



École Polytechnique de l'Université de Tours

64, Avenue Jean Portalis

37200 TOURS, FRANCE

Tél. (33)2-47-36-14-14

Fax (33)2-47-36-14-22

[www.polytech.univ-tours.fr](http://www.polytech.univ-tours.fr)

## Rapport

### Projet de Programmation et Génie Logiciel

# Création d'un logiciel standalone pour assister la préparation et la gestion de compétitions locales de BMX

Auteur(s)

Encadrant(s)

**Pierre FOURRE**

[\[pierre.fourre@etu.univ-tours.fr\]](mailto:pierre.fourre@etu.univ-tours.fr)

**Maxime SENGER**

[\[maxime.senger@etu.univ-tours.fr\]](mailto:maxime.senger@etu.univ-tours.fr)

**Christophe LENTE**

[\[christophe.lente@univ-tours.fr\]](mailto:christophe.lente@univ-tours.fr)

**Polytech Tours  
Département Informatique**

<b>Contexte de la réalisation</b>	<b>5</b>
Objectifs	5
Besoin client	6
<b>Spécification générale</b>	<b>7</b>
Langages utilisés	7
Technologies utilisées	7
flask	7
jinja2	7
sqlite	7
bootstrap 4	7
jquery 3	7
Rendu	8
Bibliothèques python utilisées	8
flask_login et werkzeug.security	8
sqlalchemy et flask_sqlalchemy	8
Justification des choix technologiques	8
Cout	8
Documentation riche	8
Simplicité de maintien	9
Existence de framework CSS	9
<b>Description des interfaces externes du logiciel</b>	<b>10</b>
Interfaces matériel/logiciel	10
Interfaces homme/machine	10
Interfaces logiciel/logiciel	10
<b>Enchaînement vue</b>	<b>10</b>
<b>Spécifications fonctionnelles</b>	<b>11</b>
Authentification	11
Maquette	11
Paramètres	11
Résultats	11
Accueil	12
Résultats	12
Clubs	12
Maquette	13
Paramètres	13
Résultats	13
Titulaires	13
Maquette	14
Paramètres	15
Résultats	15

Titulaires club	16
Maquette	16
Championnats	17
Maquette	17
Paramètres	17
Résultats	17
Catégories	17
Maquette	18
Paramètres	18
Résultats	18
Catégories	19
Maquette	19
Résultats	19
Phases	19
Maquette	20
Résultats	20
Races	20
Maquette	21
Résultats	21
<b>Spécifications non fonctionnelles</b>	<b>22</b>
Contraintes de développement et conception	22
Contraintes de fonctionnement et d'exploitation	22
Performance	22
Capacité	22
Contrôlabilité	22
Sécurité	22
<b>Structure programme</b>	<b>23</b>
Code source	23
app.py	23
website/	23
website/__init__.py	24
website/auth.py	24
website/views.py	24
website/models.py	24
website/database.db	25
website/static/	25
website/templates/	25
Rendu	25
Base de données	26
<b>Use cases</b>	<b>27</b>
Utilisateur	28

<b>Développements restants</b>	<b>29</b>
Nécessaires	29
Gestion des championnats départementaux	29
Affichage des scores entre races	29
Téléchargement des résultats d'étapes/championnats	29
Envisageable	30
Import des titulaires	30
Modification des titulaires	30
Bouton de sélection de tous les titulaires	30

# Contexte de la réalisation

## Objectifs

Notre projet s'inscrit dans la **résolution d'une vraie problématique**, ici la création d'un logiciel pour aider à faciliter la tâche des clubs et de leurs bénévoles quant à l'**organisation des différentes manches** du championnat régional et du championnat départemental UFOLEP.

Tout au long de l'année, les différents clubs accueillent en leur sein une manche du championnat (l'équivalent d'une journée dans les autres sports). En amont de ladite manche, les bénévoles doivent préparer la répartition des pilotes au sein de différents groupes en s'assurant qu'à l'intérieur de ces groupes, la répartition des couloirs soit le plus équitable possible (certains couloirs étant considérés plus avantageux que d'autres).

Ce travail est très souvent **long et fastidieux**, ainsi l'automatisation de cette tâche via notre logiciel est un vrai plus.

En plus du travail en amont, un suivi des courses est nécessaire durant la manche (calcul des points des pilotes en fonction de leur position, ventilation dans les phases finales ( $\frac{1}{4}$ ,  $\frac{1}{2}$  finale) etc...). Notre logiciel permet aussi **d'automatiser cela** : les bénévoles du club ont uniquement à rentrer les positions d'arrivée des différents pilotes sur la ligne d'arrivée dans le logiciel, celui-ci traite ces données et effectue en interne et ressort le planning de la suite de la manche.

Enfin, le logiciel permet d'obtenir une fois toutes les courses terminées le classement de la manche ainsi que le classement général.

A terme, notre logiciel devrait ainsi pouvoir assister les clubs lors des **10 manches de championnat annuel** (6 manches régionales et 4 manches départementales à l'heure actuel)

## Besoin client

Les besoins clients ont été assez faciles à comprendre. En effet, en plus d'une réunion avec un représentant de l'UFOLEP pour bien saisir les problèmes auxquels devraient répondre notre logiciel, plusieurs documents annexes nous furent transmis, document qui fut d'une grande aide.

Ainsi, dans l'un des documents annexe, on peut retrouver **les principales contraintes du logiciel** demandé :

*"Ce logiciel doit répondre aux réglementations techniques de ces deux championnats en ce qui concerne :*

- L'attribution des couloirs*
- Le brassage des pilotes*
- La ventilation des pilotes en  $\frac{1}{4}$ ,  $\frac{1}{2}$ , et finale*
- Le classement par points de la course du jour*
- Le classement général par points du championnat en cours*

*Le logiciel devra veiller à ce que chaque pilote prenne le départ dans un couloir différent lors de chaque course comme indiqué sur les « feuilles de race »."*

En plus de ces éléments essentiels, **d'autres caractéristiques** à implémenter furent discutées avec le représentant de l'UFOLEP :

- Gestion des clubs et des titulaires via le logiciel
- L'affichage des scores intermédiaire au fil de la manche
- Possibilité de changement de dernières minutes (Ajout ou suppression d'un pilote au matin d'une manche)

# Spécification générale

## Langages utilisés

Le projet se divise en deux parties, le back-end et le front-end. Le back-end regroupe toutes les fonctions non-visibles par l'utilisateur qui permettent le calcul de données. Au contraire, le front-end est l'interface graphique avec laquelle l'utilisateur interagit.

Le back-end de ce logiciel sera uniquement réalisé en **python**, en utilisant la version 3.8. Le front-end quant à lui sera fait d'une combinaison de technologies WEB que sont **HTML5**, **CSS2.1** et **Javascript**.

## Technologies utilisées

### flask

Flask est la bibliothèque fondamentale utilisée lors de ce projet. Celle-ci permet la gestion du serveur WEB.

### jinja2

Jinja2 est un moteur de templates pour python, il permet la **transformation de code python en HTML5**.

### sqlite

Sqlite est une bibliothèque écrite en langage C qui propose un moteur de base de données relationnelle accessible par le langage SQL depuis un **fichier local**.

### bootstrap 4

Bootstrap 4 est un framework CSS simplifiant la création de sites WEB esthétiques. Il est particulièrement **performant pour la création d'interfaces** "administratives".

### jquery 3

Jquery 3 est un framework javascript notamment utilisé par bootstrap, nous l'utiliserons aussi afin de faciliter la création de requêtes AJAX (ex: lors de la modification d'un titulaire).

## Rendu

Une fois le projet réalisé, le client recevra un **exécutable standalone** (ne nécessitant pas d'installation) utilisable depuis une clef USB. Il recevra aussi le **code source complet**, permettant la reprise du code par une personne tierce si nécessaire. Celui-ci sera **uniquement utilisable sous Windows**.

## Bibliothèques python utilisées

Ci-dessous une liste non-exhaustive des **bibliothèques python** les plus importantes utilisées dans le projet :

### flask\_login et werkzeug.security

Flask\_login est un **module pour la bibliothèque Flask**. Il permet la **gestion d'authentification** d'utilisateurs sécurisé. Werkzeug.security sera utilisé afin de comparer les hash présents dans la base de données avec les données fournies par l'utilisateur.

### sqlalchemy et flask\_sqlalchemy

Sqlalchemy et son module pour flask permettent **une interface ORM** (simulation de base de données orientée objet) simplifiant ainsi la gestion de données.

## Justification des choix technologiques

Nous avons fait le choix d'utiliser des **technologies WEB** pour la création de l'interface du logiciel plutôt que d'utiliser des librairies telles que QT5 ou TKinter. Ce choix a été réalisé après une **réflexion de notre équipe**.

### Cout

En prenant QT5 pour exemple, une licence commerciale revient à **\$3950 par an**. Notre solution est **gratuite**, et ce même pour une utilisation commerciale.

### Documentation riche

Les technologies WEB basiques que sont l'HTML, le CSS et JavaScript bénéficient d'une documentation **très importante** du fait de leur ancienneté et du fait que ce sont des langages **très utilisés**.



## Simplicité de maintien

Du fait de la popularité des technologies WEB, **chaque développeur connaît les bases du développement WEB** et de ces technologies. De ce fait, le maintien du logiciel par une tierce personne est **simplifié** et ne nécessite **pas l'apprentissage d'une nouvelle technologie**.

## Existence de framework CSS

Afin de faciliter encore plus le développement de l'interface, il existe des frameworks CSS permettant de réaliser une interface agréable et esthétique en **une fraction du temps** qui aurait été nécessaire en utilisant une librairie telle que QT5 ou TKinter.

# Description des interfaces externes du logiciel

## Interfaces matériel/logiciel

Le serveur est hébergé **localement** sur la machine qui utilise le logiciel. Celle-ci a pour système d'exploitation **Windows**.

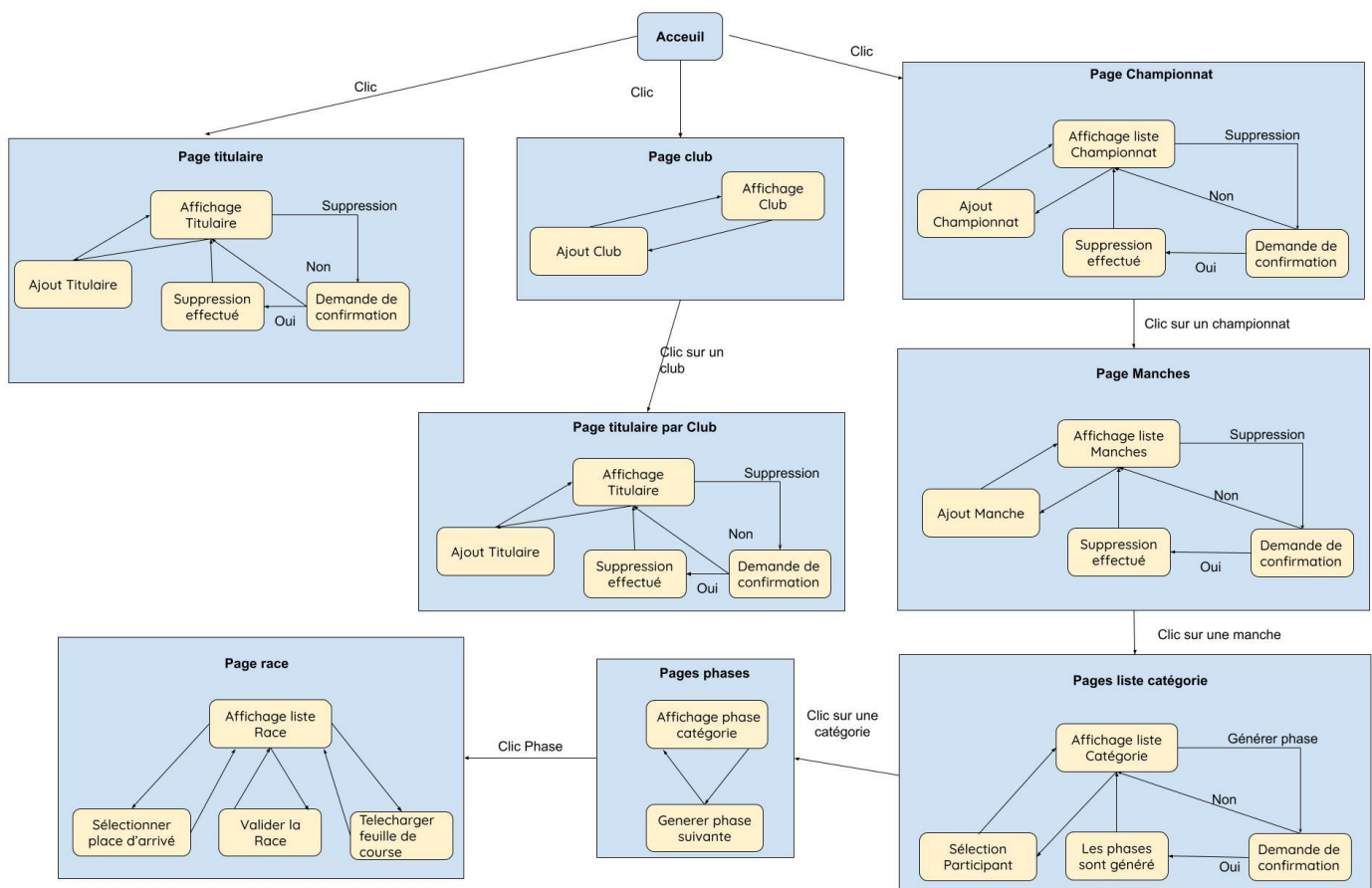
## Interfaces homme/machine

La solution est accessible aux utilisateurs via un **site internet** en utilisant un navigateur tel que Google Chrome ou Mozilla Firefox via l'url **127.0.0.1:5000**.

## Interfaces logiciel/logiciel

L'interface WEB interagit avec le serveur.

## Enchaînement vue



# Spécifications fonctionnelles

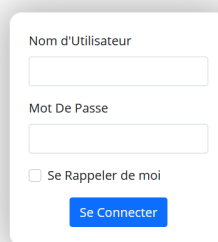
## Authentification

### Maquette

Page web contenant un formulaire de connexion. Le formulaire est composé de deux champs obligatoires :

- Nom d'Utilisateur
- Mot De Passe

Ainsi qu'un bouton de connexion permettant l'envoi du formulaire.



Maquette d'un formulaire de connexion. Le formulaire est composé de deux champs obligatoires : 'Nom d'Utilisateur' et 'Mot De Passe'. Il y a également une case à cocher 'Se Rappeler de moi' et un bouton 'Se Connecter'.

### Paramètres

Paramètre	Description	Exemple
Nom d'Utilisateur	Alphanumérique + caractères spéciaux (1-30)	ufolep
Mot De Passe	Alphanumérique + caractères spéciaux (7-30)	ufolep2021

### Résultats

La page s'affiche uniquement si l'utilisateur n'est **pas encore connecté**.

L'envoi du formulaire n'est possible que si les champs de nom d'utilisateur et de mot de passe ne sont **pas vides**.

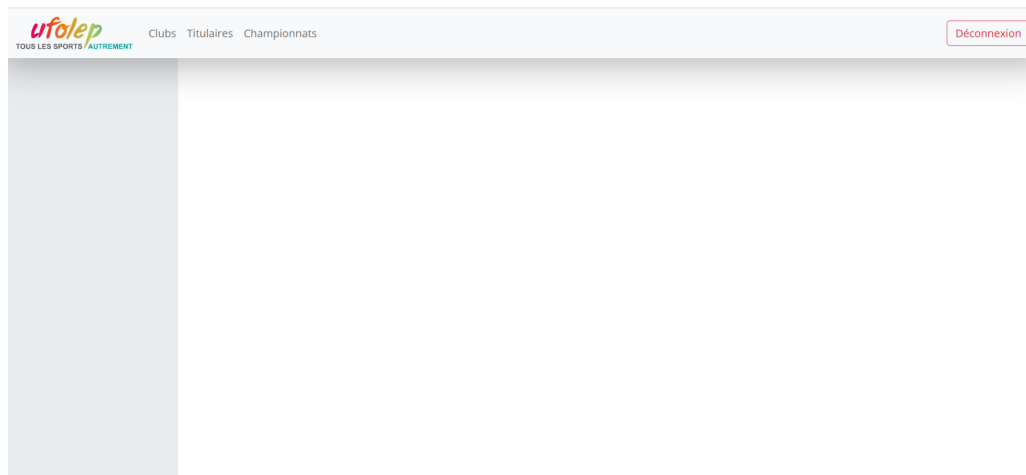
Lors de l'envoi du formulaire :

- Si aucun compte utilisateur correspondant aux données fournies n'est trouvé, un message d'erreur notifie l'utilisateur au-dessus du formulaire de connexion
- Sinon, l'utilisateur est connecté et est redirigé vers la page d'accueil

## Accueil

### Maquette

Page d'accueil de l'application, on y est redirigé lors de la connexion. On y retrouve la structure de base de toutes les autres pages dont la barre de navigation. Celle-ci permet de naviguer entre les "titulaires", les "clubs", les "championnats" et de se déconnecter.



### Résultats

La page s'affiche uniquement si l'utilisateur **est connecté**.

Le clic sur les boutons "Clubs", "Titulaires" et "Championnats" **redirigent vers les pages associées**.

Le clic sur le bouton de déconnexion redirige vers la page d'authentification et **bloque l'accès** aux pages nécessitant d'être connecté.

## Clubs

### Maquette

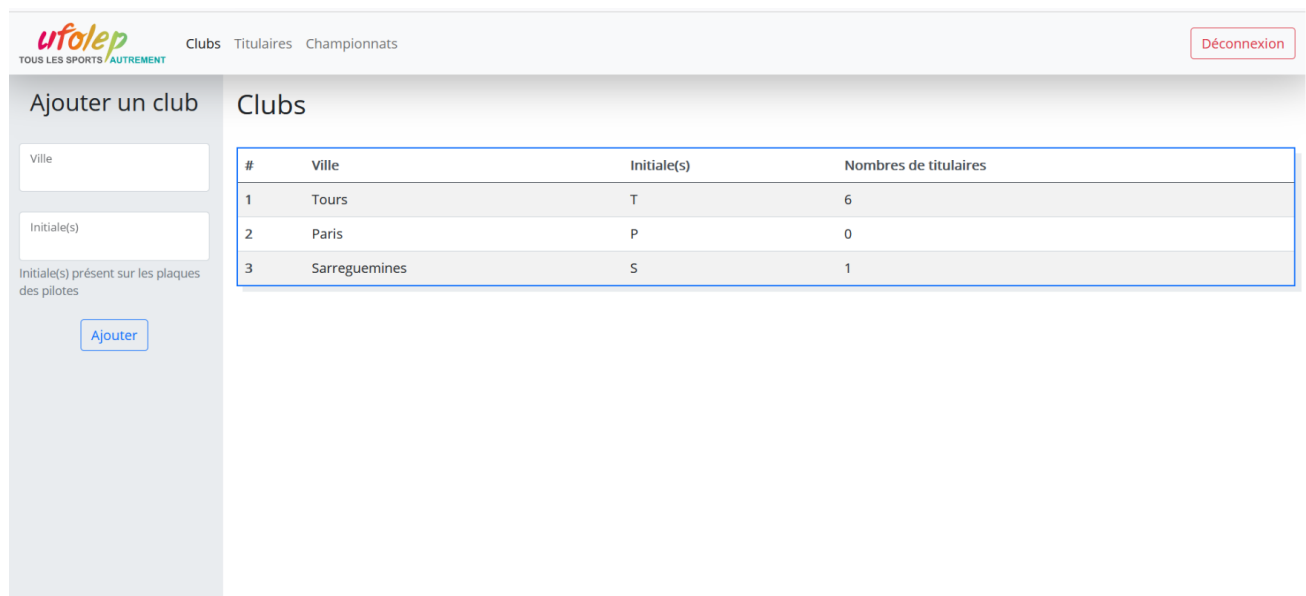
Page WEB affichant la liste des clubs partenaires. On y retrouve un formulaire permettant d'ajouter un club composé des deux champs suivants :

- Ville
- Initiale(s)

Le bouton "Ajouter" permet l'envoi du formulaire.

On retrouve aussi un tableau de la liste des clubs contenant les informations suivantes :

- Ville
- Initiale(s)
- Nombre de titulaires



#	Ville	Initiale(s)	Nombres de titulaires
1	Tours	T	6
2	Paris	P	0
3	Sarreguemines	S	1

### Paramètres

Paramètre	Description	Exemple
Ville	Alphanumérique + caractères spéciaux (1-30)	Tours
Initiale(s)	Alphanumérique (1-2)	T

### Résultats

La page s'affiche uniquement si l'utilisateur est **connecté**.

L'envoi du formulaire est possible uniquement si le club **n'existe pas** encore et si les champs ne sont **pas vides**.

Lors de l'envoi du formulaire, **le club est ajouté** et la page se recharge pour afficher la liste de clubs mise à jour.

Le clic sur l'un des clubs redirige vers la liste des titulaires du club sélectionné.

# Titulaires

## Maquette

Page WEB affichant la liste des titulaires. On y retrouve un formulaire permettant d'ajouter un titulaire composé des champs suivants :


- Nom
- Prénom
- Date de naissance
- Club
- Numéro de plaque
- Sexe

Le bouton "Ajouter" permet l'envoi du formulaire.

On retrouve aussi un tableau de la liste des titulaires contenant les informations suivantes :

- Nom
- Prénom
- Date de naissance
- Sexe
- Club
- Plaque

Ainsi qu'un bouton de suppression (pas encore mis en place)



Clubs Titulaires Championnats

Déconnexion

Ajouter un titulaire

Nom

Prénom

Date De Naissance  
jj / mm / aaaa

Club

Choisir un club

Numéro De Plaque

Sexe

Choisir un sexe

Titulaires (tous)

#	Nom	Prénom	Date de Naissance	Sexe	Club	Plaque	
1	MAXIME	SENGER	29/07/2000	Homme	Tours	T 01	✕
2	MARIE-AMÉLIE	GARCIA	29/07/2000	Homme	Tours	T 02	✕
3	PIERRE	FOUREE	29/07/2000	Homme	Tours	T 03	✕
4	DIEGO	DELOFFRE	29/07/2000	Homme	Tours	T 04	✕
5	JORIS	LOIT	29/07/2000	Homme	Tours	T 05	✕
6	JEAN	DUPOND	29/07/2000	Homme	Tours	T 06	✕
7	TIM	FRITZ	29/07/2000	Homme	Sarreguemines	S 01	✕

## Paramètres

Paramètre	Description	Exemple
Nom	Alphanumérique	Dupond
Prénom	Alphanumérique	Jean
Date De Naissance	Date (jj/mm/aaaa)	21/07/2000
Club	Menu déroulant	Tours
Numéro De Plaque	Numérique (1-99)	1
Sexe	Menu déroulant	Homme

## Résultats

La page s'affiche uniquement si l'utilisateur est **connecté**.

L'envoi du formulaire est possible uniquement si le titulaire et le numéro de plaque **n'existe pas** encore pour le club sélectionné et si les champs ne sont **pas vide**.


Lors de l'envoi du formulaire, **le titulaire est ajouté** et la page se recharge pour afficher la liste de titulaires mise à jour.



# Titulaires club

## Maquette

Même page que la page "Titulaires" ci-dessus. La différence étant que seuls les titulaires inscrits au club sélectionné sont affichés.


TOUS LES SPORTS / AUTREMENT

Clubs Titulaires Championnats

Déconnexion

Ajouter un titulaire

Nom

Prénom

Date De Naissance  
jj / mm / aaaa

Club

Choisir un club

Numéro De Plaque

Sexe

Choisir un sexe

Titulaires (Tours)

#	Nom	Prénom	Date de Naissance	Sexe	Club	Plaque	
1	MAXIME	SENGER	29/07/2000	Homme	Tours	T 01	✕
2	MARIE-AMÉLIE	GARCIA	29/07/2000	Homme	Tours	T 02	✕
3	PIERRE	FOUREE	29/07/2000	Homme	Tours	T 03	✕
4	DIEGO	DELOFFRE	29/07/2000	Homme	Tours	T 04	✕
5	JORIS	LOIT	29/07/2000	Homme	Tours	T 05	✕
6	JEAN	DUPOND	29/07/2000	Homme	Tours	T 06	✕

## Championnats

### Maquette

Page WEB affichant la liste des championnats. On y retrouve un formulaire permettant d'ajouter un championnat composé des champs suivants :

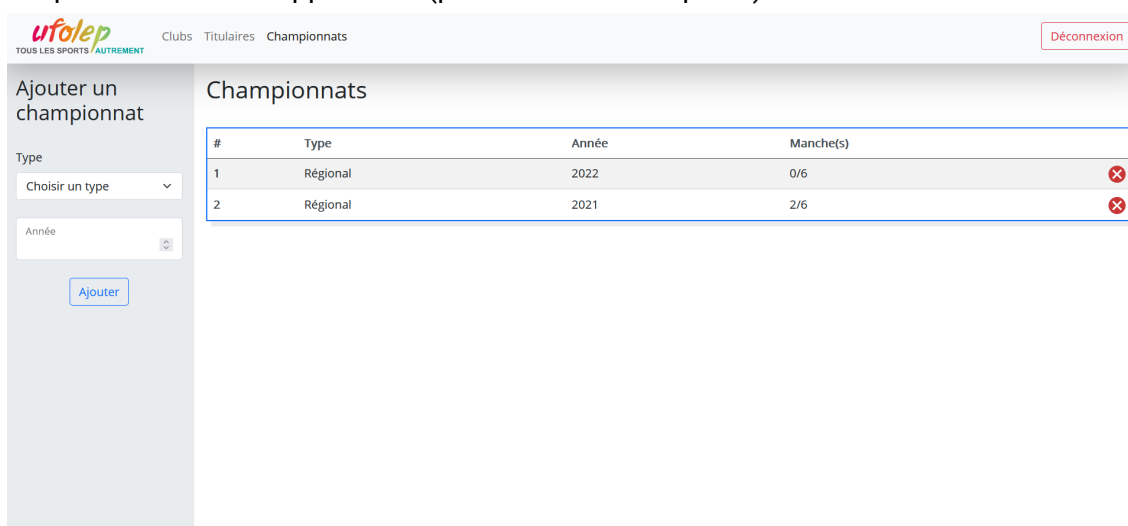
- Type
- Année

Le bouton "Ajouter" permet l'envoi du formulaire.

On retrouve aussi un tableau de la liste des championnats contenant les informations suivantes :

- Type
- Année
- Manche(s)

Ainsi qu'un bouton de suppression (pas encore mis en place)



### Paramètres

Paramètre	Description	Exemple
Type	Menu déroulant	Régional
Année	Numérique (2000-3000)	2021

### Résultats

La page s'affiche uniquement si l'utilisateur est **connecté**.

L'envoi du formulaire est possible uniquement si le type de championnat **n'existe pas** pour l'année sélectionnée et si les champs ne sont **pas vides**.

Lors de l'envoi du formulaire, **le championnat est ajouté** et la page se recharge pour afficher la liste de championnats mise à jour.

Le clic sur l'un des championnats redirige vers la liste des manches du championnat sélectionné.

## Catégories

### Maquette

Page WEB affichant la liste des manches d'un championnat. On y retrouve un formulaire permettant d'ajouter une manche composé du champ suivant :

- Club Organisateur

Le bouton "Ajouter" permet l'envoi du formulaire.

On retrouve aussi un tableau de la liste des manches contenant les informations suivantes :

- Club Organisateur
- Participants

Ajouter une étape  
Club Organisateur  
Choisir un lieu  
Ajouter

⏪ Etapes : Régional - 2021

#	Club Organisateur	Participants
1	tours	7/7

### Paramètres

Paramètre	Description	Exemple
Club Organisateur	Menu déroulant	Tours

### Résultats

La page s'affiche uniquement si l'utilisateur est **connecté**.

L'envoi du formulaire est possible uniquement si une étape dont le club organisateur est le même **n'existe pas** pour le championnat sélectionné et si le champ n'est **pas vide**.

Lors de l'envoi du formulaire, **l'étape est ajoutée** et la page se recharge pour afficher la liste des étapes mise à jour.

Le clic sur l'une des étapes redirige vers la liste des catégories de l'étape sélectionnée.

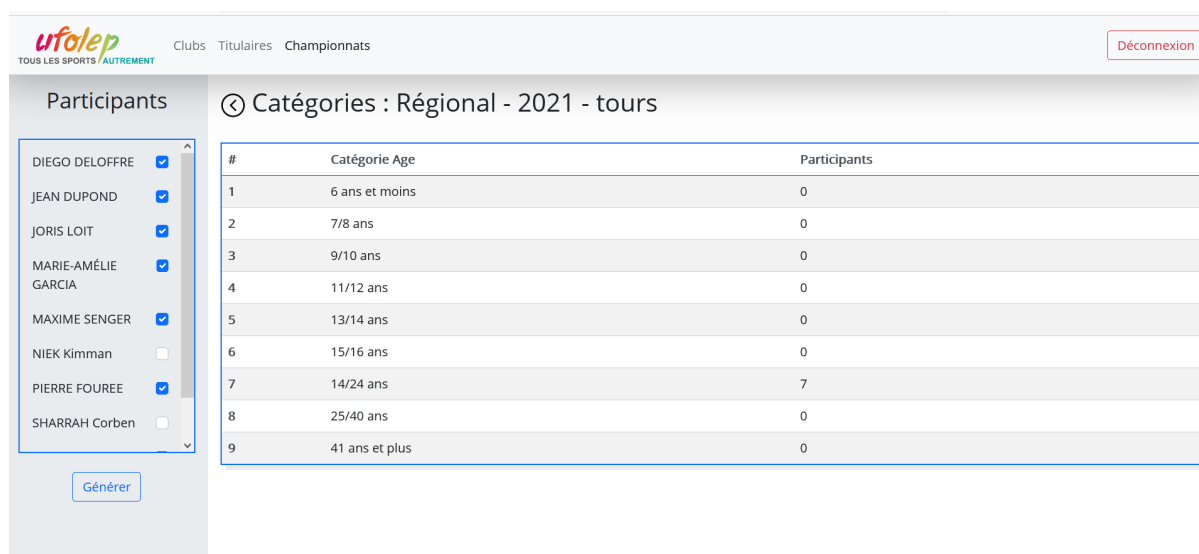
## Catégories

### Maquette

Page WEB affichant la liste des catégories d'une manche. On y retrouve un formulaire permettant la sélection des participants composé d'une case cochable par titulaire. Le bouton "Ajouter" permet l'envoi du formulaire.

On retrouve aussi un tableau de la liste des catégories contenant les informations suivantes:

- Catégorie Age
- Participants



#	Catégorie Age	Participants
1	6 ans et moins	0
2	7/8 ans	0
3	9/10 ans	0
4	11/12 ans	0
5	13/14 ans	0
6	15/16 ans	0
7	14/24 ans	7
8	25/40 ans	0
9	41 ans et plus	0

### Résultats

La page s'affiche uniquement si l'utilisateur est **connecté**.

L'envoi du formulaire est possible uniquement si l'utilisateur **confirme son choix**.

Lors de l'envoi du formulaire, **toutes les données générées précédemment sont supprimées** et **une nouvelle structure est générée**. La page est aussi actualisée afin d'afficher les nouvelles données.

Le clic sur l'une des catégories redirige vers la liste des phases de la catégorie sélectionnée.

## Phases

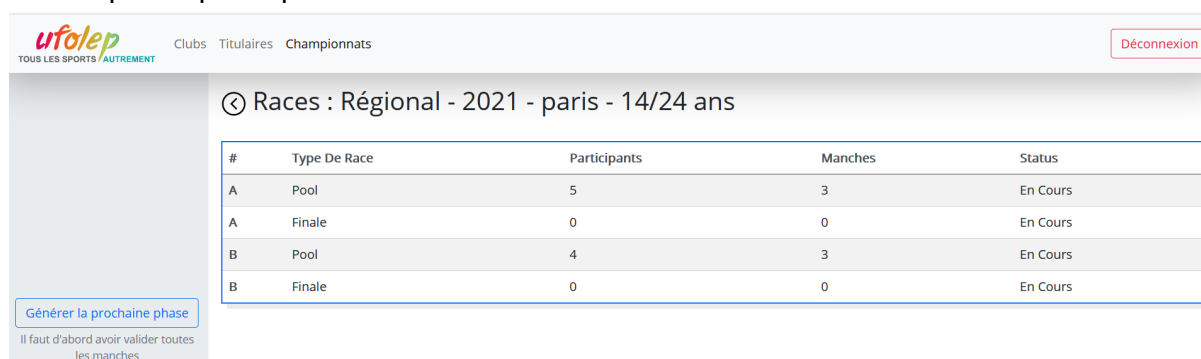
### Maquette

Page WEB affichant la liste des phases d'une catégorie. On y retrouve un bouton permettant la génération des phases suivantes.

On retrouve aussi un tableau de la liste des phases contenant les informations suivantes :

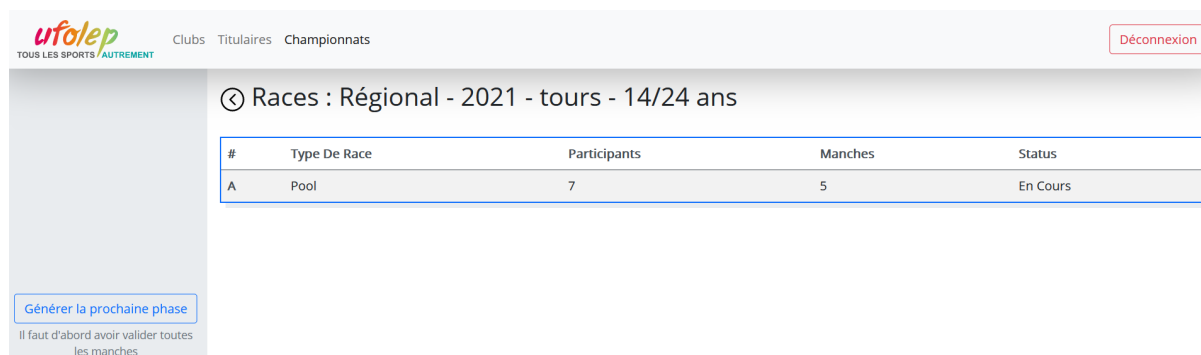
- Type De Race
- Participants
- Race(s)
- Status

Phases pour 9 participants :



#	Type De Race	Participants	Manches	Status
A	Pool	5	3	En Cours
A	Finale	0	0	En Cours
B	Pool	4	3	En Cours
B	Finale	0	0	En Cours

Phases pour 7 participants :



#	Type De Race	Participants	Manches	Status
A	Pool	7	5	En Cours

### Résultats

La page s'affiche uniquement si l'utilisateur est **connecté**.

Le clic sur "Générer la prochaine phase" **est possible que si la phase courante est "Finie"**.

La génération de la prochaine phase **calcule les scores en fonction de l'ordre d'arrivée** aux différentes races et **génère les prochaines phases**.

Le clic sur l'une des phases redirige vers la liste des races de la phase sélectionnée.

## Races

### Maquette

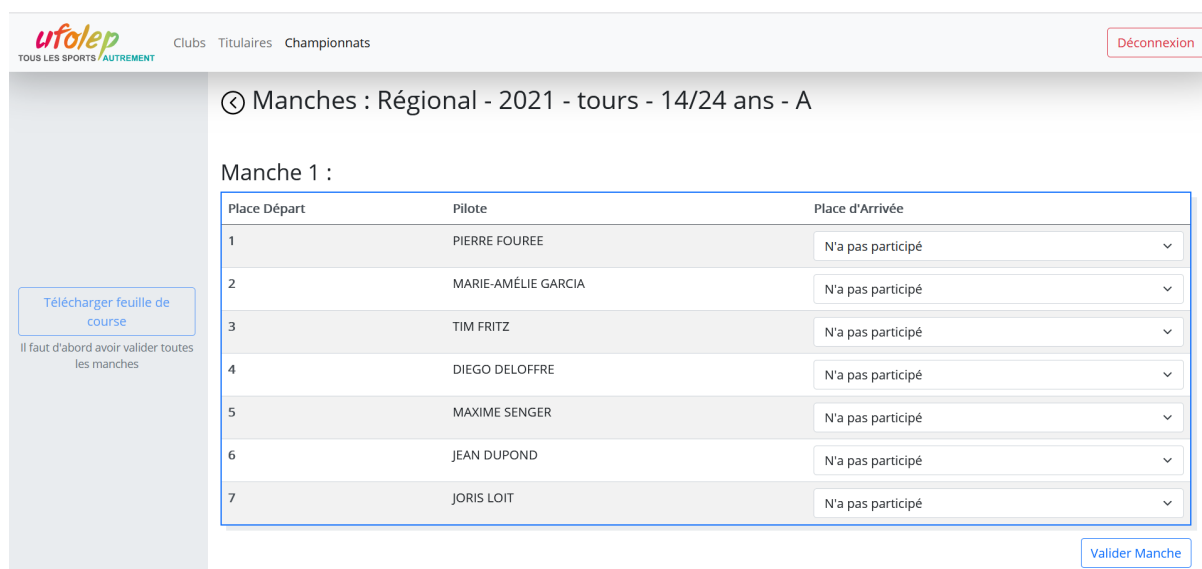
Page WEB affichant la liste des races d'une phase. On y retrouve un bouton permettant le téléchargement du tableau des scores intermédiaire (pas encore implémenté).

On retrouve aussi plusieurs tableaux, chacun représentant une race comportant les champs suivants :

- Place départ
- Pilote
- Place d'Arrivée

La place d'arrivée est un menu déroulant permettant le choix de celle-ci.

Le bouton "Valider Race" permet d'enregistrer les places d'arrivées.



ufolep TOUS LES SPORTS / AUTREMENT Clubs Titulaires Championnats Déconnexion

⏪ Manches : Régional - 2021 - tours - 14/24 ans - A

Manche 1 :

Place Départ	Pilote	Place d'Arrivée
1	PIERRE FOUREE	N'a pas participé
2	MARIE-AMÉLIE GARCIA	N'a pas participé
3	TIM FRITZ	N'a pas participé
4	DIEGO DELOFFRE	N'a pas participé
5	MAXIME SENER	N'a pas participé
6	JEAN DUPOND	N'a pas participé
7	JORIS LOIT	N'a pas participé

Télécharger feuille de course

Il faut d'abord avoir validé toutes les manches

Valider Manche

### Résultats

La page s'affiche uniquement si l'utilisateur est **connecté**.

Le bouton "Télécharger feuille de course" n'est cliquable qu'**une fois que toutes les races ont été validées**.

Le clic du bouton "Valider Race" **enregistre les places d'arrivées** des différents pilotes.

Une fois que toutes les races sont validées, le statut de la phase passe à "Finie".

# Spécifications non fonctionnelles

## Contraintes de développement et conception

L'application doit être accessible en utilisant un navigateur WEB **Google Chrome version 92** ou plus et **Mozilla Firefox version 93** ou plus.

Les données doivent être stockées dans **un fichier ou un SGBD**.

Le logiciel doit être **standalone**, et ne doit **pas nécessiter l'installation d'autres logiciels** pour fonctionner.

Le logiciel doit pouvoir être lancé depuis la plateforme **Windows**.

## Contraintes de fonctionnement et d'exploitation

### Performance

Il n'y a pas de contraintes de performance.

### Capacité

Il n'y a pas de contraintes de capacité.

### Contrôlabilité

Il n'y a pas de contraintes de contrôlabilité.

### Sécurité

L'utilisateur doit **posséder un compte**, accessible via une page de connexion, afin de gérer les données relatives aux clubs, titulaires et championnats.

L'authentification est réalisée via la combinaison de nom d'utilisateur et de mot de passe suivante : **"ufolep:ufolep2021"**.

# Structure programme

## Code source

Le code source du logiciel est structuré de manière à suivre les conventions structurelles **définies par Flask**. De ce fait le code source est séparé en **plusieurs fichiers et dossiers** dont nous expliquerons ici le but.

### app.py

Le fichier "app.py" est **la base** de tout le programme, c'est le fichier à **lancer en utilisant l'interpréteur Python**.

Le fichier permet la création du **serveur WEB local** et l'**ouverture du navigateur** par défaut de la machine.

### website/

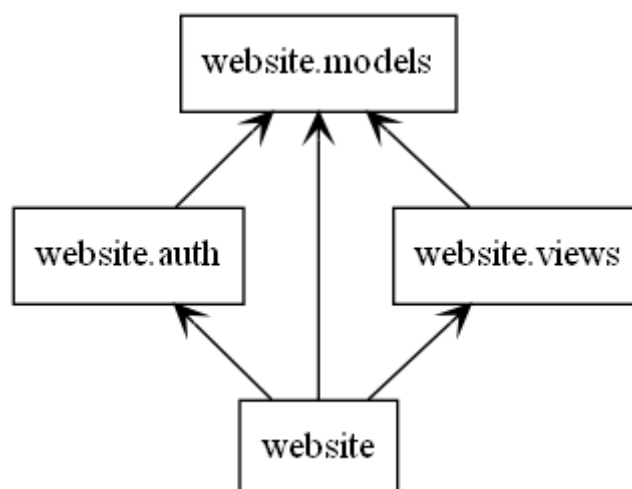
Le dossier "website" est le dossier contenant le **package nécessaire au fonctionnement** de l'application Flask. Celui-ci contient deux dossiers :

- static
- templates

Ainsi que cinq fichiers :

- \_\_init\_\_.py
- auth.py
- views.py
- models.py
- database.db

Voici la structure d'import dans le dossier website :





## website/\_\_init\_\_.py

Le fichier “\_\_init\_\_.py” est la base de notre package, il contient la **variable permettant l'accès à la base de données** ainsi que la fonction servant à **configurer l'application**. Dans cette fonction de configuration nous spécifions notamment les routes présentes dans les fichiers “views.py” et “auth.py”.

## website/auth.py

Les fichiers “auth.py” et “views.py” sont structurés de la même manière. En effet les deux fichiers **déclarent des routes** (URL d'accès pour le navigateur et protocole utilisé) et les lient avec **une fonction python** qui sera exécutée à chaque visite du navigateur sur l'URL spécifiée.

En voici un exemple :

```
190 @views.route("/clubs/")
191 @login_required
192 def clubs() :
193
194     clubs = Club.query.all()
195
196     return render_template("clubs.html", clubs=clubs)
197
```

Dans cet exemple, nous définissons la route d'URL “/clubs/” de ce fait, si l'utilisateur visite la page “127.0.0.1:5000/clubs/” c'est la fonction “clubs” qui sera exécutée.

La mention “@login\_required” **bloque l'accès aux personnes non-authentifiées**.

La fonction “render\_template” permet de **peupler le template “clubs.html” avec les variables passées en argument**.

Le fichier “auth.py” contient les routes ainsi que les fonctions permettant les actions relatives à **l'authentification de l'utilisateur**.

## website/views.py

Le fichier “views.py” définit toutes les routes et fonctions permettant l'utilisation du logiciel. **Chaque page WEB accessible y est définie**. On y retrouve aussi des fonctions nécessaires à la **récupération des données dans la base de données** lorsqu'une structure particulière est nécessaire, ainsi que la **fonction de classement des pilotes** en fonction de leur ordre d'arrivée aux races précédentes.

## website/models.py

Le fichier “models.py” contient la **structure de la base de données** y permettant l'accès au moteur ORM SQLAlchemy.

Chaque classe présente dans le fichier définit une table de la base de donnée, et chaque variable une colonne ou une relation entre colonne.

## website/database.db

Le fichier “database.db” est le fichier servant de **base de données**. C’est un fichier **SQLite** pouvant être ouvert avec un logiciel gratuit et libre tels que “DB Browser for SQLite”. On y retrouve **la même structure** que définie dans le fichier “website/models.py”.

## website/static/

Le dossier “website/static/” contient tous les fichiers dit “statiques”. On retrouve les fichiers de **style CSS**, les fichiers **javascript** ainsi que **les images** nécessaires à la création de l’interface du logiciel.

## website/templates/

Le dossier “website/templates/” contient lui tous les fichiers de **templating Jinja2**. Les fichiers sont au format “.html”. Le fichier de base est “base.html” qui définit la structure de la page WEB, tous les autres fichiers l’“étendent”. Cela permet de **limiter la quantité de code**, puisqu’on ne réécrit pas plusieurs fois la même chose.

## Rendu

Comme prévu par le cahier des charges, le rendu est un **exécutable standalone** qui peut être stocké sur une clef USB. Il n'y a qu'un fichier de visible dans le dossier de rendu, celui-ci est un raccourci à chemin relatif créé avec le logiciel "Reverse" puisque l'utilitaire windows ne permet que la création de raccourcis statiques. Ce raccourci pointe vers l'exécutable du logiciel que l'on retrouve dans **le dossier caché "app"** aux côtés des autres fichiers nécessaires créé par "Pyinstaller". Nous y avons ajouté le dossier "website" qui, comme dans le code source, contient les fichiers statiques, les templates et le fichier de base de données.

Nous n'avons pas opté pour la création d'exécutable dit "One File" car cela ne permettait pas la **persistance des données** de la base de données.

## Base de données

La base de données de ce logiciel est de structure assez **complexe**, en effet afin de faciliter l'accès à certaines informations nous avons ajouté de **nombreuses clefs étrangères**. De ce fait le schéma, bien que complexe, permet un **gain de temps** et d'itérations de requêtes considérables.

Toute la structure de la base de données est définie dans le fichier "website/models.py" selon la syntaxe nécessaire au moteur ORM Sqlalchemy. En voici un exemple :

```
class Titulaire(db.Model) :
    __tablename__ = "titulaire"

    id = db.Column(db.Integer, primary_key=True)
    nom = db.Column(db.String(100), nullable=False)
    prenom = db.Column(db.String(100), nullable=False)
    date_naissance = db.Column(db.Date, nullable=False)
    numero_plaque = db.Column(db.String(2), nullable=False)
    club_id = db.Column(db.Integer, db.ForeignKey('club.id'), nullable=False)
    sexe_id = db.Column(db.Integer, db.ForeignKey('sexe.id'), nullable=False)
    manches = db.relationship('Participant_manche', backref='titulaire', lazy=True)
```

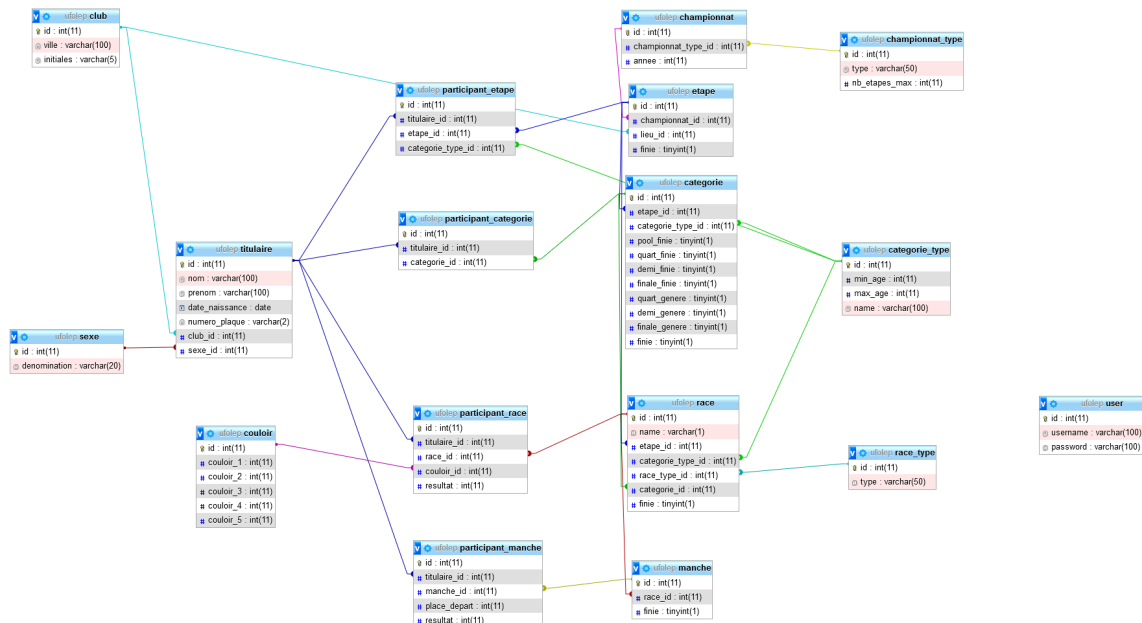
Chaque table est définie par une classe dans le fichier python. La variable "\_\_tablename\_\_" est **optionnelle**. Si non définie, le nom de la table sera le même que celui de la classe en minuscule.

Chaque variable ne commençant pas par "\_\_" et étant au singulier représente une colonne de la table. Le nom de la colonne est le même que celui de la variable.

Nous avons décidé de suivre la convention de nommage suivante :

- Chaque table à une colonne "id" qui est sa **clef primaire**.
- Les variables se terminant par "\_id" sont des **clefs secondaires** vers la table nommée.
- Les variables au pluriel sont des **listes créées par Sqlalchemy** et **n'existent pas dans la base de données**. Elle représente la liste d'éléments dont l'élément courant est clef secondaire. Ces listes sont itérables, ce qui est très pratique.

- Voici un schéma relationnel de la base de données :



Vous le trouverez au format PNG dans le dossier de rendu sous le nom : "database.png".

# Use cases

## Utilisateur

- L'utilisateur lance le programme en double-cliquant sur le raccourci de l'exécutable.
- L'utilisateur s'identifie via la page de login qui vient de s'ouvrir en utilisant le nom d'utilisateur et le mot de passe fournis (CF: Spécifications non-fonctionnelles/Contraintes de fonctionnement et d'exploitation/Sécurité)
- L'utilisateur doit ajouter un nouveau club qui vient de rejoindre l'association, il clic donc sur l'onglet "Club" dans la barre de navigation et remplit le formulaire avant de cliquer sur "Ajouter".
- L'utilisateur veut ajouter les titulaires du nouveau club, il clic dans le tableau sur la ligne correspondante au club qu'il vient de créer et remplit le formulaire d'ajout de titulaire avant de cliquer sur "Ajouter".
- Afin de vérifier que tous les titulaires ajoutés ont bien été pris en compte, l'utilisateur clic sur l'onglet "Titulaire" de la barre de navigation, ce qui affiche la liste de tous les titulaires.
- L'utilisateur se rend dans l'onglet "Championnat" de la barre de navigation afin de créer le championnat régional de l'année. Il remplit le formulaire avant de cliquer sur "Ajouter".
- En cliquant sur le nouveau championnat créé, l'utilisateur peut maintenant ajouter une nouvelle manche. Pour cela, celui-ci remplit le formulaire avant de cliquer sur "Ajouter".
- Quelques jours avant la manche de tournois, l'utilisateur clic sur la manche précédemment créer. Il doit maintenant cocher les pilotes qui seront présents à cette manche. Après quoi celui-ci peut cliquer sur "Générer" et valider son choix afin de que le logiciel prépare la structure des courses pour la manche.
- L'un des pilotes se désiste avant le début de la manche, l'utilisateur peut décocher son nom et régénérer la structure des courses.
- De même l'un des pilotes non prévus pour la manche du jour est présent. Celui-ci peut être ajouté avant le début de la manche. Attention, la génération de la structure des courses écrase toutes structures et résultats précédents pour ladite manche.
- Avant chaque race, l'utilisateur vérifie les couloirs de départ des pilotes dans le tableau de race.
- Après chaque race, l'utilisateur fournit l'ordre d'arrivée des pilotes dans le tableau de race et valide la race.
- Toutes les races d'une phase ont été courus, l'utilisateur se rend sur la page listant les phases et génère la phase suivante débloquent ainsi l'accès aux races de la phase suivante.
- En fin de manche, l'utilisateur télécharge la fiche de score au format PDF et l'imprime.

# Développements restants

## Nécessaires

### Gestion des championnats départementaux

La gestion des championnats régionaux est terminée, mais nous n'avons pas eu le temps d'implémenter les championnats départementaux.

Le principe est le même pour les deux types de championnats, seule la structure des courses est différente. Il est donc possible de s'inspirer des fonctions présentes pour les championnats régionaux afin de les implémenter pour les championnats départementaux.

### Affichage des scores entre races

Lors d'une réunion avec monsieur Delannoy, celui-ci nous a fait comprendre la nécessité de pouvoir récupérer le tableau des scores entre chaque race afin de pouvoir faciliter la transition avec le logiciel et de vérifier sa véracité.

Il faudra pour cela ajouter un bouton sur la page des races afin de télécharger un récapitulatif des points au format PDF.

### Téléchargement des résultats d'étapes/championnats

De la même manière que pour les scores entre races, il faudra ajouter la possibilité de télécharger la feuille de score à la fin de chaque étape et championnats. Il faudra aussi ajouter un bouton qui lancera un téléchargement au format PDF.

## Envisageable

### Import des titulaires

Afin de faciliter l'entrée des titulaires dans la base de données, le représentant de l'UFOLEP demande la possibilité d'importer une liste de titulaires au format Excel tous les ans.

Bien que non-nécessaire au fonctionnement du logiciel, cette fonctionnalité facilitera grandement son utilisation.

### Modification des titulaires

L'une de nos premières idées a été d'offrir la possibilité de modifier les titulaires directement via leur liste. En rendant chaque case du tableau modifiable et en ajoutant un event "onchange" il est facile de réaliser un appel API afin d'enregistrer les modifications.

### Bouton de sélection de tous les titulaires

Lors de la sélection des titulaires participants à une étape de championnat, l'ajout d'une case cochant tous les titulaires rendrait la saisie de données plus rapide et plus simple pour les membres du club.