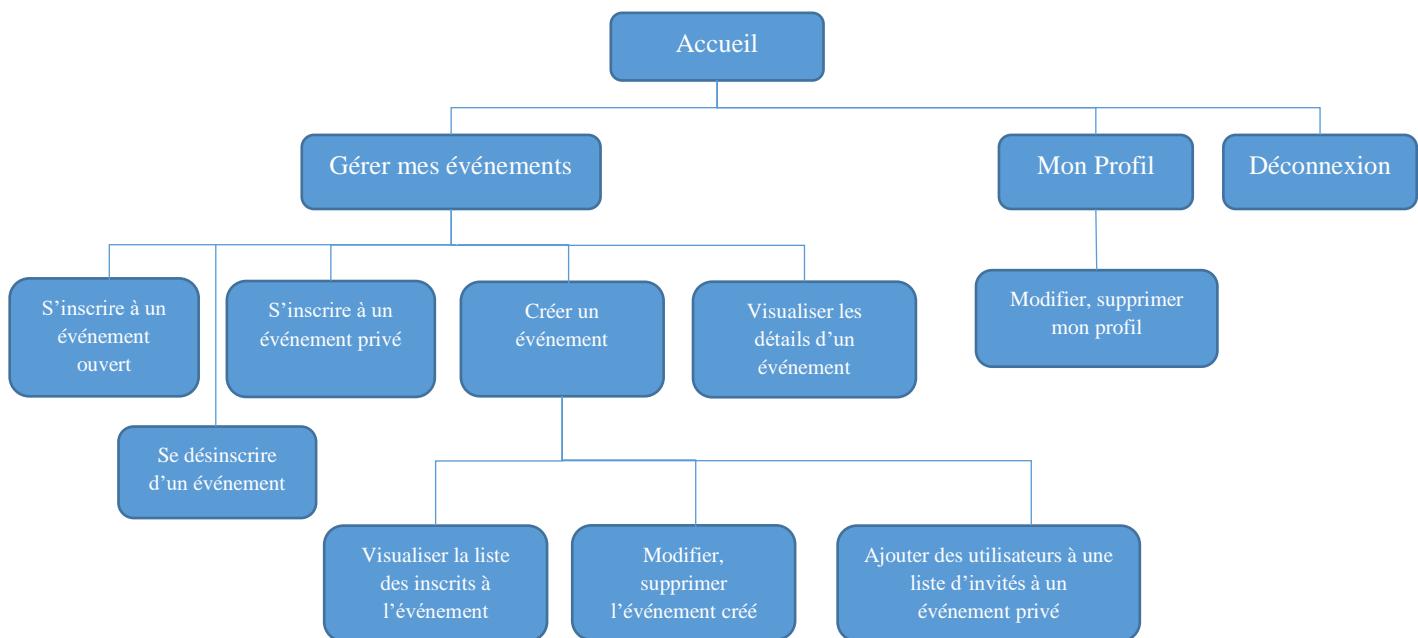
Le code doit être structuré dans différents dossiers représentant les différentes couches applicatives, cf. :

- La couche du modèle contenant les accès à la base de données
- La couche de présentation contenant les vues (HTML, CSS, images, éventuellement JavaScript)
- La couche de traitement contenant les managers/contrôleurs ou autres fichiers de traitement des informations en PHP

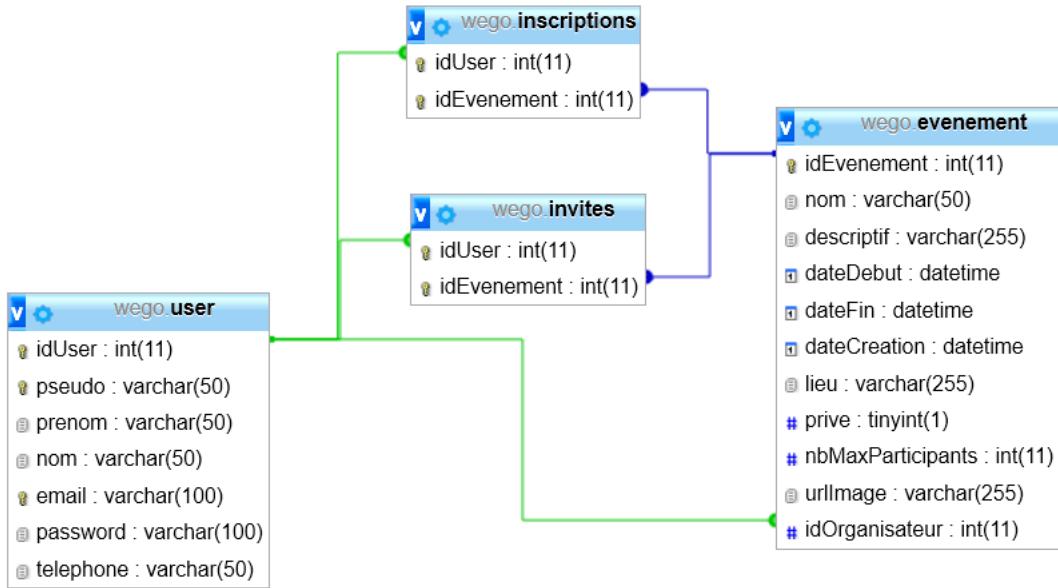
Structure du contenu accessible à l'utilisateur sans être connecté



Structure du contenu accessible pour les utilisateurs connectés



4.5 Structure de la base de données



4.6 Tests de l'application

- Chaque fonctionnalité doit faire l'objet d'une définition précise des critères d'acceptation explicités et validés dans le backlog.
- Les accès et droits réservés à l'administrateur enregistrés doivent être sécurisés et testés.

4.7 Consignes

Des consignes de travail vous ont été expliquées durant l'atelier TPI. Pour rappel :

- Vous devez sauvegarder vos sources et données chaque jour
- Vous devez ranger votre place de travail
- Vous devez être présent selon l'horaire fixé.

5. Livrables

Planning

Rapport de projet

Manuel utilisateur

Journal de travail

6. Points techniques évalués spécifiques au projet (obligatoire)

correspondants aux points A14 à A20 du formulaire d'évaluation

Éléments mesurables, servant à l'évaluation

No	Critère	Indicateurs	Point de l'énoncé	Testé et validé par	Nb Points
A14	Un utilisateur anonyme peut visualiser uniquement les événements ouverts à venir	1. Un utilisateur anonyme affiche la page d'accueil du site et aucun événement de type privé n'est affiché. 2. Un utilisateur anonyme affiche la page d'accueil du site et aucun événement antérieur à l'heure du chargement de la page + 1 n'est affiché	Un utilisateur anonyme a le droit d'afficher une page accueil présentant <ul style="list-style-type: none"> • Le nom et une image des événements ouverts à venir (heure du chargement de la page + 1 heure) • Si l'événement a une image, l'image s'affiche. Si non, une image par défaut s'affiche 	Utilisateur anonyme	3
A15	Un utilisateur peut créer un nouveau compte et s'y connecter	3. Un utilisateur anonyme enregistre (à partir d'un écran public) un nouveau compte 4. L'utilisateur créé au point 3 peut se déconnecter et se connecter à son nouveau compte	Un utilisateur anonyme a le droit d'afficher une zone lui permettant de créer un compte et une zone de connexion	<ul style="list-style-type: none"> • Utilisateur anonyme pour la création du compte • Utilisateur créé pour la déconnexion / connexion 	3
A16	Il est impossible de supprimer un profil d'un organisateur d'un événement ayant des participants inscrits	5. Un utilisateur authentifié organisant au moins un événement où le nombre de participants != 0 ne peut pas supprimer son profil 6. Un utilisateur authentifié organisant des événements où, pour chaque événement organisé, le nombre de participants == 0 supprime son profil 7. Un utilisateur authentifié n'organisant aucun événement supprime son profil	Un utilisateur authentifié a le droit d'afficher une page personnelle permettant de gérer son propre profil (lire, modifier, supprimer). Contrainte : <ul style="list-style-type: none"> • Il est impossible de supprimer un profil si celui-ci est l'organisateur d'un événement à venir où des personnes sont inscrites (la gestion des désinscriptions groupées d'autres participants que l'organisateur lui-même est hors périmètre de cette application). 	<ul style="list-style-type: none"> • Utilisateur authentifié organisant au moins un événement où le nombre de participants != 0 ➔ fonctionnalité refusée • Utilisateur authentifié organisant 1 ou plusieurs événements où le nombre de participants == 0 ➔ profil supprimé • Utilisateur authentifié n'organisant aucun événement ➔ profil supprimé 	3
A17	Seul un utilisateur authentifié peut créer un événement ouvert	8. un utilisateur authentifié saisi et enregistre dans la base de données un événement contenant <ol style="list-style-type: none"> a. nom (obligatoire) b. un court descriptif (obligatoire) c. une Date&Time de début (obligatoire) d. une Date&Time de fin (obligatoire) e. une Date&Time de création (obligatoire, date système) f. un lieu (obligatoire) g. un type ouvert (obligatoire) h. un nombre maximum de participants (obligatoire) i. une image j. l'image n'est pas obligatoire. k. Si l'organisateur ne fournit pas d'image, une image par défaut lui est attribuée 	<ul style="list-style-type: none"> • Un utilisateur authentifié a le droit d'afficher une page lui permettant de créer un événement ouvert ou privé 	<ul style="list-style-type: none"> • utilisateur authentifié saisie et enregistrement de l'événement; • admin (persistance BD) ; • Utilisateur anonyme (fonctionnalité refusée) 	3
A18	Seul un utilisateur authentifié peut créer un événement privé et créer la liste des invités à l'événement	9. un utilisateur authentifié un utilisateur authentifié 10. un utilisateur authentifié ajoute des invités à la liste d'invités de son événement privé	Un utilisateur authentifié a le droit d'afficher une page lui permettant de gérer ses inscriptions à un événement. Les événements de type privé <ul style="list-style-type: none"> • Lors de la création d'un événement privé, l'utilisateur (l'hôte), doit remplir 	<ul style="list-style-type: none"> • Utilisateur authentifié organisateur de l'événement crée & enregistre l'événement privé & remplit la liste d'invités 	3

		<p>11. L'utilisateur peut afficher la liste des invités à son événement privé</p> <p>12. L'utilisateur authentifié invité (cf. membre de la liste) voit l'événement dans sa zone contenant tous les événements privés à venir pour lesquels il a été invité</p>	<p>une liste des personnes invitées parmi les utilisateurs de l'application</p> <ul style="list-style-type: none"> • Seuls les utilisateurs faisant partie de cette liste d'invités peuvent s'inscrire à un événement privé. • Un utilisateur authentifié a le droit d'afficher une zone contenant tous les événements privés à venir pour lesquels l'utilisateur a été invité (cf. fait partie de la liste d'invités de l'événement privé) 	<ul style="list-style-type: none"> • Admin (persistance BD) ; • Utilisateur authentifié membre de la liste voit l'événement privé dans sa zone d'affichage d'événements privés • Utilisateur anonyme (fonctionnalité refusée) 	
A19	Seul un utilisateur authentifié peut s'inscrire, recevoir un email de confirmation, un email de rappel et se désinscrire à un événement (pour autant qu'il reste encore de la place)	<p>13. un utilisateur authentifié s'inscrit à un événement public pour lequel le nombre maximum de participants n'est pas encore atteint</p> <p>14. Après une inscription, le nombre de places disponibles diminue de 1 ($\text{nbMaximunParticipants} - \text{nblInscrits}$)</p> <p>15. Après l'inscription, l'utilisateur peut voir l'événement dans la zone</p> <p>16. Après l'inscription, l'utilisateur reçoit un mail de confirmation</p> <p>17. L'utilisateur inscrit reçoit un mail de rappel 1 jour avant la date de l'événement</p> <p>18. un utilisateur authentifié ne peut pas s'inscrire à un événement où le nombre maximum de participants est atteint ($\text{nblInscrits} == \text{nbParticipants}$)</p> <p>19. Un utilisateur authentifié peut se désinscrire d'un événement</p>	<ul style="list-style-type: none"> • Un utilisateur authentifié a le droit d'afficher une page lui permettant de gérer ses inscriptions à un événement. Cette page contient trois zones d'affichage <ul style="list-style-type: none"> • Une zone contenant la liste de tous les événements pour lesquels l'utilisateur est inscrit • Un bouton sur chaque événement de cette zone permet à l'utilisateur de se désinscrire. • Suite à la désinscription, l'événement n'est plus affiché dans cette zone • Lorsqu'un utilisateur s'inscrit à un événement une confirmation par mail lui est envoyée au mail enregistré dans la base de données. • Un utilisateur inscrit à un événement reçoit un mail de rappel un jour avant. 	<ul style="list-style-type: none"> • utilisateur authentifié s'inscrit à un événement où $\text{nbParticipants} < \text{nbParticipantsMaximun}$; • le client mail test enregistré dans la BD pour l'utilisateur inscrit reçoit le mail de confirmation • client mail test enregistré dans la BD pour l'utilisateur inscrit reçoit le mail de rappel 1 jour avant la date de l'événement • admin (persistance BD) ; • utilisateur anonyme (fonctionnalité refusée) 	3
A20	Seul un utilisateur authentifié peut s'inscrire à un événement privé (pour autant qu'il ait été invité)	<p>20. L'utilisateur authentifié invité (cf. membre de la liste créée au point A18) voit l'événement dans sa zone contenant tous les événements privés à venir pour lesquels il a été invité</p> <p>21. L'utilisateur authentifié peut s'inscrire à l'événement pour lequel il a été invité</p>	<ul style="list-style-type: none"> • Un utilisateur authentifié a le droit d'afficher une zone contenant tous les événements privés à venir pour lesquels il a été invité (cf. fait partie de la liste d'invités de l'événement privé) 	<ul style="list-style-type: none"> • Utilisateur authentifié membre de la liste voit l'événement privé dans sa zone d'affichage d'événements privés 	3

Journal de bord

Lundi 25 Mai 2020

Objectifs

Les objectifs de la journée sont lire l'énoncé, rédiger le backlog ainsi que les scénarios et faire la planification.

Déroulement

08h00: Je commence ma journée en lisant l'énoncé (ce qui correspond à l'étape *S'informer* de la méthode en 6 étapes). J'ai compris presque la totalité de l'énoncé sauf pour le critère A19 15) qui dit:

Après l'inscription, l'utilisateur peut voir l'événement dans la zone.

Car je ne vois pas ce qu'est la zone.

Je me demande si les invités aux événements privés reçoivent un mail les prévenant qu'ils sont invités à l'événement.

Et aussi je me demande si les utilisateurs invités à un événement privé s'il reçoit un mail.

Donc j'ai décidé d'attendre ma formatrice qui passe à 11h30.

08h20: Je finis de lire l'énoncé et je commence à rédiger le backlog.

Pour définir un ordre de priorité pour les *story* du backlog, j'ai utilisé la méthode MoSCoW.

- !! L'équivalent de **Must**: sera **Indispensable**
- ! L'équivalent de **Should**: sera **Critique**

J'ai choisi de retirer le **Could** et le **Won't** car je ne dois pas déborder du cahier des charges lors de cet examen, mais si j'avais pu déborder du cahier des charges j'aurais pu mettre en **Should** le fait d'envoyer un mail aux personnes qui sont

invitées à un événement privé.

Chaque *user story* est rédigé sous le format suivant:

Nom	S.<n° de la story> <i>Nom de la story</i>
Description	<i>Description de la story</i>
Test(s)	<i>Numéro des tests qui valide la story</i>
Priorité	<i>Priorité de la story</i>

10h45: Je finis de rédiger le backlog et je commence à rédiger mon planning.

Pour le planning prévisionnel et le planning effectif, j'ai utilisé la couleur [#3993fa](#) afin de marquer les jalons.

11h30: J'ai un entretien avec Mme Mota afin de voir mon avancement afin de voir si ma matinée s'est bien déroulée et j'ai pu lui demander pour le point A19.

Lors de l'entretien, on va vérifier si la syntaxe de mon backlog est correcte, répondre aux 2 questions que je me suis posé durant la matinée.

?) Je ne comprends pas ce qu'est la "zone" citée dans le point A19 15, pourriez-vous m'aider comprendre ?

Réponse: La zone est dans la page sur laquelle on peut gérer ses événements.

?) Est-ce que je dois envoyer un mail aux utilisateurs afin de les prévenir qu'ils sont invités à un événement privé ?

Réponse: Il ne faut pas envoyer de mail à ce moment-là, c'est seulement lors de l'inscription d'un événement.

Après l'entretien, je vais devoir rajouter des tâches dans mon backlog, car je ne recouvre pas toutes les demandes du cahier des charges.

12h35: Fin de l'entretien avec Mme Mota, on a fixé un nouvel entretien aujourd'hui à 16h.

12h50: Je prends ma pause de midi.

13h35: Je reprends mon travail sur le backlog.

14h50: Je finis de rédiger le planning prévisionnel et je l'ai implémenté avec le backlog dans la documentation. Je commence à corriger les erreurs de grammaire de mon journal de bord et de la documentation.

15h15: Je commence à rédiger les scénarios de tests sous le format suivant :

Nom	<n° du test>.<n° du cas de test> <i>Nom du test</i> (données valides / données invalides)
User Story	S.<n° de la story>
Situation	<i>Description de tout le test</i>
Résultat obtenu	<i>Résultat du test</i>
Statut	✗ K0 / ✓ OK

16h00: Entretien avec Mme Mota.

17h35: Je finis mon entretien avec Mme Mota. Normalement ma journée devrait se terminer à 17h00, mais aujourd'hui je continue à travailler.

18h10: Je finis ma journée.

Bilan

Ma première journée de TPI s'est terminée, j'avais prévu de faire trop de tâches aujourd'hui alors j'ai commencé à prendre du retard sur mon planning donc demain je vais finir de rédiger les scénarios. Durant la journée, j'ai rencontré des problèmes de compréhension avec l'énoncé, mais j'ai pu répondre à mes interrogations grâce aux deux entretiens faits avec Mme Mota.

Mardi 26 Mai 2020

Objectifs

Les objectifs de la journée sont finir de rédiger les scénarios, créer le dépôt Git, l'implémentation de la base de données, la configuration de Composer, de développer l'inscription, la connexion et la gestion des accès au site .

Déroulement

08h00: Je commence ma journée en continuant la rédaction des scénarios.

8h30: Je vois que la rédaction me prend beaucoup de temps donc j'ai décidé d'arrêter pour les scénarios et de mettre en place la base données car étant un jalon je me dois de la faire en priorités.

8h40: Je commence la création de la base de données.

9h00: J'ai fini de créer la base de données et de la mettre dans le serveur local. Je continue ma matinée en mettant la base de données dans la documentation.

9h45: Je finis de mettre les informations en lien avec la base de données dans la documentation. Ensuite je créer le dépôt Git.

9h55: Je finis initialisation du dépôt Git. Et je commence à mettre en place Composer et ses dépendances.

10h00: Je finis d'installer Composer et ses dépendances et je reprends mon travail sur les scénarios.

11h00: J'ai un entretien avec Mme Mota afin de voir mon avancement..

Durant cet entretien j'ai demandé à Mme Mota :

❓ Faut-il ajouter directement tous les utilisateurs d'un événement privé ou on peut en les ajouter en ajouter une partie et ajouter la suite après avoir créer l'événement ?

Réponse: Il faut ajouter tous les invités lors de la création de l'événement.

J'ai aussi pris le choix avec Mme Mota de ne pas rédiger tous les scénarios de tests d'un coup mais les rédiger au début de la journée et à la fin de la journée.

12h10: Fin de l'entretien avec Mme Mota et j'ai complété mon journal de bord.

12h30: Correction des scénarios rédigés.

12h40: Je prends ma pause du midi.

13h25: Je finis ma pause midi et je créer un deuxième dépôt Git afin d'en avoir un spécialement pour la documentation et un autre spécialement pour le code source.

13h40: Je finis de mettre en place les deux dépôts Git et je commence à développer l'inscription.

16h10: Je finis de développer et de tester l'inscription et je commence à développer la connexion au site.

17h00: Fin de journée.

Bilan

Aujourd'hui j'ai pu faire une grande partie de ce que je souhaitais,, malheureusement je n'ai pas eu le temps faire la gestion des accès. J'ai fait le choix de ne pas faire tous les scénarios de tests d'un coup mais quelque scénario le matin et en fin de journée.

Mercredi 27 Mai 2020

Objectifs

Les objectifs de la journée sont faire la gestion des accès, la création d'événements la création de la liste d'invités pour un événement privé, l'affichage des événements et la page qui contient les informations d'un événement.

Déroulement

08h00: Je commence ma journée en rédigeant les scénarios pour la gestion des accès, pour l'affichage des événements et pour la page qui contient les informations d'un événement.

08h30: Je finis de rédiger les scénarios prévus et je commence développer la gestion des accès

9h00: Je finis de développer la gestion des accès et je commence à développer la création d'événements.

11h30: J'ai un entretien avec Mme Mota afin de voir mon avancement.

Mme Mota est revenue sur la question :

❓ Faut-il ajouter directement tous les utilisateurs d'un événement privé ou on peut en les ajouter en ajouter une partie et ajouter la suite après avoir créer l'événement ?

En me disant qu'il faudrait gérer le nombre maximal de personnes invitées à l'événement privé de la même manière que les événements publics c'est-à-dire qu'on n'est pas obligé d'inviter toutes les personnes dès la création de l'événement.

11h55: Fin de l'entretien avec Mme Mota.

12h10: Je prends ma pause midi.

13h00: Je finis ma pause midi. Et je reprends la création d'un événement.

14h50: Je mets en pause le développement de la création d'événements pour ajouter une colonne pour le *salt* dans la modifier la base de données afin d'avoir une meilleure sécurité dans le site.

15h00: Je finis d'ajouter la colonne dans la base de données. Je vais l'implémenter dans l'inscription et pour la connexion.

15h20: Je finis d'implémenter le *salt* pour l'inscription et la connexion et je reprends la création d'événements.

15h50: Je finis de développer et de tester la création d'événement public.

16h00: Je commence à développer la création de la liste des invités pour un événement privé.

18h00: Je finis ma journée.

Bilan

Aujourd'hui j'ai pu faire que la moitié de ce que je souhaitais, car le style du site m'a pris beaucoup plus de temps que ce que j'avais prévu. Je n'ai pas pu finir la création de la liste des invités à un événement privé, car lorsque ma modal pour ajouter des invités apparaît, les données se sont déjà

envoyées. Je n'ai pas pu faire l'affichage des événements non plus donc je vais finir ces 2 tâches pour demain.

Jeudi 28 Mai 2020

Objectifs

Les objectifs de la journée sont finir la création de la liste pour des invités à un événement privé, faire l'affichage des événements, faire la page qui contient les informations de l'événement et l'inscription à un événement ouvert.

Déroulement

08h00: Je commence ma journée en travaillant sur la création de la liste des invités pour un événement privé.

9h00: J'ai un entretien avec mes experts afin de voir si mon TPI se déroulait sans encombre.

09h10: Fin de l'entretien avec mes experts.

10h20: Je finis la création de la liste d'invités et je commence l'affichage des événements.

11h00: J'ai un entretien avec Mme Mota afin de voir mon avancement.

11h20: Fin de l'entretien avec Mme Mota et je reprends l'affichage des événements.

12h35: Je finis l'affichage des événements.

12h40: Je prends ma pause midi.

14h05: Fin de ma pause midi et je reprends mon travail sur inscription à un événement ouvert.

15h35: Je finis la logique de l'inscription à un événement ouvert.

17h25: Je finis ma journée.

Bilan

Aujourd'hui j'ai presque fait tout ce que j'avais prévu.

Avec Tanguy Cavagna (un camarade de classe) on a réussi à résoudre le problème avec la liste d'invités en utilisant un *fetch()*

```
1 return fetch("url", {  
2   method: "",  
3   headers: {  
4     Accept: "",  
5     "Content-Type": "",  
6   },  
7 }).then((response) => response.json());
```

Au lieu d'un appel *ajax*

```
1 $.ajax({  
2   type: "method",  
3   url: "url",  
4   data: "data",  
5   dataType: "dataType",  
6   success: function (response) {},  
7 });
```

Car avec un appel ajax on ne peut pas mettre le *success:* en asynchrone alors qu'avec un *fetch* on peut le mettre en asynchrone.

Mais je n'ai pas pu finir à 100% l'inscription à un événement public, j'ai réussi à faire toute la logique mais je n'ai pas réussi à faire un style qui fonctionne une fois qu'on c'est inscrit à plusieurs événements, donc je vais finir cette tâche demain.

Vendredi 29 Mai 2020

Objectifs

Les objectifs de la journée sont de finir inscription à un événement public et commencer à faire la page pour la gestion des événements.

Déroulement

08h00: Je commence ma journée en reprenant le travail sur inscription à un événement public.

8h25: Je finis l'inscription à un événement privé. Et je commence l'affichage de la page de gestion des événements.

11h00: J'ai un entretien avec Mme Mota afin de voir mon avancement.

Durant cet entretien Mme Mota m'a demandé de faire un logo afin d'avoir une identité visuelle pour mon site.

11h40: Fin de l'entretien avec Mme Mota et je reprends mon travail sur la page gestion des événements.

12h25: Je prends ma pause midi.

13h20: Fin de ma pause midi et je me remets à travailler sur la page de gestion des événements.

15h00: Je pars pour mon rendez-vous chez le médecin.

16h20: Retour de chez le médecin. Et je me remets à travailler sur la page de gestion des événements.

18h30: Je finis la page de gestion des événements ce qui finit ma journée, j'ai travaillé plus tard afin rattraper le fait que j'ai eu un rendez-vous chez le médecin.

Bilan

Aujourd'hui j'ai pu faire tout ce que j'avais prévu, ce qui est une première depuis le début de mon TPI.

Mardi 02 Juin 2020

Objectifs

Les objectifs de la journée sont faire une page entière avec les informations de l'événement (actuellement il n'y a qu'une carte avec les informations), et faire la validation à un événement.

Déroulement

08h00: Je commence ma journée en travaillant sur la page qui contient les informations de l'événement.

11h00: J'ai un entretien avec Mme Mota afin de voir mon avancement et de me faire un retour sur la documentation intermédiaire rendue vendredi.

Lors de cet entretien, on est revenu sur le rendu intermédiaire que j'ai rendu vendredi 29 mai.

Dans les points que je dois revoir dans ma documentation sont:

- La catégorie *Structure du code*(page 5 de l'énoncé).

- Ajout un Rapport fichier rapport de bug pour documenter mes tests (fails)
- Faire en sorte les émoticônes du suivi journalier de tests soit plus visibles
- Lister tous les acronymes + leurs significations citées dans le glossaire
- Mettre plus en évidence que j'utilise les outils la méthodologie agile (backlog, Gherkins, rédaction des scénarios de test avant d'implémenter dans l'app)
- Mieux documenter les difficultés
- Ajouter des légendes au tableau et aux images

12h40: Fin de l'entretien avec Mme Mota et je reprends mon travail sur l'affichage des informations d'un événement.

14h10: Je prends ma pause midi.

14h45: Fin de ma pause midi et je me remets à travailler sur la page d'information d'un événement.

15h20: Je finis la page d'informations d'un événement. Et je commence à rédiger le scénario de la validation de participation à un événement.

15h40: Je finis de rédiger les scénarios et je commence à implémenter la fonctionnalité.

17h30: J'ai fini d'implémenter les mails lors de l'inscription à un événement et de le tester. Et je finis ma journée.

Bilan

Aujourd'hui j'ai pu faire toutes les tâches planifiées, j'ai eu du mal avec l'envoi de mail car je n'ai pas beaucoup utilisé mail lors de ma formation mais grâce à la documentation de PHPMailer j'ai pu y arriver. J'ai eu un problème avec l'envoi de mail car une fois que le mail c'était envoyé le message de validation du call ajax n'apparaissait plus et il n'y avait l'actualisation de la page non plus.

Mais en recherchant sur stackoverflow j'ai pu trouver où était mon problème.

```
1 try {
2     //Enable SMTP debugging.
3     $mail->SMTPDebug = 3;
```

De base la variable STMPDebug est initialisée à 3, ce qui veut dire qu'un message de validation est envoyé côté client et côté serveur et ceci interférait avec:

```
1 echo json_encode([
2     'ReturnCode' => 0,
3     'Success' => $successMessage
4 ]);
5 exit();
```

Dans ma console JavaScript il y avait ceci qui était affiché avec mon message de validation à la fin:

```
▶ about: function abort(a) {#  
▶ always: function always() {#  
▶ catch: function catch(a) {#  
▶ done: function add() {#  
▶ fail: function add() {#  
▶ getAllResponseHeaders: function getAllResponseHeaders() {#  
▶ getResponseHeader: function getResponseHeader(a) {#  
▶ overrideMimeType: function overrideMimeType(a) {#  
▶ pipe: function pipe() {#  
▶ progress: function add() {#  
▶ promise: function promise(a) {#  
readyState: 4  
responseText: "2020-06-02 16:32:09 Connection: opening to  
smtp.gmail.com:587, timeout=300, options={array}(<br>)n2020-06-02 16:32:09  
Connection: opened(<br>)n2020-06-02 16:32:09 SERVER -&gt; CLIENT: 220  
smtp.gmail.com ESMTP 118sm32409kmj.22 - gsmtp<br>n2020-06-02 16:32:09  
CLIENT -&gt; SERVER: EHLO localhost<br>n2020-06-02 16:32:09 SERVER -&gt;  
CLIENT: 250-smtp.gmail.com at your service,  
[2a02::a13:61a1:3f00::cbe:8539:2df7:735f]250-SIZE  
3582577250-88TMIM250-STARTTLS250-ENHANCEDSTATUSCODES250-PIPELINING250-  
CHUNKING250 SMTPUTF8<br>n2020-06-02 16:32:09 CLIENT -&gt; SERVER:  
STARTTLS<br>n2020-06-02 16:32:09 SERVER -&gt; CLIENT: 220 2.0.0 Ready to  
start TLS<br>n2020-06-02 16:32:09 CLIENT -&gt; SERVER: EHLO  
localhost<br>n2020-06-02 16:32:09 SERVER -&gt; CLIENT: 250-  
smtp.gmail.com at your service,  
[2a02::a13:61a1:3f00::cbe:8539:2df7:735f]250-SIZE  
3582577250-88TMIM250-AUTH LOGIN PLAIN XAUTHTH250-CLIENTTOKEN  
OAUTHBEARER XAUTHT250-ENHANCEDSTATUSCODES250-PIPELINING250-CHUNKING250  
SMTPUTF8<br>n2020-06-02 16:32:09 CLIENT -&gt; SERVER: AUTH  
LOGIN<br>n2020-06-02 16:32:09 SERVER -&gt; CLIENT: 334  
VXNlcm5hbWU8<br>n2020-06-02 16:32:09 CLIENT -&gt; SERVER: [credentials  
hidden]<br>n2020-06-02 16:32:09 SERVER -&gt; CLIENT: 334  
UGFzc3dvcmQ8<br>n2020-06-02 16:32:09 CLIENT -&gt; SERVER: [credentials  
hidden]<br>n2020-06-02 16:32:09 SERVER -&gt; CLIENT: 235 2.7.0  
Accepted<br>n2020-06-02 16:32:09 CLIENT -&gt; SERVER: MAIL  
FROM:site;nichohourau7@gmail.com&gt;<br>n2020-06-02 16:32:09 SERVER -&gt;  
CLIENT: 250 2.1.0 OK 118sm32409kmj.22 - gsmtp<br>n2020-06-02 16:32:09  
CLIENT -&gt; SERVER: RCTP TO:<nicolas.hr@edge.ch>&gt;<br>n2020-06-02  
16:32:09 SERVER -&gt; CLIENT: 250 2.1.5 OK 118sm32409kmj.22 -  
gsmtp<br>n2020-06-02 16:32:09 CLIENT -&gt; SERVER: DATA:<br>n2020-06-02  
16:32:18 SERVER -&gt; CLIENT: 354 Go ahead 118sm32409kmj.22 -  
gsmtp<br>n2020-06-02 16:32:18 CLIENT -&gt; SERVER: Date: Tue, 2 Jun 2020  
18:32:09 +0200<br>n2020-06-02 16:32:18 CLIENT -&gt; SERVER: To:  
nicolas.hr@edge.ch<br>n2020-06-02 16:32:18 CLIENT -&gt; SERVER: From:  
WEGO Service <nicohourau7@gmail.com>&gt;<br>n2020-06-02 16:32:18  
CLIENT -&gt; SERVER: Subject: WEGO : Validation de  
participation<br>n2020-06-02 16:32:18 CLIENT -&gt; SERVER: Message-ID:  
&lt;DzRZ28x00KkjwEMXAXFPnLhGz2r0Rz20M4bKGIm@localhost&gt;  
<br>n2020-06-02 16:32:18 CLIENT -&gt; SERVER: X-Mailer: PHPMailer 6.1.5  
(https://github.com/PHPMailer/PHPMailer)<br>n2020-06-02 16:32:18 CLIENT -&gt;  
SERVER: MIME-Version: 1.0<br>n2020-06-02 16:32:18 CLIENT -&gt; SERVER:  
SERVER: Content-Type: text/html; charset=iso-8859-1<br>n2020-06-02  
16:32:18 CLIENT -&gt; SERVER: Content-Transfer-Encoding:  
8bit<br>n2020-06-02 16:32:18 CLIENT -&gt; SERVER: <br>n2020-06-02  
16:32:18 CLIENT -&gt; SERVER: Vous participez &agrave;  
1&#039;&acute;s&acute;ecutive;&acute;nement &quot;albert&quot; pr&acute;eacute;vu de  
2070-10-26 06:02:00 +02:00; &grave; 2070-10-27 12:02:00<br>n2020-06-02  
16:32:19 CLIENT -&gt; SERVER: <br>n2020-06-02 16:32:19 SERVER -&gt; CLIENT: 250 2.0.0 OK  
1591115530 118sm32409kmj.22 - gsmtp<br>n2020-06-02 16:32:18 CLIENT -&gt;  
SERVER: QUIT<br>n2020-06-02 16:32:18 SERVER -&gt; CLIENT: 221  
2.0.0 closing connection 118sm32409kmj.22 - gsmtp<br>n2020-06-02  
16:32:19 Connection: closed<br>{["ReturnCode":0,"Success  
":true,"Inscription": "\u0000 1 \u000e0\u0009 \u000e0\u0009ement \u000e0\u0009ussui"]}  
setRequestHeader: function setRequestHeader(a, b) {#  
state: function state() {#  
status: 200  
statusCode: statusCode(a) {#  
length: 1  
name: "statusCode"  
prototype: Object { ... }  
},
```

Mais du coup après être allé sur StackOverflow :

```
1 try {  
2     //Enable SMTP debugging.  
3     $mail->SMTPDebug = 0;
```

J'ai pu comprendre qu'en initialisant `STMPDebug` à 0 il n'y aurait plus de problème avec mon call ajax.

Aussi aujourd'hui lors de l'entretien avec Mme Mota elle m'a fait un retour sur la documentation intermédiaire et je me suis rendu compte que j'avais mal compris la partie *Structure du code* dans l'énoncé (page 5), Mme Mota a passé environ 30 minutes à m'expliquer ce qui était attendu.

[Cliquez ici pour voir la documentation de PHPMailer](#)

Mercredi 03 Juin 2020

Objectifs

Les objectifs de la journée sont d'implémenter le rappel de la participation à un événement, la suppression d'un événement, l'affichage du profile et commencer la modification d'un événement.

Déroulement

08h00: Je commence ma journée en rédigeant les scénarios du rappel à un événement et la suppression d'un événement.

08h35: Je finis de rédiger les scénarios du rappel à un événement, de la suppression d'un événement et de l'affichage de la profile et je commence à implémenter le mail de rappel de la participation à un événement.

10h35: Je finis d'implémenter et de tester l'envoi du mail qui rappelle qu'un événement a lieu le lendemain. Et au lieu de faire la suppression d'un événement.

11h05: Je finis d'écrire les scénarios et je commence à implémenter la page profile.

12h15: Je finis l'affichage de la page profile. Et je commence à implémenter la suppression de l'événement.

13h25: Je prends ma pause midi.

14h05: Je finis ma pause midi et je reprends mon travail pour la suppression des événements.

14h45: Je finis d'implémenter et de supprimer la suppression d'événement. Et je commence la rédaction des scénarios pour la modification d'un événement.

15h15: Je finis de rédiger les scénarios de tests de la modification d'événement et je me mets à l'implémenter dans le site.

16h00: J'ai un entretien avec Mme Mota afin de voir mon avancement.

Lors de cet entretien j'ai pu demander à Mme Mota si je pouvais modifier la structure du site afin que ce soit plus pratique pour moi. Elle a accepté car aucune ressource n'est impactée par ma modification.

On a aussi vu que ma structure des scénarios était fausse, j'oubliais de mettre la partie **quand** dans la partie *Situation*.

17h20: Fin de l'entretien avec Mme Mota et je finis ma journée en même temps.

Bilan

Aujourd'hui j'ai pu faire ce qui était prévu sans rencontrer de problème. C'est lors de l'entretien avec Mme Mota qu'on s'est rendu compte que la rédaction de mes scénarios était fausse donc je vais les corriger demain.

Au début mes scénarios ressemblaient à ceci :

Nom	1.1 Crédation d'un nouveau compte (toutes les données, données valides)
User Story	S.I Inscription
Situation	Etant donné que je suis un utilisateur anonyme et que je ne possède pas de compte quand j'arrive sur la page d'accueil du site et que j'ouvre la page d'inscription et que je rentre mon pseudo Nicolas.hr et que je rentre mon prénom Nicolas et que je rentre mon nom Hoarau et que je rentre mon email nico1as.hr@eduge.ch et que je rentre mon mot de passe Super2012 et que je rentre la confirmation de mon mot de passe Super2012 et je rentre mon numéro de téléphone 123 456 78 90 et que je clique sur le bouton INSCRIPTION alors je suis redirigé vers la page Connexion .
Résultat obtenu	Un message de confirmation apparaît au centre de la page avec le texte Le compte a bien été crée. et je suis redirigé sur la page Connexion .
Statut	✓ OK

Mais à la fin il ressemblaient à ceci :

Nom	9.1 Validation par mail pour un événement privé (situation valides)
User Story	S.7 Validation participation à un événement
Situation	Etant donné que je suis un utilisateur connecté et je vais participer à l'événement Mon anniversaire et que j'appuie sur le bouton S'INSCRIRE alors un mail pour confirmer ma participation s'envoie.
Résultat obtenu	Je clique sur le bouton S'INSCRIRE et je reçois un email qui confirme ma participation disant Vous participez à l'événement "Mon anniversaire" prévu de 2020-08-01 14:00:00 à 2020-08-02 02:00:00. à l'événement un message de validation apparaît en haut à droite disant Vous êtes bien inscrit à l'événement "20km de Paris"
Statut	✓ OK

Jeudi 04 Juin 2020

Objectifs

Les objectifs de la journée sont de finir la modification d'un événement, corriger la rédaction des scénarios et commencer la suppression du profile.

Déroulement

08h00: Je commence ma journée en travaillant sur la correction des scénarios rédigés.

09h00: J'ai un entretien avec mes experts afin de voir mon avancement dans mon TPI.

9h10: Fin de l'entretien et je me remets à travailler sur la correction des scénarios.

10h10: Je finis de corriger les scénarios de tests et je reprends la modification d'un événement.

10h50: Je demande à Mme Mota:

❓ Pour la modification d'un événement est-ce qu'on peut le passer de privé à public ou de public à privé ?

Réponse: Ce n'est pas dans le cahier de charges.

12h00: Je prends ma pause midi.

13h15: Je finis ma pause midi et je reprends l'implémentation de la modification d'un événement.

14h30: Je finis l'implémentation et les tests de la modification d'un événement et je commence la rédaction de la suppression du profil.

14h55: Je finis la rédaction des scénarios pour la suppression du profil et je commence à implémenter la suppression du profile.

17h00: Je finis ma journée.

Bilan

Aujourd'hui j'ai pu faire toutes les tâches qui étaient prévues. Je n'ai pas eu de point bloquant, mais une interrogation sur la modification des événements, car je ne savais pas si on pouvait passer un événement de public à privé et inversément, mais Mme Mota m'a dit que ce n'était pas dans le cahier des charges.

Vendredi 05 Juin 2020

Objectifs

Les objectifs de la journée sont finir la suppression du profile et la faire la modification du profile.

Déroulement

08h00: Je commence ma journée en travaillant sur la suppression du profile.

09h35: Je finis la suppression du profile. Et je commence la rédaction des scénarios pour la modification du profile.

10h15: Je finis la rédaction des scénarios pour la modification du profile. Et je commence l'implémentation de la modification du profile.

11h05: J'ai un entretien avec Mme Mota afin de voir mon avancement.

Lors de l'entretien j'ai pu demandé à Mme Mota:

↗ Je ne vois pas ce qui est attendu dans le point A18 9 veut dire, pourriez-vous m'aider à comprendre ?

| Ce qui est attendu dans ce point c'est qu'un utilisateur soit pouvoir créer un événement privé.

12h00: Fin de l'entretien avec Mme Mota et je me remets à travailler sur la modification du profil.

13h10: Je prends ma pause midi.

14h05: Je finis ma pause midi et je reprends la modification.

14h50: Je finis d'implémenter et de tester la modification et je reprends la modification des événements car je n'avais pas fait en sorte que l'on puisse changer l'image de l'événement.

15h50: Je finis la modification des événements et je rajoute une quatrième zone à la page **Gérer mes événements** afin d'avoir une partie qui contient tous les événements qu'on a créé.

16h30: Je finis l'implémentation de la quatrième zone. Et je relis mon énoncé du TPI afin d'être sûr que tout ce qui est demandé est complété.

16h40: Je finis de relire mon énoncé et je vois qu'il me manque l'affichage de la liste des invités pour le créateur d'un événement privé alors, je commence à faire l'affichage de la liste des invités pour le créateur d'un événement privé.

17h10: Je finis d'implémenter l'affichage de la liste d'invités pour le créateur d'un événement privé. Et je commence à créer un logo pour le site.

17h30: Je finis le logo du site et je commence à faire une icône pour le site.

17h40: Je finis l'icône pour le site et je commence l'implémentation le logo et l'icône dans le site.

17h55: Je finis d'implémenter les le logo et l'icône dans le site et je finis ma journée.

Bilan

Aujourd'hui j'ai pu finir l'implémentation des fonctionnalités et j'ai même pu faire un logo pour le site. Après la relecture de l'énoncé j'ai pu demander à Mme Mota ce que voulais dire le point A18 9 car l'énoncé dit : **9.un utilisateur authentifié**
un utilisateur authentifié.

Lundi 08 Juin 2020

Objectifs

Les objectifs de la journée sont de faire un test de tout le site afin d'être sûr que toutes les fonctionnalités fonctionnent, de faire la partie structure dans la documentation.

Déroulement

08h00: Je commence ma journée en testant le site.

08h05: Je vois qu'on ne peut pas supprimer d'événement privé à cause de la liste d'invités.

08h20: Je finis la suppression d'un événement privé et je reprends le test de toutes les fonctionnalités du site.

09h10: Je commence à implémenter la page **But du site**.

09h55: Fin de l'implémentation de la page **But du site**. Et je commence à rédiger la partie structure dans la documentation.

11h00: J'ai un entretien avec Mme Mota afin de voir mon avancement.

11h35: Fin de l'entretien avec Mme Mota. Et je reprends la structure de la rédaction du projet.

13h35: Je finis la rédaction de la structure du projet et je prends ma pause midi.

14h05: Je finis ma pause du midi et je mets le schéma de la base de données dans la documentation.

14h35: Je finis de mettre à jour la base de données dans la documentation et je commence la description de l'application.

15h50: Je finis de mettre la description de l'application et je commence à rédiger l'analyse des fonctionnalités majeures.

17h10: Je finis ma journée.

Bilan

Aujourd'hui j'ai pu faire le test de toute l'application WEB et corriger les quelques bugs que j'ai trouvés, j'ai aussi bien avancé dans ma documentation mais je n'ai pas eu le temps de finir la rédaction des fonctionnalités majeures donc je vais finir cette tâche demain.

Mardi 09 Juin 2020

Objectifs

Les objectifs de la journée sont finir la documentation technique, la documentation utilisateur et rendre le projet.

Déroulement

08h00: Je commence ma journée en reprenant le travail sur la rédaction de l'analyse des fonctionnalités majeures.

8h15: N'arrivant pas à avancer sur ce sujet je décide de passer à la rédaction de la conclusion.

10h15: Je finis de rédiger la conclusion et rédige le glossaire.

11h00: Je finis de rédiger le glossaire et je rédige la documentation utilisateur.

12h00: J'ai un entretien avec Mme Mota afin de voir mon avancement.

12h20: Je mets en pause la rédaction du manuel utilisateur pour reprendre la rédaction l'analyse des fonctionnalités majeures.

12h50: Je finis la rédaction de l'analyse des fonctionnalités majeures et je reprends la documentation utilisateur.

13h00: Je prends ma pause midi.

13h30: Je finis ma pause midi. Et je reprends la rédaction de la documentation utilisateur.

14h45: En voulant faire une capture d'écran pour la suppression du profile un message d'erreur est apparu. Et je commence à résoudre le bug.

15h00: Je finis de corriger le bug et je reprends la documentation utilisateur.

15h05: Je finis la documentation utilisateur et je commence la correction de l'orthographe et de la grammaire pour les documentations et le journal de bord.

16h10: Je finis la correction des documentations et du journal de bord et, je commence à exporter mon code en pdf.

16h40: Je finis de mettre le code en pdf et je commence à assembler la documentation technique avec l'énoncé, le journal de bord et le code source.

17h00: Je rends la documentation technique et le manuel utilisateur à mes experts.

Bilan

Cette journée aura été stressante pour moi car, j'ai tout juste eu le temps de finaliser le projet.

Code source

```
{  
    "_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"  
            ],  
        },  
    ]  
}
```

```
51     "authors": [
52         {
53             "name": "Marcus Bointon",
54             "email": "phpmailer@synchromedia.co.uk"
55         },
56         {
57             "name": "Jim Jagielski",
58             "email": "jimjag@gmail.com"
59         },
60         {
61             "name": "Andy Prevost",
62             "email": "codeworxtech@users.sourceforge.net"
63         },
64         {
65             "name": "Brent R. Matzelle"
66         }
67     ],
68     "description": "PHPMailer is a full-featured email creation and transfer
class for PHP",
69     "funding": [
70         {
71             "url": "https://marcus.bointon.com/donations/",
72             "type": "custom"
73         },
74         {
75             "url": "https://github.com/Synchro",
76             "type": "github"
77         },
78         {
79             "url": "https://www.patreon.com/marcusbointon",
80             "type": "patreon"
81         }
82     ],
83     "time": "2020-03-14T14:23:48+00:00"
84   }
85 ],
86 "packages-dev": [],
87 "aliases": [],
88 "minimum-stability": "stable",
89 "stability-flags": [],
90 "prefer-stable": false,
91 "prefer-lowest": false,
92 "platform": [],
93 "platform-dev": [],
94 "plugin-api-version": "1.1.0"
95 }
```

```
1 | {  
2 |     "require": {  
3 |         "phpmailer/phpmailer": "^6.1"  
4 |     }  
5 | }
```

```
1 .gitignore
2 .vscode
3 /vendor/
4 /database/schema.mwb.bak
5 /views/assets/upload/
_
```

```
<?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'; ?>

    
    <div class="container">
        <div class="img">
```

```
54     
55 </div>
56 <div class="content">
57     <form method="post">
58         
59         <h2 class="title">Inscription</h2>
60         <div class="input-div one">
61             <div class="i">
62                 <i class="fas fa-user"></i>
63             </div>
64             <div class="div">
65                 <h5>Nom</h5>
66                 <input type="text" class="input" id="lastname" maxlength="50">
67             </div>
68         </div>
69         <div class="input-div one">
70             <div class="i">
71                 <i class="fas fa-user"></i>
72             </div>
73             <div class="div">
74                 <h5>Prénom</h5>
75                 <input type="text" class="input" id="firstname" maxlength="50">
76             </div>
77         </div>
78         <div class="input-div one">
79             <div class="i">
80                 <i class="fas fa-user"></i>
81             </div>
82             <div class="div">
83                 <h5>Nom d'utilisateur</h5>
84                 <input type="text" class="input" id="nickname" maxlength="50">
85             </div>
86         </div>
87         <div class="input-div one">
88             <div class="i">
89                 <i class="fa fa-envelope" aria-hidden="true"></i>
90             </div>
91             <div class="div">
92                 <h5>Email</h5>
93                 <input type="email" class="input" id="email" maxlength="100">
94             </div>
95         </div>
96         <div class="input-div one">
97             <div class="i">
98                 <i class="fas fa-phone-alt"></i>
99             </div>
100            <div class="div">
101                <h5>Téléphone</h5>
102                <input type="phone" class="input" id="phoneNumber" maxlength="50">
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" value="Inscription">
124 </form>
125 </div>
126 </div>
127
128 <!-- JAVASCRIPT INCLUDES -->
129 <?php include "./includes/footer.inc.html"; ?>
130
131 <script src="./js/register.js"></script>
132 </body>
133
134 </html>
1--
```

```
<?php
/*
 * Auteur      : Hoarau Nicolas
 * Projet     : WE GO
 * Page       : profile.php
 * Date       : JJ.03.2020
 * Description : Page qui affiche les données 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' ?>


<div class="container">
    <div class="img">
        
    </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
/*
 * 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
/*
 * 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'; ?>

    
    <div class="container">
        <div class="img">
            
```

```
54     </div>
55     <div class="content">
56         <form method="post">
57             
58             <h2 class="title">Bienvenue</h2>
59             <div class="input-div one">
60                 <div class="i">
61                     <i class="fas fa-user"></i>
62                 </div>
63                 <div class="div">
64                     <h5>Nom d'utilisateur ou Email</h5>
65                     <input type="text" class="input" id="authenticator">
66                 </div>
67             </div>
68             <div class="input-div pass">
69                 <div class="i">
70                     <i class="fas fa-lock"></i>
71                 </div>
72                 <div class="div">
73                     <h5>Mot de passe</h5>
74                     <input type="password" class="input" id="password">
75                 </div>
76             </div>
77             <input type="submit" id="btnLogUser" class="btn" value="Connexion">
78         </form>
79     </div>
80 </div>
81
82 <?php include './includes/footer.inc.html'; ?>
83
84 <script type="text/javascript" src=".js/login.js"></script>
85 </body>
86
87 </html>
~~
```

```
<?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
/*
 * 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/iconWego.svg" type="image/x-icon">

    <title></title>
</head>

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

    
    <div class="container">
        <div class="img">
            
        </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
/*
 * 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/iconWego.svg" type="image/x-icon">

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

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

    
    <div class="container">
        <div class="img">
```

```
54 |         
55 |     </div>
56 |     <div class="content">
57 |         <form method="post">
58 |             
59 |             <h2 class="title">Modifier mon profile</h2>
60 |             <div class="input-div one">
61 |                 <div class="i">
62 |                     <i class="fas fa-user"></i>
63 |                 </div>
64 |                 <div class="div">
65 |                     <h5>Nom</h5>
66 |                     <input type="text" class="input" id="lastname" maxlength="50" value=<?=
   |                         isset($_SESSION['loggedIn']['nom']) ? $_SESSION['loggedIn']['nom'] : '' ?>">
67 |                 </div>
68 |             </div>
69 |             <div class="input-div one">
70 |                 <div class="i">
71 |                     <i class="fas fa-user"></i>
72 |                 </div>
73 |                 <div class="div">
74 |                     <h5>Prénom</h5>
75 |                     <input type="text" class="input" id="firstname" maxlength="50" value=<?=
   |                         $_SESSION['loggedIn']['prenom'] ?>">
76 |                 </div>
77 |             </div>
78 |             <div class="input-div one">
79 |                 <div class="i">
80 |                     <i class="fas fa-user"></i>
81 |                 </div>
82 |                 <div class="div">
83 |                     <h5>Nom d'utilisateur</h5>
84 |                     <input type="text" class="input" id="nickname" maxlength="50" value=<?=
   |                         $_SESSION['loggedIn']['pseudo'] ?>">
85 |                 </div>
86 |             </div>
87 |             <div class="input-div one">
88 |                 <div class="i">
89 |                     <i class="fa fa-envelope" aria-hidden="true"></i>
90 |                 </div>
91 |                 <div class="div">
92 |                     <h5>Email</h5>
93 |                     <input type="email" class="input" id="email" maxlength="100" value=<?=
   |                         $_SESSION['loggedIn']['email'] ?>">
94 |                 </div>
95 |             </div>
96 |             <div class="input-div one">
97 |                 <div class="i">
98 |                     <i class="fas fa-phone-alt"></i>
99 |                 </div>
100 |                 <div class="div">
101 |                     <h5>Téléphone</h5>
102 |                     <input type="phone" class="input" id="phoneNumber" maxlength="50"
   |                         value=<?=
   |                         isset($_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
/*
 * 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">
```

```
49 <meta name="viewport" content="width=device-width, initial-scale=1.0">
50
51 <!-- STYLE DU SITE -->
52 <link rel="stylesheet" href="./css/style.css">
53 <link rel="stylesheet" href="./css/media.css">
54
55 <!-- SWEETALERT2 -->
56 <link rel="stylesheet" href="./css/sweetalert2.css">
57
58 <!-- GOOGLE FONTS -->
59 <link href="https://fonts.googleapis.com/css?family=Poppins:600&display=swap"
60   rel="stylesheet">
61
62 <!-- FONTAWESOME -->
63 <script src="https://kit.fontawesome.com/a81368914c.js"></script>
64
65 <link rel="shortcut icon" href="./assets/img/iconWego.svg" type="image/x-icon">
66
67 <title>Modifier mon événement</title>
68 </head>
69
70 <body>
71   <?php include './includes/navbar.inc.php' ?>
72
73   
74   <div class="container">
75     <div class="img">
76       
77     </div>
78
79     <div class="content">
80       <form method="post">
81         
82         <h2 class="title">Modifier mon événement</h2>
83         <div class="input-div one">
84           <div class="i">
85             <i class="fa fa-calendar" aria-hidden="true"></i>
86           </div>
87           <div class="div">
88             <h5>Nom de l'événement</h5>
89             <input type="text" class="input" id="eventName" maxlength="50" value="<?=
90 $EventData[ 'nom' ] ?>">
91             </div>
92           </div>
93           <div class="input-div one">
94             <div class="i">
95               <i class="fas fa-pen"></i>
96             </div>
97             <div class="div">
98               <h5>Descriptif de l'événement</h5>
99               <input type="text" class="input" id="eventDescription" maxlength="255"
100      value="<?=$EventData[ 'descriptif' ] ?>">
101             </div>
102           </div>
103           <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="<?= $eventBeginningDate ?>" />
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="<?= $eventBeginningTime ?>" />
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="<?=
152      $eventData['nbMaxParticipant'] ?>" />
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
/*
 * 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' ?>

    
```

```
54 | <div class="container">
55 |   <div class="img">
56 |     
57 |   </div>
58 |
59 |   <div class="content">
60 |     <form method="post">
61 |       
62 |       <h2 class="title">Créer un événement</h2>
63 |       <div class="input-div one">
64 |         <div class="i">
65 |           <i class="fa fa-calendar" aria-hidden="true"></i>
66 |         </div>
67 |         <div class="div">
68 |           <h5>Nom de l'événement</h5>
69 |           <input type="text" class="input" id="eventName" maxlength="50">
70 |         </div>
71 |       </div>
72 |       <div class="input-div one">
73 |         <div class="i">
74 |           <i class="fas fa-pen"></i>
75 |         </div>
76 |         <div class="div">
77 |           <h5>Descriptif de l'événement</h5>
78 |           <input type="text" class="input" id="eventDescription" maxlength="255">
79 |         </div>
80 |       </div>
81 |       <div class="input-div one">
82 |         <div class="i">
83 |           <i class="fas fa-map-marker-alt"></i>
84 |         </div>
85 |         <div class="div">
86 |           <h5>Lieu</h5>
87 |           <input type="text" class="input" id="place" maxlength="255">
88 |         </div>
89 |       </div>
90 |       <div class="input-div one">
91 |         <div class="i">
92 |           <i class="fas fa-calendar-alt"></i>
93 |         </div>
94 |         <div class="div">
95 |           <h5>Date de début</h5>
96 |           <input class="input" id="beginningDate" type="text" onfocus="
  (this.type='date')" value=<?= $datetime->format('Y-m-d'); ?>" />
97 |         </div>
98 |       </div>
99 |       <div class="input-div one">
100 |         <div class="i">
101 |           <i class="fas fa-clock"></i>
102 |         </div>
103 |         <div class="div">
104 |           <h5>Heure de début</h5>
105 |           <input class="input" id="beginningTime" type="text" onfocus="
  (this.type='time')" value=<?= date('H:i'); ?>" />
106 |         </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      />
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>
165
```

```
<?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'; ?>

    
    <div class="container">
        <div class="img">
            
        </div>
        <div class="content">
            <div class="container-about">
                <div class="div about">Ce site permet de créer des événements privés ou

```

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

```
1 <?php
2 /*
3  * Auteur      : Hoarau Nicolas
4  * Projet     : WE GO
5  * Page       : 404.php
6  * Date       : 02.06.2020
7  * Description : Page qui affiche l'erreur 404
8  * Version    : 1.0
9  */
10
11 require_once dirname(__DIR__) . '/App/php/backend.php';
12 ?>
13 <!DOCTYPE html>
14 <html lang="en">
15
16 <head>
17   <meta charset="UTF-8">
18   <meta name="viewport" content="width=device-width, initial-scale=1.0">
19   <!-- STYLE DU SITE -->
20   <link rel="stylesheet" href="./css/style.css">
21   <link rel="stylesheet" href="./css/media.css">
22
23   <!-- SWEETALERT2 -->
24   <link rel="stylesheet" href="./css/sweetalert2.css">
25
26   <!-- GOOGLE FONTS -->
27   <link href="https://fonts.googleapis.com/css?family=Poppins:600&display=swap"
rel="stylesheet">
28
29   <!-- FONTAWESOME -->
30   <script src="https://kit.fontawesome.com/a81368914c.js"></script>
31
32   <link rel="shortcut icon" href="./assets/img/iconewego.svg" type="image/x-icon">
33
34   <title>Page Not Found</title>
35 </head>
36
37 <body>
38   <?php include './includes/navbar.inc.php'; ?>
39
40   
41
42 </body>
43
44 </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 via 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,
```

```
54     firstname: firstname,
55     nickname: nickname,
56     email: email,
57     phoneNumber: phoneNumber,
58     password: password,
59     verifyPassword: verifyPassword
60   },
61   dataType: "json",
62   success: (response) => {
63     switch (response.ReturnCode) {
64       case 0:
65         Swal.fire({
66           title: "Inscription",
67           text: response.Success,
68           icon: "success",
69           timer: 1300,
70           showConfirmButton: false
71         }).then(() => {
72           window.location.href = './login.php'
73         });
74         break;
75       case 1:
76       case 2:
77       case 3:
78       case 4:
79       case 5:
80         Swal.fire({
81           title: "Inscription",
82           text: response.Error,
83           icon: "error",
84           timer: 1500,
85           showConfirmButton: false
86         });
87         break;
88       }
89     },
90   error: (error) => {
91     console.error(error);
92   }
93 });
^..,
```

```
$(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 informations 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
```

```
55 * @brief Fonction qui affiche les informations de l'utilisateur
56 * @param {array} data
57 */
58 function ShowProfileData(data) {
59     document.title = data.pseudo;
60
61     let html = `<div class="event-data">
62         
63         <h2 class="title">${data.pseudo}</h2>
64         <div class="div">${data.nom == null ? '' : data.nom} ${data.prenom}</div>
65         <div class="div">${data.email}</div>
66         <div class="div place">${data.telephone == null ? '' : data.telephone}</div>
67         <button class="btn btn-edit" onclick="window.location.href =
68             './editProfile.php?idUser=${data.idUser}'">Modifier</button>
69         <button class="btn btn-delete"
70             onclick="DeleteProfile(${data.idUser})">Supprimer</button>;
71
72     html += `</div>`;
73 }
74
75 /**
76 * @author Hoarau Nicolas
77 * @date 04.06.20
78 * @brief Fonction qui affiche les informations de l'utilisateur
79 * @param {int} idUser
80 */
81 function DeleteProfile(idUser) {
82     $.ajax({
83         type: "post",
84         url: "../App/php/isProfileDeletable.php",
85         data: { idUser: idUser },
86         dataType: "json",
87         success: (response) => {
88             switch (response.ReturnCode) {
89                 case 0:
90                     Swal.fire({
91                         title: "Suppression du profile",
92                         text: response.Success,
93                         icon: "question",
94                         showConfirmButton: true,
95                         confirmButtonText: 'Valider',
96                         showCancelButton: true,
97                         cancelButtonText: 'Annuler',
98                         reverseButtons: true
99                     }).then((result) => {
100                         if (result.value) {
101                             $.ajax({
102                                 type: "post",
103                                 url: "../App/php/deleteProfile.php",
104                                 data: { idUser: idUser },
105                                 dataType: "json",
106                                 success: (data) => {
107                                     switch (data.ReturnCode) {
```

```
108         case 0:
109             Swal.fire({
110                 title: "Suppression du profile",
111                 text: data.Success,
112                 icon: "success",
113                 timer: 13000,
114                 showConfirmButton: false,
115             }).then(() => {
116                 window.location.href = '../App/php/logout.php';
117             });
118             break;
119         case 1:
120             Swal.fire({
121                 title: "Suppression du profile",
122                 text: data.Error,
123                 icon: "error",
124                 timer: 1500,
125                 showConfirmButton: false,
126             });
127             break;
128         }
129     },
130     error: (error) => {
131         console.error(error);
132     }
133 );
134 }
135 );
136 break;
137 case 1:
138     Swal.fire({
139         title: "Suppression du profile",
140         text: response.Error,
141         icon: "error",
142         timer: 1800,
143         showConfirmButton: false
144     });
145     break;
146 }
147 },
148 error: (error) => {
149     console.error(error);
150 }
151 );
152 }
153 /**
154 * @author Hoarau Nicolas
155 * @date 04.06.20
156 * @brief Fonction qui affiche les informations de l'utilisateur
157 * @param {*} event
158 */
159 */
160 function EditProfile(event) {
161     if (event)
162         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 });
~~~`
```

```
1 // Constantes login
2 const inputs = document.querySelectorAll(".input");
3
4 // Constantes navbar
5 const hamburger = document.querySelector('.hamburger');
6 const navlinks = document.querySelector('.nav-links');
7 const links = document.querySelectorAll('.nav-links li');
8
9 /**
10 * @brief ajoute le focus sur l'input cliqué
11 */
12 function addcl() {
13     let parent = this.parentNode.parentNode;
14     parent.classList.add("focus");
15 }
16
17 /**
18 * @brief fonction qui retire le focus de l'input
19 */
20 function remcl() {
21     let parent = this.parentNode.parentNode;
22     if (this.value == "") {
23         parent.classList.remove("focus");
24     }
25 }
26
27 // ajoute aux input les evnt focus et blur avec addcl et remcl en fontion
28 inputs.forEach(input => {
29     input.addEventListener("focus", addcl);
30     input.addEventListener("blur", remcl);
31 });
32
33 // ajoute l'event click au hamburger et fait l'affichage de la navbar
34 hamburger.addEventListener('click', () => {
35     let content = document.getElementsByClassName('content')[0];
36
37     navlinks.classList.toggle('open');
38
39     content.style.display == 'none' ? content.style.display = 'flex' :
40     content.style.display = 'none';
41     links.forEach(link => {
42         link.classList.toggle('fade');
43     })
44 })
```

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

/**
 * @author Hoarau Nicolas
 * @date 20.03.20
 * @brief Fonction qui récupère 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";
});

/**
```

```
54 * @author Hoarau Nicolas
55 * @date 27.05.20
56 * @brief Fonction qui envoie les données pour créer un événement via un call ajax
57 * @param {*} event
58 * @version 1.0
59 */
60 async function CreateEvent(event) {
61   if (event)
62     event.preventDefault();
63
64 //##region Initialisation
65 let formdata = new FormData();
66 let eventName = $("#eventName").val();
67 let place = $("#place").val();
68 let eventDescription = $("#eventDescription").val();
69 let nbMaxGuest = $("#nbMaxGuest").val();
70 let type = $('input[name="type"]:checked').val();
71 let img = document.getElementById("img").files[0];
72 let beginningDate = new Date($("#beginningDate").val());
73 let endDate = new Date($("#endDate").val());
74 let beginningTime = $("#beginningTime").val();
75 let endTime = $("#endTime").val();
76 let now = new Date();
77
78 beginningDate.setHours(beginningTime.split(':')[0], beginningTime.split(':')[1], 0, 0);
79 endDate.setHours(endTime.split(':')[0], endTime.split(':')[1], 0, 0);
80 now.setHours(now.getHours(), now.getMinutes(), 0, 0);
81 //##endregion
82
83 //##region Conditions
84 if (eventName.length == 0) {
85   $("#eventName").focus();
86   return;
87 } else {
88   formdata.append("eventName", eventName);
89 }
90
91 if (eventDescription.length == 0) {
92   $("#eventDescription").focus();
93   return;
94 } else {
95   formdata.append("eventDescription", eventDescription);
96 }
97
98 if (place.length == 0) {
99   $("#place").focus();
100  return;
101 } else {
102   formdata.append("place", place);
103 }
104
105 if (nbMaxGuest == 0 || nbMaxGuest == "") {
106   $("#nbMaxGuest").focus();
107   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 choisie 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 choisie 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     });
164 }
```

```
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
216 * l'événement
217 * @return le tableau de guests
```

```
217 * @version 1.0
218 */
219 function GetGuestList() {
220     if (event)
221         event.preventDefault();
222
223     let guests = [];
224
225     // Récupère les invités
226     return fetch("../App/php/getGuestList.php", {
227         method: 'POST',
228         headers: {
229             'Accept': 'application/json',
230             'Content-Type': 'application/json'
231         }
232     }).then(response => response.json())
233     .then(async data => {
234         let html = `<div class="menu"><ul>`;
235
236         $.each(data, (key, user) => {
237             html += `<li>
238                 <label>
239                     <input type="checkbox" value="${user.idUser}">
240                     <span class="icon"></span>
241                     <span class="list">${user.pseudo}</span>
242                 </label>
243             </li>`;
244         });
245
246         html += `</ul></div>`;
247
248         // Attend la validation de la modal par l'utilisateur et récupère les invités
249         coché
250         guests = await Swal.fire({
251             title: "Liste d'invités",
252             html: html,
253             showCancelButton: true,
254             cancelButtonText: 'Annuler',
255             confirmButtonText: 'Valider'
256         }).then((result) => {
257             let modalGuest = [];
258             // Récupère les case cochées de la modal
259             let pathToLi =
260                 Swal.getContent().childNodes[0].childNodes[0].children[0].children;
261
262             // Pour chaque <li>
263             for (let i = 0; i < pathToLi.length; i++) {
264                 const checkbox = pathToLi[i].children[0].children[0];
265
266                 if (checkbox.checked == true) {
267                     modalGuest.push(checkbox.value)
268                 }
269             }
270             return modalGuest;
271         });
272     });
273 }
```

```
270         return guests;
271     });
272 }
273
274 /**
275 * @author Hoarau Nicolas
276 * @date 28.05.20
277 * @brief Fonction qui récupère les événement public 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 public qui ne sont pas encore passé,
298 ainsi que les événements auxquels l'utilisateur participe et est invités
299 * @version 1.0
300 */
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="${event.idEvenement}"`+
329             `onclick="window.location.href = './event.php?idEvent=${event.idEvenement}'">`+
330             `<div class="card-image"></div>`+
331             `<div class="card-text">`+
332             `<h2>${event.nom}</h2>`+
333             `<span class="date">Par ${event.pseudo} <br/> de ${event.dateDebut} à`+
334             `${event.dateFin}</span>`+
335             `<p>${event.descriptif}</p>`+
336             `</div></div>`;
337 }
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 publics qui ne sont pas encore passé,
347 ainsi que les événements auxquels l'utilisateur participe et est invités
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 créés</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
377     catégorie</p></div>`;
378
379     for (let j = 0; j < eventZone.length; j++) {
380         const event = eventZone[j];
381
382         html += `<div class="card" id="${event.idEvenement}">
383             <div class="card-image" onclick="window.location.href =
384                 './event.php?idEvent=${event.idEvenement}'"></div>
385             <div class="card-text">
386                 <h2>${event.nom}</h2>
387                 <span class="date">Par ${event.pseudo} <br/> de ${event.dateDebut} à
388                 ${event.dateFin}</span>
389                 <p>${event.descriptif}</p>
390             </div>`;
391
392         if (data.nickname != event.pseudo) {
393             if (i === 2) {
394                 html += `<button id="participate" class="btn"
395                     onclick="Participate(${event.idEvenement}, '${event.nom}', '${event.dateDebut}',
396                     '${event.dateFin}')>Se désinscrire</button></div>`;
397             } else {
398                 html += `<button id="participate" class="btn"
399                     onclick="Participate(${event.idEvenement}, '${event.nom}', '${event.dateDebut}',
400                     '${event.dateFin}')>S'inscrire</button></div>`;
401             }
402         }
403         html += `</div><div class="card-container">`;
404     }
405 }
406
407 /**
408 * @author Hoarau Nicolas
409 * @date 28.05.20
410 * @brief Fonction qui inscrit ou désinscrit 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     },
```

```
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 * @author Hoarau Nicolas
456 * @date 02.06.20
457 * @brief Fonction qui récupère les informations de l'id d'événement reçus
458 * @param {int} idEvent
459 */
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             }
479         }
480     })
481 }
```

```
479     },
480     error: (error) => {
481       console.error(error);
482     }
483   });
484 }
485
486 /**
487 * @author Hoarau Nicolas
488 * @date 02.06.20
489 * @brief Fonction qui affiche les données d'un événement choisis
490 * @param {array} data
491 */
492 function ShowEventData(data) {
493   document.title = data.nom;
494
495   let html = `<div class="event-data">
496     
497     <h2 class="title">${data.nom}</h2>
498     <div class="div place">${data.prive == 1 ? 'Privé' : 'Publique'}</div>
499     <div class="div date">De ${data.dateDebut} à ${data.dateFin}<br> Par ${data.pseudo}</div>
500     <div class="div">${data.descriptif}</div>
501     <div class="div place">${data.lieu}</div>
502     <div class="div">${data.nbGuest} sur ${data.nbMaxParticipant} participants <a id="showGuests" onclick="GetEventGuestList(${data.idEvenement})">Voir les participants</a></div>`;
503
504   if (data.isLoggedIn === true) {
505     if (data.nickname === data.pseudo) {
506       if (data.prive == 1)
507         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>`;
508
509       html += `<button class="btn btn-edit" onclick="window.location.href = './editEvent.php?idEvent=${data.idEvenement}'>Modifier</button>
510         <button class="btn btn-delete" onclick="DeleteEvent(${data.idEvenement}, '${data.nom}', '${data.prive}')>Supprimer</button>`;
511     } else {
512       if (data.participating == 1) {
513         html += `<button class="btn" onclick="Participate(${data.idEvenement}, '${data.nom}', '${data.dateDebut}', '${data.dateFin}')>Se désinscrire</button>`;
514       } else {
515         if (data.nbGuest < data.nbMaxParticipant) {
516           html += `<button class="btn" onclick="Participate(${data.idEvenement}, '${data.nom}', '${data.dateDebut}', '${data.dateFin}')>S'inscrire</button>`;
517         }
518       }
519     }
520   } else {
521     Toast.fire({
522       icon: 'info',
523       title: "Vous ne pouvez pas participer à l'événement sans être connecté"
524     })
525   }
526 }
```

```
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 voulu
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     break;
670 
```

```
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 /**
```

```
689 * @author Hoarau Nicolas
690 * @date 04.06.20
691 * @brief Fonction qui envoie les données pour modifier un événement
692 * @param {*} event
693 * @param {int} idEvent
694 */
695 function EditEvent(event, idEvent) {
696     if (event)
697         event.preventDefault();
698
699     let formdata = new FormData();
700     let eventName = $("#eventName").val();
701     let place = $("#place").val();
702     let eventDescription = $("#eventDescription").val();
703     let nbMaxGuest = $("#nbMaxGuest").val();
704     let img = document.getElementById("img").files[0];
705     let beginningDate = new Date($("#beginningDate").val());
706     let endDate = new Date($("#endDate").val());
707     let beginningTime = $("#beginningTime").val();
708     let endTime = $("#endTime").val();
709     let now = new Date();
710
711     beginningDate.setHours(beginningTime.split(':')[0], beginningTime.split(':')[1], 0,
712     0);
712     endDate.setHours(endTime.split(':')[0], endTime.split(':')[1], 0, 0);
713     now.setHours(now.getHours(), now.getMinutes(), 0, 0);
714     formdata.append('idEvent', idEvent);
715 //##endregion
716
717 //##region Conditions
718 if (eventName.length == 0) {
719     $("#eventName").focus();
720     return;
721 } else {
722     formdata.append("eventName", eventName);
723 }
724
725 if (eventDescription.length == 0) {
726     $("#eventDescription").focus();
727     return;
728 } else {
729     formdata.append("eventDescription", eventDescription);
730 }
731
732 if (place.length == 0) {
733     $("#place").focus();
734     return;
735 } else {
736     formdata.append("place", place);
737 }
738
739 if (nbMaxGuest == 0 || nbMaxGuest == "") {
740     $("#nbMaxGuest").focus();
741     return;
742 } 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 choisie 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 choisie 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
892 * à l'événement donnée
893 * @param {int} idEvent
894 */
895 function GetEditGuestList(idEvent) {
896     $.ajax({
897         type: "post",
898         url: "../App/php/getEditGuestList.php",
899         data: { idEvent: idEvent },
900         dataType: "json",
901         success: (response) => {
902             switch (response.ReturnCode) {
903                 case 0:
904                     ShowEditGuestList(response.Data, idEvent);
905                     break;
906                 case 1:
907                     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 * @author Hoarau Nicolas
923 * @date 05.06.20
924 * @brief Fonction qui affiche la modal pour la modification de la liste des invités et envoie les données de modification
925 * @param {array} data
926 * @param {int} idEvent
927 */
928 function ShowEditGuestList(data, idEvent) {
929     let html = `<div class="menu"><ul>`;
930
931     $.each(data, (key, user) => {
932         html += `<li>
933             <label>
934                 <input type="checkbox" value="${user.idUser}" ${user.invited == 1 ? 'checked' : ''}>
935                 <span class="icon"></span>
936                 <span class="list">${user.pseudo}</span>
937             </label>
938         </li>`;
939     });
940
941     html += `</ul></div>`;
942
943     Swal.fire({
944         title: "Liste d'invités",
945         html: html,
946         showCancelButton: true,
947         cancelButtonText: 'Annuler',
948         confirmButtonText: 'Valider'
949     }).then((result) => {
950         if (result.value) {
951             let editGuestlist = [];
952             // Récupère les cases cochées de la modal
953             let pathToLi =
954                 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 les 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 });
....
```

```
1 <nav>
2   <div class="hamburger">
3     <div class="line"></div>
4     <div class="line"></div>
5     <div class="line"></div>
6   </div>
7   <a href=".index.php"></a>
8   <ul class="nav-links">
9     <li><a href=".index.php">Accueil</a></li>
10    <?php if (IsLogged() == false) : ?>
11      <li><a href=".login.php">Connexion</a></li>
12      <li><a href=".register.php">Inscription</a></li>
13    <?php else : ?>
14      <li><a href=".profile.php">Mon profile</a></li>
15      <li><a href=".manageEvents.php">Gérer mes événements</a></li>
16      <li><a href=".createEvent.php">Créer un événement</a></li>
17      <li><a href="..App/php/logout.php">Déconnexion</a></li>
18    <?php endif; ?>
19      <li><a href=".about.php">But du site</a></li>
20   </ul>
21 </nav>
```

```
1 <!-- IMPORT JQUERY -->
2 <script src="https://code.jquery.com/jquery.min.js"></script>
3 <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.0/jquery.min.js">
4 </script>
5 <!-- IMPORT SWEETALERTS -->
6 <script src="./js/sweetalert2.min.js"></script>
7
8 <!-- IMPORT JS FOR STYLE -->
9 <script type="text/javascript" src="./js/main.js"></script>
```

```
1 <?php
2 /*
3  * Auteur      : Hoarau Nicolas
4  * Projet     : WE GO
5  * Page       : const.inc.php
6  * Date       : 27.05.2020
7  * Description : fichier qui contient les constantes du projet
8  * Version    : 1.0
9  */
10
11 define('UPLOAD_PATH', dirname(dirname(__DIR__)) . '/views/assets/upload/');
12 define('DEFAULT_IMG', './assets/img/default.svg');
```

```
* {
  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;
```

```
55 |   align-items: center;
56 |   text-align: center;
57 }
58
59 .img img {
60   width: 500px;
61 }
62
63 form {
64   width: 360px;
65 }
66
67 .content img {
68   height: 100px;
69 }
70
71 .content h2 {
72   margin: 5px 0;
73   color: #333;
74   text-transform: uppercase;
75   font-size: 2rem;
76 }
77
78 .content .input-div {
79   position: relative;
80   display: grid;
81   grid-template-columns: 7% 93%;
82   margin: 15px 0;
83   padding: 5px 0;
84   border-bottom: 2px solid #d9d9d9;
85 }
86
87 .content .input-div.one {
88   margin-top: 0;
89 }
90
91 .i {
92   color: #d9d9d9;
93   display: flex;
94   justify-content: center;
95   align-items: center;
96 }
97
98 .i i {
99   transition: 0.3s;
100 }
101
102 .input-div > div {
103   position: relative;
104   height: 38px;
105 }
106
107 .input-div > div > h5 {
108   position: absolute;
109   left: 10px;
```

```
110 top: 50%;  
111 transform: translateY(-50%);  
112 color: #999;  
113 font-size: 15px;  
114 transition: 0.3s;  
115 }  
116  
117 .input-div:before,  
118 .input-div:after {  
119   content: "";  
120   position: absolute;  
121   bottom: -2px;  
122   width: 0%;  
123   height: 2px;  
124   background-color: #https://www.wappalyzer.com/technologies/font-scripts;  
125   transition: 0.4s;  
126 }  
127  
128 .input-div:before {  
129   right: 50%;  
130 }  
131  
132 .input-div:after {  
133   left: 50%;  
134 }  
135  
136 .input-div.focus:before,  
137 .input-div.focus:after {  
138   width: 50%;  
139 }  
140  
141 .input-div.focus > div > h5 {  
142   top: -5px;  
143   font-size: 15px;  
144 }  
145  
146 .input-div.focus > .i > i {  
147   color: #38d39f;  
148 }  
149  
150 .input-div > div > input {  
151   position: absolute;  
152   left: 0;  
153   top: 0;  
154   width: 100%;  
155   height: 80%;  
156   border: none;  
157   outline: none;  
158   background: none;  
159   padding: 0.5rem 0.7rem;  
160   font-size: 1.2rem;  
161   color: #555;  
162   font-family: "poppins", sans-serif;  
163 }  
164
```

```
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 }
```

```
***** 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 }
```

```
1 <?php
2 /*
3  * Auteur      : Hoarau Nicolas
4  * Projet      : WE GO
5  * Page        : config.php
6  * Date        : 26.05.2020
7  * Description : Fichier qui contient les constantes de connexion à la base de donnée:
8  * Version     : 1.0
9  */
10
11 // Le Type de la base de donnée
12 define('DB_DBTYPE', "mysql");
13
14 // Connexion local
15 define('DB_HOST', 'localhost');
16 define('DB_DBNAME', 'wego');
17 define('DB_USER', 'root');
18 define('DB_PASS', 'Super');
19 define('DB_PORT', 3306);
20
21 // PHPMailer
22 define('MAIL_USERNAME', 'nicohoarau74@gmail.com');
23 define('MAIL_PASSWORD', "4<Rh#6F+}kPw(kRbGHGU");
24 define('MAIL_HOST', "smtp.gmail.com");
25 define('MAIL_PORT', 587);
26
```

```
<?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 participation";

$mailMessage = <<<EX
Vous participez à l'événement "{$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);
```

```
54     $requestParticipation->execute();
55
56     if ($isParticipating == false) {
57         if (SendMail($email, $mailMessage, $mailSubject) == false) {
58             DatabaseController::rollBack();
59             echo json_encode([
60                 'ReturnCode' => 1,
61                 'Error' => "Une erreur est survenue lors de l'envoie du mail."
62             ]);
63             exit();
64         }
65     }
66
67     DatabaseController::commit();
68
69     echo json_encode([
70         'ReturnCode' => 0,
71         'Success' => $successMessage
72     ]);
73     exit();
74 } catch (PDOException $e) {
75     DatabaseController::rollBack();
76
77     echo json_encode([
78         'ReturnCode' => 1,
79         'Error' => "Erreur est survenue lors de l'actualisation de la participation"
80     ]);
81     exit();
82 }
```

```
<?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 minimum 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([
                    'ReturnCode' => 3,
                    'Error' => 'L\'utilisateur existe déjà dans la base de données.'
                ]);
                exit();
            }
        } else {
            echo json_encode([
                'ReturnCode' => 4,
                'Error' => 'L\'adresse email fournie existe déjà dans la base de données.'
            ]);
            exit();
        }
    } else {
        echo json_encode([
            'ReturnCode' => 5,
            'Error' => 'Le mot de passe doit contenir au moins 8 caractères, une majuscule et une
chiffre.'
        ]);
        exit();
    }
}
```

```
51     'ReturnCode' => 3,
52     'Error' => 'Ce nom d\'utilisateur est déjà utilisé'
53   ]);
54   exit();
55 }
56 } else {
57   echo json_encode([
58     'ReturnCode' => 4,
59     'Error' => 'Cette adresse mail est déjà utilisé'
60   ]);
61   exit();
62 }
63 } else {
64   echo json_encode([
65     'ReturnCode' => 5,
66     'Error' => 'Le mot de passe doit faire au moins 8 caractères et doit contenir au
67     moins une majuscule et un chiffre.'
68   ]);
69   exit();
70 }
```

```
1 <?php
2 /*
3  * Auteur      : Hoarau Nicolas
4  * Projet     : WE GO
5  * Page       : logout.php
6  * Date       : 27.05.2020
7  * Description : Déconnecte l'utilisateur
8  * Version    : 1.0
9  */
10
11 if (session_status() == PHP_SESSION_NONE) {
12     session_start();
13 }
14
15 unset($_SESSION['loggedIn']);
16
17 header('Location: ../../views/index.php');
18 exit();
--
```

```
1 <?php
2 /*
3  * Auteur      : Hoarau Nicolas
4  * Projet     : WE GO
5  * Page       : login.php
6  * Date       : 26.05.2020
7  * Description : Fichier qui permet de filtrer les données reçus du call Ajax afin de
8  * connecter l'utilisateur
9  * Version    : 1.0
10 */
11 require_once __DIR__ . '/backend.php';
12
13 if (session_status() == PHP_SESSION_NONE) {
14     session_start();
15 }
16
17 // Initialisation
18 $authenticator = filter_input(INPUT_POST, "authenticator", FILTER_SANITIZE_STRING);
19 $password = filter_input(INPUT_POST, "password", FILTER_SANITIZE_STRING);
20 $userLogged = null;
21
22 if (strlen($authenticator) > 0 && strlen($password) > 0) {
23     if (strpos($authenticator, '@')) {
24         $userLogged = Login(['userEmail' => $authenticator, 'userPwd' => $password]);
25     } else {
26         $userLogged = Login(['userNickname' => $authenticator, 'userPwd' => $password]);
27     }
28
29     if ($userLogged !== false) {
30         $_SESSION['loggedIn'] = $userLogged;
31
32         echo json_encode([
33             'ReturnCode' => 0,
34             'Success' => "Connexion réussie."
35         ]);
36         exit();
37     }
38
39     echo json_encode([
40         'ReturnCode' => 1,
41         'Error' => "Pseudo/Mot de passe invalide"
42     ]);
43     exit();
44 }
```

```
1 <?php
2 /*
3  * Auteur      : Hoarau Nicolas
4  * Projet     : WE GO
5  * Page       : isProfileDeletable.php
6  * Date        : 04.06.2020
7  * Description : Fichier qui vérifie si le profile est supprimable
8  * Version    : 1.0
9  */
10
11 require_once __DIR__ . './backend.php';
12
13 $idUser = filter_input(INPUT_POST, 'idUser', FILTER_SANITIZE_NUMBER_INT);
14 $usersEvents = GetUsersEvent($idUser);
15
16 if ($usersEvents == 0 || CountUsersGuest($usersEvents) == 0) {
17     echo json_encode([
18         'ReturnCode' => 0,
19         'Success' => 'Voulez-vous vraiment supprimer votre profile ?'
20     ]);
21     exit();
22 } else {
23     echo json_encode([
24         'ReturnCode' => 1,
25         'Error' => "Il ne doit pas avoir de participant à vos événement pour supprimer
votre compte."
26     ]);
27     exit();
28 }
```

```
1 <?php
2 /*
3  * Auteur      : Hoarau Nicolas
4  * Projet      : WE GO
5  * Page        : isEventDeletable.php
6  * Date        : 03.06.2020
7  * Description  : Vérifie si on peut supprimer l'événement.
8  * Version     : 1.0
9 */
10
11 require_once __DIR__ . './backend.php';
12
13 $idEvent = filter_input(INPUT_POST, 'idEvent', FILTER_SANITIZE_NUMBER_INT);
14 $eventName = filter_input(INPUT_POST, 'eventName', FILTER_SANITIZE_STRING);
15
16 if (GetEventNbGuest($idEvent) == 0) {
17     echo json_encode([
18         'ReturnCode' => 0,
19         'Success' => 'Voulez-vous vraiment supprimer ' . $eventName . '?'
20     ]);
21     exit();
22 } else {
23     echo json_encode([
24         'ReturnCode' => 1,
25         'Error' => "L'événement \"". $eventName . "\" ne peut être supprimé car il y a des
participants"
26     ]);
27     exit();
28 }
```

```
1 <?php
2 /*
3  * Auteur      : Hoarau Nicolas
4  * Projet     : WE GO
5  * Page       : getUserData.php
6  * Date       : 03.06.2020
7  * Description : Fichier qui envoie les données de l'utilisateur
8  * Version    : 1.0
9 */
10
11 require_once dirname(__DIR__) . '/Controller/DatabaseController.php';
12
13 if (session_status() == PHP_SESSION_NONE) {
14     session_start();
15 }
16
17 $idUser = $_SESSION['loggedIn']['idUser'];
18
19 $query = <<<EX
20 SELECT idUser, pseudo, prenom, nom, email, telephone
21 FROM user
22 WHERE idUser = :idUser;
23 EX;
24
25 try {
26     $requestGetUserData = DatabaseController::prepare($query);
27     $requestGetUserData->bindParam(':idUser', $idUser, PDO::PARAM_STR);
28     $requestGetUserData->execute();
29
30     $_SESSION['loggedIn'] = $requestGetUserData->fetchAll(PDO::FETCH_ASSOC)[0];
31
32     echo json_encode([
33         'ReturnCode' => 0,
34         'Data' => $_SESSION['loggedIn']
35     ]);
36 } catch (PDOException $e) {
37     echo json_encode([
38         'ReturnCode' => 1,
39         'Error' => 'Erreur est survenue lors de la modification du compte'
40     ]);
41     exit();
42 }
```

```
1 <?php
2 /*
3  * Auteur      : Hoarau Nicolas
4  * Projet     : WE GO
5  * Page       : getPublicEvents.php
6  * Date       : 28.05.2020
7  * Description : Fichier qui récupère les événements publiques qui ne sont pas encore
8  *                passé
9  * Version    : 1.0
10 */
11 require_once dirname(__DIR__) . '/Controller/DatabaseController.php';
12 require_once __DIR__ . './backend.php';
13
14 if (session_status() == PHP_SESSION_NONE) {
15     session_start();
16 }
17
18 $dateMoreOneHour = date('Y-m-d H:i:s', strtotime(date('Y-m-d H:i:s') . '+1hours'));
19
20 $queryGetPublicEvents = <<<EX
21 SELECT e.idEvenement, e.nom, e.descriptif, e.dateDebut, e.dateFin, e.urlImage, u.pseudo
22 FROM evenement AS e
23 JOIN user AS u ON e.idOrganisateur = u.idUser
24 WHERE prive = 0
25 AND TIMESTAMP(dateDebut) >= :dateTime
26 ORDER BY dateDebut DESC
27 EX;
28
29 try {
30     $requestGetPublicEvents = DatabaseController::prepare($queryGetPublicEvents);
31     $requestGetPublicEvents->bindParam(':dateTime', $dateMoreOneHour);
32     $requestGetPublicEvents->execute();
33
34     $result['events'] = $requestGetPublicEvents->fetchAll(PDO::FETCH_ASSOC);
35
36     echo json_encode($result);
37 } catch (PDOException $e) {
38     echo json_encode([
39         'ReturnCode' => 1,
40         'Error' => 'Erreur est survenue lors de la récupération des événements'
41     ]);
42     exit();
43 }
44 }
```

```
<?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 si on est connecté sinon récupère les événements publics.
 * 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'];

$queryGetEventsParticipating = <<<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
```

```
53 WHERE idUser = :idUser )
54 AND i.idUser = :idUser
55 ORDER BY dateDebut DESC;
56 EX;
57
58 $queryGetEventCreated = <<<EX
59 SELECT e.idEvenement, e.nom, e.descriptif, e.dateDebut, e.dateFin, e.urlImage, u.pseuc
60 FROM evenement AS e
61 JOIN user AS u ON u.idUser = e.idOrganisateur
62 WHERE e.idOrganisateur = :idUser
63 AND TIMESTAMP(dateDebut) >= :dateTime
64 EX;
65
66 try {
67     $requestGetPublicEventsNotParticipating =
68         DatabaseController::prepare($queryGetPublicEventsNotParticipating);
69     $requestGetPublicEventsNotParticipating->bindParam(':idUser', $idUser,
70 PDO::PARAM_INT, 11);
71     $requestGetPublicEventsNotParticipating->bindParam(':dateTime', $dateMoreOneHour);
72
73     $requestGetPublicEventsNotParticipating->execute();
74     $result['eventsNotParticipating'] =
75         $requestGetPublicEventsNotParticipating->fetchAll(PDO::FETCH_ASSOC);
76
77     $requestGetInvitedEvents = DatabaseController::prepare($queryGetInvitedEvents);
78     $requestGetInvitedEvents->bindParam(':idUser', $idUser, PDO::PARAM_INT, 11);
79     $requestGetInvitedEvents->bindParam(':dateTime', $dateMoreOneHour);
80
81     $requestGetEventsParticipating =
82         DatabaseController::prepare($queryGetEventsParticipating);
83     $requestGetEventsParticipating->bindParam(':idUser', $idUser, PDO::PARAM_INT, 11);
84     $requestGetEventsParticipating->bindParam(':dateTime', $dateMoreOneHour);
85
86     $requestGetEventsParticipating->execute();
87     $result['eventsParticipating'] =
88         $requestGetEventsParticipating->fetchAll(PDO::FETCH_ASSOC);
89
90     $requestGetEventCreated = DatabaseController::prepare($queryGetEventCreated);
91     $requestGetEventCreated->bindParam(':idUser', $idUser, PDO::PARAM_INT, 11);
92     $requestGetEventCreated->bindParam(':dateTime', $dateMoreOneHour);
93
94     $requestGetEventCreated->execute();
95     $result['eventsCreated'] = $requestGetEventCreated->fetchAll(PDO::FETCH_ASSOC);
96
97     $result['isLogged'] = true;
98     $result['nickname'] = $_SESSION['loggedIn']['pseudo'];
99
100 } catch (PDOException $e) {
101     echo json_encode([
102         'ReturnCode' => 1,
```

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

```
1 <?php
2 /*
3  * Auteur      : Hoarau Nicolas
4  * Projet     : WE GO
5  * Page       : getInvited.php
6  * Date       : 05.06.2020
7  * Description : Fichier qui récupère les invités à un événements
8  * Version    : 1.0
9 */
10
11 require_once __DIR__ . '/backend.php';
12
13 $idEvent = filter_input(INPUT_POST, 'idEvent', FILTER_SANITIZE_NUMBER_INT);
14
15 $data = (GetInvited($idEvent) != null) ? GetInvited($idEvent) : false;
16
17 if ($data != false) {
18     echo json_encode([
19         'ReturnCode' => 0,
20         'Data' => $data
21     ]);
22     exit();
23 } else {
24     echo json_encode([
25         'ReturnCode' => 1,
26         'Error' => 'Erreur est survenue lors de la récupération des invités'
27     ]);
28     exit();
29 }
```

```
1 <?php
2 /*
3  * Auteur      : Hoarau Nicolas
4  * Projet     : WE GO
5  * Page       : getGuestList.php
6  * Date       : 27.05.2020
7  * Description : Récupère tous les utilisateurs de la base de données à part le
8  *                 créateur de l'événement
9  * Version    : 1.0
10 */
11 require_once __DIR__ . '/backend.php';
12 require_once dirname(__DIR__) . '/Controller/DatabaseController.php';
13
14 if (session_status() == PHP_SESSION_NONE) {
15     session_start();
16 }
17
18 $nickname = $_SESSION['loggedIn']['pseudo'];
19
20 $query = <<<EX
21     SELECT idUser, pseudo FROM user WHERE pseudo <> :pseudo;
22 EX;
23
24 try {
25     $requestGetUsers = DatabaseController::prepare($query);
26     $requestGetUsers->bindParam(':pseudo', $nickname, PDO::PARAM_STR, 50);
27     $requestGetUsers->execute();
28
29     $result = $requestGetUsers->fetchAll(PDO::FETCH_ASSOC);
30
31     echo json_encode($result);
32     exit();
33 } catch (PDOException $e) {
34     echo json_encode([
35         'ReturnCode' => 1,
36         'Error' => 'Erreur est survenue lors de la récupération de la liste des invités'
37     ]);
38     exit();
39 }
```

```
1 <?php
2 /*
3  * Auteur      : Hoarau Nicolas
4  * Projet     : WE GO
5  * Page       : getEventGuestList.php
6  * Date       : 02.06.2020
7  * Description : Fichier qui récupère participant d'un événement données.
8  * Version    : 1.0
9 */
10
11 require_once dirname(__DIR__) . '/Controller/DatabaseController.php';
12
13 $idEvent = filter_input(INPUT_POST, 'idEvent', FILTER_SANITIZE_NUMBER_INT);
14
15 $query = <<<EX
16 SELECT u.pseudo
17 FROM inscriptions AS i
18 JOIN user AS u ON i.idUser = u.idUser
19 WHERE idEvenement = :idEvent;
20 EX;
21
22 try {
23     $requestGetGuestList = DatabaseController::prepare($query);
24
25     $requestGetGuestList->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
26     $requestGetGuestList->execute();
27     $result = $requestGetGuestList->fetchAll(PDO::FETCH_ASSOC);
28
29     echo json_encode([
30         'Data' => $result
31     ]);
32     exit();
33 } catch (PDOException $e) {
34     echo json_encode([
35         'ReturnCode' => 1,
36         'Error' => 'Erreur est survenue lors de la récupération de la liste des
participants'
37     ]);
38     exit();
39
40     exit();
41 }
```

```
<?php
/*
 * Auteur      : Hoarau Nicolas
 * Projet     : WE GO
 * Page       : getEventData.php
 * Date       : 02.06.2020
 * Description : Fichier qui récupère les 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;
        }
    }
}
```

```
53     echo json_encode([
54         'ReturnCode' => 0,
55         'Data' => $data
56     ]);
57     exit();
58 }
59 }
60 } else {
61     echo json_encode([
62         'ReturnCode' => 404
63     ]);
64     exit();
65 }
```

```
1 <?php
2 /*
3  * Auteur      : Hoarau Nicolas
4  * Projet     : WE GO
5  * Page       : getEditGuestList.php
6  * Date       : 05.06.2020
7  * Description : Fichier qui récupère les utilisateurs en disant s'il participe ou non
8  *                à l'événement donnée
9  * Version    : 1.0
10 */
11 require_once dirname(__DIR__) . '/Controller/DatabaseController.php';
12
13 if (session_status() == PHP_SESSION_NONE) {
14     session_start();
15 }
16
17 $idEvent = filter_input(INPUT_POST, 'idEvent', FILTER_SANITIZE_NUMBER_INT);
18 $nickname = $_SESSION['loggedIn']['pseudo'];
19 $query = <<<EX
20 SELECT DISTINCT u.idUser, u.pseudo,
21   CASE WHEN i.idUser IS NOT NULL AND i.idEvenement = :idEvent THEN TRUE ELSE FALSE END
22 AS invited
23 FROM user AS u
24 LEFT JOIN invites AS i ON i.idUser = u.idUser
25 WHERE i.idEvenement = :idEvent
26 OR i.idEvenement IS NULL
27 OR i.idUser NOT IN (
28   SELECT idUser FROM invites WHERE idEvenement = :idEvent
29 )
30 AND u.pseudo <> :nickname;
31 EX;
32 try {
33     $requestGetInvitedAndNotInvited = DatabaseController::prepare($query);
34     $requestGetInvitedAndNotInvited->bindParam(':nickname', $nickname, PDO::PARAM_STR);
35     $requestGetInvitedAndNotInvited->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
36     $requestGetInvitedAndNotInvited->execute();
37     $result = $requestGetInvitedAndNotInvited->fetchAll(PDO::FETCH_ASSOC);
38
39     echo json_encode([
40         'ReturnCode' => 0,
41         'Data' => $result
42     ]);
43     exit();
44 } catch (PDOException $e) {
45     echo json_encode([
46         'ReturnCode' => 1,
47         'Error' => 'Erreur est survenue lors de la récupération des utilisateurs'
48     ]);
49     exit();
50 }
```

```
1 <?php
2 /*
3  * Auteur      : Hoarau Nicolas
4  * Projet      : WE GO
5  * Page        : eventReminder.php
6  * Date        : 03.06.2020
7  * Description  : Fichier qui vérifie si un événement est dans 24h et si c'est le cas
8  * envoie un mail de rappel
9  * Version     : 1.0
10 */
11 require_once __DIR__ . './backend.php';
12
13 $events = GetAllFutureEvent();
14 $now = date('Y-m-d');
15 $mailSubject = "WEGO: Rappel de participation";
16
17 foreach ($events as $event) {
18     $idEvent = $event['idEvenement'];
19     $eventName = $event['nom'];
20     $beginingDate = $event['dateDebut'];
21     $dayBeforeEvent = date('Y-m-d', strtotime($beginingDate . '-1 days'));
22     $mailMessage = <<<EX
23 N'oubliez pas que l'événement "{$eventName}" se déroule demain.
24 EX;
25
26     if (strtotime($now) >= strtotime($dayBeforeEvent) && strtotime($now) <
27         strtotime($beginingDate)) {
28         if (GetEventNbGuest($idEvent) > 0) {
29             $eventUsersMail = GetEventUsersMail($idEvent);
30
31             foreach ($eventUsersMail as $mail) {
32                 if (SendMail($mail['email'], $mailMessage, $mailSubject) == false) {
33                     echo json_encode([
34                         'ReturnCode' => 1,
35                         'Error' => "Une erreur est survenue lors de l'envoie du mail."
36                     ]);
37                     exit();
38                 }
39             }
40         }
41     }
-->
```

```
<?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 minimum 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,
```

```
51 | $phoneNumber, $idUser, $oldEmail)) {
52 |     echo json_encode([
53 |         'ReturnCode' => 0,
54 |         'Success' => 'Le compte a bien été modifié.'
55 |     ]);
56 |     exit();
57 | } else {
58 |     echo json_encode([
59 |         'ReturnCode' => 1,
60 |         'Error' => 'Erreur est survenue lors de la modification du compte'
61 |     ]);
62 |     exit();
63 | }
64 | } else {
65 |     echo json_encode([
66 |         'ReturnCode' => 2,
67 |         'Error' => 'Ce nom d\'utilisateur est déjà utilisé'
68 |     ]);
69 |     exit();
70 | }
71 | } else {
72 |     echo json_encode([
73 |         'ReturnCode' => 3,
74 |         'Error' => 'Cette adresse mail est déjà utilisée'
75 |     ]);
76 |     exit();
77 | }
```

```
1 <?php
2 /*
3  * Auteur      : Hoarau Nicolas
4  * Projet     : WE GO
5  * Page       : editGuestList.php
6  * Date       : 05.06.2020
7  * Description : fichier qui modifie la liste des invités
8  * Version    : 1.0
9 */
10
11 require_once __DIR__ . '/backend.php';
12
13 $idEvent = filter_input(INPUT_POST, 'idEvent', FILTER_SANITIZE_NUMBER_INT);
14 $editGuestList = $_POST['editGuetsList'];
15
16 foreach ($editGuestList as $user) {
17     if ($user[1] == "false") {
18         if (DeleteInvitation($user[0], $idEvent) == false) {
19             echo json_encode([
20                 'ReturnCode' => 1,
21                 'Error' => 'Erreur est survenue lors de la modification de la base de données'
22             ]);
23             exit();
24         }
25     } else {
26         if (CreateInvite($user[0], $idEvent) == false) {
27             echo json_encode([
28                 'ReturnCode' => 1,
29                 'Error' => 'Erreur est survenue lors de la modification de la base de données'
30             ]);
31             exit();
32         }
33     }
34 }
35
36 echo json_encode([
37     'ReturnCode' => 0,
38     'Success' => 'La modification de la liste des invités a bien été effectué'
39 ]);
40 exit();
41
```

```
<?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(__DIR__) . '/views/includes/const.inc.php';

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

$info = 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'])) {
                $mimeType = finfo_file($info, $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"
                            ]);
                        }
                    }
                }
            }
        }
    }
}
```

```
50        ]);
51        exit();
52    }
53 } else {
54     echo json_encode([
55     'ReturnCode' => 1,
56     'Error' => "Un problème est survenue lors de l'upload de l'image"
57 ]);
58     exit();
59 }
60 } else {
61     echo json_encode([
62     'ReturnCode' => 2,
63     'Error' => "Le fichier données n'est pas une image"
64 ]);
65     exit();
66 }
67 }
68
69 if (EditEvent($eventName, $eventDescription, $place, $beginningDate, $endDate,
70 $nbMaxGuest, null, $idEvent)) {
71     echo json_encode([
72     'ReturnCode' => 0,
73     'Success' => "L'événement " . $eventName . " a bien été modifié"
74 ]);
75     exit();
76 }
77 } else {
78     echo json_encode([
79     'ReturnCode' => 3,
80     'Error' => "L'événement ne peut pas se terminer avant d'avoir commencé"
81 ]);
82     exit();
83 }
84 } else {
85     echo json_encode([
86     'ReturnCode' => 4,
87     'Error' => "L'événement ne peut pas commencer avant aujourd'hui"
88 ]);
89     exit();
90 }
```

```
<?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.'
                ]);
            }
        }
    }
}
```

```
55     }
56 }
57 }
58 }
59
60 foreach ($userEventCreated as $event) {
61     $getUserInvitedToTheEvent = GetInvited($event['idEvenement']);
62
63     if (count($getUserInvitedToTheEvent) > 0) {
64         foreach ($getUserInvitedToTheEvent as $user) {
65             if (DeleteInvitation($user['idUser'], $event['idEvenement']) == false) {
66                 echo json_encode([
67                     'ReturnCode' => 1,
68                     'Error' => 'Une erreur est survenue lors de la suppression de votre profile.
69                 ]);
70             }
71         }
72     }
73
74     if (DeleteEvent($event['idEvenement']) == false) {
75         echo json_encode([
76             'ReturnCode' => 1,
77             'Error' => 'Une erreur est survenue lors de la suppression de votre profile.'
78         ]);
79     }
80 }
81
82 try {
83     DatabaseController::beginTransaction();
84
85     $requestDeleteEvent = DatabaseController::prepare($query);
86     $requestDeleteEvent->bindParam(':idUser', $idUser, PDO::PARAM_INT);
87     $requestDeleteEvent->execute();
88
89     DatabaseController::commit();
90
91     echo json_encode([
92         'ReturnCode' => 0,
93         'Success' => "Votre compte à bien été supprimé."
94     ]);
95 } catch (PDOException $e) {
96     DatabaseController::rollBack();
97     echo json_encode([
98         'ReturnCode' => 1,
99         'Error' => 'Une erreur est survenue lors de la suppression de votre profile.'
100    ]);
101 }
102 }
```

```
1 <?php
2 /*
3  * Auteur      : Hoarau Nicolas
4  * Projet     : WE GO
5  * Page       : deleteEvent.php
6  * Date       : 03.06.2020
7  * Description : Fichier qui supprime un événement donnée
8  * Version    : 1.0
9 */
10
11 require_once __DIR__ . '/backend.php';
12
13 $idEvent = filter_input(INPUT_POST, 'idEvent', FILTER_SANITIZE_NUMBER_INT);
14 $eventName = filter_input(INPUT_POST, 'eventName', FILTER_SANITIZE_STRING);
15 $private = filter_input(INPUT_POST, 'private', FILTER_SANITIZE_NUMBER_INT);
16
17 if ($private == 1) {
18     $invited = GetInvited($idEvent);
19     foreach ($invited as $user) {
20         if (DeleteInvitation($user['idUser'], $idEvent) == false) {
21             echo json_encode([
22                 'ReturnCode' => 1,
23                 'Error' => "Erreur est survenue lors de la suppression de l'événement"
24             ]);
25             exit();
26         }
27     }
28 }
29
30 if (DeleteEvent($idEvent)) {
31     echo json_encode([
32         'ReturnCode' => 0,
33         'Success' => "L'événement \" . $eventName . "\" est bien supprimé"
34     ]);
35     exit();
36 } else {
37     echo json_encode([
38         'ReturnCode' => 1,
39         'Error' => "Erreur est survenue lors de la suppression de l'événement"
40     ]);
41     exit();
42 }
```

```
<?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(__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,

```

```
51 | $endDate, $nbMaxGuest, './assets/upload/' . $filename, $type, $idUser, $guestlist)) {
52 |     echo json_encode([
53 |         'ReturnCode' => 0,
54 |         'Success' => "L'événement " . $eventName . " a bien été crée"
55 |     ]);
56 |     exit();
57 | }
58 | } else {
59 |     echo json_encode([
60 |         'ReturnCode' => 1,
61 |         'Error' => "Un problème est survenue lors de l'upload de l'image"
62 |     ]);
63 |     exit();
64 | }
65 | } else {
66 |     echo json_encode([
67 |         'ReturnCode' => 2,
68 |         'Error' => "Le fichier données n'est pas une image"
69 |     ]);
70 |     exit();
71 | }
72 |
73 | if (CreateEvent($eventName, $eventDescription, $place, $beginningDate, $endDate,
74 | $nbMaxGuest, $img, $type, $idUser, $guestlist)) {
75 |     echo json_encode([
76 |         'ReturnCode' => 0,
77 |         'Success' => "L'événement " . $eventName . " a bien été crée"
78 |     ]);
79 |     exit();
80 | }
81 | } else {
82 |     echo json_encode([
83 |         'ReturnCode' => 3,
84 |         'Error' => "L'événement ne peut pas se terminer avant d'avoir commencé"
85 |     ]);
86 |     exit();
87 | }
88 | } else {
89 |     echo json_encode([
90 |         'ReturnCode' => 4,
91 |         'Error' => "L'événement ne peut pas commencer avant aujourd'hui"
92 |     ]);
93 | }
94 }
```

```
<?php
/*
 * Auteur      : Hoarau Nicolas
 * Projet     : WE GO
 * Page       : backend.php
 * Date       : 26.05.2020
 * Description : Fichier qui contient toutes les fonctions du 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 variable son
    nom de colonne

    $clauseField = "";
}
```

```
54     $clauseValue = "";
55
56     if ($userEmail !== null) {
57         $clauseField = "email";
58         $clauseValue = $userEmail;
59     } elseif ($userNickname) {
60         $clauseField = "pseudo";
61         $clauseValue = $userNickname;
62     }
63
64     $query = <<<EX
65     SELECT salt
66     FROM user
67     WHERE `{$clauseField}` = :clauseValue;
68     EX;
69
70     try {
71         $requestSalt = DatabaseController::prepare($query);
72         $requestSalt->bindParam(':clauseValue', $clauseValue, PDO::PARAM_STR);
73         $requestSalt->execute();
74
75         $result = $requestSalt->fetch(PDO::FETCH_ASSOC);
76
77         return $result !== false ? $result['salt'] : null;
78     } catch (PDOException $e) {
79         return null;
80     }
81 }
82
83 /**
84 * @author Hoarau Nicolas
85 * @date 26.05.2020
86 * @brief Fonction qui créer un nouvel utilisateur dans la base de données
87 * @param string $nickname
88 * @param string $lastname
89 * @param string $firstname
90 * @param string $email
91 * @param string $phoneNumber
92 * @param string $password
93 * @return boolean
94 * @version 1.0
95 *      Inscription sans salt
96 * @version 1.1
97 *      Inscription avec salt
98 */
99 function Register(string $nickname, string $firstname, string $email, string
100 $password, string $lastname = null, string $phoneNumber = null): bool
101 {
102     $query = "";
103     $salt = hash('sha256', microtime());
104     $userPassword = hash('sha256', $password . $salt);
105     if ($lastname != null && $phoneNumber != null) {
106         $query = <<<EX
107             INSERT INTO user (pseudo, prenom, nom, email, password, telephone, salt)
```

```
108     VALUES (:nickname, :firstname, :lastname, :email, :password, :phoneNumber,
109     :salt);
110     EX;
111 } elseif ($phoneNumber != null) {
112     $query = <<<EX
113     INSERT INTO user (pseudo, prenom, nom, email, password, telephone, salt)
114     VALUES (:nickname, :firstname, :email, :password, :phoneNumber, :Salt);
115     EX;
116 } elseif ($lastname != null) {
117     $query = <<<EX
118     INSERT INTO user (pseudo, prenom, nom, email, password, salt)
119     VALUES (:nickname, :firstname, :lastname, :email, :password, :salt);
120     EX;
121 } else {
122     $query = <<<EX
123     INSERT INTO user (pseudo, prenom, email, password, salt)
124     VALUES (:nickname, :firstname, :email, :password, :salt);
125     EX;
126 }
127 try {
128     DatabaseController::beginTransaction();
129
130     $requestRegister = DatabaseController::prepare($query);
131     $requestRegister->bindParam(':nickname', $nickname, PDO::PARAM_STR, 50);
132     $requestRegister->bindParam(':firstname', $firstname, PDO::PARAM_STR, 50);
133
134     if ($lastname != null)
135         $requestRegister->bindParam(':lastname', $lastname, PDO::PARAM_STR, 50);
136
137     $requestRegister->bindParam(':email', $email, PDO::PARAM_STR, 100);
138     $requestRegister->bindParam(':password', $userPassword, PDO::PARAM_STR, 100);
139
140     if ($phoneNumber != null)
141         $requestRegister->bindParam(':phoneNumber', $phoneNumber, PDO::PARAM_STR, 50);
142
143     $requestRegister->bindParam(':salt', $salt, PDO::PARAM_STR, 64);
144
145     $requestRegister->execute();
146
147     DatabaseController::commit();
148     return true;
149 } catch (PDOException $e) {
150     DatabaseController::rollBack();
151     return false;
152 }
153 }
154 /**
155 * @author Hoarau Nicolas
156 * @date 26.05.2020
157 * @brief Fonction qui connecte l'utilisateur
158 * @param array les données de connexion de l'utilisateur
159 * @return array|false
160 * @version 1.0
161 *
```

```
162 *      Inscription sans salt
163 * @version 1.1
164 *      Inscription avec salt
165 */
166 function Login(array $args)
167 {
168     // initialise à null les colonnes du tableau vide/inexistante
169     $args += [
170         'userEmail' => null,
171         'userNickname' => null,
172         'userPwd' => null,
173     ];
174
175     extract($args); // extrait les données du tableau avec comme nom de variable son
176     nom de colonne
177     $loginField = "";
178     $authenticator = "";
179     $salt = "";
180
181     if ($userEmail !== null) {
182         $salt = GetSalt(['userEmail' => $userEmail]);
183         $authenticator = $userEmail;
184         $loginField = "email";
185     } elseif ($userNickname !== null) {
186         $salt = GetSalt(['userNickname' => $userNickname]);
187         $authenticator = $userNickname;
188         $loginField = "pseudo";
189     } else {
190         return false;
191     }
192
193     if ($userPwd == null)
194         return false;
195
196     $pwd = hash('sha256', $userPwd . $salt);
197
198     $query = <<<EX
199     SELECT idUser, pseudo, prenom, nom, email, telephone
200     FROM user
201     WHERE `{$loginField}` = :wayToConnectValue
202     AND password = :pwd;
203 EX;
204
205     try {
206         $requestLogin = DatabaseController::prepare($query);
207         $requestLogin->bindParam(':wayToConnectValue', $authenticator, PDO::PARAM_STR);
208         $requestLogin->bindParam(':pwd', $pwd, PDO::PARAM_STR);
209         $requestLogin->execute();
210
211         $result = $requestLogin->fetch(PDO::FETCH_ASSOC);
212
213         return $result !== false > 0 ? $result : false;
214     } catch (PDOException $e) {
215         return false;
```

```
216     }
217 }
218
219 /**
220 * @author Hoarau Nicolas
221 * @date 26.05.2020
222 * @brief Fonction qui vérifie si le paramètre donnée(email, nickname) est déjà
223 * utilisé ou non
224 * @param array $args
225 * @return boolean|null
226 * @version 1.0
227 */
228 function IsTaken(array $args): ?bool
229 {
230     $args += [
231         'userEmail' => null,
232         'userNickname' => null,
233         'idUser' => null
234     ];
235     extract($args);
236
237     $authenticator = "";
238     $field = "";
239     if ($userEmail != null) {
240         $authenticator = $userEmail;
241         $field = "email";
242     } elseif ($userNickname != null) {
243         $authenticator = $userNickname;
244         $field = "pseudo";
245     }
246
247     $queryWithoutidUser = <<<EX
248     SELECT `{$field}`
249     FROM user
250     WHERE `{$field}` = :authenticator;
251 EX;
252
253     $queryWithidUser = <<<EX
254     SELECT `{$field}`
255     FROM user
256     WHERE `{$field}` = :authenticator
257     AND idUser <> :idUser;
258 EX;
259
260
261     $query = ($idUser != null) ? $queryWithidUser : $queryWithoutidUser;
262
263     try {
264         $requestIsUsed = DatabaseController::prepare($query);
265         $requestIsUsed->bindParam(':authenticator', $authenticator, PDO::PARAM_STR);
266
267         if ($idUser != null)
268             $requestIsUsed->bindParam(':idUser', $idUser, PDO::PARAM_INT);
269     }
```

```
270     $requestIsUsed->execute();
271     $result = $requestIsUsed->fetch(PDO::FETCH_ASSOC);
272 
273     return $result !== false ? true : false;
274 } catch (PDOException $e) {
275     return null;
276 }
277 }
278 
279 /**
280 * @author Hoarau Nicolas
281 * @date 27.05.2020
282 * @brief Fonction qui crée un événement
283 * @param string $eventName
284 * @param string $eventDescription
285 * @param string $place
286 * @param Date $beginningDate
287 * @param Date $endDate
288 * @param integer $nbMaxGuest
289 * @param FILES $img
290 * @param integer $idUser
291 * @return boolean
292 * @version 1.0
293 */
294 function CreateEvent(string $eventName, string $eventDescription, string $place,
295     $beginningDate, $endDate, int $nbMaxGuest, $img, int $type, int $idUser, array
296     $guestlist = null): bool
297 {
298     $creationDate = date('Y-m-d H:i');
299     $beginningDate = date("Y-m-d H:i", strtotime($beginningDate));
300     $endDate = date("Y-m-d H:i", strtotime($endDate));
301 
302     $query = <<<EX
303         INSERT INTO evenement (nom, descriptif, dateDebut, dateFin, dateCreation, lieu,
304         prive, nbMaxParticipant, urlImage, idOrganisateur)
305         VALUES (:name, :description, :beginningDate, :endDate, :creationDate, :place,
306         :type, :nbMaxGuest, :urlImg ,:idUser);
307         EX;
308 
309     try {
310         DatabaseController::beginTransaction();
311 
312         $requestCreateEvent = DatabaseController::prepare($query);
313         $requestCreateEvent->bindParam(':name', $eventName, PDO::PARAM_STR, 45);
314         $requestCreateEvent->bindParam(':description', $eventDescription, PDO::PARAM_STR,
315             255);
316         $requestCreateEvent->bindParam(':beginningDate', $beginningDate);
317         $requestCreateEvent->bindParam(':endDate', $endDate);
318         $requestCreateEvent->bindParam(':creationDate', $creationDate);
319         $requestCreateEvent->bindParam(':place', $place, PDO::PARAM_STR, 255);
320         $requestCreateEvent->bindParam(':type', $type, PDO::PARAM_INT, 1);
321         $requestCreateEvent->bindParam(':nbMaxGuest', $nbMaxGuest, PDO::PARAM_INT, 11);
322         $requestCreateEvent->bindParam(':urlImg', $img, PDO::PARAM_STR, 255);
323         $requestCreateEvent->bindParam(':idUser', $idUser, PDO::PARAM_INT, 11);
324     }
```

```
320     $requestCreateEvent->execute();
321
322     if ($guestlist != null) {
323         $lastInsertId = DatabaseController::getInstance()->lastInsertId();
324
325         for ($i = 0; $i < count($guestlist); $i++) {
326             if ($idUser != $guestlist[$i]) {
327                 if (CreateInvite($guestlist[$i], $lastInsertId) == false) {
328                     DatabaseController::rollBack();
329                 }
330             }
331         }
332
333         DatabaseController::commit();
334         return true;
335     }
336
337     DatabaseController::commit();
338     return true;
339 } catch (PDOException $e) {
340     DatabaseController::rollBack();
341     return false;
342 }
343 }
344
345 /**
346 * @author Hoarau Nicolas
347 * @date 27.05.2020
348 * @brief Fonction qui crée une invitation pour un utilisateur à un événement
349 * @param integer $idGuest
350 * @param integer $idEvent
351 * @return boolean
352 * @version 1.0
353 */
354 function CreateInvite(int $idGuest, int $idEvent): bool
355 {
356     $query = <<<EX
357     INSERT INTO invites (idUser, idEvenement)
358     VALUES (:idGuest, :idEvent);
359     EX;
360
361     try {
362         DatabaseController::beginTransaction();
363
364         $requestCreateInvite = DatabaseController::prepare($query);
365         $requestCreateInvite->bindParam(':idGuest', $idGuest, PDO::PARAM_INT, 11);
366         $requestCreateInvite->bindParam(':idEvent', $idEvent, PDO::PARAM_INT, 11);
367         $requestCreateInvite->execute();
368
369         DatabaseController::commit();
370
371         return true;
372     } catch (PDOException $e) {
373         DatabaseController::rollBack();
374         return false;
375     }
376 }
```

```
375     }
376 }
377
378 /**
379 * @author Hoarau Nicolas
380 * @date 05.06.2020
381 * @brief Fonction qui supprime l'invitation d'un utilisateur à un événement
382 * @param integer $idGuest
383 * @param integer $idEvent
384 * @return boolean
385 * @version 1.0
386 */
387 function DeleteInvite(int $idGuest, int $idEvent): bool
388 {
389     $query = <<<EX
390 DELETE FROM invites WHERE idUser = :idUser AND idEvenement = :idEvent;
391 EX;
392
393 try {
394     DatabaseController::beginTransaction();
395
396     $requestDeleteInvite = DatabaseController::prepare($query);
397     $requestDeleteInvite->bindParam(':idGuest', $idGuest, PDO::PARAM_INT, 11);
398     $requestDeleteInvite->bindParam(':idEvent', $idEvent, PDO::PARAM_INT, 11);
399     $requestDeleteInvite->execute();
400
401     DatabaseController::commit();
402
403     return true;
404 } catch (PDOException $e) {
405     DatabaseController::rollBack();
406     return false;
407 }
408 }
409
410 /**
411 * @author Hoarau Nicolas
412 * @date 02.06.2020
413 * @brief Fonction qui vérifie si l'événement existe
414 * @param integer $idEvent
415 * @return boolean
416 * @version 1.0
417 */
418 function EventExist(int $idEvent): bool
419 {
420     $query = <<<EX
421 SELECT COUNT(idEvenement) AS exist
422 FROM evenement
423 WHERE idEvenement = :idEvent
424 EX;
425
426 try {
427     $requestExist = DatabaseController::prepare($query);
428     $requestExist->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
429     $requestExist->execute();
```

```
430
431     $result = $requestExist->fetch(PDO::FETCH_ASSOC);
432
433     return $result['exist'] == 0 ? false : true;
434 } catch (PDOException $e) {
435     return false;
436 }
437 }
438
439 /**
440 * @author Hoarau Nicolas
441 * @date 02.06.2020
442 * @brief Fonction qui vérifie si l'utilisateur participe à l'événement
443 * @param integer $idEvent
444 * @param integer $idUser
445 * @return boolean
446 * @version 1.0
447 */
448 function IsParticipating(int $idEvent, int $idUser): bool
449 {
450     $query = <<<EX
451     SELECT COUNT(*) AS participating
452     FROM inscriptions
453     WHERE idUser = :idUser
454     AND idEvenement = :idEvent
455     EX;
456
457     try {
458         $requestIsParticipating = DatabaseController::prepare($query);
459         $requestIsParticipating->bindParam(':idUser', $idUser, PDO::PARAM_INT);
460         $requestIsParticipating->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
461         $requestIsParticipating->execute();
462
463         $result = $requestIsParticipating->fetch(PDO::FETCH_ASSOC);
464
465         return $result['participating'] == 0 ? false : true;
466     } catch (PDOException $e) {
467         return false;
468     }
469 }
470
471 /**
472 * @author Hoarau Nicolas
473 * @date 02.06.2020
474 * @brief Fonction qui vérifie si l'événement est privé
475 * @param integer $idEvent
476 * @return boolean
477 * @version 1.0
478 */
479 function IsPrivate(int $idEvent): bool
480 {
481     $query = <<<EX
482     SELECT prive
483     FROM evenement
484     WHERE idEvenement = :idEvent;
```

```
485 EX;
486
487 try {
488     $requestIsPrivate = DatabaseController::prepare($query);
489     $requestIsPrivate->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
490     $requestIsPrivate->execute();
491
492     $result = $requestIsPrivate->fetch(PDO::FETCH_ASSOC);
493
494     return $result['prive'] == 0 ? false : true;
495 } catch (PDOException $e) {
496     return false;
497 }
498 }
499
500 /**
501 * @author Hoarau Nicolas
502 * @date 02.06.2020
503 * @brief Fonction qui vérifie si l'utilisateur est invité à l'événement donnée
504 * @param integer $idEvent
505 * @param integer $idUser
506 * @return boolean
507 * @version 1.0
508 */
509 function IsInvited(int $idEvent, int $idUser): bool
510 {
511     $query = <<<EX
512     SELECT COUNT(*) AS isInvited
513     FROM invites
514     WHERE idUser = :idUser
515     AND idEvenement = :idEvent;
516     EX;
517
518     try {
519         $requestIsInvited = DatabaseController::prepare($query);
520         $requestIsInvited->bindParam(':idUser', $idUser, PDO::PARAM_INT);
521         $requestIsInvited->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
522         $requestIsInvited->execute();
523         $result = $requestIsInvited->fetch(PDO::FETCH_ASSOC);
524
525         return $result['isInvited'] == 1 ? true : false;
526     } catch (PDOException $e) {
527         return false;
528     }
529 }
530
531 /**
532 * @author Hoarau Nicolas
533 * @date 02.06.2020
534 * @brief Fonction qui vérifie si l'utilisateur est le créateur de l'événement donnée
535 * @param integer $idEvent
536 * @param integer $idUser
537 * @return boolean
538 * @version 1.0
539 */
```

```
540 function IsCreator(int $idEvent, int $idUser): bool
541 {
542     $query = <<<EX
543     SELECT COUNT(*) AS isCreator
544     FROM evenement
545     WHERE idOrganisateur = :idUser
546     AND idEvenement = :idEvent;
547     EX;
548
549     try {
550         $requestIsCreator = DatabaseController::prepare($query);
551         $requestIsCreator->bindParam(':idUser', $idUser, PDO::PARAM_INT);
552         $requestIsCreator->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
553         $requestIsCreator->execute();
554
555         $result = $requestIsCreator->fetch(PDO::FETCH_ASSOC);
556
557         return $result['isCreator'] == 1 ? true : false;
558     } catch (PDOException $e) {
559         return false;
560     }
561 }
562
563 /**
564 * @author Hoarau Nicolas
565 * @date 02.06.2020
566 * @brief Fonction qui récupère les information de l'événement
567 * @param integer $idEvent
568 * @param integer $idUser
569 * @return array|false
570 * @version 1.0
571 */
572 function GetEventData(int $idEvent, int $idUser = null)
573 {
574     $queryLogged = <<<EX
575     SELECT e.idEvenement, e.nom, e.descriptif, e.dateDebut, e.dateFin, e.lieu,
576     e.nbMaxParticipant, e.prive, e.urlImage, u.pseudo, (SELECT COUNT(*) FROM inscriptions
577     WHERE idUser = :idUser AND idEvenement = :idEvent) AS participating, (SELECT
578     COUNT(idUser) FROM inscriptions WHERE idevenement = :idEvent) AS nbGuest
579     FROM evenement AS e
580     JOIN user AS u ON e.idOrganisateur = u.idUser
581     WHERE e.idEvenement = :idEvent;
582     EX;
583
584     $queryNotLogged = <<<EX
585     SELECT e.idEvenement, e.nom, e.descriptif, e.dateDebut, e.dateFin, e.lieu,
586     e.nbMaxParticipant, e.prive, e.urlImage, u.pseudo, (SELECT COUNT(idUser) FROM
587     inscriptions WHERE idEvenement = :idEvent) AS nbGuest
588     FROM evenement AS e
589     JOIN user AS u ON e.idOrganisateur = u.idUser
590     WHERE e.idEvenement = :idEvent;
591     EX;
592
593     $query = $idUser !== null ? $queryLogged : $queryNotLogged;
594 }
```

```
590     try {
591         $requestGetEventData = DatabaseController::prepare($query);
592
593         if ($idUser !== null)
594             $requestGetEventData->bindParam(':idUser', $idUser, PDO::PARAM_INT);
595
596         $requestGetEventData->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
597         $requestGetEventData->execute();
598
599         $result = $requestGetEventData->fetch(PDO::FETCH_ASSOC);
600
601         return $result !== false ? $result : null;
602     } catch (PDOException $e) {
603         return false;
604     }
605 }
606
607 /**
608 * @author Hoarau Nicolas
609 * @date 02.06.2020
610 * @brief Fonction qui envoie un mail à un destinataire précit
611 * @param string $recipientMail
612 * @param string $message
613 * @return bool
614 * @version 1.0
615 */
616 function SendMail(string $recipientMail, string $message, string $subject): bool
617 {
618     // Instantiation and passing `true` enables exceptions
619     $mail = new PHPMailer(true);
620
621     try {
622         //Enable SMTP debugging.
623         $mail->SMTPDebug = 0;
624         //Set PHPMailer to use SMTP.
625         $mail->isSMTP();
626
627         //Set SMTP host name
628         $mail->Host = MAIL_HOST;
629
630         //Set this to true if SMTP host requires authentication to send email
631         $mail->SMTPAuth = true;
632
633         //Provide username and password
634         $mail->Username = MAIL_USERNAME;
635         $mail->Password = MAIL_PASSWORD;
636
637         //If SMTP requires TLS encryption then set it
638         $mail->SMTPSecure = "tls";
639
640         //Set TCP port to connect to
641         $mail->Port = MAIL_PORT;
642         $mail->From = MAIL_USERNAME;
643         $mail->FromName = "WEGO Service";
644 }
```

```
645     $mail->addAddress($recipientMail);
646
647     $mail->isHTML(true);
648
649     $mail->Subject = $subject;
650     $mail->Body = $message;
651
652     if ($mail->send())
653         return true;
654     else
655         return false;
656 } catch (Exception $e) {
657     return false;
658 }
659 }
660
661 /**
662 * @author Hoarau Nicolas
663 * @date 03.06.2020
664 * @brief Récupère tous les événements publiques pas encore passé
665 * @return array|null
666 * @version 1.0
667 */
668 function GetAllFutureEvent(): ?array
669 {
670     $dateMoreOneHour = date('Y-m-d H:i:s', strtotime(date('Y-m-d H:i:s') . '+1hours'));
671
672     $query = <<<EX
673     SELECT idEvenement, nom, dateDebut
674     FROM evenement
675     WHERE TIMESTAMP(dateDebut) >= :dateTime
676     ORDER BY dateDebut DESC
677 EX;
678
679     try {
680         $requestGetAllFutureEvent = DatabaseController::prepare($query);
681         $requestGetAllFutureEvent->bindParam(':dateTime', $dateMoreOneHour);
682         $requestGetAllFutureEvent->execute();
683         $result = $requestGetAllFutureEvent->fetchAll(PDO::FETCH_ASSOC);
684         return $result;
685     } catch (PDOException $e) {
686         return null;
687     }
688 }
689
690 /**
691 * @author Hoarau Nicolas
692 * @date 03.06.2020
693 * @brief Fonction qui récupère le nombre de participant à un événement
694 * @param integer $idEvent
695 * @return integer|null
696 * @version 1.0
697 */
698 function GetEventNbGuest(int $idEvent): ?int
699 {
```

```
700 $query = <<<EX
701 SELECT COUNT(idUser) AS nbGuest
702 FROM inscriptions
703 WHERE idEvenement = :idEvent
704 EX;
705
706 try {
707     $requestGetEventNbGuest = DatabaseController::prepare($query);
708     $requestGetEventNbGuest->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
709     $requestGetEventNbGuest->execute();
710
711     $result = $requestGetEventNbGuest->fetch(PDO::FETCH_ASSOC);
712
713     return $result !== false ? $result['nbGuest'] : 0;
714 } catch (PDOException $e) {
715     return null;
716 }
717 }
718
719 /**
720 * @author Hoarau Nicolas
721 * @date 03.06.2020
722 * @brief Fonction qui récupère les addresses mail des participant à l'événement donn
723 * @param integer $idEvent
724 * @return array|null
725 * @version 1.0
726 */
727 function GetEventUsersMail(int $idEvent): ?array
728 {
729     $query = <<<EX
730     SELECT u.email
731     FROM inscriptions AS i
732     JOIN user AS u on i.idUser = u.idUser
733     WHERE idEvenement = :idEvent;
734     EX;
735
736     try {
737         $requestGetEventUsersMail = DatabaseController::prepare($query);
738         $requestGetEventUsersMail->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
739         $requestGetEventUsersMail->execute();
740
741         $result = $requestGetEventUsersMail->fetchAll(PDO::FETCH_ASSOC);
742
743         return $result;
744     } catch (PDOException $e) {
745         return null;
746     }
747 }
748
749 /**
750 * @author Hoarau Nicolas
751 * @date 04.06.2020
752 * @brief Fonction qui modifie un événement donné
753 * @param string $eventName
754 * @param string $eventDescription
```

```
755 * @param string $place
756 * @param Date $beginningDate
757 * @param Date $endDate
758 * @param integer $nbMaxGuest
759 * @param integer $idUser
760 * @param integer $idEvent
761 * @return boolean
762 * @version 1.0
763 */
764 function EditEvent(string $eventName, string $eventDescription, string $place,
765 $beginningDate, $endDate, int $nbMaxGuest, $img = null, int $idEvent): bool
765 {
766     $query = <<<EX
767     UPDATE evenement
768     SET nom = :eventName, descriptif = :eventDescription, lieu = :place, dateDebut =
769     :beginningDate, dateFin = :endDate, nbMaxParticipant = :nbMaxGuest
770     EX;
771     if ($img != null)
772         $query .= ", urlImage = :urlImg";
773
774     $query .= " WHERE idEvenement = :idEvent";
775
776     try {
777         DatabaseController::beginTransaction();
778         $requestEditEvent = DatabaseController::prepare($query);
779         $requestEditEvent->bindParam(':eventName', $eventName, PDO::PARAM_STR);
780         $requestEditEvent->bindParam(':eventDescription', $eventDescription,
781             PDO::PARAM_STR);
782         $requestEditEvent->bindParam(':place', $place, PDO::PARAM_STR);
783         $requestEditEvent->bindParam(':beginningDate', $beginningDate);
784         $requestEditEvent->bindParam(':endDate', $endDate);
785         $requestEditEvent->bindParam(':nbMaxGuest', $nbMaxGuest, PDO::PARAM_INT);
786
787         if ($img != null)
788             $requestEditEvent->bindParam(':urlImg', $img, PDO::PARAM_STR, 255);
789
790         $requestEditEvent->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
791         $requestEditEvent->execute();
792         DatabaseController::commit();
793         return true;
794     } catch (PDOException $e) {
795         DatabaseController::rollBack();
796         return false;
797     }
798
799 /**
800 * @author Hoarau Nicolas
801 * @date 04.06.2020
802 * @brief Fonction qui récupère l'id des événements d'un utilisateur
803 * @param integer $idUser
804 * @return array|int|null
805 * @version 1.0
806 */
```

```
807 function GetUsersEvent(int $idUser)
808 {
809     $query = <<<EX
810     SELECT idEvenement FROM evenement WHERE idOrganisateur = :idUser;
811     EX;
812
813     try {
814         $requestGetUsersEvents = DatabaseController::prepare($query);
815         $requestGetUsersEvents->bindParam(':idUser', $idUser, PDO::PARAM_INT);
816         $requestGetUsersEvents->execute();
817
818         $result = $requestGetUsersEvents->fetchAll(PDO::FETCH_ASSOC);
819         return $result == false ? [] : $result;
820     } catch (PDOException $e) {
821         return null;
822     }
823 }
824
825 /**
826 * @author Hoarau Nicolas
827 * @date 05.06.2020
828 * @brief Fonction qui compte le nombre de participant aux événement de l'utilisateur
829 * @param array $usersEvent
830 * @return integer
831 * @version 1.0
832 */
833 function CountUsersGuest(array $usersEvent): int
834 {
835     $result = 0;
836
837     foreach ($usersEvent as $event) {
838         $result += GetEventNbGuest($event['idEvenement']);
839
840         if ($result > 0)
841             break;
842     }
843
844     return $result;
845 }
846
847 /**
848 * @author Hoarau Nicolas
849 * @date 05.06.2020
850 * @brief Fonction qui récupère les événements auxquels un utilisateur participe
851 * @param integer $idUser
852 * @return array|null
853 * @version 1.0
854 */
855 function GetEventsParticipating(int $idUser): ?array
856 {
857     $query = <<<EX
858     SELECT idEvenement FROM inscriptions WHERE idUser = :idUser;
859     EX;
860
861     try {
```

```
862     $requestGetEventsParticipating = DatabaseController::prepare($query);
863     $requestGetEventsParticipating->bindParam(':idUser', $idUser, PDO::PARAM_INT);
864     $requestGetEventsParticipating->execute();
865 
866     $result = $requestGetEventsParticipating->fetchAll(PDO::FETCH_ASSOC);
867 
868     return $result;
869 } catch (PDOException $e) {
870     return null;
871 }
872 }
873 
874 /**
875 * @author Hoarau Nicolas
876 * @date 05.06.2020
877 * @brief Fonction qui supprime la participation d'un utilisateur à un événement
878 * @param integer $idUser
879 * @param integer $idEvent
880 * @return boolean
881 * @version 1.0
882 */
883 function DeleteParticipation(int $idUser, int $idEvent): bool
884 {
885     $query = <<<EX
886 DELETE FROM inscriptions WHERE idEvenement = :idEvent AND idUser = :idUser;
887 EX;
888 
889     try {
890         DatabaseController::beginTransaction();
891 
892         $requestDeleteParticipation = DatabaseController::prepare($query);
893         $requestDeleteParticipation->bindParam(':idUser', $idUser, PDO::PARAM_INT);
894         $requestDeleteParticipation->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
895         $requestDeleteParticipation->execute();
896 
897         DatabaseController::commit();
898 
899         return true;
900     } catch (PDOException $e) {
901         DatabaseController::rollBack();
902         return false;
903     }
904 }
905 
906 /**
907 * @author Hoarau Nicolas
908 * @date 05.06.2020
909 * @brief Fonction qui supprime l'invitation d'un utilisateur à un événement
910 * @param integer $idUser
911 * @param integer $idEvent
912 * @return boolean
913 * @version 1.0
914 */
915 function DeleteInvitation(int $idUser, int $idEvent): bool
916 {
```

```
917 $query = <<<EX
918 DELETE FROM invites WHERE idEvenement = :idEvent AND idUser = :idUser;
919 EX;
920
921 try {
922     DatabaseController::beginTransaction();
923
924     $requestDeleteDeleteInvitation = DatabaseController::prepare($query);
925     $requestDeleteDeleteInvitation->bindParam(':idUser', $idUser, PDO::PARAM_INT);
926     $requestDeleteDeleteInvitation->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
927     $requestDeleteDeleteInvitation->execute();
928
929     DatabaseController::commit();
930
931     return true;
932 } catch (PDOException $e) {
933     DatabaseController::rollBack();
934     return false;
935 }
936 }
937
938 /**
939 * @author Hoarau Nicolas
940 * @date 05.06.2020
941 * @brief Fonction qui récupère les événements auxquels un utilisateur est invités
942 * @param integer $idUser
943 * @return array|null
944 * @version 1.0
945 */
946 function GetEventsInvited(int $idUser): ?array
947 {
948     $query = <<<EX
949     SELECT idEvenement FROM invites WHERE idUser = :idUser;
950     EX;
951
952     try {
953         $requestGetEventsParticipating = DatabaseController::prepare($query);
954         $requestGetEventsParticipating->bindParam(':idUser', $idUser, PDO::PARAM_INT);
955         $requestGetEventsParticipating->execute();
956
957         $result = $requestGetEventsParticipating->fetchAll(PDO::FETCH_ASSOC);
958
959         return $result;
960     } catch (PDOException $e) {
961         return null;
962     }
963 }
964
965 /**
966 * @author Hoarau Nicolas
967 * @date 05.06.2020
968 * @brief Fonction qui modifie un utilisateur donné
969 * @param string $nickname
970 * @param string $firstname
971 * @param string $email
```

```
972 * @param string $password
973 * @param string $lastname
974 * @param string $phoneNumber
975 * @param integer $idUser
976 * @param string $oldEmail
977 * @return boolean
978 * @version 1.0
979 */
980 function EditProfile(string $nickname, string $firstname, string $email, string
981     $password = null, string $lastname = null, string $phoneNumber = null, int $idUser,
982     string $oldEmail): bool
983 {
984     $salt = "";
985     $userPassword = null;
986
987     $query = <<<EX
988     UPDATE user
989     SET pseudo = :nickname, prenom = :firstname, email = :email
990     EX;
991
992     if ($lastname != null || $lastname != "") {
993         $query .= ", nom = :lastname";
994
995     if ($phoneNumber != null || $phoneNumber != "") {
996         $query .= ", telephone = :phoneNumber";
997
998     if ($password != null || $password != "") {
999         $salt = GetSalt(['userEmail' => $oldEmail]);
1000        $userPassword = hash('sha256', $password . $salt);
1001        $query .= ", password = :password";
1002    }
1003
1004    try {
1005        DatabaseController::beginTransaction();
1006
1007        $requestEditEvent = DatabaseController::prepare($query);
1008        $requestEditEvent->bindParam(':nickname', $nickname, PDO::PARAM_STR);
1009        $requestEditEvent->bindParam(':firstname', $firstname, PDO::PARAM_STR);
1010        $requestEditEvent->bindParam(':email', $email, PDO::PARAM_STR);
1011
1012        if ($lastname != null)
1013            $requestEditEvent->bindParam(':lastname', $lastname, PDO::PARAM_STR);
1014
1015        if ($phoneNumber != null)
1016            $requestEditEvent->bindParam(':phoneNumber', $phoneNumber, PDO::PARAM_STR);
1017
1018        if ($userPassword != null)
1019            $requestEditEvent->bindParam(':password', $userPassword, PDO::PARAM_STR);
1020
1021        $requestEditEvent->bindParam(':idUser', $idUser, PDO::PARAM_INT);
1022        $requestEditEvent->execute();
1023        DatabaseController::commit();
1024        return true;
1025    }
1026
1027    catch (Exception $e) {
1028        DatabaseController::rollBack();
1029        throw $e;
1030    }
1031
1032    return false;
1033}
```

```
1025 } catch (PDOException $e) {
1026     DatabaseController::rollBack();
1027     return false;
1028 }
1029 }
1030
1031 function GetInvited(int $idEvent): ?array
1032 {
1033     $query = <<<EX
1034     SELECT u.pseudo, i.idUser
1035     FROM invites AS i
1036     JOIN user AS u ON i.idUser = u.idUser
1037     WHERE i.idEvenement = :idEvent
1038 EX;
1039
1040     try {
1041         $requestGetInvited = DatabaseController::prepare($query);
1042         $requestGetInvited->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
1043         $requestGetInvited->execute();
1044         $result = $requestGetInvited->fetchAll(PDO::FETCH_ASSOC);
1045
1046         return $result;
1047     } catch (PDOException $e) {
1048         return null;
1049     }
1050 }
1051
1052 /**
1053 * @author Hoarau Nicolas
1054 * @date 08.06.2020
1055 * @brief Fonction qui supprime un événement données
1056 * @param integer $idEvent
1057 * @return boolean
1058 */
1059 function DeleteEvent(int $idEvent): bool
1060 {
1061     $query = <<<EX
1062     DELETE FROM evenement WHERE idEvenement = :idEvent;
1063 EX;
1064
1065     try {
1066         DatabaseController::beginTransaction();
1067         $requestDeleteEvent = DatabaseController::prepare($query);
1068         $requestDeleteEvent->bindParam(':idEvent', $idEvent, PDO::PARAM_INT);
1069         $requestDeleteEvent->execute();
1070         DatabaseController::commit();
1071         return true;
1072     } catch (PDOException $e) {
1073         DatabaseController::rollBack();
1074         return false;
1075     }
1076 }
1077
```



```
<?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
     */
}
```

```
54 |     public function commit()
55 | {
56 |     $this->_transactionDepth--;
57 |
58 |     if($this->_transactionDepth == 0 || !$this->hasSavepoint()) {
59 |         parent::commit();
60 |     } else {
61 |         $this->exec("RELEASE SAVEPOINT LEVEL{$this->_transactionDepth}");
62 |     }
63 |
64 |
65 | /**
66 | * Rollback current transaction,
67 | *
68 | * @throws PDOException if there is no transaction started
69 | * @return bool|void
70 | */
71 | public function rollBack()
72 | {
73 |
74 |     if ($this->_transactionDepth == 0) {
75 |         throw new PDOException('Rollback error : There is no transaction started');
76 |     }
77 |
78 |     $this->_transactionDepth--;
79 |
80 |     if($this->_transactionDepth == 0 || !$this->hasSavepoint()) {
81 |         parent::rollBack();
82 |     } else {
83 |         $this->exec("ROLLBACK TO SAVEPOINT LEVEL{$this->_transactionDepth}");
84 |     }
85 |
86 |
87 }
```

```
<?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;
    }
}
```

```
51     }
52
53     # end method
54     /**
55      * @brief Passes on any static calls to this class onto the singleton PDO instance
56      * @param $chrMethod    The method to call
57      * @param $arrArguments The method's parameters
58      * @return $mix         The method's return value
59     */
60     final public static function __callStatic($chrMethod, $arrArguments)
61     {
62         $pdo = self::getInstance();
63         return call_user_func_array(array($pdo, $chrMethod), $arrArguments);
64     }
65
66     # end method
67 }
68
```