GIROUD Antoine

BOIDIN Benoît

Projet de Programmation Web

MIASHS L3

Table des matières

[II. Introduction 3](#_Toc71986387)

[III. Architecture du site 3](#_Toc71986388)

[IV. Schéma conceptuel de la base de données 5](#_Toc71986389)

[A. Modèle conceptuel 5](#_Toc71986390)

[B. Modèle logique 5](#_Toc71986391)

[V. Méthodes et algorithmes significatifs 5](#_Toc71986392)

[A. Requêtes afficher. 6](#_Toc71986393)

[B. Ajout d’un site (avec l’image etc…). 6](#_Toc71986394)

[C. Connexion. 6](#_Toc71986395)

[D. Inscription. 6](#_Toc71986396)

[E. Affichage du template des commentaires avec les boutons « supprimer » et « afficher ». 6](#_Toc71986397)

[VI. Conclusion 6](#_Toc71986398)

[A. Problèmes rencontrés. 6](#_Toc71986399)

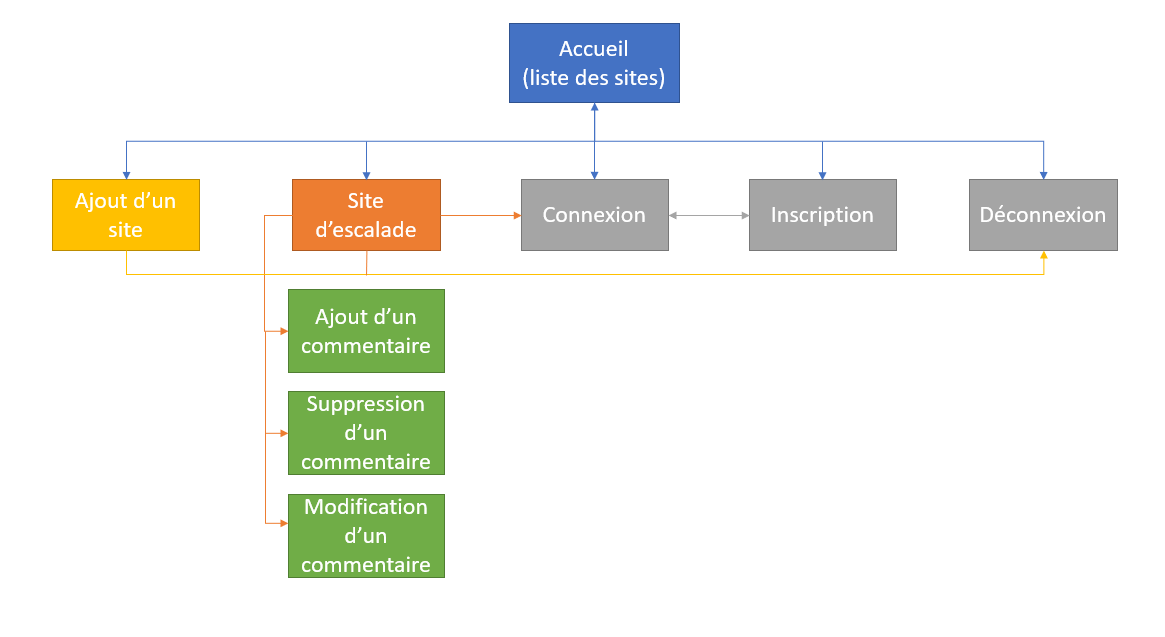
[B. Bilan. 6](#_Toc71986400)

# Introduction

Notre première réflexion lors de ce projet était de créer un site web qui pourrait nous être utile même après le rendu final. Étant tous deux passionnés d’escalade, nous avons alors eu l’idée de créer un site web répertoriant les sites d’escalade présents aux alentours de Lyon. En effet, lors de l’organisation de sortie d’escalade, il est souvent difficile de repérer le site idéal et de s’accorder sur celui-là. Notre site web regroupant la majorité des salles et falaises aux alentours, cela simplifie alors grandement la recherche. De plus, il est possible de visionner les détails de ces lieux tels que leur localisation, la difficulté des voies ainsi que leur nombre, et enfin le type d’escalade qui permet de connaitre le matériel à emporter.

Les utilisateurs ont la possibilité de rajouter des sites dans le cas où certains ont été oubliés. De plus, un espace commentaire est dédié à chacun d’eux. De ce fait, les utilisateurs peuvent donner leur avis sur les voies qu’ils ont grimpé afin d’indiquer par exemple si le niveau indiqué est correct, si le lieu est agréable ou encore d’organiser des sorties à l’avance avec les autres utilisateurs.

# Architecture du site



Le graphique ci-dessus représente l’architecture de notre site web, avec les différentes parties étant séparées sous forme de cluster.

On peut donc observer dans un premier temps l’accueil (en bleu), comprenant la liste de tous les sites d’escalades de notre base de données. Il est ensuite possible d’accéder à toutes les autres pages du site web à partir de cet accueil.

En jaune est représenté la page permettant d’ajouter, si l’utilisateur s’est identifié, un nouveau site d’escalade, au sein de laquelle nous pouvons donc retrouver un formulaire où il est possible de renseigner toutes les données nécessaires (soit le nom du site, le lieu, le niveau, le nombre et le type de voies, ainsi qu’une image).

En orange, on peut retrouver la page qui permet donc d’afficher toutes ces données. Nous pouvons donc cliquer sur un des sites affichés à l’accueil, où nous serons alors renvoyés sur la page de ce site. A partir de là, il sera possible pour un utilisateur connecté d’ajouter un commentaire, qu’il pourra par la suite supprimer ou modifier.

Ensuite, nous avons affiché en gris les pages connexion, inscription et déconnexion. Il est possible de se rendre sur ces pages à partir d’un menu qui est présent sur toutes les autres pages du site. Il est donc possible de se rendre sur la page de connexion, d’inscription ou de déconnexion à partir de l’accueil ou d’un site d’escalade. Pour l’ajout d’un site, il faut forcément être connecté, donc seule la page déconnexion est accessible d’ici.

Pendant toute la navigation, l’utilisateur a accès à un menu dynamique, qui s’adapte en fonction de la connexion pour proposer les options cohérentes, ainsi que le pied de page avec les mentions légales.

Pour l’affichage dynamique des pages, nous avons utilisé le même moteur de template que lors des séances de TD.

Nous avons utilisé, pour la forme de notre projet, un template CSS qui donne un aspect professionnel au site.

# Schéma conceptuel de la base de données

## Modèle conceptuel

## Modèle logique

Site (Idsite, Nomsite, Localisation, Niveau, Nbvoies, Image, #Idtype)

Type (Idtype, Nomtype)

Message (Idmessage, Date, Contenu, #Idgrimpeur, #Idsite)

Grimpeur (Idgrimpeur, Login, Passwd)

# Méthodes et algorithmes significatifs

J’ai mis ce à quoi je pensais mais si tu trouves que ce n’est pas pertinent ou qu’il manque des choses n’hésite pas à modifier.

## Requêtes afficher.

Le site requiert un grand nombre de requêtes, en particulier pour l’affichage des sites, informations et commentaires. Pour éviter les répétitions, nous avons donc créé une classe « Requete », qui premet de créer un objet et d’exécuter les fonctions suivantes :

* L’exécution consiste à se connecter à la base, et à préparer la requête.
* La requête d’affichage prend comme argument le type de données à afficher dans le modèle, et qui associe des noms et valeurs en fonction du template voulu.

## Ajout d’un site (avec l’image etc…).

* Déplacement du fichier original dans un dossier d’images.
* Insertion des informations et du nom de l’image dans la base.

L’ajout d’un site fait appel à la classe Requete pour entrer les informations dans la base de données. Pour ce qui est de l’image, le fichier est transféré dans un répertoire, et son nom est conservé dans la base de données.

## Connexion.

Lorsque l’utilisateur entre ses identifiants, la page créé un objet qui permet de vérifier les arguments, et les transférer dans les variables de sessions s’ils sont corrects. Après cela, l’utilisateur est redirigé vers l’index.

## Inscription.

La classe d’inscription contient une requête pour l’ajout des nouveaux identifiants, si les mots de passe sont similaires, et fait appel à la classe de connexion pour connecter l’utilisateur dès son inscription.

## Affichage du template des commentaires avec les boutons « supprimer » et « afficher ».

Afin que l’utilisateur puisse modifier ou supprimer les commentaires qu’il a postés, nous avons intégré des liens et un formulaire dans le template. La fonction afficher de la classe Requete qui correspond aux commentaires se charge donc de détecter quels commentaires ont été postés par l’utilisateur connecté. Lorsqu’il fait une action, il est redirigé sur la même page pour pouvoir observer ce qu’il vient de changer.

# Conclusion

## Problèmes rencontrés.

Nous avons rencontré plusieurs problèmes lors du développement de notre site.

Premièrement, le template de style que nous utilisons comprend un grand nombre de commentaires pour guider les développeurs dans leur travail. Cependant, ces commentaires commencent par BEGIN et terminent par END, ce qui est entré en conflit avec le moteur de template que nous utilisons en TD.

Nous avons décidé de placer la connexion PDO dans la classe qui se chargeait d’effectuer les requêtes, pour éviter les répétitions, mais ceci nous a empêché d’utiliser un seul fichier avec les identifiants de connexion. Ceci présente donc un inconvénient pour la mise en ligne du site et le changement des identifiants de connexion à la base de données.

## Bilan.

Lors de ce projet, nous avons compris qu’un site dynamique demande un nombre important de traitements, et qu’il est nécessaire de séparer les différents langages pour éviter les confusions, et surtout afin de séparer les paradigmes.

La programmation orienté objet permet de limiter drastiquement le nombre de lignes de code, en réutilisant les mêmes fonctions pour plusieurs usages. Un des avantages que nous avons rapidement remarqué est que, lorsqu’un changement ou une amélioration est apportée, elle se répercute automatiquement sur tous les éléments concernés.