

Web architecture

Décembre 2019

SMITS Victor 16107

Ecole centrale des arts et métiers

MIN1

Année académique 2019-2020

Table des matières

1	Spécification	5
1	Objectif du site	5
2	Acteurs	5
3	User Stories	5
3.1	Compatible Symfony-Twig et Symfony-Angular	5
3.2	Compatible Symfony-Angular uniquement	6
4	Contraintes non fonctionnelles	6
5	Mockup	7
5.1	Login	7
5.2	Accueil	7
5.3	Inscription	8
5.4	Month	8
5.5	Admin - Session	9
5.6	Admin - NoHrefle session	9
5.7	Admin - Abonnement	10
2	Base de données	11
1	Diagramme Entité Relation	11
2	Diagramme Relationnel	12
3	Description des tables et des relations entre tables	12
4	Description précise des champs	13
4.1	Table : Person	13
4.2	Table : Session	13
4.3	Table : Type_Session	14
4.4	Table : inscription	14
4.5	Table : lien_person_type_session	14
3	MVC	15
1	User Stories compatible Angular - Twig	15
1.1	En tant qu'utilisateur, je dois pouvoir me connecter sur le site	15
1.2	En tant qu'utilisateur, je dois pouvoir me créer un compte sur le site	17
1.3	En tant qu'utilisateur, je dois pouvoir sélectionner le mois et l'année pour afficher les sessions qui m'intéressent	19
1.4	En tant qu'utilisateur, je dois pouvoir m'inscrire à une session un certain jour	20

1.5	En tant qu'utilisateur, je dois pouvoir être inscrit automa- tiquement à une session	21
1.6	En tant qu'utilisateur, je dois pouvoir me désinscrire à une session	22
1.7	En tant qu'utilisateur, je dois pouvoir voir qui est inscrit pour chaque session	22
1.8	En tant qu'utilisateur, je veux pouvoir voir le nombre de séances qui restent dans mon abonnement	24
1.9	En tant qu'utilisateur, je veux pouvoir modifier mon profil . .	25
1.10	En tant qu'administrateur, je dois pouvoir annuler une session	26
1.11	En tant qu'administrateur, je dois pouvoir créer une nouvelle session peu importe la date	28
1.12	En tant qu'administrateur, je dois pouvoir gérer les abon- nement des utilisateurs	31
4	API	32
1	Tableau recapitulatif	32
2	GET : /api	34
2.1	Requête	34
2.2	Réponse	34
3	GET : /api/month	35
3.1	Requête	35
3.2	Réponse	36
4	GET : /api/profile	37
4.1	Requête	37
4.2	Réponse	38
5	GET : /api/admin/abonnement	40
5.1	Requête	40
5.2	Réponse	40
6	GET : /api/TypeSession	43
6.1	Requête	43
6.2	Réponse	43
7	PUT : /api/admin/renewAbo	44
7.1	Requête	44
7.2	Réponse	45
8	PUT : /api/admin/editAbo	45
8.1	Requête	45
8.2	Réponse	46
9	PUT : /api/admin/cancel	47
9.1	Requête	47
9.2	Réponse	48
10	PUT : /api/admin/recreate	48
10.1	Requête	48
10.2	Réponse	49
11	PUT : /api/admin/autocreate	50

11.1	Requête	50
11.2	Réponse	50
12	PUT : /api/editProfile	51
12.1	Requête	51
12.2	Réponse	52
13	PUT : /api/TypeSession	53
13.1	Requête	53
13.2	Réponse	54
14	DELETE : /api/admin/session	55
14.1	Requête	55
14.2	Réponse	55
15	DELETE : /api/Desinscription	56
15.1	Requête	56
15.2	Réponse	56
16	DELETE : /api/TypeSession	57
16.1	Requête	57
16.2	Réponse	57
17	DELETE : /api/admin/user	58
17.1	Requête	58
17.2	Réponse	58
18	POST : /api/login	59
18.1	Requête	59
18.2	Réponse	59
19	POST : /api/admin/session	61
19.1	Requête	61
19.2	Réponse	61
20	POST : /api/Inscription	62
20.1	Requête	62
20.2	Réponse	63
21	POST : /api/TypeSession	63
21.1	Requête	63
21.2	Réponse	64
22	POST : /api/register	65
22.1	Requête	65
22.2	Réponse	66
5	Séparation backend frontend	67
1	User Stories compatible Angular uniquement	68
1.1	En tant qu'administrateur, je dois pouvoir créer un type de session	68
1.2	En tant qu'administrateur, je veux pouvoir modifier un type de session	71
1.3	En tant qu'administrateur, je veux pouvoir supprimer un type de session	73

1.4	En tant qu'administrateur, je veux pouvoir supprimer un utilisateur du système	74
1.5	En tant qu'administrateur, je dois pouvoir générer les sessions pour un nombre d'année	76
2	User Stories compatible Symfony-Angular	79
2.1	En tant qu'utilisateur, je dois pouvoir me connecter sur le site	79
2.2	En tant qu'utilisateur, je dois pouvoir me créer un compte sur le site	82
2.3	En tant qu'utilisateur, je dois pouvoir sélectionner le mois et l'année pour afficher les sessions qui m'intéressent	83
2.4	En tant qu'utilisateur, je dois pouvoir m'inscrire à une session un certain jour	83
2.5	En tant qu'utilisateur, je dois pouvoir être inscrit automatiquement à une session	84
2.6	En tant qu'utilisateur, je dois pouvoir me désinscrire à une session	87
2.7	En tant qu'utilisateur, je dois pouvoir voir qui est inscrit pour chaque session	88
2.8	En tant qu'utilisateur, je veux pouvoir voir le nombre de séances qui restent dans mon abonnement	89
2.9	En tant qu'utilisateur, je veux pouvoir modifier mon profil . .	90
2.10	En tant qu'administrateur, je dois pouvoir annuler une session	91
2.11	En tant qu'administrateur, je dois pouvoir créer une nouvelle session peu importe la date	91
2.12	En tant qu'administrateur, je dois pouvoir gérer les abonnements des utilisateurs	92
6	Conclusion	93
1	Objectifs atteints et non atteints	93
1.1	Compatibles Symfony-Twig et Symfony-Angular	93
1.2	Compatibles Symfony-Angular uniquement	95
2	Pistes d'amélioration	95
7	Annexe	97
1	Code source	97

1 | Spécification

1 Objectif du site

Le site doit permettre à des personnes de s'inscrire et/ou se désinscrire de leur groupe à des sessions de sport, en l'occurrence des sessions d'aquabike.

2 Acteurs

- Utilisateur : personne qui utilise le site et s'inscrit à des sessions de sport
- Administrateur : personne administrant le site et ayant tout contrôle sur les sessions de sport organisées.

3 User Stories

3.1 Compatible Symfony-Twig et Symfony-Angular

User Stories	Priority
En tant qu'utilisateur, je dois pouvoir me créer un compte sur le site	2
En tant qu'utilisateur, je dois pouvoir me connecter sur le site	2
En tant qu'utilisateur, je dois pouvoir sélectionner le mois et l'année pour afficher les sessions qui m'intéressent	2
En tant qu'utilisateur, je dois pouvoir m'inscrire à une session un certain jour	1
En tant qu'utilisateur, je dois pouvoir être inscrit automatiquement à une session	1
En tant qu'utilisateur, je dois pouvoir me désinscrire à une session	1
En tant qu'utilisateur, je dois pouvoir voir qui est inscrit pour chaque session	2

En tant qu'utilisateur, je veux pouvoir voir le nombre de séances qui restent dans mon abonnement	2
En tant qu'utilisateur, je veux pouvoir modifier mon profil	1
En tant qu'utilisateur, je veux pouvoir m'inscrire à une session uniquement si il me reste des séances à disposition.	2
En tant qu'administrateur, je dois pouvoir annuler une session	2
En tant qu'administrateur, je dois pouvoir créer une nouvelle session peu importe la date	2
En tant qu'administrateur, je dois pouvoir gérer les abonnements des utilisateurs	2

3.2 Compatible Symfony-Angular uniquement

En tant qu'administrateur, je dois pouvoir créer un type de session	2
En tant qu'administrateur, je dois pouvoir générer les sessions pour un nombre d'années	1
En tant qu'administrateur, je veux pouvoir supprimer un utilisateur du système	1
En tant qu'administrateur, je veux pouvoir gérer les types de sessions disponibles pour l'inscription automatique	2
En tant qu'administrateur, je veux pouvoir modifier un type de session	1
En tant qu'administrateur, je veux pouvoir supprimer un type de session	1

4 Contraintes non fonctionnelles

- Le site doit être "user-friendly" pour des personnes non adeptes de la technologie.
- Le site doit être "responsive" afin de pouvoir l'utiliser sur tous types de d'écran.

5 Mockup

5.1 Login

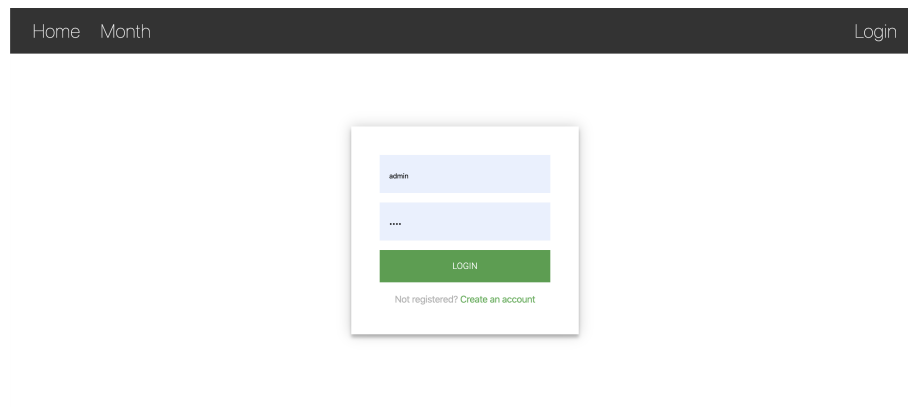


Figure 1.5.1: Page de connection

5.2 Accueil

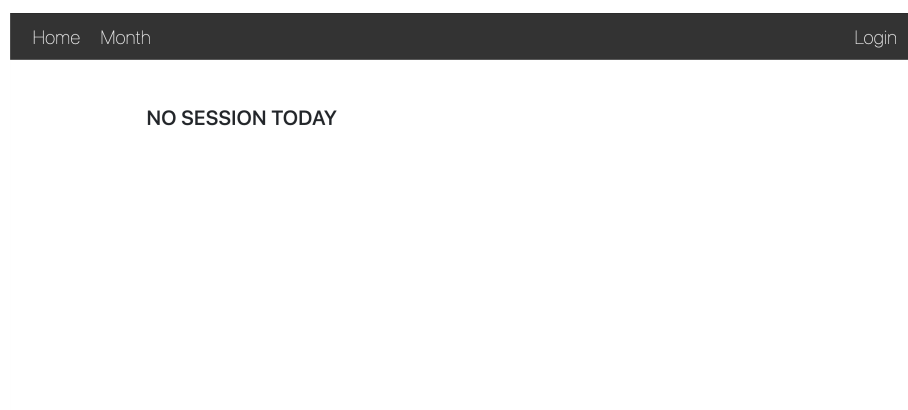
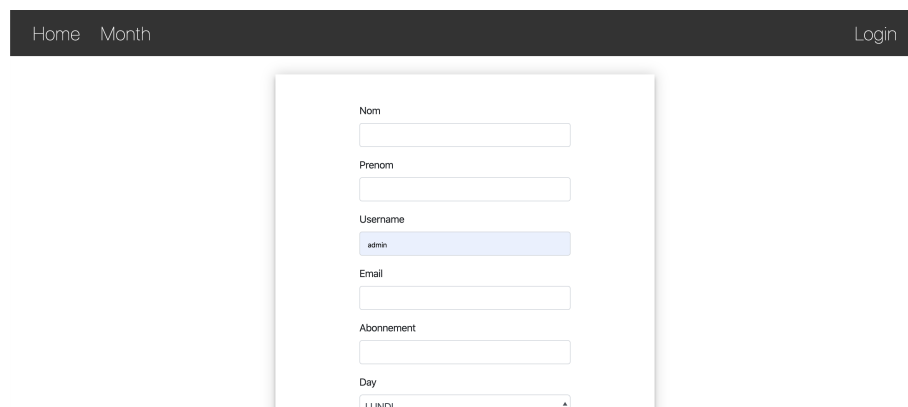


Figure 1.5.2: Page d'accueil

5.3 Inscription

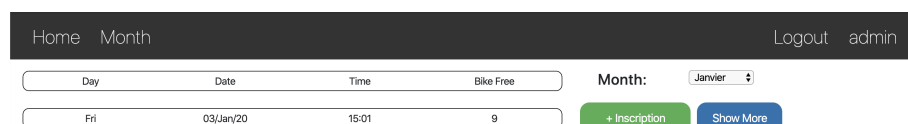


The registration form is displayed within a modal window. The header bar at the top contains 'Home' and 'Month' on the left, and 'Login' on the right. The form fields are as follows:

- Nom:
- Prenom:
- Username:
- Email:
- Abonnement:
- Day:

Figure 1.5.3: Page d'inscription au site

5.4 Month



The 'Month' page interface includes a header bar with 'Home' and 'Month' on the left, and 'Logout' and 'admin' on the right. Below the header, there is a table of sessions and a 'Month:' selector.

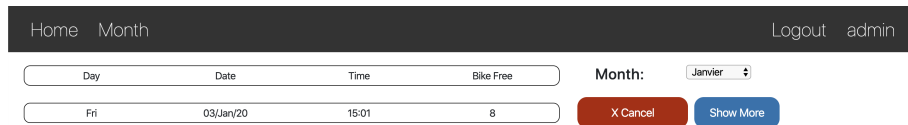
Day	Date	Time	Bike Free
Fri	03/Jan/20	15:01	9

Month:

[+ Inscription](#) [Show More](#)

Figure 1.5.4: Page d'affichage des sessions du mois

5.5 Admin - Session



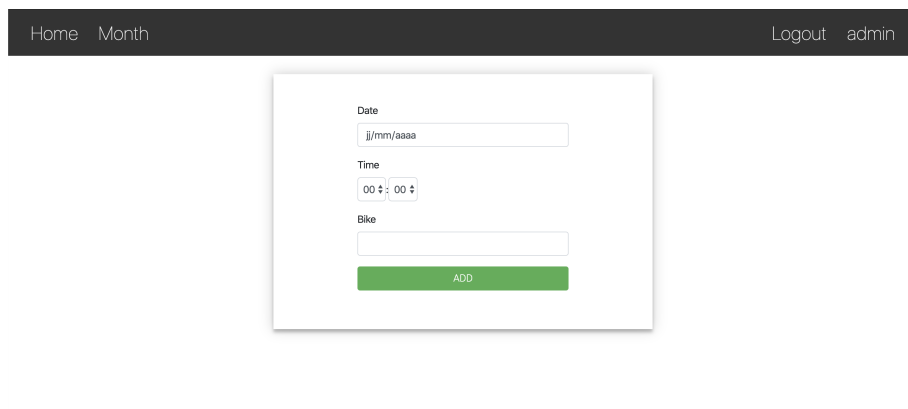
Day	Date	Time	Bike Free
Fri	03/Jan/20	15:01	8

Month: Janvier

X Cancel Show More

Figure 1.5.5: Page de gestion des sessions du mois pour l'administrateur

5.6 Admin - NoHrefle session



Date

jj/mm/aaaa

Time

00 : 00

Bike

ADD

Figure 1.5.6: Page de création d'une nouvelle session

5.7 Admin - Abonnement

Home Month				Logout admin	
Nom	Prenom	Email	Abonnement		
victor	smits	victor-smi@hotmail.fr	0	+ Renouvellement	

Figure 1.5.7: Page de gestion des abonnements des utilisateurs

2 | Base de données

1 Diagramme Entité Relation

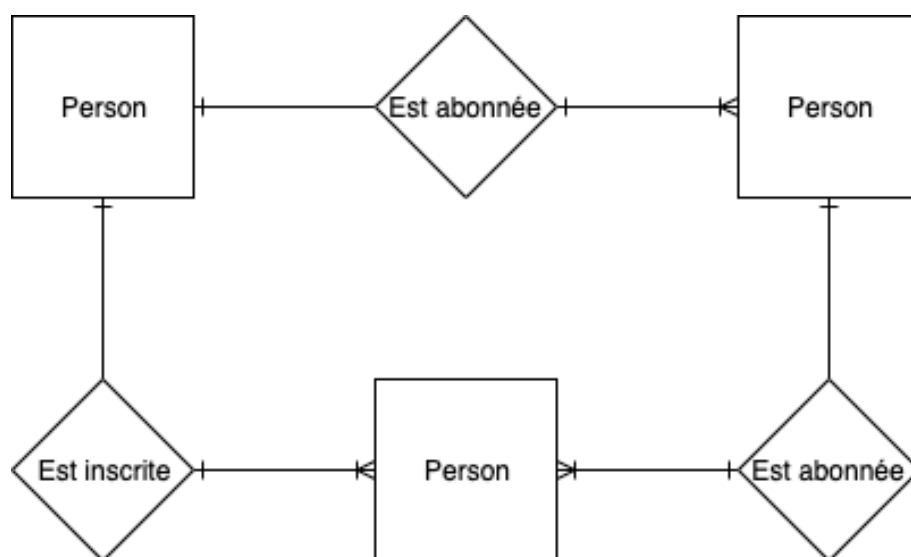


Figure 2.1.1: Diagramme Relationnel

2 Diagramme Relationnel

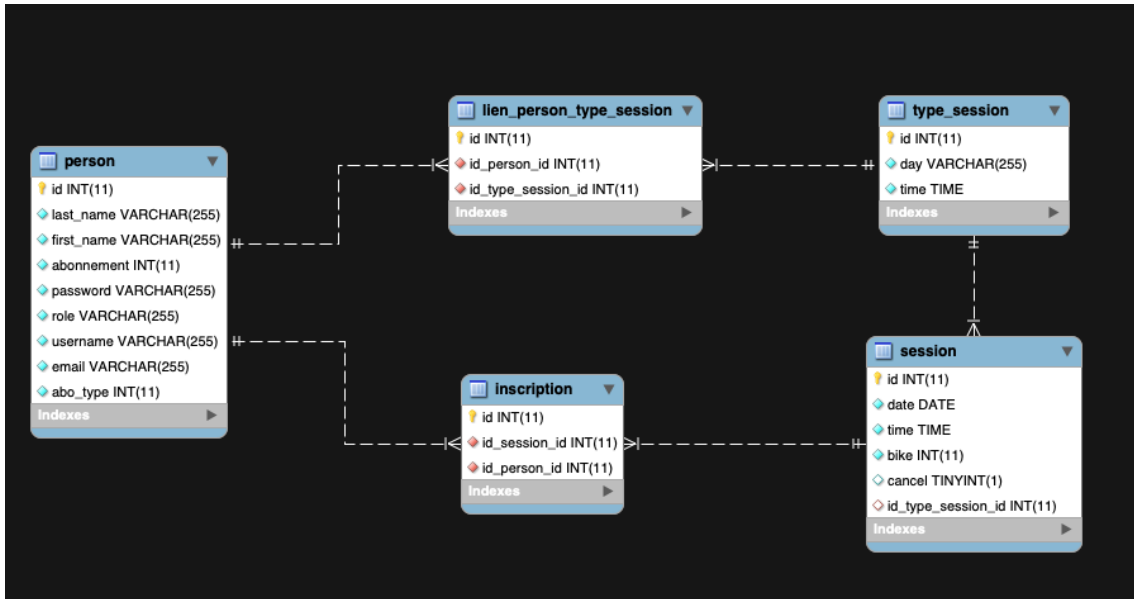


Figure 2.2.2: Diagramme Relationnel

3 Description des tables et des relations entre tables

Tables	Tables	Types	Description
Person	Inscription	1 à N	Une personne possède plusieurs inscriptions.
Session	Inscription	1 à N	Une session a plusieurs inscriptions.
Session	Type Session	N à 1	Plusieurs sessions correspondent à un type de sessions, un type de session peut correspondre à plusieurs sessions.
Person	Lien person type session	1 à N	Une Personne possède plusieurs liens, plusieurs liens corresponde à une personne.
Type Session	Lien person type session	1 à N	Un type de session possède plusieurs liens, plusieurs liens correspondent à un seul type de session.

4 Description précise des champs

4.1 Table : Person

Nom	Key	NN	Type	Référence	Intitulé	Commentaire
id	PK	O	int		Identifiant	
last_name		O	varchar(45)		Nom	
frist_name		O	varchar(45)		Prenom	
abonnement		O	int		Nombre de session restante	
password		O	varchar(45)		Mot de passe crypté	
role		O	varchar(45)		Role sur le site	Default = role_user
username		O	varchar(45)		Nom d'utilisateur	Unique
email		O	varchar(45)		Adresse Email	Unique
abo_type		O	int		Abonnement origine	

4.2 Table : Session

Nom	Key	NN	Type	Référence	Intitulé	Commentaire
id	PK	O	int		Identifiant	
date		O	DATE		date	
time		O	Time		heure	
bike		O	int		Nombre de vélos restants	
cancel		O	Boolean		Statut de la session	Default = Null
id type session	FK	O	int	id / typeSession	Lien avec le type de session	Default = Null

4.3 Table : Type_Session

Nom	Key	NN	Type	Référence	Intitulé	Commentaire
id	PK	O	int		Identifiant	
day		O	varchar(45)		code du jour	
time		O	Time		heure	

4.4 Table : inscription

Nom	Key	NN	Type	Référence	Intitulé	Commentaire
id	PK	O	int		Identifiant	
id_session	FK	O	int	id / session	lien avec la session	
id_person	FK	O	int	id / person	lien avec la person	

4.5 Table : lien_person_type_session

Nom	Key	NN	Type	Référence	IntitHref	Commentaire
id	PK	O	int		Identifiant	
id_type_session	FK	O	int	id / type_session	lien avec le type de session	
id_person	FK	O	int	id / person	lien avec la person	

3 | MVC

1 User Stories compatible Angular - Twig

Seul l'aspect graphique changera entre la version Twig et la version Angular. Pour plus de détails rendez-vous au chapitre 5

1.1 En tant qu'utilisateur, je dois pouvoir me connecter sur le site

1. L'utilisateur se rend sur la page de connection en cliquant sur *Login*
2. Il entre son nom d'utilisateur et son mot de passe.
3. Après vérification, l'utilisateur est redirigé vers la page d'accueil.

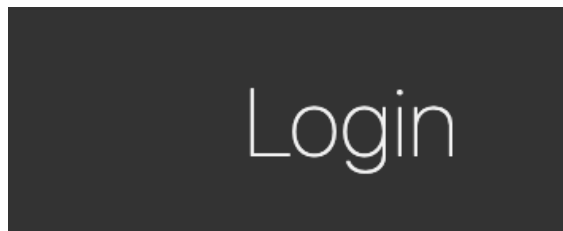


Figure 3.1.1: Bouton de connection

A login form with a white background and a subtle drop shadow. It contains three main elements: a light blue rectangular input field for the username containing the text 'admin', a second light blue rectangular input field for the password containing four dots '....', and a solid green rectangular button with the text 'LOGIN' in white. Below the button, the text 'Not registered? [Create an account](#)' is displayed in a smaller, grey font, with 'Create an account' being a green link.

Figure 3.1.2: Formulaire de connection

Gestion des erreurs et des risques

En cas d'erreur dans les informations de connection fournis par l'utilisateur, une erreur sera produite et affiché sur le site.

Diagramme de séquence

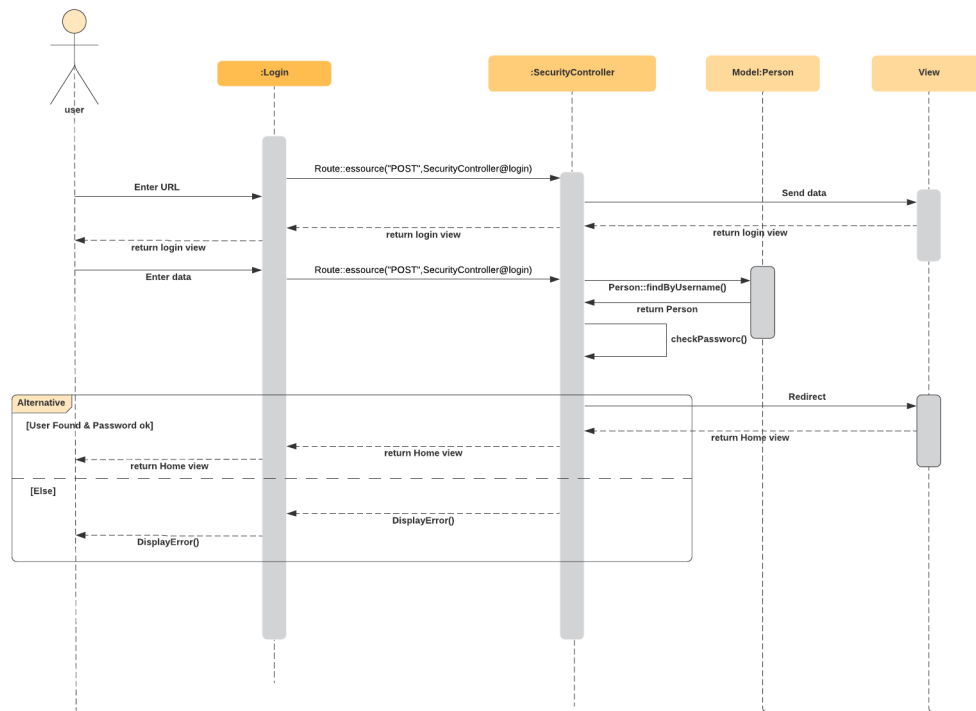


Figure 3.1.3: Diagramme de séquence de la connections d'un utilisateur

Scripts concernés

- URL: [login.html.twig](#)
- URL: [Person.php](#)
- URL: [SecurityController.php](#)

1.2 En tant qu'utilisateur, je dois pouvoir me créer un compte sur le site

1. L'utilisateur rentre ses information dans le formulaire.
2. L'utilisateur sélectionne le type de session à laquelle il est inscrit.
3. L'utilisateur clique sur *Register* et attend la redirection.

The image shows a registration form with the following fields and elements:

- Nom**: A text input field.
- Prenom**: A text input field.
- Username**: A text input field containing the value "admin".
- Email**: A text input field.
- Abonnement**: A text input field.
- Day**: A dropdown menu showing "LUNDI".
- Password**: A text input field containing four dots "....".
- Password**: A second text input field for password confirmation.
- REGISTER**: A green button at the bottom.

Figure 3.1.4: Formulaire d'inscription

Gestion des erreurs et sécurité

- L'utilisateur doit remplir tous les champs. Si un champ est manquant, une erreur apparaîtra.
- Le nom d'utilisateur et l'adresse Email sont uniques dans le système, s'il entre un Email ou nom d'utilisateur déjà existant, une erreur lui dira de corriger.
- L'utilisateur doit confirmer son mot de passe, si les mots de passe sont différents ou plus petits que 6 caractères il y aura un message d'erreur.

Diagramme de séquence

Scripts concernés

- URL: [RegistrationController.php](#)
- URL: [register.html.twig](#)
- URL: [Person.php](#)

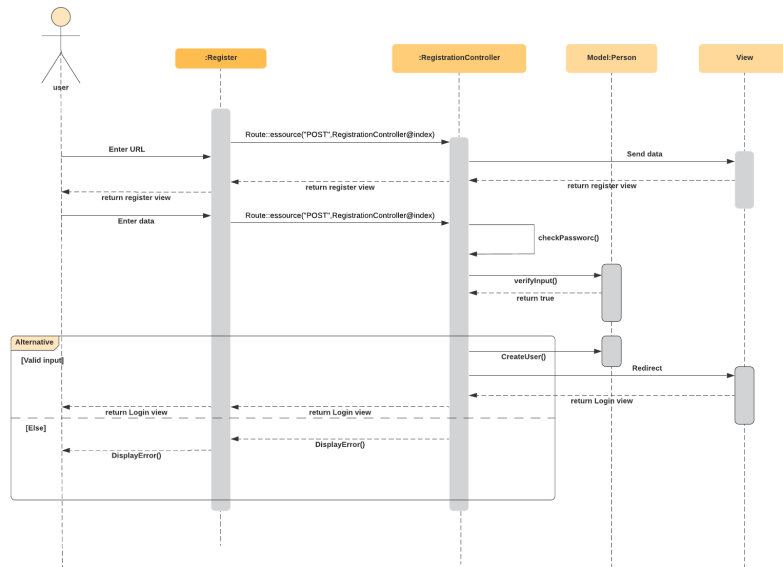


Figure 3.1.5: Diagramme de séquence de l'enregistrement d'un nouvelle utilisateur.

1.3 En tant qu'utilisateur, je dois pouvoir sélectionner le mois et l'année pour afficher les sessions qui m'intéressent

La sélection de l'année n'est pas disponible sur la version Symfony-Twig mais est implémenté dans la version Symfony-Angular.

1. L'utilisateur choisit le mois et l'année.
2. la liste se rafraîchit afin de correspondre à la selection effectuée.

Day	Date	Time	Bike Free	Month: Decembre
Mon	23/Dec/19	19:01	9	+ Inscription Show More
Mon	23/Dec/19	20:01	9	+ Inscription Show More
Tue	31/Dec/19	20:01	9	+ Inscription Show More

Figure 3.1.6: selection du mois et de l'année

1.4 En tant qu'utilisateur, je dois pouvoir m'inscrire à une session un certain jour

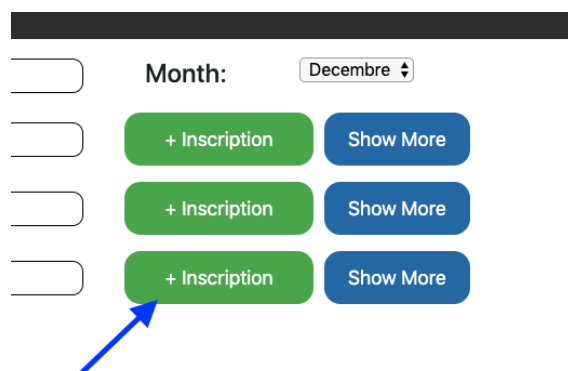


Figure 3.1.7: bouton d'inscription

1. l'utilisateur choisit la session correspondant à sa demande.
2. l'utilisateur clique sur le bouton *+ Inscription*

Gestion des erreurs

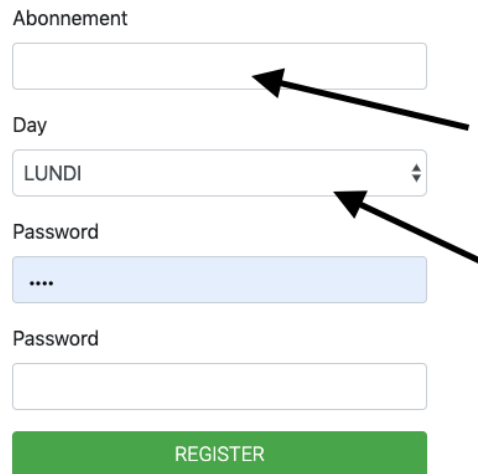
- Si l'utilisateur ne possède plus d'abonnement, une erreur sera affichée lui signalant le problème.
- Si la session est complète un message d'erreur sera affiché.

Scripts concerné

- URL: [MonthController.php](#)
- URL: [month.html.twig](#)
- URL: [Session.php](#)

1.5 En tant qu'utilisateur, je dois pouvoir être inscrit automatiquement à une session

L'inscription automatique n'est pas fonctionnelle dans la partie Symfony-Twig mais la possibilité de choisir son abonnement est disponible.



The image shows a registration form with the following elements:

- Abonnement**: A text input field.
- Day**: A dropdown menu currently showing "LUNDI".
- Password**: A password input field with masked characters (dots).
- Password**: A second, empty password input field.
- REGISTER**: A green button at the bottom.

Two black arrows point to the "Abonnement" and "Day" fields, indicating they are the focus of the automatic registration feature.

Figure 3.1.8: Selection pour l'inscription automatique

1. L'utilisateur introduit le nombre de sessions pour les quelles il paye.
2. Il selectionne les sessions pour les-quelles il souhaite s'inscrire.
3. Il clique sur *Register*.

Gestion des erreurs

Si une erreur arrive durant l'inscription, l'utilisateur ne sera pas redirigé et aura un message d'erreur.

Scripts concernés

- URL: [RegistrationController.php](#)
- URL: [register.html.twig](#)
- URL: [Person.php](#)
- URL: [Inscription.php](#)

1.6 En tant qu'utilisateur, je dois pouvoir me désinscrire à une session



Figure 3.1.9: Bouton de désinscription à une session

1. L'utilisateur trouve la session qui l'intéresse.
2. Le check vert indique qu'il est inscrit.
3. Il clique sur le bouton pour se désinscrire.

Scripts concernés

- URL: [MonthController.php](#)
- URL: [month.html.twig](#)
- URL: [Person.php](#)
- URL: [Inscription.php](#)

1.7 En tant qu'utilisateur, je dois pouvoir voir qui est inscrit pour chaque session

1. L'utilisateur trouve la session qui l'intéresse.
2. L'utilisateur clique sur le bouton *Show More* dans la dernière colonne.
3. Un tableau s'affiche avec la liste des inscrit(e)s.

Day	Date	Time	Bike Free
Mon	23/Dec/19	19:01	8

Month: Decembre

X Desinscription Show More

Figure 3.1.10: Bouton d'affichage

Nom	Prenom	Date	
Smits	Victor	23/Dec/19	
		Time	19:01
		Bike Free	8

Figure 3.1.11: Liste Participant(e)s

Scripts concernés

- URL: [MonthController.php](#)
- URL: [month.html.twig](#)
- URL: [Person.php](#)
- URL: [Inscription.php](#)

1.8 En tant qu'utilisateur, je veux pouvoir voir le nombre de séances qui restent dans mon abonnement

1. L'utilisateur clique sur son nom dans la barre de navigation.
2. L'utilisateur est redirigé vers sa page de profil.
3. L'utilisateur retrouve l'information dans le cadre affiché sur la page.



Figure 3.1.12: Bouton de navigation vers le profil

Day	Date	Time	Bike Free		
Thu	10/Oct/19	19:01	8	X Desinscription	Show More
Wed	16/Oct/19	19:01	8	X Desinscription	Show More
Wed	16/Oct/19	20:01	8	X Desinscription	Show More
Thu	17/Oct/19	19:01	7	X Desinscription	Show More
Thu	17/Oct/19	20:01	7	X Desinscription	Show More
Mon	23/Dec/19	19:01	8	X Desinscription	Show More

Nom

Prenom

Email

Abonnement

Jour

Edit Profile

Smits

Victor

victor-smi@hotmail.fr

13

Mon

Figure 3.1.13: Nombre de séances restantes dans l'abonnement

Scripts concernés

- URL: [ProfileController.php](#)
- URL: [profile.html.twig](#)
- URL: [Person.php](#)

1.9 En tant qu'utilisateur, je veux pouvoir modifier mon profil

1. L'utilisateur clique sur son nom dans la barre de navigation.
2. L'utilisateur clique sur le bouton edit en dessous de son profil.
3. Il est redirigé vers la page de modification.
4. L'utilisateur rentre les informations qu'il souhaite changer.
5. L'utilisateur confirme en cliquant sur *Edit*



Figure 3.1.14: Bouton de navigation vers le profil

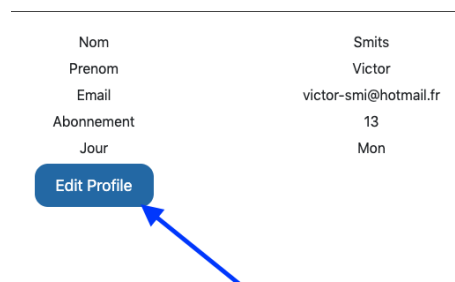
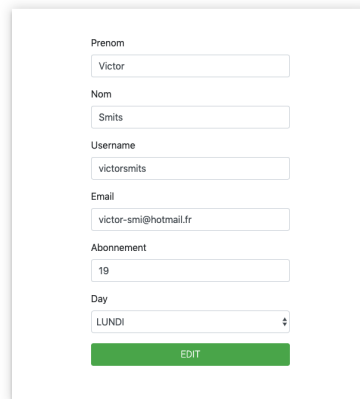


Figure 3.1.15: Bouton de modification du profil



Prenom
Victor

Nom
Smits

Username
victorsmits

Email
victor-smi@hotmail.fr

Abonnement
19

Day
LUNDI

EDIT

Figure 3.1.16: Fenêtre de modification des données du profil

Scripts concernés

- URL: [ProfileController.php](#)
- URL: [profile.html.twig](#)
- URL: [Person.php](#)

1.10 En tant qu'administrateur, je dois pouvoir annuler une session

1. L'administrateur se connecte au site avec ses identifiants.
2. Un nouveau bouton de navigation apparait dans la barre de navigation.
3. L'administrateur clique sur le bouton *Admin*
4. Il sélectionne *Session* dans le menu déroulant.
5. L'administrateur atterrit sur la page de gestion des sessions.
6. Il clique sur la croix correspondant à la session qu'il souhaite annuler.

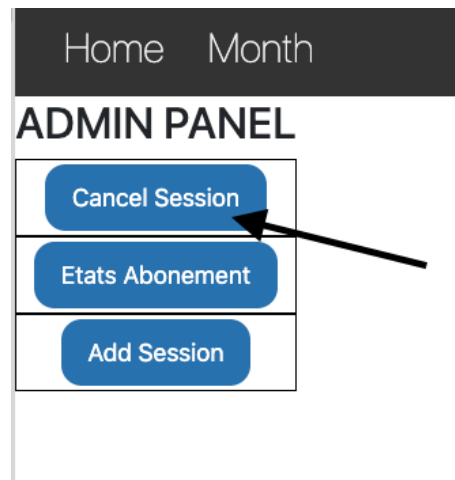


Figure 3.1.17: Bouton de navigation de l'administrateur

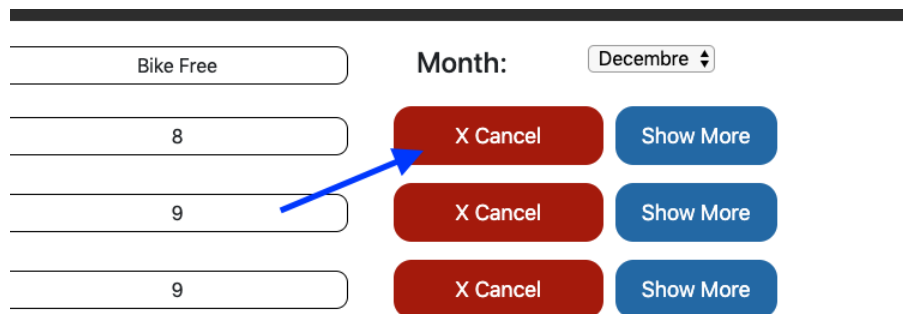


Figure 3.1.18: Bouton d'annulation de la session

Scripts concernés

- URL: [SessionAdministrationController.php](#)
- URL: [SessionAdmin.html.twig](#)
- URL: [Session.php](#)

1.11 En tant qu'administrateur, je dois pouvoir créer une nouvelle session peu importe la date

1. L'administrateur se connecte au site avec ses identifiants.
2. Un nouveau bouton de navigation apparait dans la barre de navigation.
3. L'administrateur clique sur le bouton *Admin*
4. Il sélectionne *Nouvelle Session* dans le menu déroulant.
5. L'administrateur atterrit sur la page d'ajout de sessions.
6. Il remplit le formulaire avec les bonnes informations.
7. Il clique sur *Add*

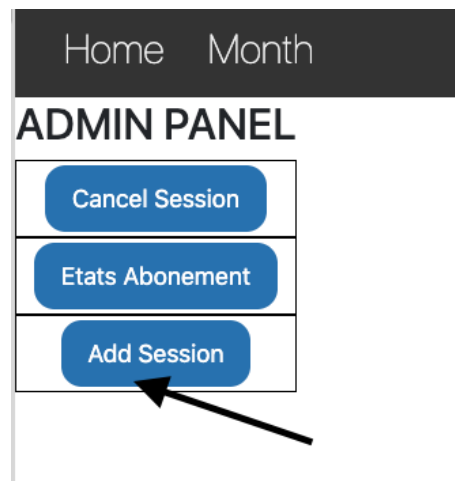
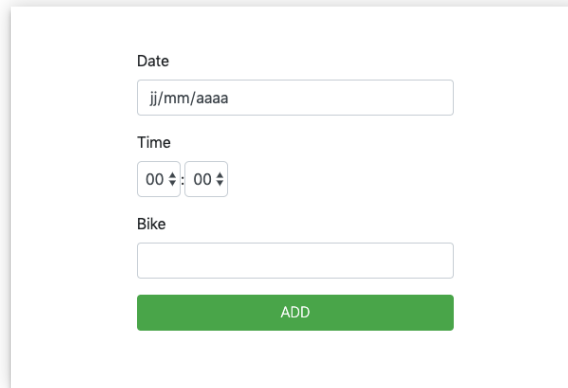


Figure 3.1.19: Panel de navigation de l'administrateur

The image shows a web form for adding a session. It is contained within a white box with a subtle drop shadow. At the top, there is a label 'Date' above a text input field with a placeholder 'jj/mm/aaaa'. Below this is a label 'Time' above two spinners, each showing '00' with up and down arrows. Underneath is a label 'Bike' above a text input field. At the bottom of the form is a green button with the text 'ADD' in white capital letters.

Date

jj/mm/aaaa

Time

00 : 00

Bike

ADD

Figure 3.1.20: Formulaire d'ajout d'une session

Gestion des erreurs

- Tous les champs doivent être remplis. Si ce n'est pas le cas, il sera impossible d'envoyer le formulaire et l'administrateur aura une erreur affichée.

Diagramme de séquence

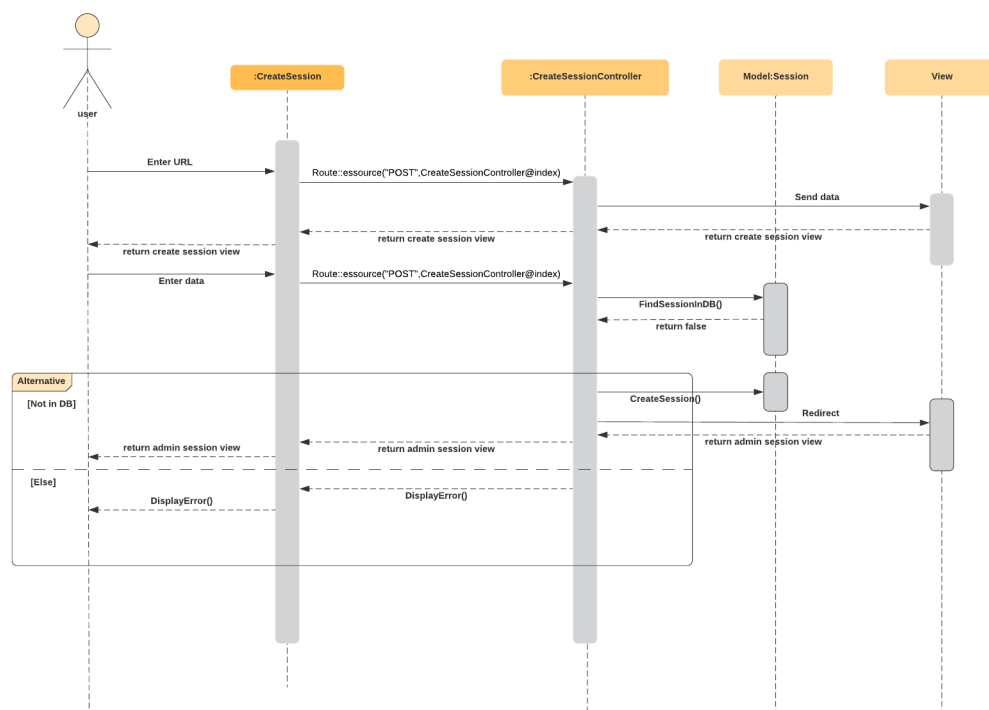


Figure 3.1.21: Diagramme de séquence de l'ajout d'une session

Scripts concerné

- URL: [Session.php](#)
- URL: [CreateSessionController.php](#)
- URL: [session.html.twig](#)

1.12 En tant qu'administrateur, je dois pouvoir gérer les abonnements des utilisateurs

1. L'administrateur se rend sur le panel admin.
2. Il sélectionne *Etats Abonnement* dans le menu.
3. L'administrateur atterrit sur la page de gestion des abonnements
4. Il clique sur *Renouveler* pour relancer l'abonnement d'un utilisateur.

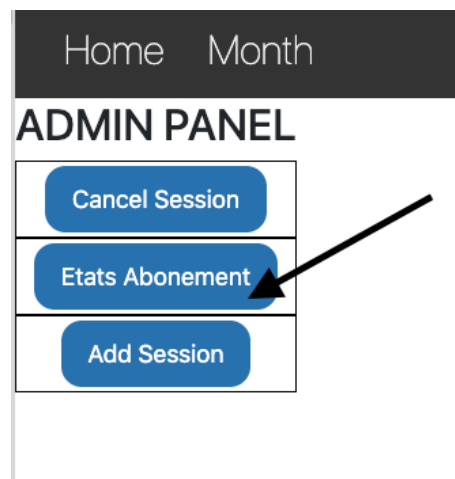


Figure 3.1.22: Bouton Etats Abonnement

Nom	Prénom	Email	Abonnement	
Smits	Victor	victor-sm@hotmail.fr	14	+ Renouvellement
Smits	Victor	victor-sm@hotmail.com	0	+ Renouvellement
smits	victor	16107@ecm.be	18	+ Renouvellement
Isabelle	Smits	test@test.com	20	+ Renouvellement

Figure 3.1.23: Bouton de renouvellement de l'abonnement

Scripts concernés

- URL: [Person.php](#)
- URL: [AbonnementController.php](#)
- URL: [abonnement.html.twig](#)

4 | API

1 Tableau recapitulatif

Méthode HTTP	Requête	Contrôleur	Méthode	Action
GET	/api	HomeControllerAPI	index	renvoie les informations des sessions d'aujourd'hui.
GET	/api/month/{month?}/{year?}	MonthControllerAPI	index	renvoie la liste des sessions du mois.
GET	/api/profile/{username?}	ProfileControllerAPI	index	Renvoie le profil de l'utilisateur.
GET	/api/admin/abonnement	AbonnementControllerAPI	index	renvoie la liste des utilisateur.
GET	/api/TypeSession	RegistrationControllerApi	getTypeSession	Renvoie la liste des types de session.
PUT	/api/admin/renewAbo	AbonnementControllerAPI	RenewAbo	renouvelle l'abonnement de l'utilisateur et l'inscrit à ses sessions.
PUT	/api/admin/editAbo	AbonnementControllerAPI	editAbo	Change l'abonnement de l'utilisateur.
PUT	/api/admin/Cancel	SessionAdministration-ControllerApi	cancel-Session	Annule la session et désinscrit les utilisateurs.
PUT	/api/admin/recreate	SessionAdministration-ControllerApi	recreate-Session	Recrée la session.
PUT	/api/admin/autocreate	CreateSessionControllerApi	createSession	Génère des sessions pour x années.

PUT	/api/editProfile	ProfileControllerApi	editProfile	Modifie les données de l'utilisateur.
PUT	/api/TypeSession	RegistrationControllerApi	editTypeSession	Modifie les données du type de session.
DELETE	/api/admin/session/{id?}	SessionAdministration-ControllerApi	deleteSession	Supprime la session.
DELETE	/api/Desinscription/{username?}/{id?}	MonthControllerAPI	remove-Inscription	Désinscrit l'utilisateur de la session.
DELETE	/api/TypeSession/{id?}	RegistrationControllerApi	Delete-TypeSession	Supprime le type de session.
DELETE	/api/admin/user/{id?}	AbonnementControllerApi	delAbo	Supprime l'utilisateur et ses références.
POST	/api/login	SecurityControllerApi	login	Authentifie l'utilisateur.
POST	/api/admin/session	CreateSessionControllerApi	index	Créer une nouvelle session.
POST	/api/Inscription	MonthControllerAPI	create-Inscription	Inscrit l'utilisateur à la session.
POST	/api/TypeSession	RegistrationControllerApi	setType-Session	Créer un nouveau type de session.
POST	/api/register	RegistrationControllerApi	register	Créer un nouvel utilisateur et l'inscrit a ses sessions.

2 GET : /api

Renvoie un JSON contenant toutes les sessions du jour ainsi que les inscriptions.

2.1 Requête

paramètre du path

Pas de Paramètre pour le path.

Content type

text/plain

Body

Pas de body pour la requête.

2.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
id	int	O	Id de la session.
Date	DateTime	O	Date de la session.
Time	DateTime	O	Heure de la session.
Bike	int	O	Nombre de vélos disponibles pour la session.
Cancel	Boolean	O	Etat de la session.
IdTypeSession	int	O	id du type de session correspondant.
idInscription	List	O	Liste des participants.
idPerson	Person	O	Identité du participant.
lastName	String	O	Nom du participant.
firstName	String	O	Prénom du participant.

Exemple JSON

```
1 {
2   id: 15,
3   Date: "2019/12/23 00:00",
4   time: "1970/01/01 19:00",
5   bike: 8,
6   Cancel: false,
7   IdTypeSession: 1,
8   idInscription: [{
9     0: {
10       id: 45,
11       idSession: 15,
12       idPerson: {
13         id: 49,
14         lastName: "Isabelle",
15         firstName: "Smits"
16       }
17     },
18   }]
19 }
```

Code

URL: [HomeControllerApi.php](#)

3 GET : /api/month

Renvoie une liste de JSON contenant toute les sessions du mois.

3.1 Requête

paramètre du path

Propriété	Type	Obligatoire	Description
Month	int	O	numéro du mois sélectionné.
Year	int	O	Année sélectionnée.

Content type

text/plain

Body

Pas de body pour la requête.

3.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
id	int	O	Id de la session.
Date	DateTime	O	Date de la session.
Time	DateTime	O	Heure de la session.
Bike	int	O	Nombre de vélos disponibles pour la session.
Cancel	Boolean	O	Etat de la session.
idInscription	List	O	Liste des participants.
idPerson	Person	O	Identité du participant.
lastName	String	O	Nom du participant.
firstName	String	O	Prénom du participant.

Exemple JSON

```
1 {
2   id: 15,
3   Date: "2019/12/23 00:00",
4   time: "1970/01/01 19:00",
5   bike: 8,
6   Cancel: false,
7   idInscription: [{
8     0: {
9       id: 45,
10      idSession: 15,
11      idPerson: {
12        id: 49,
13        lastName: "Isabelle",
14        firstName: "Smits"
15      }
16    },
17  ]
18 }
```

Code

URL: [MonthControllerApi.php](#)

4 GET : /api/profile

Renvoie un de JSON contenant toutes les informations de l'utilisateur

4.1 Requête

paramètre du path

Propriété	Type	Obligatoire	Description
Username	String	O	Nom d'utilisateur à afficher.

Content type

text/plain

Body

Pas de body pour la requête.

4.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
id	int	O	Id de l'utilisateur.
lastName	String	O	Nom du participant.
firstName	String	O	Prénom du participant.
Abonnement	int	O	Nombre de sessions restant dans l'abonnement.
UserName	String	O	Nom d'utilisateur.
Email	String	O	Email de l'utilisateur.
AboType	int	O	Type d'abonnement de l'utilisateur.
Role	String	O	Role donnant les accès a l'utilisateur.
idInscription	List<Session>	O	Liste des sessions auxquelles l'utilisateur est inscrit.
id	int	O	Id de la session.
Date	DateTime	O	Date de la session.
Time	DateTime	O	Heure de la session.
Bike	int	O	Nombre de vélos disponibles pour la session.
Cancel	Boolean	O	Etat de la session.

Exemple JSON

```
1 {  
2   id: 49,  
3   LastName: "Isabelle",  
4   FirstName: "Smits",  
5   Abonnement: 0,  
6   role: "ROLE_USER",  
7   Username: "isa",  
8   Email: "isa_smi@hotmail.com",  
9   AboType: 20,  
10  idInscription: [{  
11    0: {  
12      id: 45,  
13      idSession: {  
14        id: 15,  
15        date: "2019/12/23 00:00",  
16        time: "1970/01/01 19:00",  
17        bike: 8,  
18        cancel: false  
19      }  
20    }  
21  }]  
22 }
```

Code

URL: [ProfileControllerApi.php](#)

5 GET : /api/admin/abonnement

Renvoie une liste de JSON contenant toute les informations des l'utilisateurs.

5.1 Requête

paramètre du path

Pas de Paramètre pour le path.

Content type

text/plain

Body

Pas de body pour la requête.

5.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
id	int	O	Id de l'utilisateur.
lastName	String	O	Nom du participant.
firstName	String	O	Prénom du participant.
Abonnement	int	O	Nombre de sessions restant dans l'abonnement.
UserName	String	O	Nom d'utilisateur.
Email	String	O	Email de l'utilisateur.
AboType	int	O	Type d'abonnement de l'utilisateur.
Role	String	O	Role donnant les accès a l'utilisateur.
idTypeSession	List<LienPerson- TypeSession>	O	Liste des sessions auxquelles l'utilisateur est inscrit.
id	int	O	Id du lien entre la personne et le type de session.
idPerson	int	O	Id de la personne.
IdTypeSession	JSON<TypeSession>	O	Type de session.
id	int	O	Id du type de session.
day	String	O	Jour du type de session.
Time	DateTime	O	Heure du type de session.

Exemple JSON

```
1 {
2   id: 49,
3   LastName: "Isabelle",
4   FirstName: "Smits",
5   Abonnement: 0,
6   role: "ROLE_USER",
7   Username: "isa",
8   Email: "isa_smi@hotmail.com",
9   AboType: 20,
10  idTypeSession: [{
11    0: {
12      id: 55
13      IdPerson: 1
14      IdTypeSession: {
15        id: 2
16        day: "Mon"
17        time: "1970/01/01 20:10"
18      }
19    }
20  }]
21 }
```

Code

URL: [AbonnementControllerApi.php](#)

6 GET : /api/TypeSession

Renvoie une liste de JSON contenant tous les types de sessions.

6.1 Requête

paramètre du path

Pas de Paramètre pour le path.

Content type

text/plain

Body

Pas de body pour la requête.

6.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
id	int	O	Id du type de session.
day	String	O	Jour du type de session.
Time	DateTime	O	Heure du type de session.
idTypeSession	List<LienPerson- TypeSession>	O	Liste des sessions auxquelles l'utilisateur est inscrit.
id	int	O	Id du lien entre la personne et le type de session.
idPerson	int	O	Id de la personne.
id	int	O	Id de l'utilisateur.
lastName	String	O	Nom du participant.
firstName	String	O	Prénom du participant.

Exemple JSON

```
1 {
2   id: 2
3   day: "Mon"
4   time: "1970/01/01 20:10"
5   idTypeSession: [{
6     0: {
7       id: 55
8       IdPerson: {
9         id: 1,
10        LastName: "Isabelle",
11        FirstName: "Smits"
12      }
13    }
14  }]
15 }
```

Code

URL: [ProfileControllerApi.php](#)

7 PUT : /api/admin/renewAbo

Requête PUT pour le renouvellement de l'abonnement d'un utilisateur.

7.1 Requête

paramètre du path

Pas de Paramètre pour le path.

Content type

application/json

Body

Propriété	Type	Obligatoire	Description
id	int	O	Id de l'utilisateur.

Exemple JSON

```
1 {  
2   id: 1  
3 }
```

7.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
result	Boolean	O	Status de la requête.

Exemple JSON

```
1 {  
2   result: true  
3 }
```

Code

URL: [AbonnementControllerApi.php](#)

8 PUT : /api/admin/editAbo

Requête PUT pour modifier l'abonnement d'un utilisateur.

8.1 Requête

paramètre du path

Pas de Paramètre pour le path.

Content type

text/plain

Body

Propriété	Type	Obligatoire	Description
id	int	O	Id de l'utilisateur.
FirstName	String	N	Prénom de l'utilisateur.
LastName	String	N	Nom de l'utilisateur.
aboType	int	O	Type d'abonnement de l'utilisateur.

Exemple JSON

```
1 {  
2   FirstName: "Victor"  
3   LastName: "Smits"  
4   Id: 1  
5   aboType: 0  
6 }
```

8.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
result	Boolean	O	Status de la requête.

Exemple JSON

```
1 {  
2   result: true  
3 }
```

Code

URL: [AbonnementControllerApi.php](#)

9 PUT : /api/admin/cancel

Requête PUT pour annuler une session et automatiquement désinscrire toute les personnes inscrites.

9.1 Requête

paramètre du path

Pas de Paramètre pour le path.

Content type

application/json

Body

Propriété	Type	Obligatoire	Description
id	int	O	Id de la session.

Exemple JSON

```
1 {  
2   id: 1  
3 }
```


9.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
result	Boolean	O	Status de la requête.

Exemple JSON

```
1 {  
2   result: true  
3 }
```

Code

URL: [AbonnementControllerApi.php](#)

10 PUT : /api/admin/recreate

Requête PUT pour recréer une session préalablement annulée.

10.1 Requête

paramètre du path

Pas de Paramètre pour le path.

Content type

application/json

Body

Propriété	Type	Obligatoire	Description
id	int	O	Id de la session.
bike	int	O	Nombre de vélos dans la session.

Exemple JSON

```
1 {  
2   id: 1,  
3   Bike : 9,  
4 }
```

10.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
result	Boolean	O	Statut de la requête.

Exemple JSON

```
1 {  
2   result: true  
3 }
```

Code

URL: [SessionAdministrationControllerApi.php](#)

11 PUT : /api/admin/autocreate

Requête PUT pour générer automatiquement les sessions, correspondant au type de sessions sélectionnées, sur x années.

11.1 Requête

paramètre du path

Pas de Paramètre pour le path.

Content type

application/json

Body

Propriété	Type	Obligatoire	Description
year	int	O	Nombre d'années pour la génération automatique.
bike	int	O	Nombre de vélos dans la session.
idTypeSession	List<int>	O	Liste des id des types de sessions à générer.

Exemple JSON

```
1 {  
2   Bike: 9  
3   year: 1  
4   idTypeSession: [1,2,3,4]  
5 }
```

11.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
result	Boolean	O	Status de la requête.

Exemple JSON

```
1 {  
2   result: true  
3 }
```

Code

URL: [CreateSessionControllerApi.php](#)

12 PUT : /api/editProfile

Requête PUT permettant à un utilisateur de modifier son profil.

12.1 Requête

paramètre du path

Pas de Paramètre pour le path.

Content type

application/json

Body

Propriété	Type	Obligatoire	Description
id	int	O	Id de l'utilisateur.
lastName	String	O	Nom de l'utilisateur.
firstName	String	O	Prénom de l'utilisateur.
password	String	O	Nouveau mot de passe de l'utilisateur.
typeSessions	List<TypeSession>	O	Liste des types de sessions auxquelles l'utilisateur est inscrit.
id	int	O	Id du type de session.
Day	String	O	Jour du type de session.
Time	String	O	Heure du type de session.

Exemple JSON

```
1 {
2   id: 1
3   lastName: "Smits"
4   firstName: "Victor"
5   Email: "victor-smi@hotmail.com"
6   password: null
7   typeSessions: [{
8     0: {
9       Id: 2
10      Day: "Lundi"
11      Time: "20:10"
12    }
13  }]
14 }
```

12.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
result	Boolean	O	Status de la requête.

Exemple JSON

```
1 {
2   result: true
3 }
```

Code

URL: [EditProfileControllerApi.php](#)

13 PUT : /api/TypeSession

Requête PUT pour modifier un type de session. Après modification, les sessions correspondant à l'ancien type de session sont supprimées. Pour pouvoir en créer de nouvelles, il faut passer par la génération automatique.

13.1 Requête

paramètre du path

Pas de Paramètre pour le path.

Content type

application/json

Body

Propriété	Type	Obligatoire	Description
id	int	O	Id de l'utilisateur.
Day	String	O	Jour du type de la session.
Time	String	O	Heure du type de la session.

Exemple JSON

```
1 {  
2   id: 1  
3   Day: "MON"  
4   Time: "20:10"  
5 }
```

13.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
result	Boolean	O	Statut de la requête.

Exemple JSON

```
1 {  
2   result: true  
3 }
```

Code

URL: [RegistrationControllerApi.php](#)

14 DELETE : /api/admin/session

Requête permettant d'effacer une session du système.

14.1 Requête

paramètre du path

Propriété	Type	Obligatoire	Description
id	int	O	id de la session à effacer.

Content type

text/plain

Body

Pas de body pour la requête.

14.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
result	Boolean	O	Statut de la requête.

Exemple JSON

```
1 {  
2   result: true  
3 }
```

Code

URL: [SessionAdministrationControllerApi.php](#)

15 DELETE : /api/Desinscription

Requête permettant de désinscrire un utilisateur d'une session.

15.1 Requête

paramètre du path

Propriété	Type	Obligatoire	Description
username	String	O	Nom d'utilisateur.
id	int	O	Id de la session.

Content type

text/plain

Body

Pas de body pour la requête.

15.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
result	Boolean	O	Statut de la requête.

Exemple JSON

```
1 {  
2   result: true  
3 }
```

Code

URL: [MonthControllerApi.php](#)

16 DELETE : /api/TypeSession

Requête permettant de supprimer un type de session et, automatiquement, les sessions qui y sont liées.

16.1 Requête

paramètre du path

Propriété	Type	Obligatoire	Description
id	int	O	Id du type de session.

Content type

text/plain

Body

Pas de body pour la requête.

16.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
result	Boolean	O	Statut de la requête.

Exemple JSON

```
1 {  
2   result: true  
3 }
```

Code

URL: [RegistrationControllerApi.php](#)

17 DELETE : /api/admin/user

Requête permettant de supprimer un utilisateur et, automatiquement, le désinscrire des sessions qui lui sont liées.

17.1 Requête

paramètre du path

Propriété	Type	Obligatoire	Description
id	int	O	Id de l'utilisateur.

Content type

text/plain

Body

Pas de body pour la requête.

17.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
result	Boolean	O	Statut de la requête.

Exemple JSON

```
1 {  
2   result: true  
3 }
```

Code

URL: [AbonnementControllerApi.php](#)

18 POST : /api/login

Requête POST permettant à un utilisateur de se connecter au site.

18.1 Requête

paramètre du path

Pas de Paramètre pour le path.

Content type

application/json

Body

Propriété	Type	Obligatoire	Description
Username	String	O	Nom d'utilisateur servant à la connexion.
password	String	O	Mot de passe de l'utilisateur.

Exemple JSON

```
1 {  
2   Username: "victor-smi@hotmail.com"  
3   password: "0000"  
4 }
```

18.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
result	Boolean	O	Statut de la requête.

Exemple JSON

```
1 {  
2   result: true  
3 }
```

Code

URL: [SecurityControllerApi.php](#)

19 POST : /api/admin/session

Requête POST permettant l'ajout d'une nouvelle session dans le système.

19.1 Requête

paramètre du path

Pas de Paramètre pour le path.

Content type

application/json

Body

Propriété	Type	Obligatoire	Description
id	int	O	Id de l'utilisateur.
Date	String	O	Date de la session.
Time	String	O	Heure de la session.
Bike	int	O	Nombre de vélos disponibles.
Cancel	Boolean	O	Etat de la session.

Exemple JSON

```
1 {  
2   id: 0,  
3   Day: "Mon Dec 23 2019",  
4   Time: "20:10",  
5   Bike : 9,  
6   Cancel : false  
7 }
```

19.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
result	Boolean	O	Status de la requête.

Exemple JSON

```
1 {  
2   result: true  
3 }
```

Code

URL: [CreateSessionControllerApi.php](#)

20 POST : /api/Inscription

Requête POST permettant d'ajouter une nouvelle session dans le système.

20.1 Requête

paramètre du path

Pas de Paramètre pour le path.

Content type

application/json

Body

Propriété	Type	Obligatoire	Description
id	int	O	Id de la session.
Username	String	O	Nom d'utilisateur.

Exemple JSON

```
1 {  
2   Username : "admin",  
3   Id : 262  
4 }
```

20.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
result	Boolean	O	Status de la requête.

Exemple JSON

```
1 {  
2   result: true  
3 }
```

Code

URL: [CreateSessionControllerApi.php](#)

21 POST : /api/TypeSession

Requête POST permettant d'ajouter un nouveau type de session.

21.1 Requête

paramètre du path

Pas de Paramètre pour le path.

Content type

application/json

Body

Propriété	Type	Obligatoire	Description
id	int	O	Id de la session.
Day	String	O	Jour du type de la session.
Time	String	O	Heure du type de la session.

Exemple JSON

```
1 {  
2   id: 1  
3   Day: "MON"  
4   Time: "20:10"  
5 }
```

21.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
result	Boolean	O	Statut de la requête.

Exemple JSON

```
1 {  
2   result: true  
3 }
```

Code

URL: [RegistrationControllerApi.php](#)

22 POST : /api/register

Requête POST permettant l'enregistrement d'un nouvel utilisateur dans le système.

22.1 Requête

paramètre du path

Pas de Paramètre pour le path.

Content type

application/json

Body

Propriété	Type	Obligatoire	Description
email	String	O	Email de l'utilisateur.
username	String	O	Nom d'utilisateur.
lastName	String	O	Nom de l'utilisateur.
firstName	String	O	Prénom de l'utilisateur.
Abonnement	int	O	Nombre de sessions disponibles dans son abonnement.
password	String	O	mot de passe de l'utilisateur.
passwordConfirmation	String	O	vérification du mot de passe de l'utilisateur.
typeSession	List<TypeSession>	O	Liste des types de session auxquelles l'utilisateur s'est inscrit.
id	int	O	Id du type de session.
Day	String	O	Jour du type de session.
Time	String	O	Heure du type de session.

Exemple JSON

```
1 {
2   username : "victorsmits",
3   lastName: "Smits",
4   firstName: "Victor",
5   Email: "victor-smi@hotmail.com",
6   password: "123456",
7   passwordConfirmation : "123456",
8   typeSessions: [{
9     0: {
10      Id: 2
11      Day: "Lundi"
12      Time: "20:10"
13    }
14  }]
15 }
```

22.2 Réponse

Réussite

Retourne code 200

Content-type

application/json

Body

Propriété	Type	Obligatoire	Description
result	Boolean	O	Statut de la requête.

Exemple JSON

```
1 {
2   result: true
3 }
```

Code

URL: [RegistrationControllerApi.php](#)

5 | Séparation backend frontend

Pour la séparation backend-frontend nous avons utilisé l'api REST qui nous permet de séparer les actions sur la base de données et l'interface de l'application web tout en permettant la transmission des données grâce au format JSON. L'ensemble des requêtes réalisées ci-dessous ainsi que leurs explications sont expliquées dans le chapitre 4.

La majorité des User Stories sont comparables et applicables aux 2 architectures, cependant certaines de ces User Stories sont implémentées uniquement dans la partie Angular. Ces User Stories sont expliquées et illustrées ci-dessous.

1 User Stories compatible Angular uniquement

C'est User Stories on été ajouter part après la création du twig, soit pour faciliter l'exécution d'élément, soit dans le but d'ajouter de nouvelle fonctionnalité.

1.1 En tant qu'administrateur, je dois pouvoir créer un type de session

1. L'administrateur se connecte au site avec ses identifiant.
2. Un nouveau bouton de navigation apparait dans la bar de navigation.
3. L'administrateur clique sur le bouton *Admin*
4. Il sélectionne *Type Session* dans le menu déroulant.
5. L'administrateur atterris sur la page de gestion des type de sessions.
6. Il clique sur *Ajout d'une session +*
7. Un formulaire apparait et il le remplis avec les bonne information (jour - heure [hh:mm])
8. Il clique sur *ok*



Figure 5.1.1: Bouton de navigation de l'administrateur

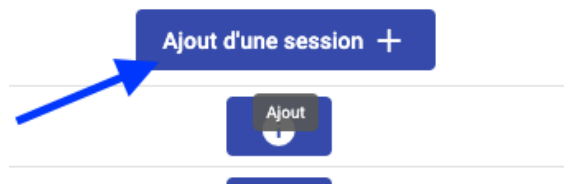


Figure 5.1.2: Bouton d'ajout d'un type de session

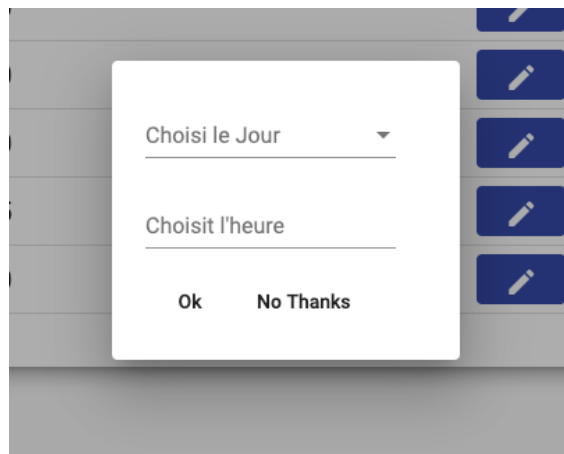


Figure 5.1.3: Page d'ajout du type de session

Gestion des erreurs

- 2 types de sessions identiques ne peuvent exister, lors de la creation d'un type de session, une vérification se fait et, si le type de session existe deja, une erreur apparait.
- Si une erreur apparait durant la création du type de session, elle s'affichera au-dessus du formulaire.

Diagramme de séquence

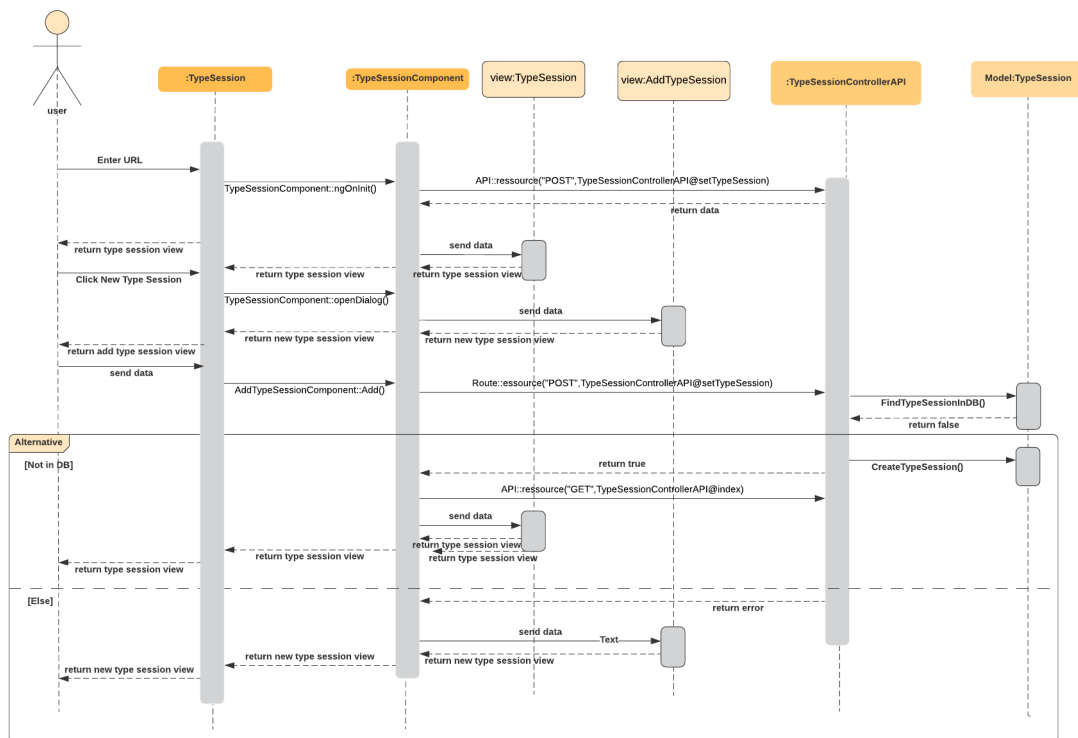


Figure 5.1.4: Diagramme de séquence de l'ajout d'un type de session

Script concerné

- URL: [RegistrationControllerApi.php](#)
- URL: [TypeSession.php](#)
- URL: [add-type-session.component.ts](#)
- URL: [add-type-session.component.html](#)

1.2 En tant qu'administrateur, je veux pouvoir modifier un type de session

1. L'administrateur se connecte au site avec ses identifiants.
2. Un nouveau bouton de navigation apparait dans la barre de navigation.
3. L'administrateur clique sur le bouton *Admin*
4. Il sélectionne *Type Session* dans le menu déroulant.
5. L'administrateur atterrit sur la page de gestion des types de sessions.
6. Il clique sur le bouton edit
7. Un formulaire apparait et il le remplit avec les bonnes informations (jour - heure [hh:mm])
8. Il clique sur *ok*

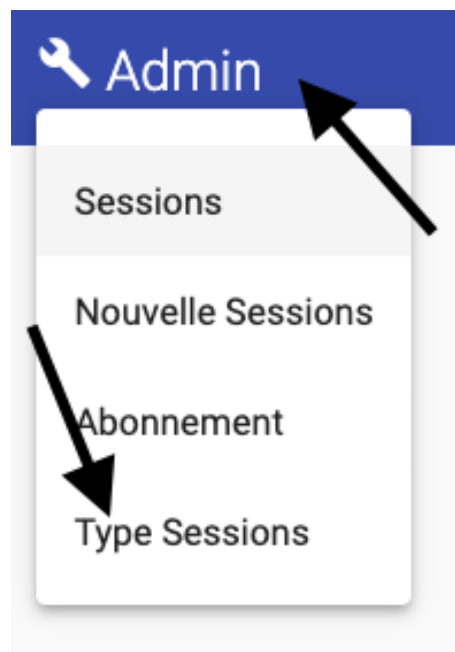


Figure 5.1.5: Menu de l'administrateur

Jour	Heure	Modification / Suppression	Ajout d'une session +
Lundi	19:00	 	
Lundi	20:10	 	

Figure 5.1.6: Bouton de modification du type de session

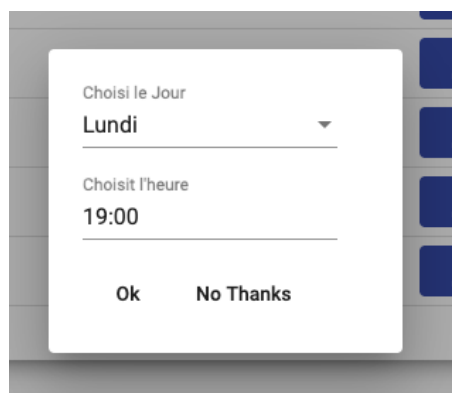


Figure 5.1.7: Formulaire de modification du type de session

Gestion des erreurs

Chaque type de session est unique, si l'administrateur rentre des informations correspondant à un type de session déjà existant, une erreur apparaîtra au dessus du formulaire.

Diagramme de séquence

Le diagramme de séquence de la modification d'un type de session est identique à celui de la création de session. Seule la première méthode appelé dans le backend change pour devenir *editTypeSession*.

Scripts concernés

- URL: [RegistrationControllerApi.php](#)
- URL: [TypeSession.php](#)
- URL: [edit-type-session.component.ts](#)
- URL: [edit-type-session.component.html](#)

1.3 En tant qu'administrateur, je veux pouvoir supprimer un type de session

1. L'administrateur se connecte au site avec ses identifiants.
2. Un nouveau bouton de navigation apparait dans la barre de navigation.
3. L'administrateur clique sur le bouton *Admin*
4. Il sélectionne *Type Session* dans le menu déroulant.
5. L'administrateur atterrit sur la page de gestion des types de sessions.
6. Il clique sur le bouton supprimer
7. Une fenêtre de confirmation apparait.
8. Il clique sur *OUI!*

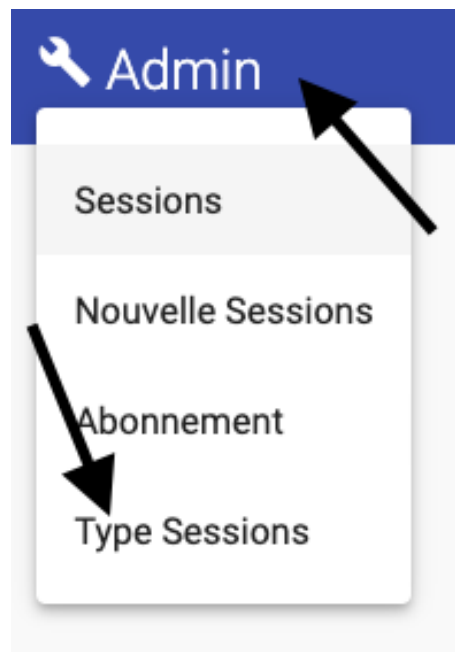


Figure 5.1.8: Menu de l'administrateur







Jour	Heure	Modification / Suppression	Ajout d'une session +
Lundi	19:00	 	
Lundi	20:10	 	

Figure 5.1.9: Bouton de suppression du type de session

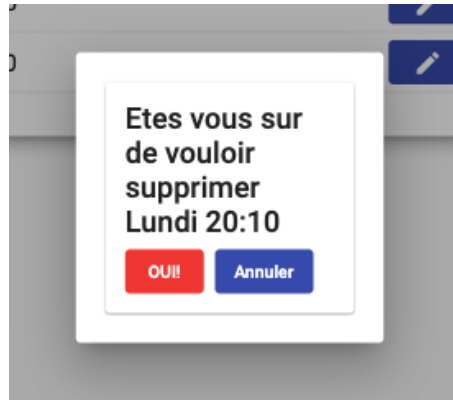


Figure 5.1.10: Fenêtre de confirmation de suppression

Scripts concernés

- URL: [RegistrationControllerApi.php](#)
- URL: [TypeSession.php](#)
- URL: [del-type-session.component.ts](#)
- URL: [del-type-session.component.html](#)

1.4 En tant qu'administrateur, je veux pouvoir supprimer un utilisateur du système

1. L'administrateur se connecte au site avec ses identifiants.
2. Un nouveau bouton de navigation apparait dans la barre de navigation.
3. L'administrateur clique sur le bouton *Admin*
4. Il sélectionne *Abonnement* dans le menu déroulant.
5. L'administrateur atterrit sur la page de gestion des types des utilisateurs.
6. Il clique sur le bouton supprimer
7. Une fenêtre de confirmation apparait.
8. Il clique sur *OUI!*

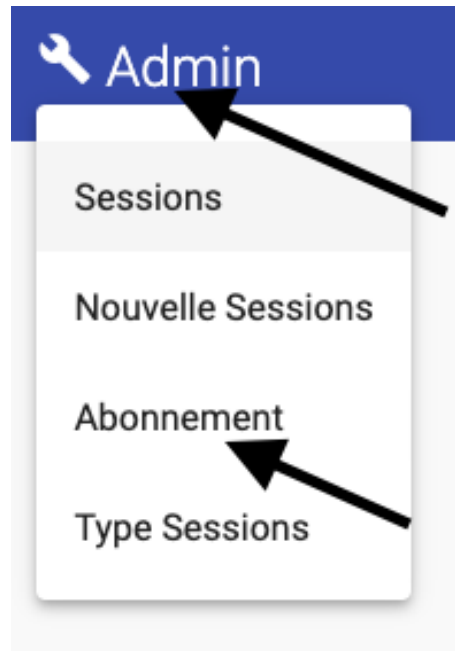


Figure 5.1.11: Menu de l'administrateur

Nom	Prenom	Abonnement	Type Abonnement	
Smits	Victor	0	0	  
Isabelle	Smits	0	20	  

Figure 5.1.12: Bouton de suppression de l'utilisateur



Figure 5.1.13: Fenêtre de confirmation de suppression

Scripts concernés

- URL: [AbonnementControllerApi.php](#)
- URL: [Person.php](#)
- URL: [del-abo.component.ts](#)
- URL: [del-abo.component.html](#)

1.5 En tant qu'administrateur, je dois pouvoir générer les sessions pour un nombre d'année

1. L'administrateur se connecte au site avec ses identifiants.
2. Un nouveau bouton de navigation apparait dans la barre de navigation.
3. L'administrateur clique sur le bouton *Admin*
4. Il sélectionne *Nouvelle session* dans le menu déroulant.
5. L'administrateur atterrit sur la page de création de session.
6. Il clique sur *Génération auto*
7. Un formulaire lui demandant sur combien d'années et le type de session il souhaite générer apparait.
8. Il sélectionne les types de session qu'il souhaite générer.
9. Il indique le nombre d'années voulus
10. Il clique sur *Je confirme la génération*

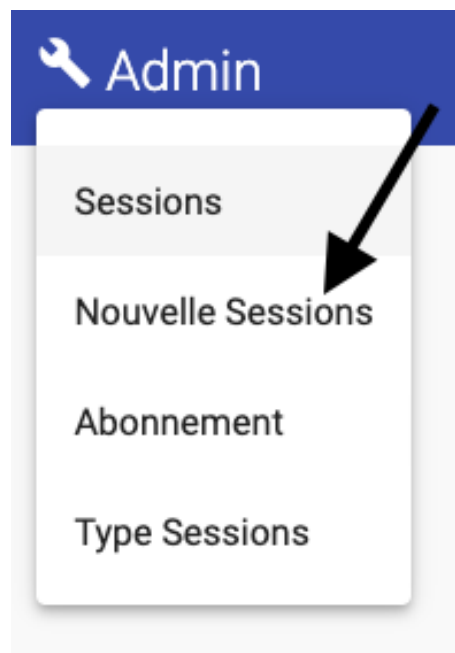


Figure 5.1.14: Menu de l'administrateur

Choose a date *

Pick a time *

Number of bike *

Add

Generation auto

Figure 5.1.15: Bouton de génération automatique

Génération automatique de sessions

- ☐ Lundi 20:10
- ☐ Mercredi 09:00
- ☐ Mercredi 10:10
- ☐ Jeudi 17:30
- ☐ Jeudi 18:45
- ☐ Jeudi 19:50
- ☐ Lundi 19:00
- ☐ Lundi 10:00

nombre d'année

je confirme la génération

annuler

Figure 5.1.16: Formulaire pour la génération automatique de sessions

Gestion des erreurs

- La génération automatique suppose que tous les types de session on été préalablement configurés. Si ce n'est pas le cas, une erreur apparaîtra à la confirmation.
- Si une erreur se crée durant la génération, elle est interrompue et un message d'erreur est affiché à l'administrateur.

Diagramme de séquence

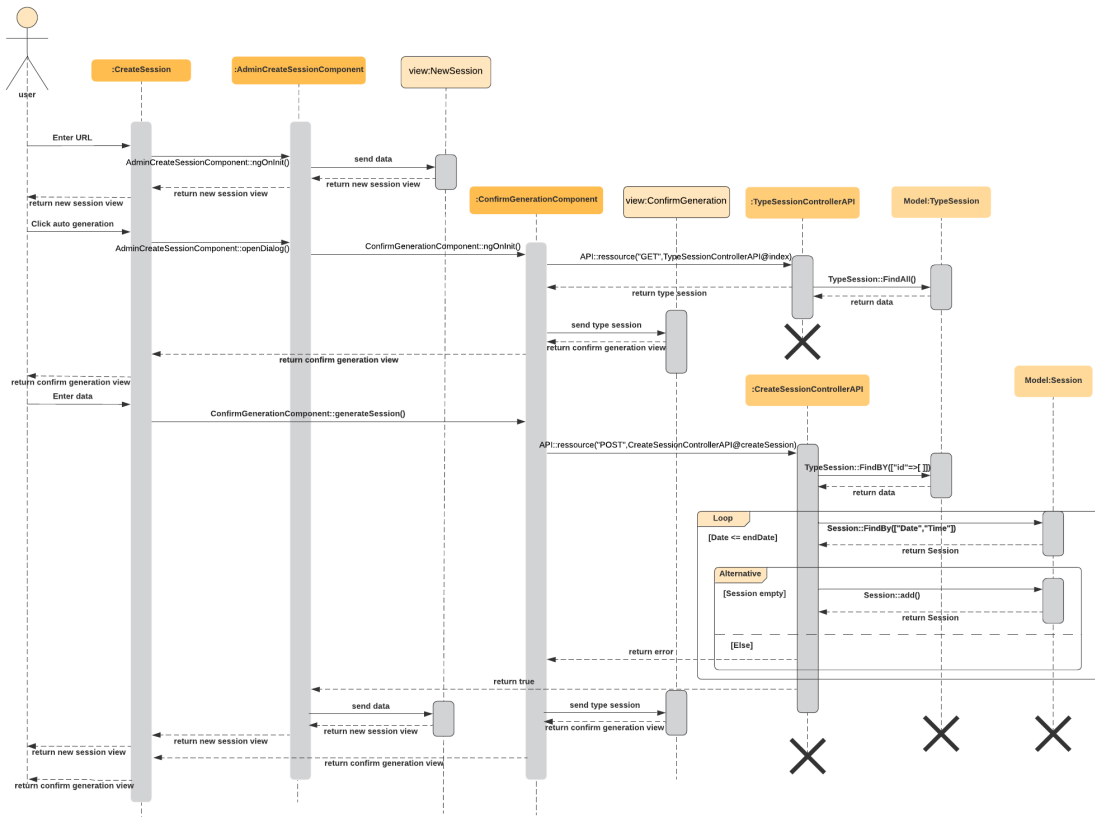


Figure 5.1.17: Diagramme de séquence de la génération automatique de session

Scripts concernés

- URL: [Session.php](#)
- URL: [CreateSessionControllerApi.php](#)
- URL: [admin-create-session.component.ts](#)
- URL: [admin-create-session.component.html](#)

2 User Stories compatible Symfony-Angular

2.1 En tant qu'utilisateur, je dois pouvoir me connecter sur le site

Différences

- Réalisation d'une requête POST via l'api REST permettant d'effectuer l'authentification dans le backend.
- Différences au niveau de l'aspect graphique de la page de connection.
- Connection via l'adresse Email et non le nom d'utilisateur.
- Redirection automatique de l'utilisateur vers la page de connection lors de l'accès au site si l'utilisateur n'est pas connecté.
- Utilisation d'un LoginAuthenticator afin de pouvoir authentifier l'utilisateur dans le backend.
- Utilisation de cookies pour préserver les informations de l'utilisateur.

Diagramme de séquence

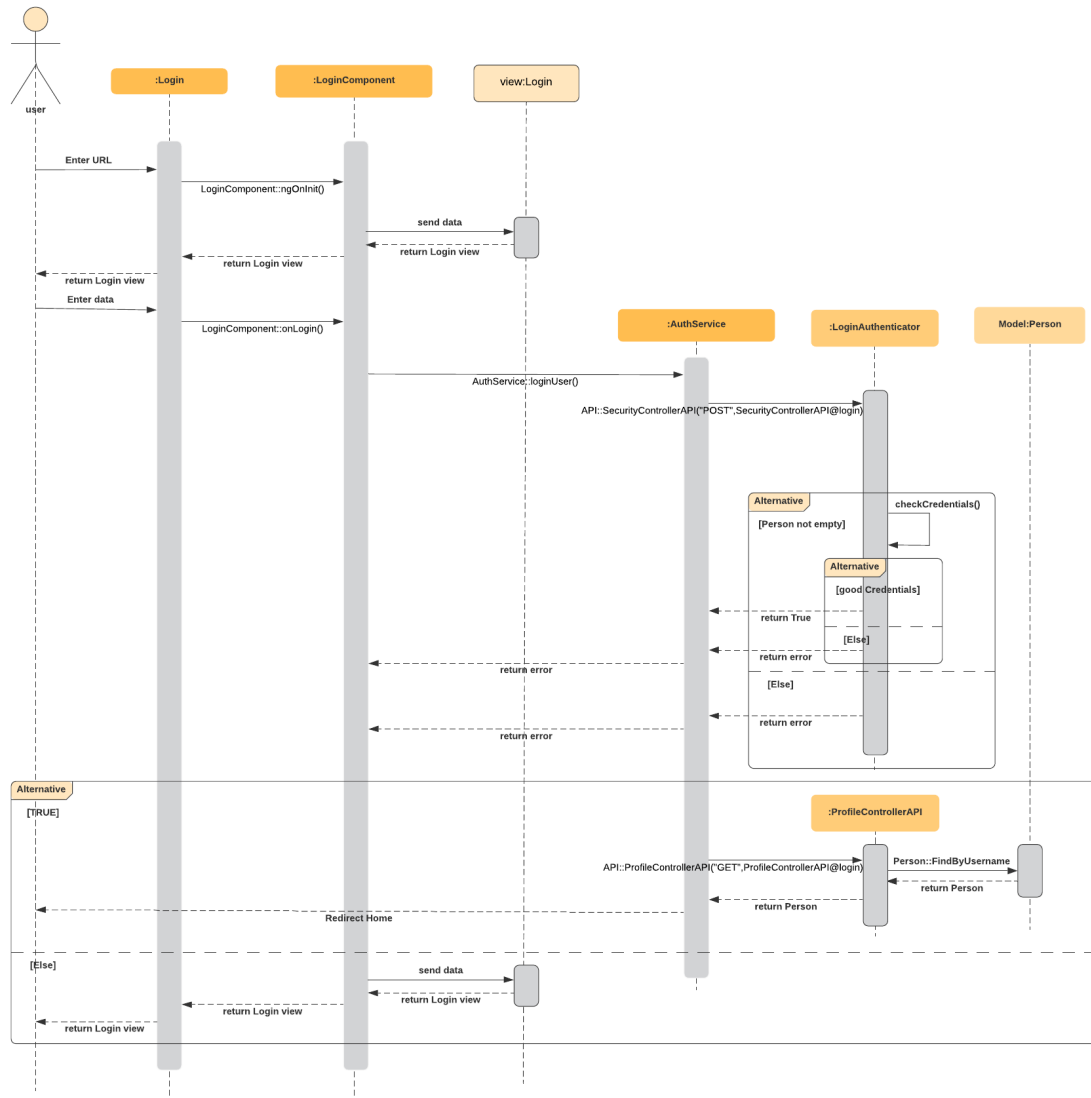


Figure 5.2.18: Diagramme de séquence de la connexion d'un utilisateur

Scripts concernés

- URL: [api.service.ts](#)
- URL: [auth.service.ts](#)
- URL: [SecurityControllerAPI.php](#)
- URL: [ProfileControllerAPI.php](#)
- URL: [login.component.ts](#)
- URL: [login.component.html](#)
- URL: [Person.php](#)

2.2 En tant qu'utilisateur, je dois pouvoir me créer un compte sur le site

Différences

- Réalisation d'une requête PUT via l'api REST.
- Amélioration de l'aspect graphique.
- Utilisation du *SecurityAuthenticator* du backend pour ajouter l'utilisateur dans le système.
- Ajout de l'inscription automatique de l'utilisateur au séance (voir 2.4).

Diagramme de séquence

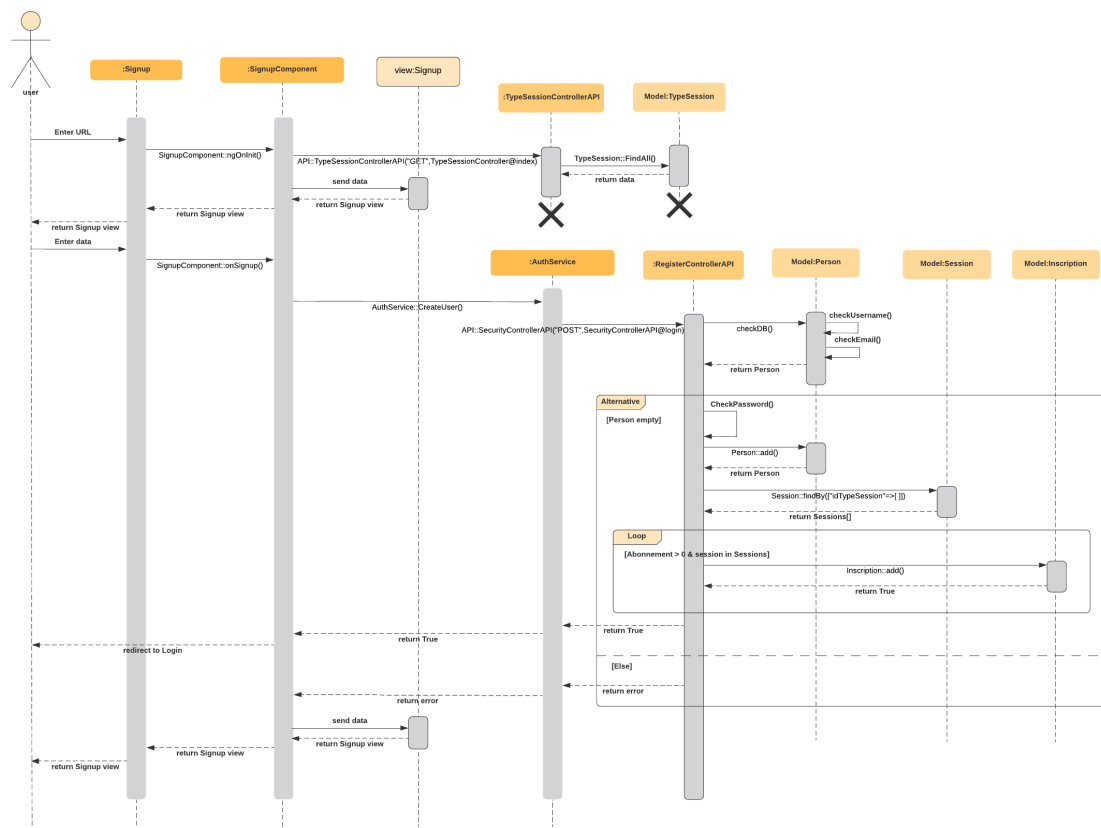


Figure 5.2.19: Diagramme de séquence de l'inscription d'un utilisateur.

Scripts concernés

- URL: [auth.service.ts](#)
- URL: [SecurityControllerAPI.php](#)
- URL: [Person.php](#)
- URL: [signup.component.ts](#)
- URL: [signup.component.html](#)

2.3 En tant qu'utilisateur, je dois pouvoir sélectionner le mois et l'année pour afficher les sessions qui m'intéressent

Différences

- Réorganisation de l'affichage.
- Ajout de la possibilité de sélectionner l'année à afficher.

Scripts concernés

- URL: [api.service.ts](#)
- URL: [MonthControllerAPI.php](#)
- URL: [Session.php](#)
- URL: [month.component.ts](#)
- URL: [month.component.html](#)

2.4 En tant qu'utilisateur, je dois pouvoir m'inscrire à une session un certain jour

Différences

- Modification de l'aspect des boutons d'inscription.

Scripts concernés

- URL: [api.service.ts](#)
- URL: [MonthControllerAPI.php](#)
- URL: [Session.php](#)
- URL: [month.component.ts](#)
- URL: [month.component.html](#)

2.5 En tant qu'utilisateur, je dois pouvoir être inscrit automatiquement à une session

Différences

- Réalisation de l'inscription automatique aux sessions lors de l'enregistrement de l'utilisateur.
- Utilisation de type de sessions afin de faciliter et de multiplier le choix des abonnements possibles.
- Possibilité pour un utilisateur de s'inscrire à plus de un type de session par semaine.
- Sélection du type de session parmi une liste.

E-Mail *

Username *

Nom *

Prenom *

Abonnement *

- ☐ Lundi 20:10
- ☐ Mercredi 09:00
- ☐ Mercredi 10:10
- ☐ Jeudi 17:30
- ☐ Jeudi 18:45
- ☐ Jeudi 19:50
- ☐ Lundi 19:00
- ☐ Lundi 10:00

Password *

Repeat Password *

S'inscrire

* : champs requis

Figure 5.2.20: Formulaire d'inscription version angular

Diagramme de séquence

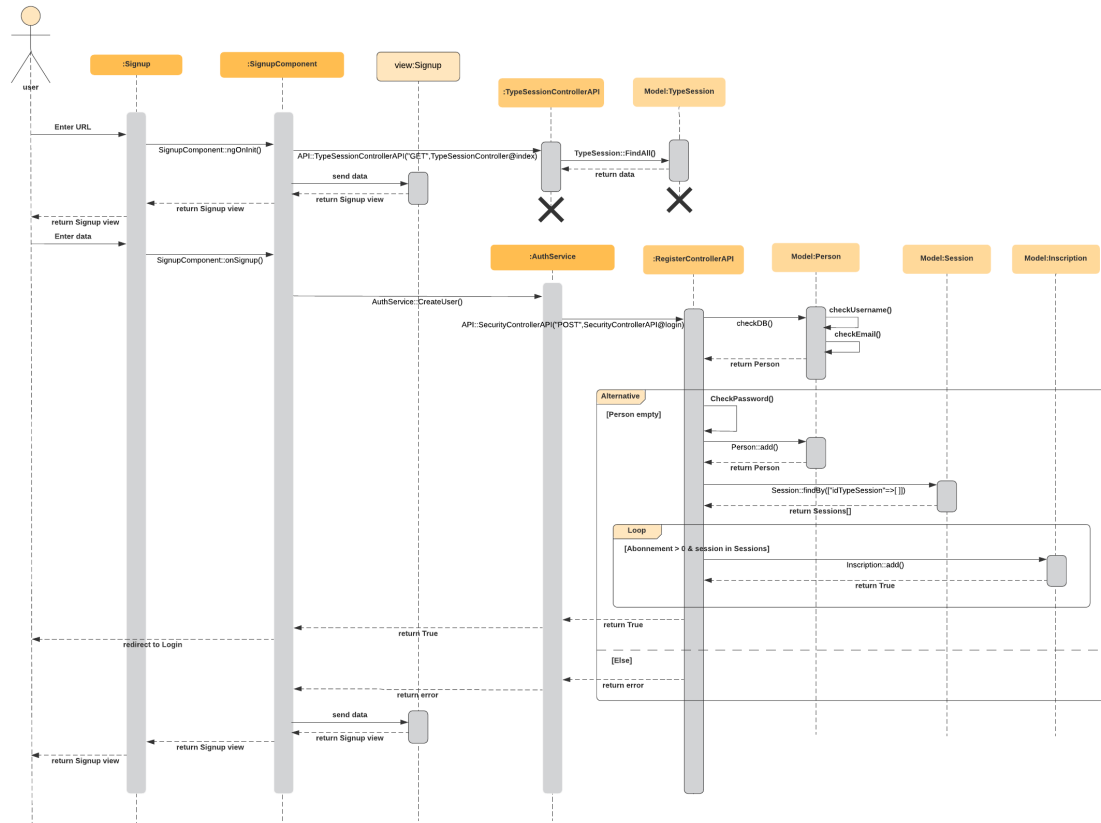


Figure 5.2.21: Diagramme de séquence de l'inscription automatique d'un utilisateur.

Scripts concernés

- URL: [auth.service.ts](#)
- URL: [SecurityControllerAPI.php](#)
- URL: [TypeSession.php](#)
- URL: [Session.php](#)
- URL: [Inscription.php](#)
- URL: [Person.php](#)
- URL: [signup.component.ts](#)
- URL: [signup.component.html](#)

2.6 En tant qu'utilisateur, je dois pouvoir me désinscrire à une session

Différences

- Modification du bouton de désinscription à un session.

Scripts concernés

- URL: [api.service.ts](#)
- URL: [MonthControllerAPI.php](#)
- URL: [Session.php](#)
- URL: [month.component.ts](#)
- URL: [month.component.html](#)

2.7 En tant qu'utilisateur, je dois pouvoir voir qui est inscrit pour chaque session

Différences

- Affichage de la liste des participant(e)s ne se fait plus sur une page séparée mais sur une fenêtre de dialogue qui apparaît devant le tableau.
- Modification du bouton permettant l'affichage des participant(e)s.

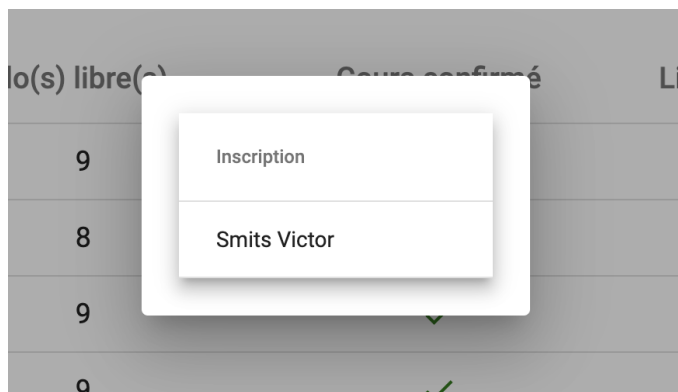


Figure 5.2.22: liste des participant(e)s

Scripts concernés

- URL: [api.service.ts](#)
- URL: [MonthControllerAPI.php](#)
- URL: [Session.php](#)
- URL: [month.component.ts](#)
- URL: [month.component.html](#)

2.8 En tant qu'utilisateur, je veux pouvoir voir le nombre de séances qui restent dans mon abonnement

Différences

- Modification graphique de l'affichage du profil et des informations.

Scripts concernés

- URL: [api.service.ts](#)
- URL: [ProfileControllerAPI.php](#)
- URL: [Person.php](#)
- URL: [profile.component.ts](#)
- URL: [profile.component.html](#)

2.9 En tant qu'utilisateur, je veux pouvoir modifier mon profil

Différences

- Modification graphique du formulaire.
- Possibilité de s'abonner à plusieurs types de sessions différentes.

Email
victor-smi@hotmail.com

Nom
Smits

Prenom
Victor

Mot de passe

☒ Lundi 20:10

☐ Mercredi 09:00

☐ Mercredi 10:10

☐ Jeudi 17:30

☐ Jeudi 18:45

☐ Jeudi 19:50

☐ Lundi 19:00

☐ Lundi 10:00

No Thanks Ok

Figure 5.2.23: Formulaire de modification du profil de l'utilisateur

Scripts concernés

- URL: [ProfileController.php](#)
- URL: [profile.html.twig](#)
- URL: [Person.php](#)

2.10 En tant qu'administrateur, je dois pouvoir annuler une session

Différences

- Modification graphique du bouton d'annulation de la session.

Scripts concernés

- URL: [api.service.ts](#)
- URL: [SessionAdministrationControllerApi.php](#)
- URL: [Session.php](#)
- URL: [admin-session.component.ts](#)
- URL: [admin-session.component.html](#)

2.11 En tant qu'administrateur, je dois pouvoir créer une nouvelle session peu importe la date

Différences

- Modification graphique du formulaire de création de session.

Scripts concernés

- URL: [api.service.ts](#)
- URL: [CreateSessionControllerApi.php](#)
- URL: [Session.php](#)
- URL: [admin-create-session.component.ts](#)
- URL: [admin-create-session.component.html](#)

2.12 En tant qu'administrateur, je dois pouvoir gérer les abonnements des utilisateurs

Différences

- Modification graphique de l'affichage des utilisateurs enregistrés dans le système.
- Possibilité de changer le nombre de séances de l'utilisateur.
- Au renouvellement de l'abonnement, l'utilisateur sera inscrit automatiquement aux sessions correspondant aux types de sessions de l'utilisateur.

Scripts concernés

- URL: [api.service.ts](#)
- URL: [AbonnementControllerApi.php](#)
- URL: [Person.php](#)
- URL: [admin-abo.component.ts](#)
- URL: [admin-abo.component.html](#)

6 | Conclusion

1 Objectifs atteints et non atteints

1.1 Compatibles Symfony-Twig et Symfony-Angular

User Stories	Priorité	Réalisation	Reste à faire
En tant qu'utilisateur, je dois pouvoir me créer un compte sur le site	2	100%	
En tant qu'utilisateur, je dois pouvoir sélectionner le mois et l'année pour afficher les sessions qui m'intéressent	2	80%	Sélection de l'année dans la partie symfony-twig et affichage des messages d'erreur sur le site Angular.
En tant qu'utilisateur, je dois pouvoir m'inscrire à une session un certain jour	1	90%	Afficher le message d'erreur sur le site Angular.
En tant qu'utilisateur, je dois pouvoir être inscrit automatiquement à une session	1	80%	Effectuer la boucle d'inscription pour la partie Symfony-Twig.
En tant qu'utilisateur, je dois pouvoir me désinscrire à une session	1	90%	Afficher le message d'erreur sur le site Angular.
En tant qu'utilisateur, je dois pouvoir voir qui est inscrit pour chaque session	2	100%	

En tant qu'utilisateur, je veux pouvoir voir dynamiquement le nombre de séances qui restent dans mon abonnement	2	100%	
En tant qu'utilisateur, je veux pouvoir modifier mon profil	1	100%	
En tant qu'utilisateur, je veux pouvoir m'inscrire à une session uniquement si il me reste des séances a disposition.	2	100%	
En tant qu'administrateur, je dois pouvoir annuler une session	2	90%	Afficher les messages d'erreur sur le site Angular.
En tant qu'administrateur, je dois pouvoir gérer les abonnements des utilisateurs	2	100%	
En tant qu'administrateur, je dois pouvoir créer une nouvelle session peu importe la date	2	80%	Affichage des erreurs sur Twig et vérification de l'existence de la session créée.

1.2 Compatibles Symfony-Angular uniquement

En tant qu'administrateur, je dois pouvoir générer les sessions pour un nombre d'années	1	100%	
En tant qu'administrateur, je dois pouvoir créer un type de session	2	100%	
En tant qu'administrateur, je veux pouvoir gérer les types de sessions disponibles pour l'inscription automatique	2	100%	
En tant qu'administrateur, je veux pouvoir modifier un type de session	1	100%	
En tant qu'administrateur, je veux pouvoir supprimer un type de session	1	90%	Afficher le message d'erreur sur le site Angular.
En tant qu'administrateur, je veux pouvoir supprimer un utilisateur du système	1	90%	Afficher le message d'erreur sur le site Angular.

2 Pistes d'amélioration

- Optimisation des requêtes de base de données.
- Nettoyage du code source (duplication, réorganisation, ...).
- Amélioration/Finition de la responsivité du site.
- Triage des sessions reçues dans angular pour un affichage trié en fonction du jour et de l'heure.
- Renommage des clés des json envoyées au frontend.
- Renommage de certains champs de la base de données.
- Utilisation d'une base de données orienté graph.

- Lors de la creation d'une session, trouver un type de session compatible si existant, sinon mettre le paramètre à NULL.

7 | Annexe

1 Code source

Retrouvez l'entièreté du code source sur le Github suivant : URL: [Aquabike](#)