

Mathématiques et Informatique Appliquées aux Sciences Humaines et Sociales (MIASHS)

Technologie et handicap (HANDI)

UE Interface Homme - Machine / Programmation Web Accessible

>>>

## Elaboration d'un site web accessible Opéra national de Paris

Bilo Boury, Charles Cascio, Nicolas Roelandt, Nassim Yousfi

26 juin 2017





# Sommaire

Titre . . . . .	1
Sommaire . . . . .	3
1    Introduction . . . . .	4
2    Conception . . . . .	4
2.1    Répartition des tâches . . . . .	4
2.2    maquette préliminaire . . . . .	4
2.3    Modèle Conceptuel de Données . . . . .	5
3    Réalisation de la base de données . . . . .	5
3.1    Design général . . . . .	5
3.2    Accessibilité . . . . .	10
4    PHP . . . . .	10
4.1    Gestion des sessions . . . . .	10
4.2    Requêtes en base . . . . .	10
5    Gestion des réservations . . . . .	10
5.1    Plan de salle . . . . .	10
6    Versionnement . . . . .	10
6.1    Logiciels de gestion du versionnement . . . . .	10
6.1.1    GIT . . . . .	10
6.1.2    GitKraken . . . . .	11
6.2    Hébergement . . . . .	11
6.3    avantages . . . . .	11
6.4    inconvénients . . . . .	12
6.5    problèmes rencontrés . . . . .	12
7    Evolutions possibles / points non traités . . . . .	12
7.1    Fonctionnalités / Interface du site . . . . .	12
7.2    Page gestion.php . . . . .	12
7.3    Page spectacle.php . . . . .	12
8    Conclusion . . . . .	12
1    Annexes . . . . .	12
1.1    Documentation de la base de données . . . . .	12
Liste des figures . . . . .	23
Liste des tables . . . . .	24

## 1 Introduction

Ce projet nous a été confié dans le cadre du Master *Mathématiques et Informatique Appliquées aux Sciences Humaines et Sociales (MIASHS)* parcours *Technologie et handicap (HANDI)*. Il s'agit d'un projet associant les cours d'Interface Homme Machine (Dominique Archambault) et Programmation Web Accessible (Isis Truck) de la première année.

L'objectif est de développer un site web permettant la réservation de place de spectacle. Le site doit être accessible et donc fonctionnel pour des personnes en situation de handicap (visuel ou moteur).

Nous avons voulu travailler sur un cas le plus concret possible, c'est pourquoi nous nous sommes inspirés du site de l'Opéra national de Paris.

Si les spectacles présentés dans ce projet ont réellement eu lieu, soit à l'Opéra Bastille ou au Palais Garnier, l'inspiration s'arrête là. Le plan de salle, les dates de représentation, les clients sont totalement fictifs.

Le site présenté ici a été développé par une équipe de 4 étudiants :

- Bilo Boury
- Charles Cascio
- Nicolas Roelandt
- Nassim Yousfi

## 2 Conception

Ce site a été développé majoritairement le soir et le week-end, nous avons donc voulu le garder le plus simple et fonctionnel possible. D'une part car dans un soucis d'économie de temps et d'énergie, et d'autre part dans un souci d'accessibilité.

### 2.1 Répartition des tâches

1. maquette et design du site (HTML/CSS) : Charles Cascio
2. MCD et gestion des places : Bilo Boury
3. Implémentation de la base MySQL et PHP : Nicolas Roelandt
4. Implémentation PHP, gestion des sessions : Nassim Yousfi

### 2.2 maquette préliminaire

## 2.3 Modèle Conceptuel de Données

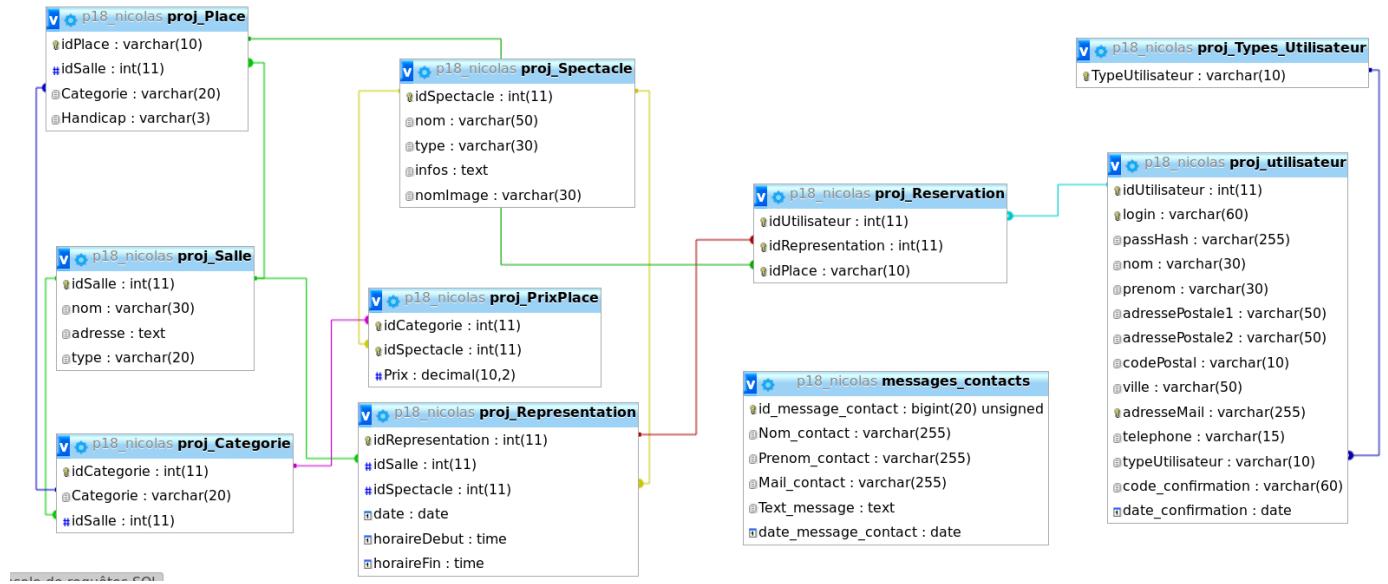


Figure 1. Modèle Conceptuel de Données

## 3 Réalisation de la base de données

La base de données a été implémentée avec MySQL, car cette implémentation est courante pour des sites web. Il est donc aisément de trouver de la documentation à ce sujet ainsi que des packs logiciels fournissant la base de données et le serveur web pour un usage local.

Enfin, il s'agit de la base de données installée sur le serveur Handiman. # Design du site {#design-du-site}

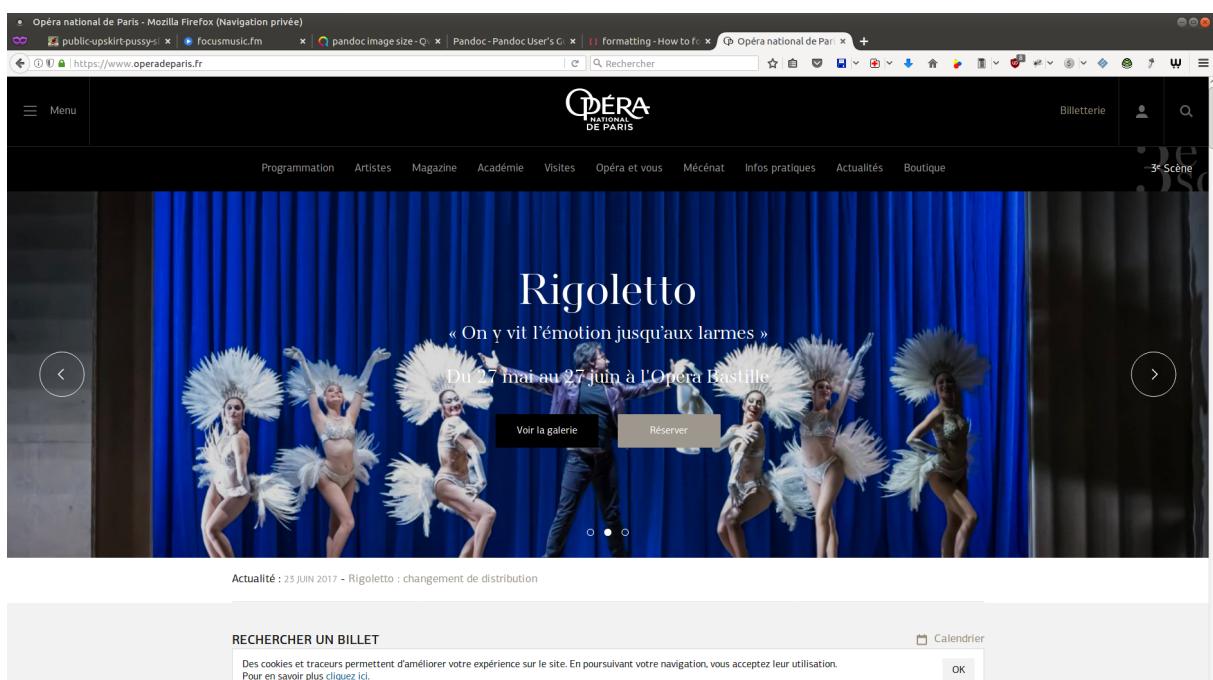
### 3.1 Design général

Si le site officiel<sup>1</sup> est superbe, nous ne nous en sommes pas servi pour développer le design de notre site.

Comme le montre les figures 3 et 4, le design a beaucoup évolué entre les différentes versions.

1. <https://www.operadeparis.fr>

## Elaboration d'un site web accessible Opéra national de Paris



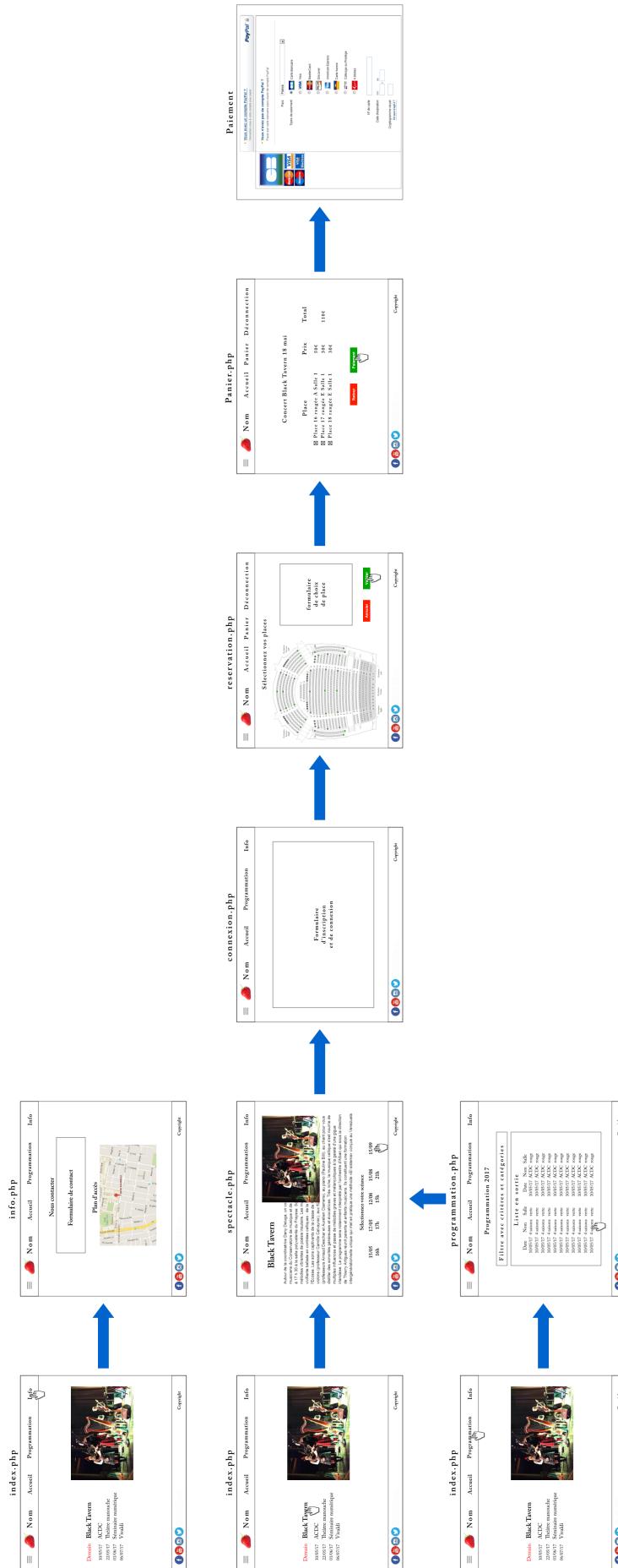
**Figure 2.** Site officiel de l'Opéra national de Paris



# Elaboration d'un site web accessible

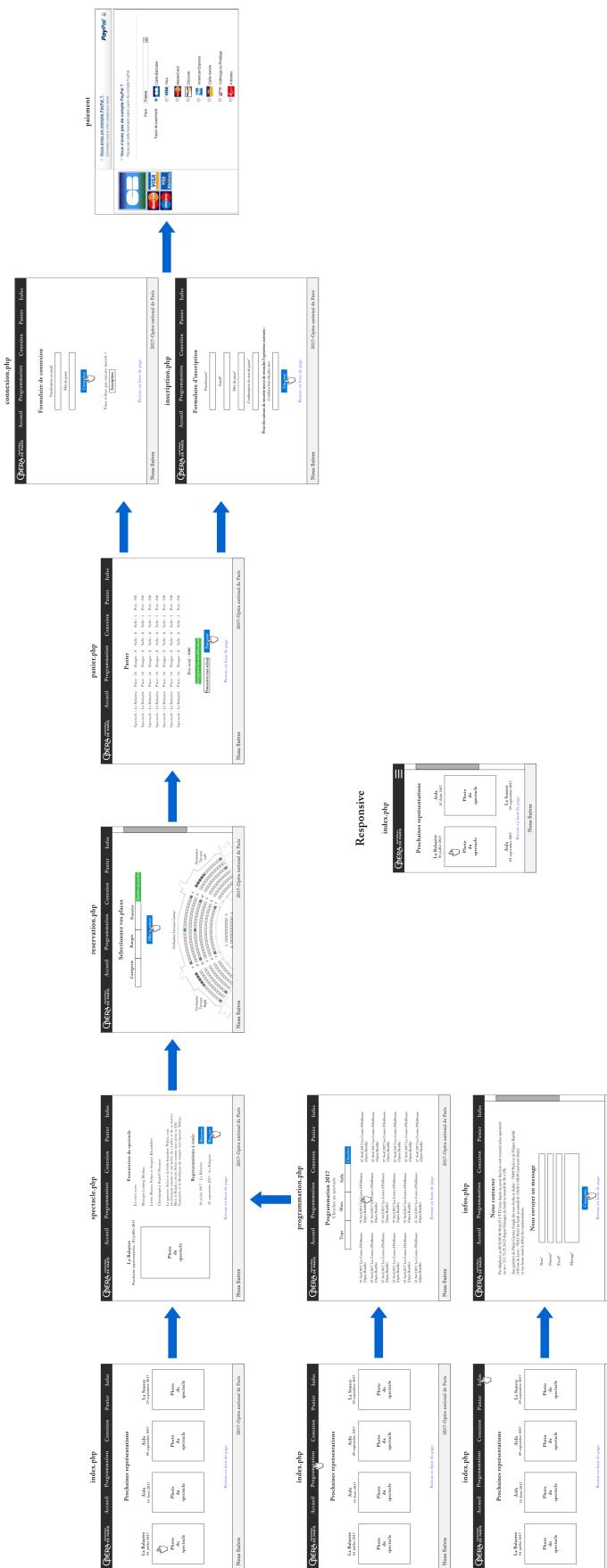
## Opéra national de Paris

---



**Figure 3.** Première ébauche du design

# Elaboration d'un site web accessible Opéra national de Paris



**Figure 4.** Design actuel

### 3.2 Accessibilité

## 4 PHP

De façon général, nous avons essayé de conserver un maximum d'accessibilité pour notre site. Ainsi, nous avons volontairement favoriser l'usage de *PHP* par rapport à *Javascript*. Ceci afin de faciliter l'usage du site avec les lecteurs d'écrans.

De cette manière, la page de réservation de place comporte deux manières de choisir un siège :

- soit par un formulaire ;
- soit en cliquant sur la place voulue de la représentation de la salle.

Cette deuxième option a nécessité l'utilisation de *Javascript* mais peut être aisément contournée.

### 4.1 Gestion des sessions

La variable de session contient deux index :

1. l'index `$_SESSION["auth"]` (authentification), qui contient tout les attributs liés à l'utilisateur inscrit dans la page d'inscription (*register.php*) ainsi que les informations nécessaires à la validation de son compte affectées dans le fichier *confirmation.php*.
2. L'index `$_SESSION["flash"]`, qui contient tous les messages d'erreurs et de succès relatifs à la gestion des formulaires et des redirections.

Pour factoriser l'ouverture de la super variable dans toutes les pages, nous avons effectué cette ouverture dans le fichier *menu.php* qui est présent dans toutes les pages du site, néanmoins nous avons, à cause des nombreuses inclusions de fichiers PHP dont la variable de session est déjà déclarée, du prévenir l'éventualité d'une double ouverture de la session, ce qui engendrerait une erreur.

En utilisant cette instruction,

```
<?php if (session_status() == PHP_SESSION_NONE) session_start();?>
```

, nous vérifions d'abord si la variable de session existe déjà, dans le cas contraire et seulement dans ce cas là, la session est ouverte.

### 4.2 Requêtes en base

## 5 Gestion des réservations

### 5.1 Plan de salle

## 6 Versionnement

### 6.1 Logiciels de gestion du versionnement

#### 6.1.1 GIT

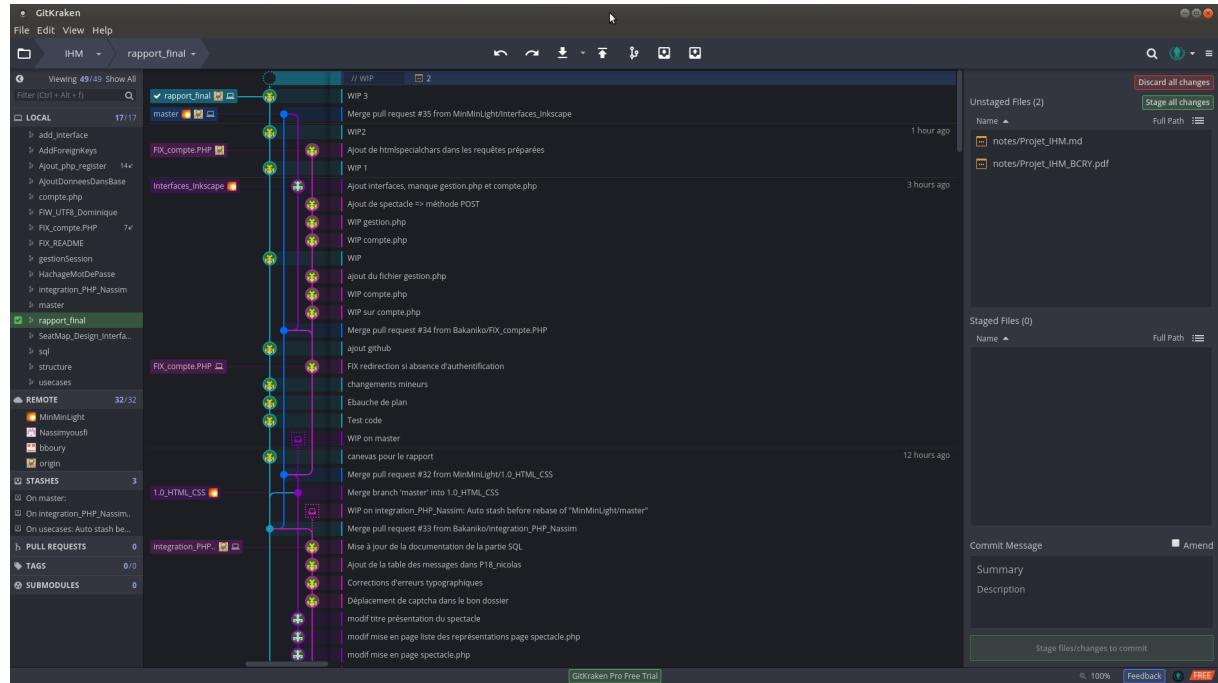
Dans le cadre du développement logiciel, la plupart des entreprises utilisent un logiciel de versionnement. Nous avons nous aussi souhaiter versionner notre travail. Nous avons choisi GIT car il est de plus en plus employé dans le monde professionnel et qu'il est open-source. Cela nous a permis d'acquérir des compétences de gestion de projet et de développement susceptibles d'intéresser un employeur.

De plus, en combinaison avec Github,

Toutefois seu l'un d'entre nous avait déjà une expérience de ce logiciel de versionnement. L'apprentissage n'a pas été aisément pour tous mais nous avons tous progresser dans notre connaissance de ce logiciel. Afin de faciliter cet apprentissage, nous avons eu recours au logiciel **GitKraken** qui propose une interface graphique à git.

### 6.1.2 GitKraken

Pour nous aider à nous retrouver dans les branches et les dépôts, nous avons eu recours au logiciel **GitKraken** édité par la société Axosoft<sup>2</sup>.



**Figure 5.** Interface de GitKraken

Outre une esthétique très travaillée, il permet de réaliser facilement plusieurs opérations courantes sans avoir à recourir à la ligne de commande (COMMIT, PULL, PUSH, ADD REMOTE). Il permet aussi de visualiser les branches des collaborateurs et leur avancement.

## 6.2 Hébergement

L'hébergement s'est fait principalement sur nos machines personnelles, puis le site a été déployé sur Handiman.

La version déployée est visible à cette adresse : <http://handiman.univ-paris8.fr/~nicolas/>  
La base de code est hébergée et visible sur github : <https://github.com/MinMinLight/IHM>.

Nous aurions pu utiliser d'autres plate-formes tel que Bitbucket ou Gitlab / Framagit; mais GitKraken s'intègre mieux avec Github et Bitbucket. De plus, beaucoup de projets libres ont recours à Github, et c'était l'occasion de se familiariser avec son interface.

La création et le lancement d'un projet nous ont semblé plus complexe sur des plate-formes telles que Gitlab / Framagit. Nous avons donc écarté ces solutions.

Avec du recul, il aurait probablement été plus intéressant d'héberger le code sur Bitbucket. En effet, c'est gratuit jusqu'à 5 collaborateurs (nous étions 4) et il est possible de rendre le projet privé. En effet, Github propose un hébergement gratuit pour les projets open-source, . Tout ce que nous publions dessus est donc librement accessible sur Internet. Ce qui fait que nous avons dû faire attention à ne pas publier nos codes d'accès ou nos mails personnels.

## 6.3 avantages

Permet une communication entre les

2. <https://www.gitkraken.com/>

## 6.4 inconvénients

- sécurité

## 6.5 problèmes rencontrés

- envoi de mail, fonctionne sur l'ordinateur de Nassim, ne fonctionne pas sur l'ordinateur de Nicolas ou sur Handiman. La *SendMail* n'est pas installé sur le serveur.

# 7 Evolutions possibles / points non traités

## 7.1 Fonctionnalités / Interface du site

- gestion des achats (panier), paiement de la commande
- renvoi d'un mot de passe temporaire en cas d'oubli
- charger directement l'image depuis l'interface de gestion
- si un administrateur est connecté, remplacer panier par gestion qui renvoie vers la page gestion.php

## 7.2 Page gestion.php

- vérifier les saisies administrateur : format de date et horaires
- pouvoir dupliquer les données d'une représentation ou d'un spectacle pour gagner du temps de saisie
- insérer un message demandant la confirmation de l'action de suppression d'un spectacle ou d'une représentation
- dans l'onglet suppression, trouver un moyen de filtrer les représentations : par spectacle, par salle, par mois, par année. Éventuellement en reprenant le code de la page programmation.php. Il faudra peut-être avoir recours à Javascript et JQuery/Ajax.

## 7.3 Page spectacle.php

- gérer le cas où l'idSpectacle n'existe pas dans la base,
- si toutes les dates du spectacles sont passées, afficher quand même les informations, plus représentations passées (sans bouton réservé)

# 8 Conclusion

## 1 Annexes

### 1.1 Documentation de la base de données

## Contents

<b>Notes sur le SQL</b>	<b>2</b>
proj_TypeUtilisateur . . . . .	2
<b>TypeUtilisateur</b> . . . . .	2
proj_Utilisateur . . . . .	2
<b>IDU</b> . . . . .	2
login . . . . .	3
passHash . . . . .	3
nom . . . . .	3
prenom . . . . .	3
adressePostale1 . . . . .	3
adressePostale2 . . . . .	3
codePostal . . . . .	4
ville . . . . .	4
adresseMail . . . . .	4
telephone . . . . .	4
<i>typeUtilisateur</i> . . . . .	4
proj_Spectacle . . . . .	4
<b>idSpectacle</b> . . . . .	4
nom . . . . .	5
type . . . . .	5
infos . . . . .	5
proj_Salle . . . . .	5
<b>idSalle</b> . . . . .	5
nom . . . . .	5
adresse . . . . .	6
type . . . . .	6
proj_Representation . . . . .	6
<b>idRepresentation</b> . . . . .	6
<i>idSalle</i> . . . . .	6
<i>idSpectacle</i> . . . . .	6
date . . . . .	6
HoraireDebut . . . . .	6
HoraireFin . . . . .	7
proj_Categorie . . . . .	7
<b>idCategorie</b> . . . . .	7
Categorie . . . . .	7
<i>idSalle</i> . . . . .	7
proj_Place . . . . .	7
<b>idPlace</b> . . . . .	7
<i>idSalle</i> . . . . .	7
<i>idCategorie</i> . . . . .	8
proj_PrixPlace . . . . .	8
<i>idCategorie</i> . . . . .	8
<i>idSpectacle</i> . . . . .	8

prix . . . . .	8
proj_Reservation . . . . .	8
<i>IDU</i> . . . . .	8
<i>idRepresentation</i> . . . . .	8
<i>idPlace</i> . . . . .	8
messages_contacts . . . . .	9
id_message_contact . . . . .	9
Nom_contact . . . . .	9
Prenom_contact . . . . .	9
Mail_contact . . . . .	9
Text_message . . . . .	9
date_message_contact . . . . .	9
<b>Disponibilité des places</b>	<b>10</b>
<b>Clés étrangères</b>	<b>10</b>

## Notes sur le SQL

Notes table par table

Toutes les tables commencent par le préfix “proj\_”

### **proj\_TypeUtilisateur**

#### **TypeUtilisateur**

Défini les types d'utilisateurs possibles:

- admin : administre le site et la base de données
- manager : peut gérer les spectacles et les réservations mais pas le site et la base
- user : client du site, peut créer un compte et faire des réservations

Un utilisateur du site doit obligatoirement avoir un TypeUtilisateur, seul un admin peut changer le type d'un utilisateur.

varchar 10

### **proj\_Utilisateur**

#### **IDU**

Identifiant Unique utilisateur, **clé primaire**

type numérique auto-incrémenté

**login**

Identifiant de l'utilisateur (pseudo, adresse mail)

Varchar 30, **UNIQUE**

Il ne doit pas y avoir 2 login identiques dans la base

**passHash**

Mot de passe haché par la fonction password\_hash() de PHP; doit être utilisé directement par password\_verify().

varchar 255 NOT NULL

**nom**

Nom de l'utilisateur

Varchar 30

**prenom**

Prenom de l'utilisateur

Varchar 30

**adressePostale1**

Première ligne de l'adresse postale

Varchar 50

**adressePostale2**

Deuxième ligne de l'adresse postale

Varchar 50

**codePostal**

Code postal : traité comme string pour les client étrangers  
Varchar 10

**ville**

Ville de résidence  
Varchar 15

**adresseMail**

email de contact, *devra correspondre à un certain format unique*  
varchar 50

**telephone**

telephone de contact, traité comme string pour les numéros étrangers, *devra correspondre à un certain format*  
varchar 15

***typeUtilisateur***

référence le type d'utilisateur  
Par défaut, réglé sur *user*, devra être changé ultérieurement par un admin  
Clé étrangère sur **proj\_TypeUtilisateur.TypeUtilisateur**  
varchar 10

**proj\_Spectacle**

Référence tous les spectacles proposés (faits, en cours, à venir)

**idSpectacle**

Identifiant unique auto-incrémenté, **clé primaire**  
Int

**nom**

Nom du spectacle

varchar 50

(unique ?)

**type**

Référence le type de spectacle: concert, pièce de théâtre, danse ...

Suggestion: Les valeurs devraient venir d'une table externe

varchar 30

**infos**

contient les informations du spectacle:

- nom du metteur en scène / chorégraphe
- noms des acteurs/musiciens
- description courte

suggestion : peut être divisé en plusieurs champs dans un vrai site

TEXT

**proj\_Salle**

Une salle est composée de places et appartient à un lieu

**idSalle**

Identifiant unique

Numérique INT auto-incrémenté **clé primaire**

**nom**

nom de la salle (salle n°3; Théâtre Odéon; etc)

varchar 30

**adresse**

adresse postale

TEXT

**type**

type de salle: salle de concert, théâtre, piste de cirque...

varchar 20

**proj\_Representation**

Une représentation est identifiée par un spectacle (auteur, acteurs, etc), une salle, un horaire et une date. Un spectacle peut avoir plusieurs représentations dans la même salle.

**idRepresentation**

Identifiant unique

Numérique INT auto-incrémenté **clé primaire**

***idSalle***

proj\_Salle.idSalle, **clé étrangère**

***idSpectacle***

proj\_Spectacle.idSpectacle, **clé étrangère**

**date**

date de la représentation, séparée du temps pour être plus simple à dupliquer

DATE

**HoraireDebut**

Début de la représentation

TIME

### **HoraireFin**

Fin de la représentation

TIME

### **proj\_Categorie**

Catégories disponibles pour une salle

#### **idCategorie**

Identifiant unique

Numérique INT auto-incrémenté **clé primaire**

#### **Categorie**

Nom de la categorie

varchar 20

#### ***idSalle***

Les salles ont certaines catégories mais pas toutes. Il faut séparer les catégories de chaque salle, car la même catégorie n'aura pas le même prix dans un théâtre que dans un opéra.

proj\_Salle.idSalle, **clé étrangère**

### **proj\_Place**

#### **idPlace**

Identifiant unique

Numérique INT auto-incrémenté **clé primaire**

#### ***idSalle***

Une place appartient toujours à une salle

proj\_Salle.idSalle, **clé étrangère**

*idCategorie*

proj\_Categorie.idCategorie, **clé étrangère**

**proj\_PrixPlace**

Le prix est fonction du spectacle, peut importe la représentation, et de la catégorie. La catégorie permet de remonter à la salle.

*idCategorie*

proj\_Categorie.idCategorie, **clé étrangère; clé primaire**

*idSpectacle*

proj\_Spectacle.idSpectacle, **clé étrangère; clé primaire**

**prix**

Prix en euros

Décimal

**proj\_Reservation**

Une réservation prend en compte 1 et 1 seule place, pour une unique représentation, par un unique utilisateur /client.

Avec le couple idPlace et idRepresentation, il est possible de retrouver le prix

***IDU***

proj\_Utilisateur.IDU, **clé étrangère; clé primaire**

*idRepresentation*

proj\_Representation.idRepresentation, **clé étrangère; clé primaire**

*idPlace*

proj\_Place.idPlace, **clé étrangère; clé primaire**

## **messages\_contacts**

Cette table est destinée à contenir les messages envoyés à travers la page infos.phpMyAdmin

### **id\_message\_contact**

Id unique du message, généré à l'insertion  
type *SERIAL*

### **Nom\_contact**

Nom de l'expéditeur  
*varchar 30*

### **Prenom\_contact**

Prenom de l'expéditeur  
*varchar 255*

### **Mail\_contact**

email de l'expéditeur  
*varchar 255*

### **Text\_message**

Texte du message  
type *TEXT*

### **date\_message\_contact**

Date d'enregistrement du message  
type *date*

## Disponibilité des places

Pas besoin de créer une liste spéciale:

Il suffit de trier les places pour le spectacle concerné où il y a (ou pas) de réservations de faite.

## Clés étrangères

```
proj_Utilisateur(typeUtilisateur) -> proj_TypeUtilisateur(typeUtilisateur)
proj_Representation(idSalle) -> proj_Salle(idSalle)
proj_Representation(idSpectacle) -> proj_Spectacle(idSpectacle)
proj_Categorie(idSalle) -> proj_Salle(idSalle)
proj_Place(idSalle) -> proj_Salle(idSalle)
proj_Place(idCategorie) -> proj_Categorie(idCategorie)
proj_PrixPlace(idCategorie) -> proj_Categorie(idCategorie)
proj_PrixPlace(idSpectacle) -> proj_Spectacle(idSpectacle)
proj_Reservation(IDU) -> proj_Utilisateur(IDU)
proj_Reservation(idRepresentation) -> proj_Representation(idRepresentation)
proj_Reservation(idPlace) -> proj_Place(idPlace)
```

## Liste des figures

1	Modèle Conceptuel de Données . . . . .	5
2	Site officiel de l'Opéra national de Paris . . . . .	6
3	Première ébauche du design . . . . .	8
4	Design actuel . . . . .	9
5	Interface de GitKraken . . . . .	11

## Liste des tables