Documentation technique

*[Attirez l’attention du lecteur avec une citation du document ou utilisez cet espace pour mettre en valeur un point clé. Pour placer cette zone de texte n’importe où sur la page, faites-la simplement glisser.]*

*CourtSquash*

Mai 2019

Auteur : Burnand Loïc  
Version 1.0

Table des matières

[1. Analyse préliminaire 4](#_Toc9518471)

[1.1 Introduction 4](#_Toc9518472)

[1.2 Organisation 4](#_Toc9518473)

[1.3 Objectifs 4](#_Toc9518474)

[2. Maquettage 4](#_Toc9518475)

[2.1 Introduction 4](#_Toc9518476)

[2.2 Barre de menu 4](#_Toc9518477)

[2.3 Login 5](#_Toc9518478)

[2.4 Inscription 5](#_Toc9518479)

[2.5 Accueil (Admin + User) 6](#_Toc9518480)

[2.5.1 Réservation (Admin + User) 7](#_Toc9518481)

[2.6 Profils d’un utilisateur (Admin + User) 8](#_Toc9518482)

[2.7 Vue de tous les utilisateurs (Admin) 9](#_Toc9518483)

[2.8 Paramètres administrateurs (Admin) 10](#_Toc9518484)

[3. Modèle de données 10](#_Toc9518485)

[3.1 Users 10](#_Toc9518486)

[3.2 Tokens 11](#_Toc9518487)

[3.3 Reservations 12](#_Toc9518488)

[3.4 Role 12](#_Toc9518489)

[3.5 Courts 12](#_Toc9518490)

[3.6 Preferences 12](#_Toc9518491)

[4. Analyse organique 14](#_Toc9518492)

[4.1 Objets conteneurs 14](#_Toc9518493)

[EUser 14](#_Toc9518494)

[EResevation 15](#_Toc9518495)

[ECourt 15](#_Toc9518496)

[ERole 16](#_Toc9518497)

[EPreferences 17](#_Toc9518498)

[EEmail 17](#_Toc9518499)

[EToken 18](#_Toc9518500)

[4.2 Les managers 19](#_Toc9518501)

[ModelUsers 19](#_Toc9518502)

[ModelRoles 20](#_Toc9518503)

[ModelCourts 21](#_Toc9518504)

[ModelReservations 21](#_Toc9518505)

[ModelPreferences 22](#_Toc9518506)

[ModelTokens 22](#_Toc9518507)

[ESession 23](#_Toc9518508)

[ModelEmailSender 23](#_Toc9518509)

[5. Librairies externe 24](#_Toc9518510)

[5.1 Bootstrap 24](#_Toc9518511)

[5.2 JQuery 24](#_Toc9518512)

[5.3 Swiftmailer5 24](#_Toc9518513)

[5.4 Dhtmlx 24](#_Toc9518514)

[5.5 FullCalendar 24](#_Toc9518515)

[Tests 25](#_Toc9518516)

[Conclusion 29](#_Toc9518517)

[Améliorations possible 29](#_Toc9518518)

[Points bloquants 29](#_Toc9518519)

[Bilan Personnel 29](#_Toc9518520)

[Planning 30](#_Toc9518521)

[Planning Prévisionnel 30](#_Toc9518522)

[Planning réel 30](#_Toc9518523)

# Analyse préliminaire

## 1.1 Introduction

Le site de réservation de courts de squash permet aux membres de réserver un court de squash. Pour pouvoir s’enregistrer, la personne doit spécifier : une adresse email valide, un mot de passe et un « nickname ». La validation de l’enregistrement se fait à l’aide d’un email d’activation envoyé à l’adresse email utilisée lors de l’enregistrement. L’administrateur du site est également informé par email de l’inscription. Un utilisateur peut compléter/modifier ses informations personnelles : Nickname, Nom, Prénom, Téléphone de contact, Description et Image du profil. L’accès à l’application est régi selon deux niveaux d’habilitation : Membre et Administrateur

Le code respecte un design MVC. Les technologies utilisées sont : PHP, HTML 5, CSS 3, Javascript.

## 1.2 Organisation

Analyse 16h  
Implémentation 40h  
Test 16h  
Documentations 18h

## 1.3 Objectifs

Réaliser un site de réservation de courts Squash.

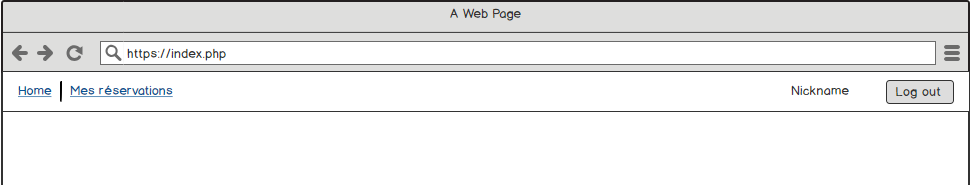
# Maquettage

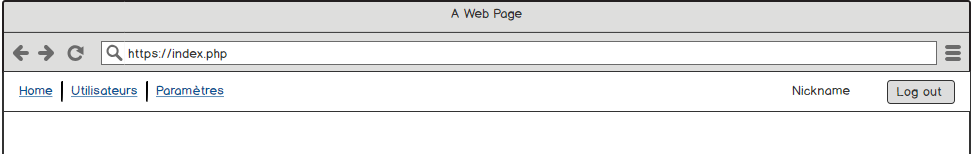
## Introduction

Dans cette section vous trouverez la description des fonctionnalités de l’application au travers de maquettes d’écran

## Barre de menu

La barre de menu s’affiche en fonction du profil de l’utilisateurs

Pour les membres : 

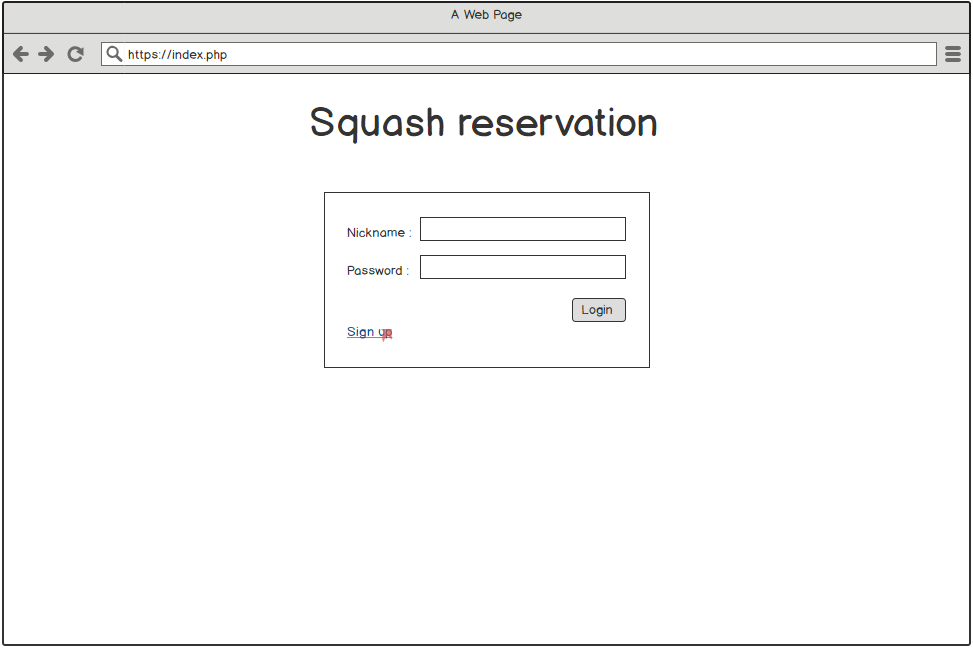
Pour les administrateurs : 

## Login

En arrivant dans le site, l’utilisateur étant non connecté, il est redirigé directement sur la page de login, tel ci-dessous

La page principale du site présente un formulaire de connexion du fait que le site est accessible uniquement aux utilisateurs enregistrés et authentifiés.

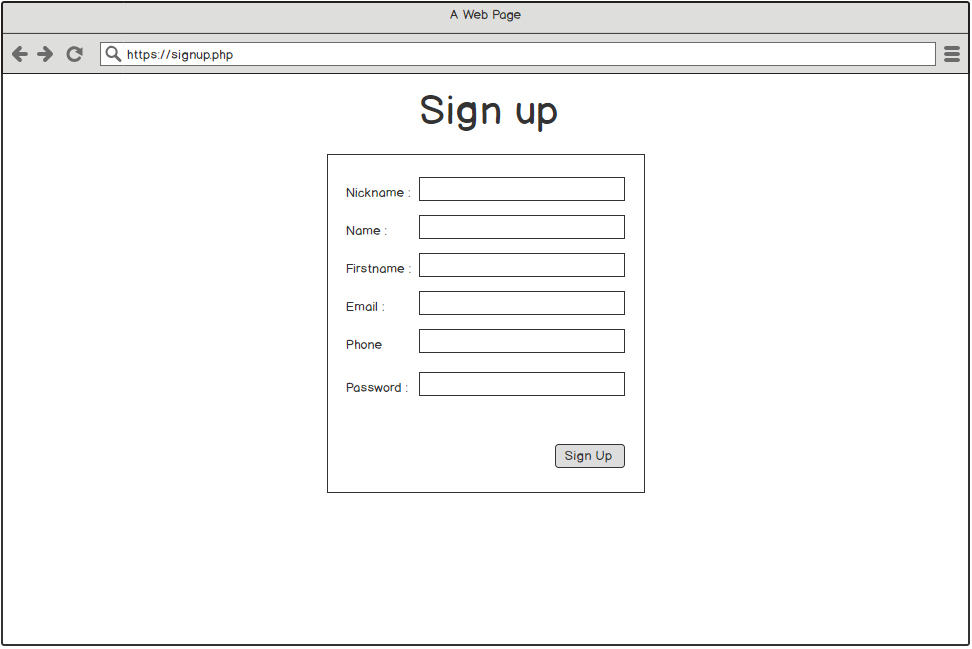
Si l’utilisateur n’est pas encore enregistré, il peut le faire depuis le lien Sign up.



Pour s’inscrire, l’utilisateur peut le faire en appuyant sur le lien Sign Up.

## Inscription

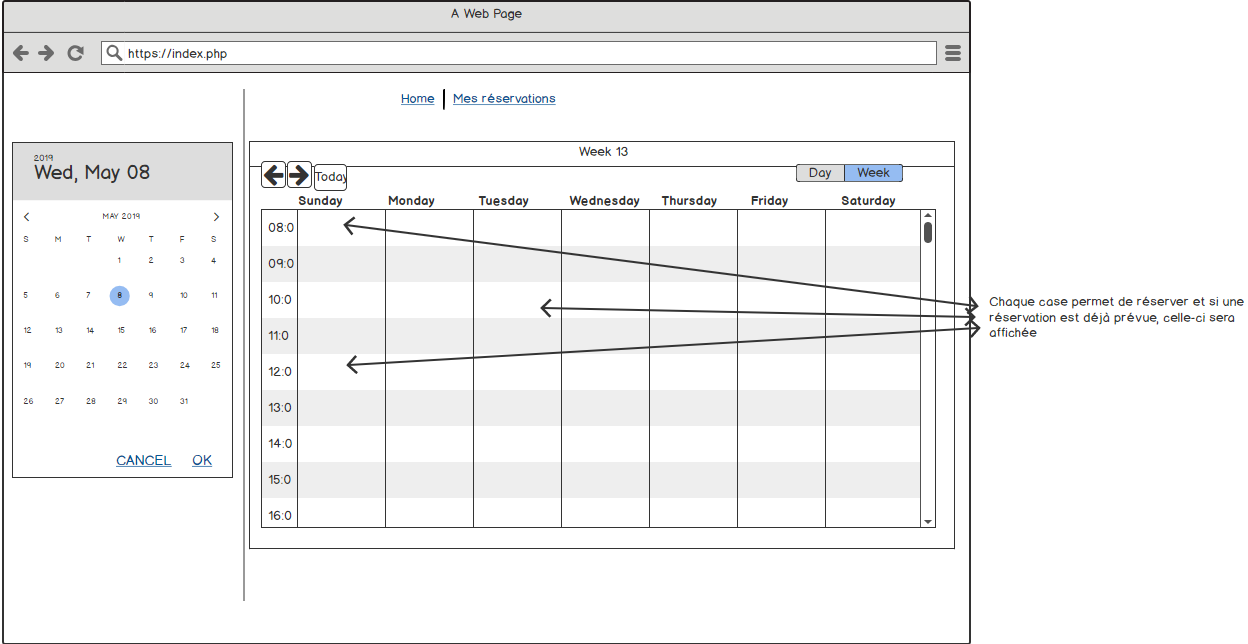
Si l’utilisateur n’est pas encore inscrit, il peut s’inscrire grâce à cette page :



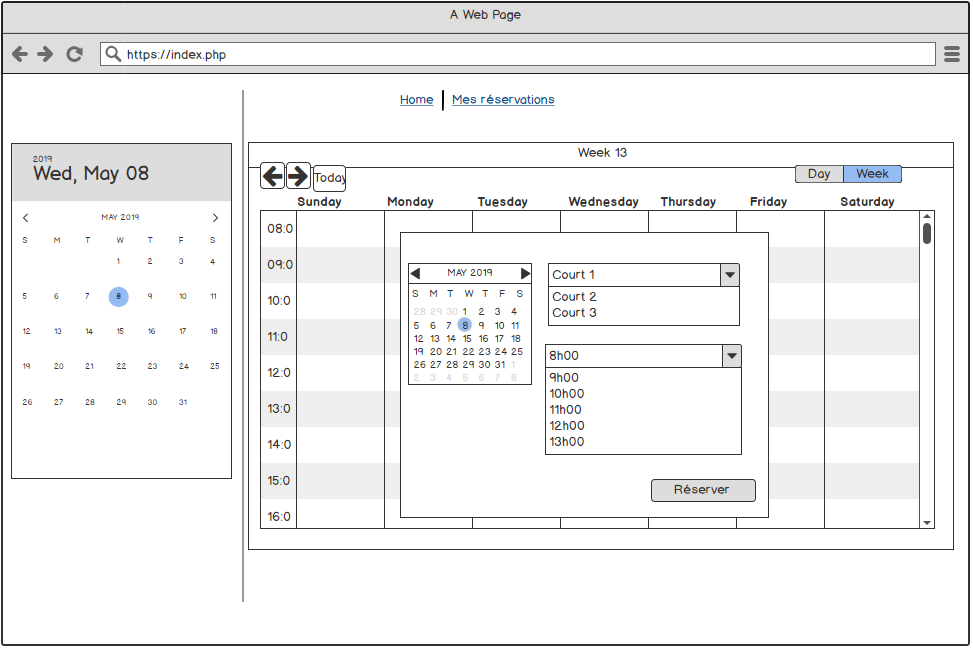
Attention, pour valider son inscription, l’utilisateur doit confirmer son compte grâce à l’email qui aura été envoyé à l’adresse indiqué dans le champ « email ». Après avoir cliqué sur le bouton pour s’inscrire, l’utilisateur est redirigé vers la page ci-dessous pour qu’il aille confirmer son compte par email

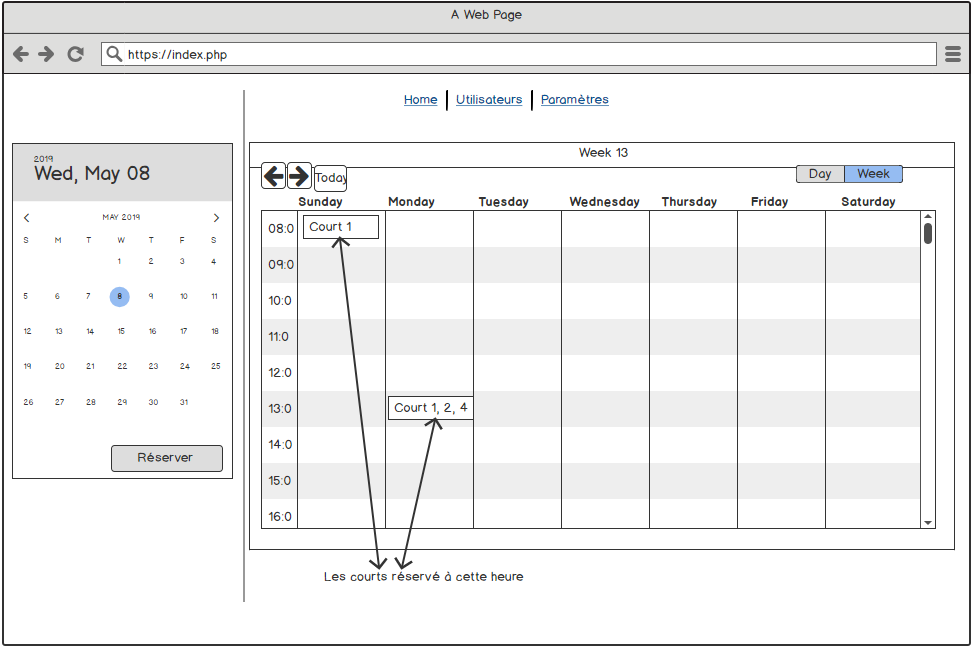
## Accueil (Admin + User)

En se connectant, un utilisateur arrivera sur la page suivante :

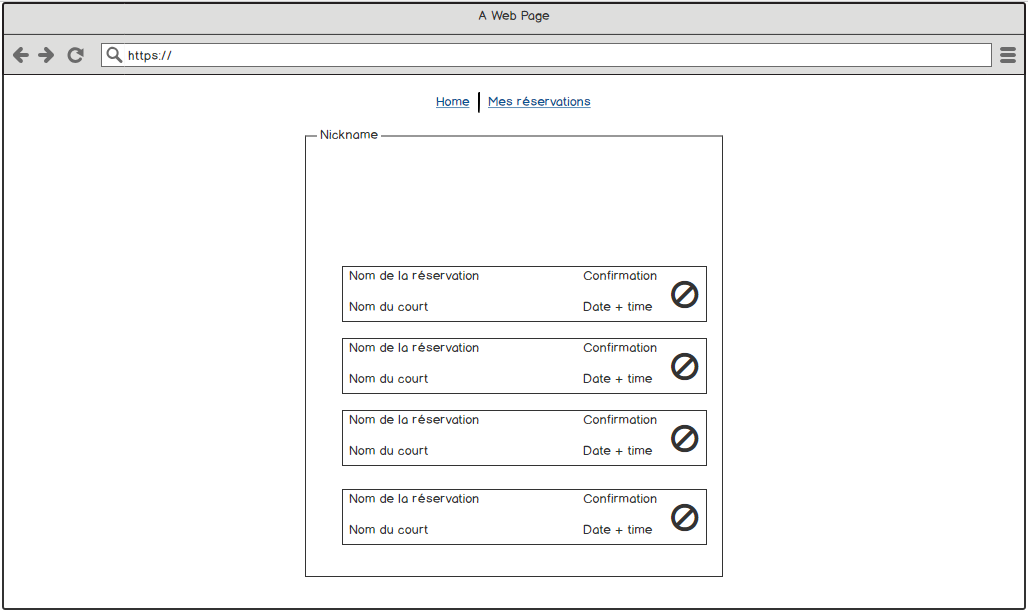
La page présente les réservations des différents courts pour la semaine courante. L’utilisateur peut sélectionner une semaine spécifique en utilisant les boutons |<|>|. Il peut également réserver un court pour un jour et une heure donnée

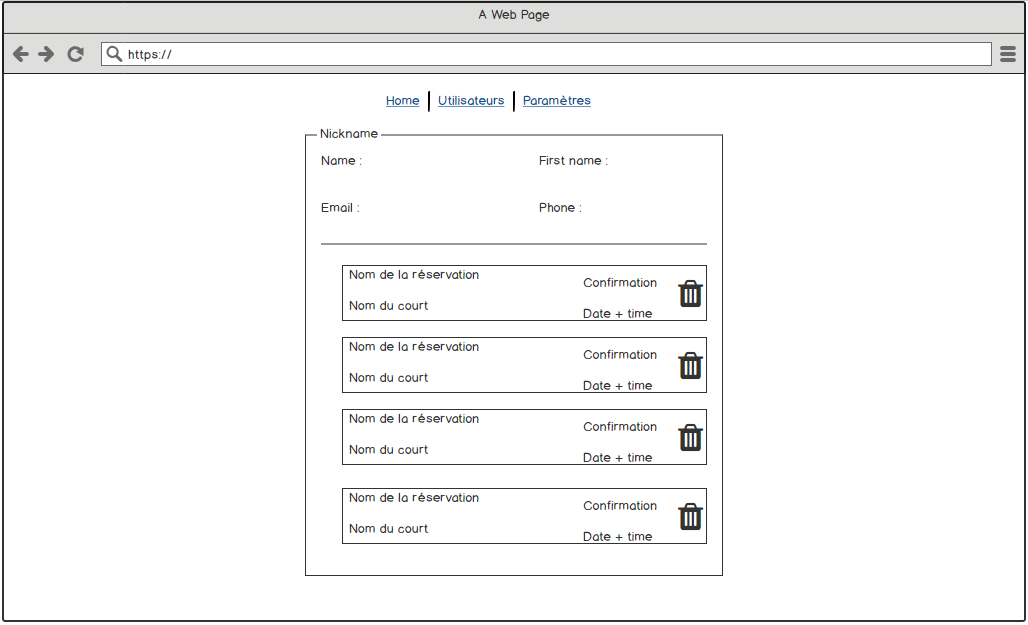
## Réservation (Admin + User)

Quand on clique sur une case disponible, une pop-up s’ouvre et nous permet de faire une réservation (La pop-up permet de changer la date et l’heure mais par défaut ces informations seront mis selon les données de la case sur laquelle on a cliqué) 

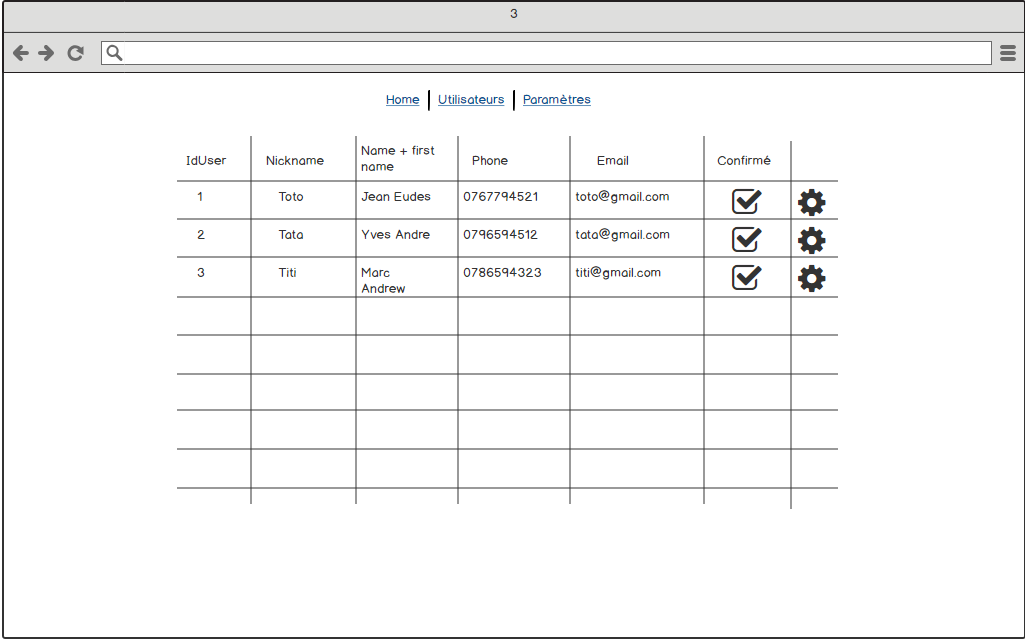
Les réservations sont représentées au sein du calendrier pour l’heure donnée et afficher le nom du court 

## Profils d’un utilisateur (Admin + User)

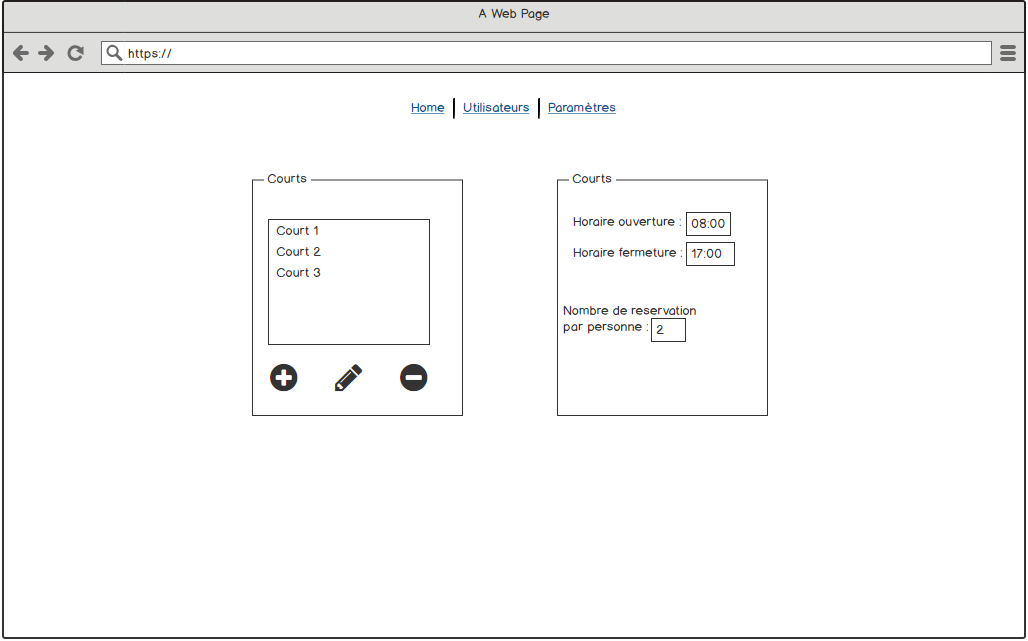
Pour annuler une réservation, il doit cliquer sur l’icône à côté de la réservation. Si la réservation est dans moins de 24h, l’icône d’annulation n’est pas affichée.

Cette page est aussi utilisée par les administrateurs qui, en allant sur un profil, peuvent supprimer les réservations. Il n’y a pas de restriction de temps pour la suppression d’une réservation pour l’administrateur

## Vue de tous les utilisateurs (Admin)

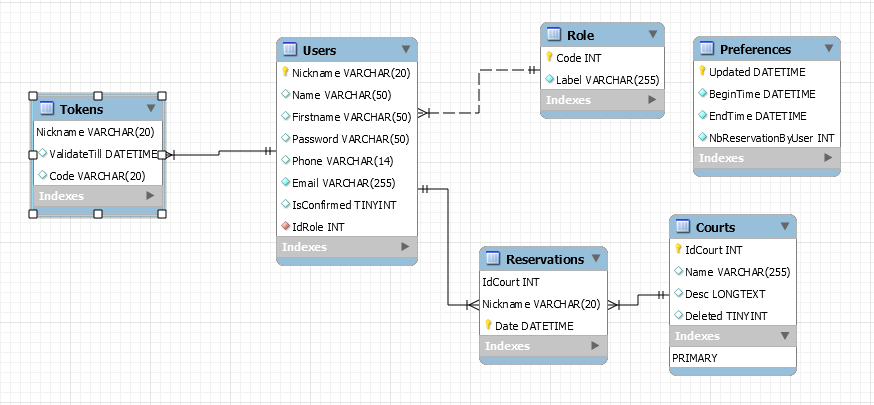
Disponible seulement en ayant les droits administrateurs, cette page permet de voir tous les utilisateurs. Il permet également de confirmer l’inscription et si l’on veut le supprimer la confirmation. Cela revient à pour bloquer l’utilisateur. Enfin depuis cette page il peut accéder à n’importe quels profils et ainsi pouvoir supprimer les réservations. 

## Paramètres administrateurs (Admin)

Cette page est accessible seulement aux administrateurs. Elle permet de gérer les différentes options du système. 

Il est possible de créer, modifier ou supprimer des courts du club de squash. Attention les courts

# Modèle de données



## Users

Cette table contient les utilisateurs du site

|  |  |  |  |
| --- | --- | --- | --- |
| Nom | Type | Clés | Description |
| Nickname | VARCHAR(20) | PK | Nom unique servant de clé primaire |
| Name | VARCHAR(50) | NN | Nom de l’utilisateur |
| Firstname | VARCHAR(50) | NN | Prénom de l’utilisateur |
| Password | VARCHAR(50) | NN | Le password encodé en sha1 |
| Phone | VARCHAR(14) |  | Le numéro de téléphone de l’utilisateur |
| Email | VARCHAR(255) | NN | L’email permettant d’envoyer des emails à l’utilisateur |
| IsConfirmed | TINYINT | Default : 0 | Le booléan pour savoir si l’utilisateur peut se connecter |
| IdRole | INT | FK : Role.IdRole | Permet de joindre la table Role |

## Tokens

Cette table contient les tokens permettant de valider l’inscription d’un utilisateur

|  |  |  |  |
| --- | --- | --- | --- |
| Nickname | VARCHAR(20) | PK, FK | Nom unique servant de clé primaire et étrangère |
| ValidateTill | DATETIME | NN | La date d’éxpiration du token |
| Code | VARCHAR(20) | NN | Chaine de cractère aélatoire permettant de validé le token |

## Reservations

Cette table contient les clés étrangères pour la liaison entre la table Courts et la table Users

|  |  |  |  |
| --- | --- | --- | --- |
| IdCourt | INT | PK, FK | Id du court pour la liaison avec la table Courts |
| Nickname | VARCHAR(20) | PK,FK | Nom de l’utilisateur servant de liaison à la table Users |
| Date | DATETIME | PK | Date de la réservation |

## Role

Cette table contient les rôles

|  |  |  |  |
| --- | --- | --- | --- |
| Code | INT | PK | Code pour identifier le code |
| Label | VARCHAR(255) | NN | Nom du role |

## Courts

Cette table contient Les différents courts disponible

|  |  |  |  |
| --- | --- | --- | --- |
| Nom | Type | Clés | Description |
| IdCourts | INT | PK, AI | Id servant de clé primaire au Court |
| Name | VARCHAR(50) | NN | Nom du Court |
| Desc | VARCHAR(255) | NN | Description du court |
| Deleted | TINYINT | NN, Défaut : 0 | Booléan servant à la supression du court |

## Preferences

Cette table ne contient qu’un seul enregistrement qui ne fait qu’être updater.

|  |  |  |  |
| --- | --- | --- | --- |
| Nom | Type | Clés | Description |
| Updated | DATETIME | PK | Dernière fois modifié, sert de clé primaire |
| Name | VARCHAR(50) | NN | Nom du Court |
| Desc | VARCHAR(255) | NN | Description du court |
| Deleted | TINYINT | NN, Défaut : 0 | Booléan servant à la supression du court |

# Analyse organique

## Objets conteneurs

J’utilise des classes conteneurs pour interfacer les données de la base de données

### EUser

#### Exemple d’utilisation

Cet objet est utilisé comme conteneur

//Exemple d'utilisation 1

$u = new EUser();

$u->Email = "loic@burnand.com";

$u->Nickname = "Test";

$u->Name = "BURNAND";

$u->FirstName = "Loïc";

$u->Phone = "0767756644";

$u->Role = new ERole();

$u->IsConfirmed = false;

//Exemple d'utilisation 2

$u = new EUser("loic@burnand.com", "Test", "BURNAND", "Loïc", "0767756644", new ERole(), false);

#### Propriété

##### Email

String, L'email de l'utilisateur

##### Nickname

String, Le pseudo de l'utilisateur

##### Name

String, Le nom de l'utilisateur

##### FirstName

String, Le prénom de l'utilisateur

##### Phone

String, le numéro de téléphone de l’utilisateur

##### IdRole

Int, l'id du rôle de l'utilisateur

##### IsConfirmed

Bool, Confirmation de l’inscription de l’utilisateur

#### Méthode

##### \_\_construct(string InEmail, string InNickname, string InName, string InFirstName, string InPhone, ERole InRole, Bool InCofirmation)

Le Constructor appelé au moment de la création de l'objet. new EUser();

@param InEmail L'email de l'utilisateur. (Optionel) Defaut ""

@param InNickname Le nickname de l'utilisateur. (Optionel) Defaut ""

@param InName Le nom de l'utilisateur. (Optionel) Defaut ""

@param InFirstName Le prénom de l'utilisateur. (Optionel) Defaut ""

@param InPhone Le numéro de téléphone de l'utilisateur. (Optionel) Defaut ""

@param InRole Le rôle de l'utilisateur. (Optionel) Defaut null

@param InCofirmation Est-ce que l'utilisateur est confirmé. (Optionel) Defaut false

### EResevation

#### Exemple d’utilisation

Cet objet est utilisé comme conteneur

//Exemple d'utilisation 1

$u = new EResevation();

$u->Court = new ECourt();

$u->User = new EUser();

$u->IsConfirmed = false;

$u->Date = "";

//Exemple d'utilisation 2

$u = new EResevation(new ECourt(), new EUser(), false, "");

#### Propriété

##### Court

ECourt, le terrain qui a été réservé

##### User

EUser, l’utilisateur qui a réservé

##### Date

DateTime, la date et l’heure du début de la réservation

#### Méthode

##### \_\_construct(ECourt InCourt, EUser InUser, bool InConfirmed, DateTime InDate)

Le Constructor appelé au moment de la création de l'objet. new EResevation();

@param InCourt Le terrain réservé. (Optionel) Defaut null

@param InUser L'utilisateur qui a réservé. (Optionel) Defaut null

@param InConfirmed La confirmation de la réservation. (Optionel) Defaut false

@param InDate La date et l'heure de la réservation. (Optionel) Defaut ""

### ECourt

#### Exemple d’utilisation

Cet objet est utilisé comme conteneur

//Exemple d'utilisation 1

$u = new ECourt();

$u->Name = "Court Test";

$u->Desc = "Ce terrain est un terrain reserver au entrainement";

$u->Deleted = true;

//Exemple d'utilisation 2

$u = new ECourt("Court Test", "Ce terrain est un terrain reserver au entrainement", true);

#### Propriété

##### Id

Int, L’Id du court

##### Name

String, Le nom du court

##### Desc

String, La description du court

##### Deleted

Bool, est-ce que le court est fermé ou pas

#### Méthode

##### \_\_construct(string InName, string InDesc, bool InDeleted)

Le Constructor appelé au moment de la création de l'objet. new ECourt();

@param InEmail L'email du Court. (Optionel) Defaut ""

@param InDesc La description du court. (Optionel) Defaut ""

@param InDeleted Est-ce que le court est fermé ou pas. (Optionel) Defaut false

### ERole

#### Exemple d’utilisation

Cet objet est utilisé comme conteneur

//Exemple d'utilisation 1

$u = new ERole();

$u->IdRole = 1;

$u->Name = "Admin";

//Exemple d'utilisation 2

$u = new ERole(1, "Admin");

#### Propriété

##### IdRole

Int, L’id du rôle

##### Name

String, le nom du rôle

#### Méthode

##### \_\_construct(Int InIdRole, String InName)

Le Constructor appelé au moment de la création de l'objet. new ERole();

@param InIdRole L'Id du rôle

@param InName Le nom du rôle. (Optionel) Defaut ""

### EPreferences

#### Exemple d’utilisation

Cet objet est utilisé comme conteneur

//Exemple d'utilisation 1

$u = new EPreferences();

$u->Updated = new DateTime('2011-01-01T15:03');

$u->BeginTime = new DateTime('2019-01-01T15:00');

$u->EndTime = new DateTime('2019-01-01T16:00');

$u->NbReservation = 2;

#### Propriétés

##### Updated

DateTime, Date et heure de la dernière update

##### BeginTime

DateTime, L'heure d'ouverture des courts

##### EndTime

DateTime l'heure de fermeture des courts

##### NbReservations

Int, Le nombre de réservation que l'utilisateur peut faire par jour

#### Méthode

##### \_\_construct()

### EEmail

#### Exemple d’utilisation

//Exemple d'utilisation 1

$u = new EEmail();

$u->Subject = "Sujet de l'email";

$u->From = "tpiSquash@gmail.com";

$u->To = "test@example.com";

$u->Body = "<p>Corps du message</p>"

//Exemple d'utilisation 2

$u = new ECourt("test@example.com", "Sujet de l'email", "<p>Corps du message</p>", "tpiSquash@gmail.com");

#### Propriétés

Subject

String, Le sujet de l'email

From

String, L'email de l'envoyeur

##### To

String, L'email du destinataire

##### Body

String, Le corps de l'email

##### Méthode

###### \_\_construct(string InTo, string InSubject, string InBody, string InFrom)

Le Constructor appelé au moment de la création de l'objet EEmail();

@param InSubject Le sujet de l'email. (Optionel) Defaut ""

@param InFrom L'email de l'envoyeur. (Optionel) Defaut ""

@param InTo Le destinataire du mail

@param InBody Le corps du message. (Optionel) Defaut ""

### EToken

#### Exemple d’utilisation

//Exemple d'utilisation 1

$u = new EToken();

$u->Nickname = "Test";

$u->ValidateTill = "2000-00-00 00:00:00";

//Exemple d'utilisation 2

$u = new EToken("Test", "2000-00-00 00:00:00");

#### Propriétés

##### Nickname

String, Le nickname du user associé à ce token

##### ValidateTill

String, La date d'éxpiration du token

##### Code

String, Le code du token

## Les managers

Ce sont des classes qui permettent de manipuler les données de la base de données et les objets conteneurs.

### ModelUsers

#### Exemple d’utilisation

Cette classe est static et permet de gérer la table User

//Exemple d'utilisation 1

$users = ModelUsers::GetAllUsers();

//Exemple d'utilisation 2

$u = new EUser();

$u->Email = "loic@burnand.com";

$u->Nickname = "Test";

$u->Name = "BURNAND";

$u->FirstName = "Loïc";

$u->Phone = "0767756644";

$u->Role = new ERole();

$u->IsConfirmed = false;

ModelUsers::AddUser($u);

#### Méthodes

##### GetAllUsers ()

Retourne le tableau des utilisateurs de type EUser

##### GetUserByNickname (String nickname)

Cherche un utilisateur par son Id dans la table. Retourne le premier utilisateur du résultat de la requête

###### Paramètres

Nickname Le nickname sous forme de string a recherché dans la base de donnée

##### AddUser (EUser User)

Ajoute un utilisateur dans la base de donnée.

###### Exemple

$u = new EUser();

$u->Email = "loic@burnand.com";

$u->Nickname = "Test";

$u->Name = "BURNAND";

$u->FirstName = "Loïc";

$u->Phone = "0767756644";

$u->Role = new ERole();

$u->IsConfirmed = false;

ModelUsers::AddUser($u);

##### UpdateUser (EUser User)

Permet de metter à jours les donnée d'un utilisateur. Retourne true si l'update est réussie, sinon retourne false

###### Paramètres

User l'utilisateur que l'on souhaite modifier

##### GetUserRole (EUser User)

Retourne le nom du rôle de l'utilisateur. La recherche se fait d’abord s’il y a, sur l’id du rôle stocké dans le paramètre User. S’il n’y est pas, il va chercher l’id en ayant cherché l’utilisateur grâce à la fonction GetUserByNickname pour ensuite chercher le rôle.

###### Exemple

// Exemple 1

$u = new EUser();

$u->IdRole = "1";

$roleName = ModelUsers::GetUserRole($u);

// Exemple 2

$u = new EUser();

$u->nickname = "Test";

$roleName = ModelUsers::GetUserRole($u);

##### CheckLogin(EUser User)

Retourne true si le password et le nickname sont dans la base et correspondent. Le paramètre User doit contenir le nickname et le password

##### UpdateConfirmation(String nickname, Bool isConfirmed)

Update le champ IsConfirmed d'un utilisateur. nickname De type string, contient le nickname de l'utilisateur à update. isConfirmed De type Boolean, contient la valeur du champ à update. Retourne true si l'update a réussie.

##### ConfirmUser(string nickname)

Update le champ IsConfirmed d'un utilisateur à 1. nickname De type string, contient le nickname de l'utilisateur à update. Retourne true si l'update a réussie.

### ModelRoles

#### Exemple d’utilisation

// Exemple d'utilisation 1

$roles = ModelRoles::GetAllRoles();

// Exemple d'utilisation 2

$role = ModelRoles::GetRolesById(1);

#### Méthodes

##### GetAllRoles ()

Retourne un tableau de ERole contenant tous les rôles stockés en base de donnée

##### GetRolesByCode (Code)

Chercher un role par son code dans la table. Retourne le premier rôle du résultat de la requête.

###### Paramètres

Code Le code a recherché dans la base de donnée

### ModelCourts

#### Exemple d’utilisation

#### Méthodes

##### GetAllCourts()

Retourne le tableau des utilisateurs de type ECourt

##### GetCourtById($Id)

Cherche un court par son Id dans la table

###### Paramètres

Id L'id a recherché dans la base de donnée

##### AddCourt($Court)

Ajoute un court dans la base de donnée. Retourne true si l'ajout est réussi, sinon retourne false

###### Paramètres

Court le court a ajouté sous forme de ECourt

##### UpdateCourt($Court)

Permet de metter à jours les donnée d'un court. Retourne true si l'update est réussie, sinon retourne false

###### Paramètres

Court le court que l'on souhaite modifier

##### DeleteCourt(ECourt Court)

Supprime un Court. Retourne true si la suppression est réussie, sinon retourne false.

###### Paramètres

Court le court que l'on souhaite supprimer

##### IsCourtReference()

Retourne true si le court est lié à une ou plusieurs reservation sinon false.

###### Paramètres

Court le court que l'on souhaite recherché

### ModelReservations

#### Exemple d’utilisation

#### Méthodes

##### GetAllReservations()

Retourne le tableau des réservations de type EReservation.

##### GetReservationsForADate($date)

Retourne un tableau de EReservation qui ont lieu à la date donnée.

###### Paramètres

date La date des réservations que l'on souhaite récupérer

##### GetReservationsByUser($User)

Cherche les réservations pour un utilisateur. Retourne les réservations du résultat de la requête

###### Paramètres

User L'utilisateur (le nickname est obligatoire) a recherché dans la base de donnée

##### GetReservationsByCourt($Court)

Cherche les réservations pour un court. Retourne les réservations du résultat de la requête

###### Paramètres

Court Le court (l'id est obligatoire) a recherché dans la base de donnée

##### DeleteReservation($Reservation)

Supprime une réservation. Retourne true si la suppression est réussie, sinon retourne false

###### Paramètres

Reservation (EReservation) La reservation à supprimer

##### AddReservation($Reservation)

Ajoute une réservation dans la base de donnée. Retourne true si l'ajout est réussi, sinon retourne false

###### Paramètres

Reservation la réservation a ajouté sous forme de EReservation

### ModelPreferences

#### Méthodes

##### GetPreferences()

Retourne un objet EPreferences contenant les préférences.

##### UpdatePreferences(EPreferences preferences)

Permet de metter à jours les préférences. Preferences, Les nouvelles préférences dans un objet EPreference. Retourne true si l'update est réussie, sinon retourne false.

##### GetBeginTime()

Retourne le début des horaires d'ouvertur dans le format "H".

##### GetEndTime()

Retourne la fin des horaires d'ouverture dans le format "H".

### ModelTokens

##### Méthodes

###### GetTokenByNickname(string nickname)

Cherche un Token par son nickname dans la table. Retourne le premier token du résultat de la requête.

Paramètres

Nickname Le nickname a recherché dans la base de donnée

###### GetTokenByCode(string code)

Cherche un Token par son code dans la table. Retourne le premier token du résultat de la requête.

Paramètres

Code Le code a recherché dans la base de donnée

###### AcceptToken(string code)

Accepte un utilisateur grâce à son token. Retourne true si l'acceptation est réussie, sinon retourne false.

Paramètres

Code Le code du token a validé

###### IsTokenValable(string code)

Vérifie si le token est toujours valable. Retourne true si le token est valable sinon false.

Paramètres

Code le code du token a vérifié

###### CreateToken(string Nickname)

Créer un token et appelle une fonction pour l'insérer en bd. Retourne true si la création est réussie, sinon retourne false.

Paramètres

Nickname Le nickname de l'utilisateur possédant le token

###### InsertToken(EToken token)

Ajoute un token dans la base de donnée. Retourne true si l'ajout est réussi, sinon retourne false.

Paramètres

Token le token a ajouté sous forme de EToken

###### generateToken(int len = 20)

Génère le code d’un token aléatoirement. Retourne la chaine de caractère aléatoire.

Paramètres

Len la longueur de la chaine de caractère aléatoire

### ESession

#### Méthodes

##### SetUser(EUser user)

Assigner un objet EUser dans la session.

###### Paramètres

User L’objet EUser a ajouté en session

##### GetUser()

Retourne l'objet EUser stocké en session.

### ModelEmailSender

#### Exemple d’utilisation

$Email->Body = '<html>' .

' <head></head>' .

' <body>'.

' <p>Veuillez ouvrir ce lien pour confirmer votre compte : http://localhost/index.php?action=EmailConfirmation&token=' .

'vDzs59AkoO7n4NA76Ttb</p>' .

' </body>' .

'</html>';

ModelEmailSender::SendEmail($Email);

#### Méthodes

##### SendEmail(EEmail email)

Permet d'envoyer un email grâce à l'api Swiftmailer.

###### Paramètres

Email Un objet EEmail contenant les données pour envoyer un email.

# Librairies externe

## 5.1 Bootstrap

Bootstrap est une collection d'outils utiles à la création du design (graphisme, animation et interactions avec la page dans le navigateur, etc.) de sites et d'applications web. C'est un ensemble qui contient des codes HTML et CSS, des formulaires, boutons, outils de navigation et autres éléments interactifs, ainsi que des extensions JavaScript en option. C'est l'un des projets les plus populaires sur la plate-forme de gestion de développement GitHub.[[1]](#footnote-1)

## 5.2 JQuery

JQuery est une bibliothèque JavaScript libre et multiplateforme créée pour faciliter l'écriture de scripts côté client dans le code HTML des pages web2. La première version est lancée en janvier 2006 par John Resig.

Le but de la bibliothèque étant le parcours et la modification du DOM (y compris le support des sélecteurs CSS 1 à 3 et un support basique de XPath), elle contient de nombreuses fonctionnalités ; notamment des animations, la manipulation des feuilles de style en cascade (accessibilité des classes et attributs), la gestion des évènements, etc. L'utilisation d'Ajax est facilité et de nombreux plugins sont présents.[[2]](#footnote-2)

## 5.3 Swiftmailer5

Swiftmailer est une librairie php qui permet d’envoyer très facilement des emails. Elle est utilisée par le framework Symfony, preuve de sa qualité.

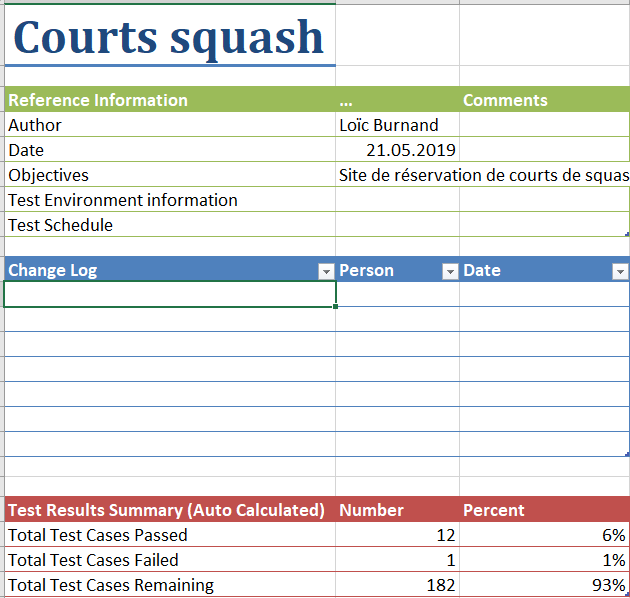
## 5.4 Dhtmlx

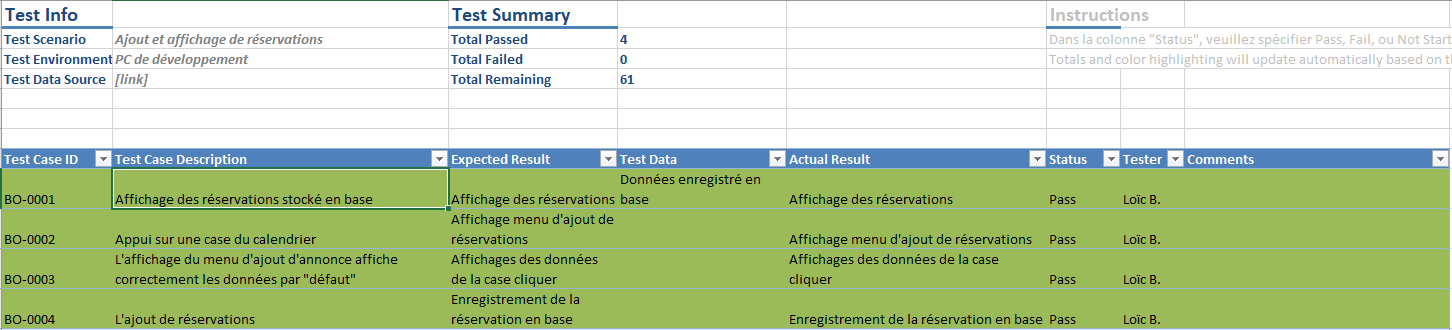
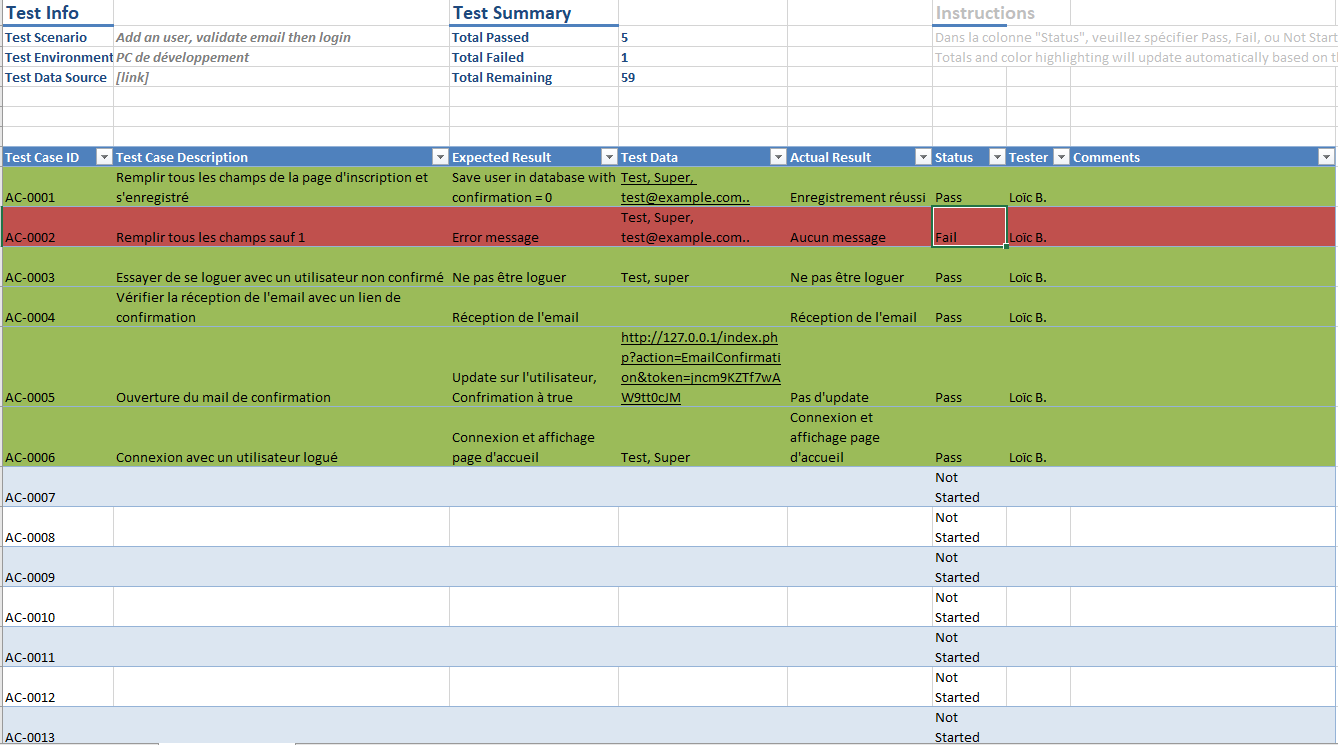
Dhtmlx est une librairire php qui permet d’utiliser facilement des composants tels que des calendriers, dans mon cas.

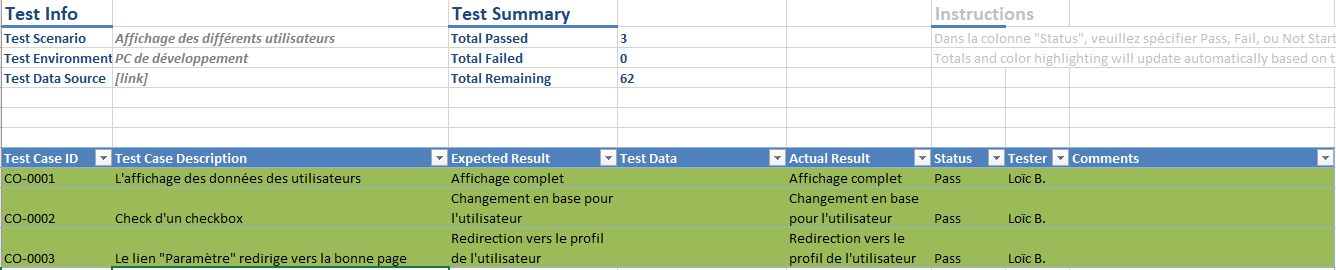
## 5.5 FullCalendar

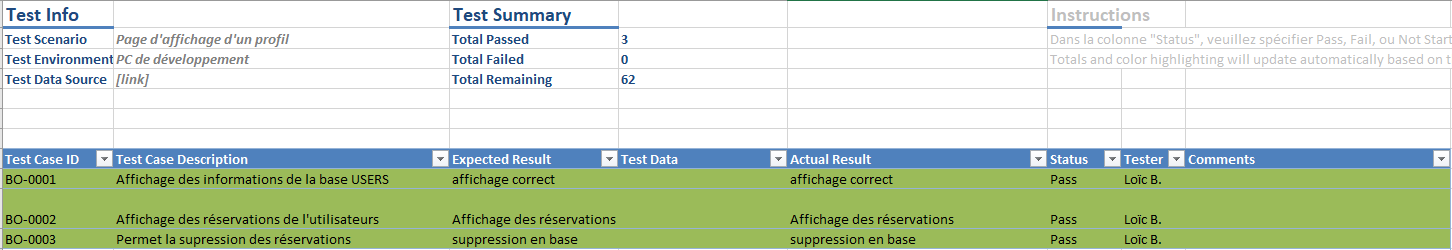
FullCalendar Scheduler est une collection de plugins premium pour FullCalendar.

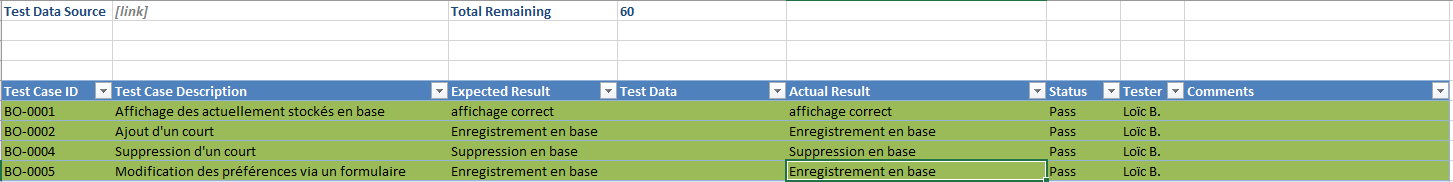
# Tests











# Conclusion

## Améliorations possible

* Ajout de la vision des détails des réservations directement depuis la page accueil
* Suppression des réservations une fois la date passée
* Annuler la possibilité de réserver un court pour une date antérieure à aujourd’hui

## Points bloquants

Pendant ce projets plusieurs points m’ont un peu bloqué et ont ralenti le développement de mon site. Le premier a été l’affichage du calendrier pour la page principale. N’ayant jamais utilisé de calendrier dans le cadre de ma formation, apprendre à utiliser l’api FullCalendar a été un peu bloquant et m’a pris pas mal de temps. Ensuite un deuxième point a été l’envoi d’emails qui était tout d’abord bloquer par le réseau, je n’ai donc été en mesure de résoudre ce problème seulement à la fin quand nous avons été passé sur un autre réseau qui permettait l’envoi d’emails.

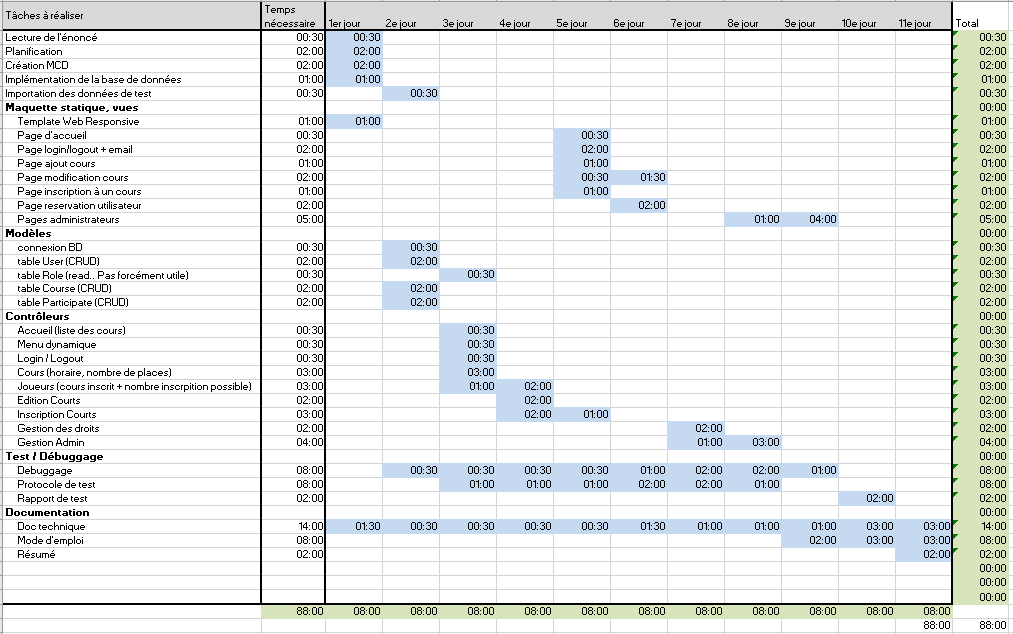
## Bilan Personnel

Ce projet sur trois semaines a été très intéressants a réalisé. Il m’a permis de mettre en œuvre tout ce que j’avais appris ces quatre dernières années et j’ai aimé me rendre compte tout le progrès que j’avais fait depuis que je suis rentré dans cette école. Toutefois celui-ci aura quand même été très éprouvant et m’auras pris énormément d’énergie. Je suis content de le finir et de voir les résultats obtenus au bout de ces trois semaines de travails.

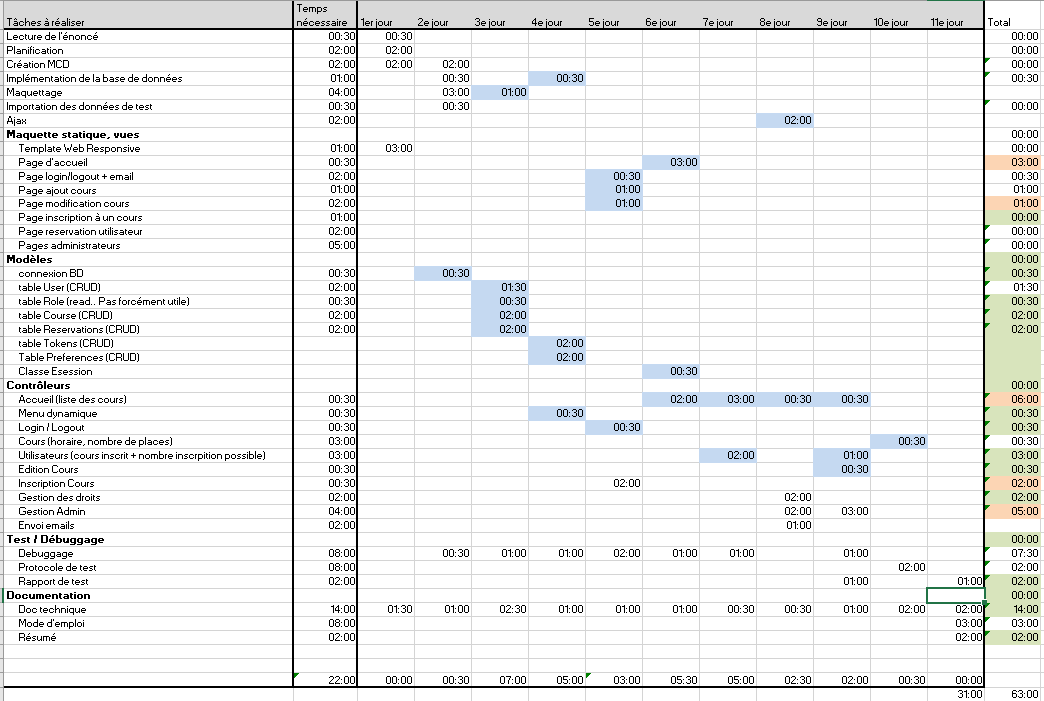
# Planning

Pour ce travail, selon les recommandations de mon enseignant, la méthode de gestion de projet utilisée est la méthode dite "modèle en cascade".

## Planning Prévisionnel



## Planning réel



1. Source : <https://fr.wikipedia.org/wiki/Bootstrap_(framework)> [↑](#footnote-ref-1)
2. Source : <https://fr.wikipedia.org/wiki/JQuery> [↑](#footnote-ref-2)