

DevSkills

Documentation technique



CFPT-I

BENEY Florent

Version 1.0 - 06/2018

Table des matières

[Introduction 3](#_Toc516669164)

[Rappel du cahier des charges 4](#_Toc516669165)

[Objectif du projet 4](#_Toc516669166)

[Description détaillée 4](#_Toc516669167)

[Étude d’opportunité 5](#_Toc516669168)

[Pourquoi ce sujet ? 5](#_Toc516669169)

[Ce qui existe déjà 5](#_Toc516669170)

[Avantage de mon projet 5](#_Toc516669171)

[Analyse fonctionnelle 5](#_Toc516669172)

[Fonctionnalités disponible 5](#_Toc516669173)

[En tant d’utilisateur normal 5](#_Toc516669174)

[En tant qu’administrateur 6](#_Toc516669175)

[Description détaillée des fonctionnalités 6](#_Toc516669176)

[Analyse organique 7](#_Toc516669177)

[Architecture du projet 7](#_Toc516669178)

[Description des fonctions 8](#_Toc516669179)

[Fonction « connecteur » 8](#_Toc516669180)

[Fonction « EmailDispo » 8](#_Toc516669181)

[Fonction « NomDispo » 8](#_Toc516669182)

[Fonction « Login » 8](#_Toc516669183)

[Fonction « GetInfosUtilisateur » 8](#_Toc516669184)

[Fonction « GetCategories » 8](#_Toc516669185)

[Fonction « GetCategorie » 9](#_Toc516669186)

[Fonction « GetTutoriels » 9](#_Toc516669187)

[Fonction « GetTutoriel » 9](#_Toc516669188)

[Fonction « GetTutorielByCategorie » 9](#_Toc516669189)

[Fonction « GetTutorielAvecAbo » 9](#_Toc516669190)

[Fonction « GetAbonnement » 10](#_Toc516669191)

[Fonction « InsererUtilisateur » 10](#_Toc516669192)

[Fonction « InsererCategorie » 10](#_Toc516669193)

[Fonction « InsererTutoriel » 10](#_Toc516669194)

[Fonction « InsererAbonnement » 10](#_Toc516669195)

[Fonction « MajCompte » 10](#_Toc516669196)

[Fonction « MajCompteMDP » 11](#_Toc516669197)

[Fonction « MajCategorie » 11](#_Toc516669198)

[Fonction « MajTutoriel » 11](#_Toc516669199)

[Fonction « SupprimerCategorie » 11](#_Toc516669200)

[Fonction « SupprimerTutoriel » 11](#_Toc516669201)

[Fonction « SupprimerAbonnement » 11](#_Toc516669202)

[Tests et protocole de tests 12](#_Toc516669203)

[Plan de tests 12](#_Toc516669204)

[Tests sur la page « inscription.php » 12](#_Toc516669205)

[Tests sur la page « index.php » 12](#_Toc516669206)

[Tests sur la page « monCompte.php » 13](#_Toc516669207)

[Tests sur la page « editerCategorie.php » 14](#_Toc516669208)

# Introduction

Le projet « DevSkills » consiste à créer un site Internet qui permettra aux personnes qui l’utilisent de consulter des tutoriels centrés sur l’informatique, et ce dans le but d’acquérir plus de compétences dans le domaine souhaité. Les tutoriels du site concerneront surtout les différents langages enseignés au CFPT.

Les tutoriels appartiendront à une certaine catégorie, qui sera défini lors de la création du tutoriel et modifiable par la suite. De par ce fait, les tutoriels pourront être triés par catégorie pour permettre de trouver plus facilement le tutoriel souhaité.

Le nom « DevSkills » est composé de deux termes, « dev » et « skills ». Le premier terme amène l’idée du développement (en anglais comme en français), tandis que le deuxième terme est un mot anglais qui pourrait se traduire par « compétences » ou « aptitudes ». Ce site est donc destiné à l’acquisition de compétences dans le développement, ou même dans l’informatique au sens plus large.

L’utilisateur qui a créé son compte sur le site pourra, après s’y être connecté :

* Modifier ses informations
* Trier les tutoriels en fonction de leurs catégories
* Consulter les tutoriels
* S’abonner aux tutoriels qu’il veut suivre
* Voir les tutoriels auxquels il est abonné
* Consulter la page d’aide qui contiendra la documentation utilisateur (FAQ)

L’administrateur pourra faire les mêmes actions que l’utilisateur, mais aura en plus accès aux fonctionnalités suivantes :

* Gérer les catégories
  + Ajouter une catégorie
  + Modifier une catégorie
  + Supprimer une catégorie
* Gérer les tutoriels
  + Ajouter un tutoriel
  + Modifier un tutoriel
  + Supprimer un tutoriel

Le contenu du tutoriel sera édité à l’aide d’un éditeur WYSIWYG[[1]](#footnote-1). J’ai choisi un éditeur nommé « Trumbowyg », qui peut facilement être adapté à des besoins spécifiques en raisons de ses nombreux plugins. Il est disponible gratuitement ici : <https://alex-d.github.io/Trumbowyg>

Pour créer ce site, j’ai utilisé un template Bootstrap pour rendre le site adapté à l’écran de l’utilisateur, que ce soit sur un mobile, un ordinateur, une tablette, etc. Il est nommé « Folio » et disponible à l’adresse suivante : <https://bootstrapmade.com/folio-bootstrap-portfolio-template>

Ce document permet de mieux comprendre le fonctionnement du site, et comment il a été développé.

# Rappel du cahier des charges

## Objectif du projet

Le site DevSkills permet aux utilisateurs de consulter des tutoriels informatiques pour apprendre les différents langages enseignés au CFPT.

Les visiteurs peuvent consulter des tutoriels fait de texte et d'images et l'administrateur du site peut en créer.

## Description détaillée

Le projet contiendra les points suivants :

* Une page de login pour l'administrateur et les utilisateurs
* Une page d’inscription pour les utilisateurs
* Une page pour que l'administrateur puisse ajouter une catégorie
* Une page pour que l'administrateur puisse voir toutes les catégories et en supprimer/modifier une
* Une page pour que l'administrateur puisse ajouter/modifier un tutoriel
* Une page pour que l'administrateur puisse voir tous les tutoriels et en supprimer/modifier un
* Une page pour qu’un utilisateur puisse voir les tutoriels auxquels il est abonné
* Une page d'accueil avec un message de bienvenue et un menu des catégories dans lequel on pourra retrouver les tutoriels listés par catégorie
* Une page d'aide avec la documentation utilisateur (FAQ)
* Un menu avec au minimum un bouton pour accueil, pour le login/déconnexion et pour voir la documentation
* Un affichage dynamique s’adaptant aux catégories et affichant les tutoriels dans chacune d’entre elles
* Une fonctionnalité pour qu’un utilisateur puisse s’abonner à un tutoriel

Les tutoriels seront enregistrés dans une base de données et seront composés de :

* Un grand champs texte pouvant contenir du HTML / Javascript / CSS (dont des liens pour les fichiers multimédia) afin d’afficher ce que l'administrateur souhaite dans un tutoriel via un éditeur WYSIWYG
* Un titre qui est un champs texte
* Un champs catégorie qui permet à un tutoriel de faire partie d'une catégorie (une seule uniquement)

Les catégories sont composées d’un titre, d’une description et d’un lien vers une image permettant d’identifier la catégorie.

Les utilisateurs ainsi que leurs abonnements sont enregistrée dans deux tables qui permettent de faire le lien entre les utilisateurs et les tutoriels. Ils pourront s’inscrire et se connecter / déconnecter.

Si une catégorie contenant des tutoriels est supprimée, les tutoriels n’apparaissent plus sur le site mais restent en base de données.

Les images et autres contenus spéciaux sont ajoutés par l'administrateur dans le site dans un répertoire prévu à cet effet (pas de fonctionnalité d’Upload).

Le site sera responsif et doit s’adapter aux éléments à afficher sur mobile ou PC. Entre autres le menu doit afficher les catégories de façon élégante.

# Étude d’opportunité

## Pourquoi ce sujet ?

Lors du début de mon TPI, je n’avais pas d’idée de sujet, mais je savais que je voulais faire un site web. Lors de discussions avec le mentor de mon TPI, monsieur Aliprandi, il m’a donné une idée de sujet : développer un site web, regroupant différents tutoriels concernant l’informatique. J’ai accepté cette idée avec plaisir, et c’est comme ceci que le choix de mon sujet fut fait.

## Ce qui existe déjà

Il existe déjà des sites similaires à « DevSkills », comme par exemple « OpenClassrooms[[2]](#footnote-2) » ou bien « Codecademy[[3]](#footnote-3) », qui proposent également de suivre des cours en ligne sur des sujets concernant (principalement) l’informatique.

## Avantage de mon projet

Mais la différence principale entre ces sites et le mien et le modèle économique : dans ces deux sites, il y a une option « Premium », mais payante. Tandis que « DevSkills » est un site totalement gratuit !

# Analyse fonctionnelle

Cette partie de la documentation technique explique ce que le projet doit pouvoir faire, conformément au cahier des charges. Comme il y a deux types de comptes (administrateur et utilisateur normal), les fonctionnalités seront par conséquent séparées en deux également.

## Fonctionnalités disponible

### En tant d’utilisateur normal

* S’inscrire au site
* Se connecter à son compte
* Consulter les différents tutoriels, trié par catégorie
* S’abonner et se désabonner aux tutoriels (pour pouvoir les suivre plus facilement)
* Voir les tutoriels auxquels il est abonné
* Modifier les informations de son compte¨
* Consulter la documentation utilisateur (FAQ)

### En tant qu’administrateur

L’administrateur peut faire les mêmes choses que l’utilisateur normal, mais en plus, il peut faire les choses suivantes :

* Gérer les catégories (ajout, modification, suppression et visualisation)
* Gérer les tutoriels (ajout, modification, suppression et visualisation)

## Description détaillée des fonctionnalités

(À compléter)

# Analyse organique

## Architecture du projet

Les fichiers du site sont séparés entre plusieurs dossiers :

* assets : contient les fichiers nécessaires au bon fonctionnement du template Bootstrap ainsi que le fichier « Readme.txt » du template
* functPHP : contient les fichiers « PDO.php » et « functions.php »
  + PDO.php : contient les différentes constantes du site et la connexion à la base de données
  + functions.php : contient toutes les fonctions du site
* img : contient les différentes images affichées sur le site, notamment celles des catégories
* inc : abréviation de « include », contient des fichiers qui sont appelés dans d’autres fichiers du site. Découper le site de cette manière permet de faire une modification dans un des fichiers include, et la modification se fera dans tous les fichiers où le fichier en question est appelé.
  + footer.inc.php : le pied de page du site, qui contient une flèche vers le haut pour revenir en haut de la page, et diverses informations sur le projet
  + header.inc.php : l’en-tête du site, la partie qui s’affiche tout en haut de la page (quand la barre de navigation n’est pas affichée)
  + navbar.inc.php : la barre de navigation du site, contenant divers liens pour naviguer dans le site et qui s’adapte en fonction de l’utilisateur qui est connecté
  + templateCSS.inc.php : fichier contenant les liens pour appeler le CSS du template, à mettre entre les balises « head » de la page
  + templateJS.inc.php : fichier contenant les liens pour appeler le JavaScript du template, à mettre avant la fin de la balise « body » de la page
* trumbowyg : ce dossier contient les fichiers pour faire fonctionner l’éditeur WYSIWYG que j’ai choisi d’utiliser, appelé « Trumbowyg ». Parmi les dossiers qu’il contient, il y a le dossier « inc » :
  + inc : ce dossier contient les fichiers suivants :
    - trumbowygCSS.inc.php : fichier contenant les liens pour appeler le CSS de Trumbowyg, nécessaire lorsqu’on veut utiliser l’éditeur WYSIWYG, à mettre entre les balises « head » de la page
    - trumbowygJS.inc.php : fichier contenant les liens pour appeler le JavaScript de Trumbowyg, nécessaire lorsqu’on veut utiliser l’éditeur WYSIWYG, à mettre avant la fin de la balise « body » de la page
* À la racine du site, il y a toutes les différentes pages du site qui sont atteignable par les personnes naviguant sur le site

## Description des fonctions

### Fonction « connecteur »

Cette fonction est déclarée dans le fichier « PDO.php » et sert à établir la connexion à la base de données. Elle ne reçoit pas de paramètre, mais utilise quatre constantes déclarée dans le même fichier. Ces constantes sont « DB\_HOST » pour l’hôte du serveur, « DB\_NAME » pour le nom de la base de données, « DB\_USER » pour l’utilisateur qui se connecte et « DB\_PASSWORD » pour le mot de passe utilisé lors de la connexion. Cette fonction retourne une instance de l’objet PDO.

### Fonction « EmailDispo »

Cette fonction, ainsi que toutes les suivantes, sont déclarées dans le fichier « functions.php ». Elle sert à indiquer si un email est disponible ou pas. Elle est utilisée lors de l’inscription ou de la modification du compte de l’utilisateur, par exemple. Elle reçoit un paramètre, nommé « email », qu’elle utilise pour construire une requête SQL (un SELECT avec l’email dans la clause WHERE) qui lui permet de savoir si l’email est déjà utilisé. En effet, si le SQL retourne une ligne, c’est que l’email est déjà utilisé et donc pas disponible. Elle retourne « True » si l’email est disponible, « False » sinon.

### Fonction « NomDispo »

Cette fonction sert à indiquer si un nom est disponible ou pas. Elle est utilisée lors de l’inscription ou de la modification du compte de l’utilisateur, par exemple. Elle reçoit un paramètre, nommé « nom », qu’elle utilise pour construire une requête SQL (un SELECT avec le nom dans la clause WHERE) qui lui permet de savoir si le nom est déjà utilisé. En effet, si le SQL retourne une ligne, c’est que le nom est déjà utilisé et donc pas disponible. Elle retourne « True » si le nom est disponible, « False » sinon.

### Fonction « Login »

Cette fonction permet de savoir si un couple nom d’utilisateur / mot de passer est correct. Elle est utilisée lors de la connexion. Elle reçoit deux paramètres, le nom d’utilisateur ainsi que le mot de passe, qu’elle utilise pour construire une requête SQL (un SELECT avec le nom et le mot de passe dans la clause WHERE) qui lui permet de savoir si le couple nom d’utilisateur / mot de passe est correct, et donc si le login est correct. En effet, si le SQL retourne une ligne, c’est que les données correspondent. Elle retourne un tableau contenant l’id de l’utilisateur, s’il est admin ou pas, et son email (des données qui pourront être réutilisées plus tard dans le site) si le login est juste, sinon « False ».

### Fonction « GetInfosUtilisateur »

Cette fonction permet de récupérer les informations d’un utilisateur. Elle est utilisée lors de l’affichage des informations de l’utilisateur, dans la page « monCompte.php ». Elle reçoit un paramètre, nommé « id », qui correspond à l’id de l’utilisateur dont on veut récupérer les informations. Elle utilise ce paramètre pour construire une requête SELECT du nom, du mot de passe ainsi que de l’email, avec l’id dans la clause WHERE. Elle retourne un tableau contenant le nom d’utilisateur, le mot de passe et l’email de l’utilisateur.

### Fonction « GetCategories »

Cette fonction permet de récupérer toutes les catégories présentes dans la base de données. Elle est utilisée notamment dans la page « accueil.php » et « gererCategories.php ». Elle ne reçoit pas de paramètre, étant donné qu’il s’agit simplement d’un SELECT effectué sur la table « categorie ». Elle retourne un tableau à deux dimensions, dont chaque ligne contient l’id d’une catégorie, son nom, sa description ainsi que le lien de son image.

### Fonction « GetCategorie »

Cette fonction récupère les informations d’une catégorie. Elle est utilisée sur la page « editerCategorie.php », lorsqu’il faut modifier une catégorie. Elle est donc très similaire à la fonction précédente (GetCategories), à la différence près que celle-ci reçoit un paramètre. Il est nommé « id » et est utilisé dans la clause WHERE de la requête SQL SELECT pour récupérer une catégorie en fonction de son id. Elle retourne un tableau contenant le nom de la catégorie, sa description ainsi que le lien de son image.

### Fonction « GetTutoriels »

Cette fonction permet de récupérer tous les tutoriels contenus dans la base de données. Elle est appelée dans la page « gererTutoriels.php », lorsqu’il faut afficher tous les tutoriels pour choisir lequel modifier. Elle ne reçoit pas de paramètre étant donné qu’il s’agit d’un SELECT sur toute la table « tutoriel ». Elle retourne un tableau à deux dimensions, dont chaque ligne contient l’id du tutoriel, son titre, le nom de la catégorie à laquelle il appartient ainsi que le lien de l’image de sa catégorie.

### Fonction « GetTutoriel »

Cette fonction récupère les informations d’un tutoriel. Elle est appelée notamment sur la page « editerTutoriel.php ». Elle est très similaire à la fonction précédente (GetTutoriels), mais celle-ci reçoit un paramètre, nommé « id », qui est utilisé dans la clause WHERE de la requête SLQ SELECT pour récupérer un tutoriel en fonction de son id. Elle retourne un tableau contenant le titre du tutoriel, son contenu ainsi que l’id de la catégorie à laquelle il appartient.

### Fonction « GetTutorielByCategorie »

Cette fonction permet de récupérer les tutoriels qui appartiennent à une certaine catégorie. Elle est appelée sur la page « categorie.php », lorsqu’il faut afficher les tutoriels en fonction de la catégorie sélectionnée. Comme paramètre, elle reçoit l’id de la catégorie, nommé « id ». Il est utilisé dans la clause WHERE de la requête SQL SELECT pour récupérer les informations des tutoriels appartenant à une certaine catégorie. Elle retourne un tableau à deux dimensions, contenant l’id du tutoriel, son titre ainsi le nom de la catégorie à laquelle il appartient.

### Fonction « GetTutorielAvecAbo »

Cette fonction permet de récupérer les informations d’un tutoriel et de savoir si l’utilisateur est abonné au tutoriel en question. Elle est utilisée sur la page « tutoriel.php », lors de l’affichage du tutoriel, et pour savoir si le bouton doit proposer de s’abonner ou bien de se désabonner du tutoriel. Elle est très semblable à la fonction GetTutoriel, mais cette fonction permet en plus de savoir si l’utilisateur qui consulte le tutoriel y est abonné. Cette fonction reçoit deux paramètres : le premier, nommé « idTutoriel », permet d’identifier le tutoriel choisi, et le deuxième, nommé « idUtilisateur », correspond à l’utilisateur qui consulte le tutoriel. Ces deux paramètres permettent de construire une requête SQL SELECT similaire à celle de la fonction GetTutoriel, avec en plus une sous-requête qui permet de récupérer si l’utilisateur est abonné au tutoriel en question. Elle retourne un tableau contenant le titre du tutoriel, son contenu, l’id de la catégorie à laquelle il appartient et une variable bool appelée « abonne » qui contient « True » si l’utilisateur est abonné au tutoriel et « False » sinon.

### Fonction « GetAbonnement »

Cette fonction permet de récupérer les tutoriels auxquels un utilisateur est abonné. Elle est utilisée sur la page « abonnement.php », lorsqu’il faut afficher les tutoriels auxquels l’utilisateur est abonné. Elle reçoit un paramètre, nommé « idUtilisateur », qui correspond à l’id de l’utilisateur en question. Ce paramètre est utilisé dans la clause WHERE de la requête SQL SELECT pour sélectionner l’utilisateur dont on veut récupérer les informations. Elle retourne un tableau à deux dimensions, dont chaque ligne contient l’id du tutoriel, son titre ainsi que le nom de la catégorie à laquelle il appartient.

### Fonction « InsererUtilisateur »

Cette fonction permet d’ajouter un nouvel utilisateur dans la base de données. Elle est utilisée sur la page « inscription.php », lorsqu’une personne s’inscrit sur le site. Elle reçoit trois paramètres, « nom », « email » et « motDePasse ». Ces trois paramètres sont utilisés pour créer une requête SQL INSERT INTO, qui ajoute un enregistrement dans la table « utilisateur », avec les données reçues en paramètre. Cette fonction ne retourne donc rien.

### Fonction « InsererCategorie »

Cette fonction permet d’ajouter une nouvelle catégorie dans la base de données. Elle est utilisée sur la page « editerCategorie.php », lorsque l’administrateur ajoute une catégorie sur le site. Elle reçoit trois paramètres, « nomCategorie », « lienImageCategorie » et « descriptionCategorie ». Ces trois paramètres sont utilisés pour créer une requête SQL INSERT INTO, qui ajoute un enregistrement dans la table « categorie », avec les données reçues en paramètre. Cette fonction ne retourne donc rien.

### Fonction « InsererTutoriel »

Cette fonction permet d’ajouter un nouveau tutoriel dans la base de données. Elle est utilisée sur la page « editerTutoriel.php », », lorsque l’administrateur ajoute un tutoriel sur le site. Elle reçoit trois paramètres, « titreTutoriel », « contenuTutoriel » et « categorieTutoriel ». Ces trois paramètres sont utilisés pour créer une requête SQL INSERT INTO, qui ajoute un enregistrement dans la table « tutoriel », avec les données reçues en paramètre. Cette fonction ne retourne donc rien.

### Fonction « InsererAbonnement »

Cette fonction permet d’ajouter un nouvel abonnement dans la base de données. Elle est utilisée sur la page « tutoriel.php », lorsqu’une personne s’abonne à un tutoriel. Elle reçoit deux paramètres, « idTutoriel » et « idUtilisateur ». Ces deux paramètres sont utilisés pour créer une requête SQL INSERT INTO, qui ajoute un enregistrement dans la table « abonnement », avec les données reçues en paramètre. Cette fonction ne retourne donc rien.

### Fonction « MajCompte »

Cette fonction permet de mettre à jour le compte d’un utilisateur, sans le mot de passe. Elle est utilisée sur la page « monCompte.php », lorsqu’un utilisateur met à jour son compte. Elle reçoit trois paramètre, « id », « nom » et « email ». Ces trois paramètres sont utilisés pour créer une requête SQL UPDATE, qui met à jour un enregistrement de la table « utilisateur » avec les données reçues en paramètres. L’enregistrement concerné est celui qui correspond à l’id reçu en paramètre. Cette fonction ne retourne donc rien.

### Fonction « MajCompteMDP »

Cette fonction permet de mettre à jour le compte d’un utilisateur, avec le mot de passe. Elle est utilisée sur la page « monCompte.php », lorsqu’un utilisateur met à jour son compte. Elle reçoit quatre paramètre, « id », « nom », « email » et « motDePasse ». Ces quatre paramètres sont utilisés pour créer une requête SQL UPDATE, qui met à jour un enregistrement de la table « utilisateur » avec les données reçues en paramètres. L’enregistrement concerné est celui qui correspond à l’id reçu en paramètre. Cette fonction ne retourne donc rien.

### Fonction « MajCategorie »

Cette fonction permet de mettre à jour une catégorie. Elle est utilisée sur la page « editerCategorie.php », lorsque l’administrateur modifie une catégorie du site. Elle reçoit quatre paramètre, « id », « nom », « description » et « lienImage ». Ces quatre paramètres sont utilisés pour créer une requête SQL UPDATE, qui met à jour un enregistrement de la table « categorie » avec les données reçues en paramètres. L’enregistrement concerné est celui qui correspond à l’id reçu en paramètre. Cette fonction ne retourne donc rien.

### Fonction « MajTutoriel »

Cette fonction permet de mettre à jour un tutoriel. Elle est utilisée sur la page « editerTutoriel.php », », lorsque l’administrateur modifie un tutoriel du site. Elle reçoit quatre paramètre, « id », « titreTutoriel », « contenuTutoriel » et « categorieTutoriel ». Ces quatre paramètres sont utilisés pour créer une requête SQL UPDATE, qui met à jour un enregistrement de la table « tutoriel » avec les données reçues en paramètres. L’enregistrement concerné est celui qui correspond à l’id reçu en paramètre. Cette fonction ne retourne donc rien.

### Fonction « SupprimerCategorie »

Cette fonction permet de supprimer une catégorie. Elle est utilisée sur la page « supprimerCategorie.php », lorsqu’un administrateur décide de supprimer une catégorie du site. Elle reçoit un seul paramètre, « idCategorie », qui correspond à l’id de la catégorie à supprimer. Ce paramètre est utilisé dans la clause WHERE d’une nouvelle requête SQL DELETE FROM, qui supprime la catégorie qui correspond à l’id reçu en paramètre. Cette fonction ne retourne donc rien.

### Fonction « SupprimerTutoriel »

Cette fonction permet de supprimer un tutoriel. Elle est utilisée sur la page « supprimerTutoriel.php », lorsqu’un administrateur décide de supprimer un tutoriel du site. Elle reçoit un seul paramètre, « idTutoriel », qui correspond à l’id du tutoriel à supprimer. Ce paramètre est utilisé dans la clause WHERE d’une nouvelle requête SQL DELETE FROM, qui supprime le tutoriel qui correspond à l’id reçu en paramètre. Cette fonction ne retourne donc rien.

### Fonction « SupprimerAbonnement »

Cette fonction permet de supprimer l’abonnement d’un utilisateur à un tutoriel. Elle est utilisée sur la page « tutoriel.php », lorsqu’un utilisateur décide de se désabonner d’un tutoriel. Elle reçoit deux paramètres, « idTutoriel », qui correspond à l’id du tutoriel, et « idUtilisateur », qui correspond à l’id de l’utilisateur. Ces paramètres sont utilisés dans la clause WHERE d’une nouvelle requête SQL DELETE FROM, qui supprime l’abonnement qui contient l’id du tutoriel et celui de l’utilisateur. Cette fonction ne retourne donc rien.

# Tests et protocole de tests

## Plan de tests

### Tests sur la page « inscription.php »

|  |  |  |
| --- | --- | --- |
| N° | Scénario | Résultat attendu |
| 01 | Modifier le code HTML de la page et retirer les attributs « required » des champs nom, email, mot de passe et répéter le mot de passe  Appuyer sur le bouton « Inscription » | Un message d’erreur devrait s’afficher, indiquant qu’il faut remplir tous les champs |
| 02 | Modifier le code HTML de la page et mettre le type du champ « email » à « text »  Remplir le champ « email » avec une valeur autre qu’un email  Remplir les autres champs avec des valeurs correctes  Appuyer sur le bouton « Inscription » | Un message d’erreur devrait s’afficher |
| 03 | Remplir le champ « nom » avec un nom déjà existant  Remplir les autres champs avec des valeurs correctes  Appuyer sur le bouton « Inscription » | Un message d’erreur devrait s’afficher, indiquant que le nom n’est pas disponible |
| 04 | Remplir le champ « email » avec un email déjà existant  Remplir les autres champs avec des valeurs correctes  Appuyer sur le bouton « Inscription » | Un message d’erreur devrait s’afficher, indiquant que l’email n’est pas disponible |
| 05 | Remplir les deux champs « mot de passe » avec des valeurs différentes  Remplir les autres champs avec des valeurs correctes  Appuyer sur le bouton « Inscription » | Un message d’erreur devrait s’afficher, indiquant que les deux mots de passe doivent correspondre |
| 06 | Remplir les tous les champs avec des valeurs correctes (pas un nom déjà utilisé, pas un email déjà utilisé, mots de passes correspondants)  Appuyer sur le bouton « Inscription » | Un message de succès devrait s’afficher, indiquant que le compte a été créé et proposant un lien pour se connecter |

### Tests sur la page « index.php »

|  |  |  |
| --- | --- | --- |
| N° | Scénario | Résultat attendu |
| 07 | Modifier le code HTML de la page et retirer les attributs « required » des champs nom et mot de passe  Appuyer sur le bouton « Connexion » | Un message d’erreur devrait s’afficher, indiquant qu’il faut remplir tous les champs |
| 08 | Remplir le champ « nom » ou le champ « mot de passe » avec une valeur correcte  Remplir l’autre champ avec une valeur quelconque  Appuyer sur le bouton « Connexion » | Un message d’erreur devrait s’afficher, indiquant que le nom ou le mot de passe n’est pas correct |
| 09 | Remplir le champ « nom » et « mot de passe » avec les informations d’un administrateur  Appuyer sur le bouton « Connexion » | Vous devriez être redirigé sur la page d’administration |
| 10 | Remplir le champ « nom » et « mot de passe » avec les informations d’un utilisateur normal  Appuyer sur le bouton « Connexion » | Vous devriez être redirigé sur la page d’accueil |

### Tests sur la page « monCompte.php »

|  |  |  |
| --- | --- | --- |
| N° | Scénario | Résultat attendu |
| 11 | Modifier le code HTML de la page et retirer les attributs « required » des champs nom et email  Appuyer sur le bouton « Modifier » | Un message d’erreur devrait s’afficher, indiquant qu’il faut au moins remplir le champ « nom » et « email » |
| 12 | Modifier le code HTML de la page et mettre le type du champ « email » à « text »  Remplir le champ « email » avec une valeur autre qu’un email  Remplir les autres champs avec des valeurs correctes  Appuyer sur le bouton « Modifier » | Un message d’erreur devrait s’afficher |
| 13 | Remplir le champ « nom » avec un nom déjà utilisé  Remplir les autres champs avec des valeurs correctes  Appuyer sur le bouton « Modifier » | Un message d’erreur devrait s’afficher, indiquant que le nom est déjà utilisé |
| 14 | Remplir le champ « email » avec un email déjà utilisé  Remplir les autres champs avec des valeurs correctes  Appuyer sur le bouton « Modifier » | Un message d’erreur devrait s’afficher, indiquant que l’email est déjà utilisé |
| 15 | Remplir le champ « mot de passe actuel » avec un mot de passe différent de l’actuel  Remplir les autres champs avec des valeurs correctes  Appuyer sur le bouton « Modifier » | Un message d’erreur devrait s’afficher, indiquant que le mot de passe actuel est incorrect |
| 16 | Remplir les champs « nouveau mot de passe » et « répéter le mot de passe » avec des valeurs différentes  Remplir les autres champs avec des valeurs correctes  Appuyer sur le bouton « Modifier » | Un message d’erreur devrait s’afficher, indiquant que les mots de passe doivent correspondre |
| 17 | Remplir tous les champs avec des valeurs correctes  Appuyer sur le bouton « Modifier » | Un message de succès devrait s’afficher, indiquant que les informations ont été modifiée avec succès |

### Tests sur la page « editerCategorie.php »

|  |  |  |
| --- | --- | --- |
| N° | Scénario | Résultat attendu |
| 18 | Modifier le code HTML de la page et retirer les attributs « required » de tous les champs  Appuyer sur le bouton « Ajouter » | Un message d’erreur devrait s’afficher, indiquant qu’il faut remplir tous les champs |
|  | Remplir tous les champs avec des valeurs correctes  Appuyer sur le bouton « Ajouter » | Un message de succès devrait s’afficher, indiquant que la catégorie a bien été ajoutée |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

1. WYSIWYG : <https://fr.wikipedia.org/wiki/What_you_see_is_what_you_get> [↑](#footnote-ref-1)
2. Lien du site d’OpenClassrooms : <https://openclassrooms.com> [↑](#footnote-ref-2)
3. Lien du site de Codecademy : <https://www.codecademy.com/fr> [↑](#footnote-ref-3)