



DOSSIER-PROJET

Nom de naissance ► ITO
Nom d'usage ►
Prénom ► MIKA
Adresse ► 272 RUE NICOLAS PARENT 73000 CHAMBERY

Titre professionnel visé

Concepteur Développeur d'Applications

Sommaire

Remerciements	4
Résumé du projet en français	5
Résumé du projet en anglais	6
Cahier des charges ou expressions des besoins de l'application à développer	7
Raisons du lancement du projet	7
Spécifications fonctionnelles	8
Fonctionnalité principale	8
1. Première partie	8
2. Seconde partie	9
a. Settings	9
b. Tickets & Price	10
c. Schedule	10
d. Lesson room	11
e. Reservation	13
f. Lesson history	14
g. Students	14
h. Ticket purchase history	14
3. Troisième partie	15
a. S'inscrire	15
b. Page d'accueil	16
c. Mon compte	18
d. Historique des leçons	18
e. Faire une réservation	19
f. Voir ma réservation	20
g. Acheter des tickets	20
h. Historique des achats de ticket	22
Spécifications techniques	
Structure des pages	23
Choix de technologies	24
1. Language de programmation	24
2. Framework	24
3. Plugin	24
4. API	24
5. Outil de développement	24

Schéma de données	25
Architecture	26
1. Structure globale du projet	26
2. Controller	26
3. Entity	27
4. Service	27
Exemples de codes	28
1. Contrôle du fuseau horaire	28
2. Contrôle de l'activation et la désactivation des créneaux horaires	33
3. Vidéo call	36
Réalisations	39
Organisation du travail	39
1. Commencer par inconnu	39
2. Méthode agile	39
3. Utilisation de commande “make” de Symfony	40
Difficultés rencontrées	40
1. Vérifier mail fonction	40
2. Stripe webhook	40
3. Agora vidéo call	41
Ce qu'il me reste à faire	41
Conclusion	42

REMERCIEMENTS

Je tiens à remercier mes formateurs, mes collègues de formation cda Simplon, et mon tuteur d'entreprise qui ont soutenu mon développement au cours de l'année.

RÉSUMÉ DU PROJET EN FRANÇAIS

J'ai développé une application qui permet à toute personne souhaitant enseigner quelque chose en ligne de créer facilement une plateforme de cours en ligne.

En raison de l'impact de la pandémie, de plus en plus de personnes apprennent en ligne. De nombreuses personnes souhaitent utiliser leur temps libre pour partager leurs compétences particulières avec d'autres personnes en ligne.

Il existe un large éventail de services en ligne, mais il faut du temps pour se démarquer parmi les nombreux sites comptant un grand nombre d'utilisateurs enregistrés et pour attirer des clients.

Si les personnes voulant enseigner quelque chose en ligne peuvent avoir leur propre site web, elles peuvent faire le marketing et le référencement elles-mêmes, ce qui attireraient plus de clients. Cependant, cela coûterait très cher.

J'ai donc créé une application qui permet à ces personnes de créer facilement leur propre classe en ligne.

Cette application dispose de toutes les fonctions requises pour les cours en ligne. De nombreuses caractéristiques, telles que la réservation d'une leçon, la prise d'une leçon en ligne par appel vidéo, le paiement du ticket de la leçon et le contact entre les élèves et l'enseignant peuvent être effectués sur la même plateforme.

Pour réaliser mon projet, j'utilise php avec le framework symfony qui me permet d'implémenter les différentes fonctionnalités.

J'ai aussi utilisé Agora Vidéo API pour implémenter la fonctionnalité de vidéo call, Stripe API pour le paiement.

RÉSUMÉ DU PROJET EN ANGLAIS

I developed an application that allows anyone who wants to teach something online to easily create an online course platform.

Due to the impact of the pandemic, more and more people are learning online.
Many people want to use their free time to share their particular skills with others online.

There is a wide range of online services available, but it takes time to stand out among the many sites with large numbers of registered users and to attract customers.

If people who want to teach something online can have their own website, they can do the marketing and SEO themselves, which will attract more customers. However, this would be very expensive.

So I created an application that allows these people to easily create their own online class.

This app has all the features required for online classes.
Many features, such as booking a lesson, taking an online lesson via video call, paying for the lesson ticket and contact between students and teacher can be done on the same platform.

To realize my project, I use php with the symfony framework which allows me to implement the different features.

I also used Agora Video API to implement the video call functionality, Stripe API for the payment.

CAHIER DES CHARGES OU EXPRESSIONS DES BESOINS DE L'APPLICATION À DÉVELOPPER

Raisons du lancement du projet

De plus en plus de personnes apprennent en ligne, notamment en raison de la pandémie.

Il existe un large éventail de services en ligne, mais il faut du temps pour se démarquer parmi les nombreux sites comptant un grand nombre d'utilisateurs enregistrés, et attirer des clients.

Si vous avez votre propre site web, vous pourrez faire votre propre marketing, votre propre référencement et attirer des clients plus facilement, mais cela vous coûtera plus cher.

J'ai donc mis au point un système qui permet à quiconque d'enseigner sans peine quelque chose en ligne. Si vous voulez enseigner vos compétences particulières en ligne, cette application permet de démarrer aisément.

SPÉCIFICATIONS FONCTIONNELLES

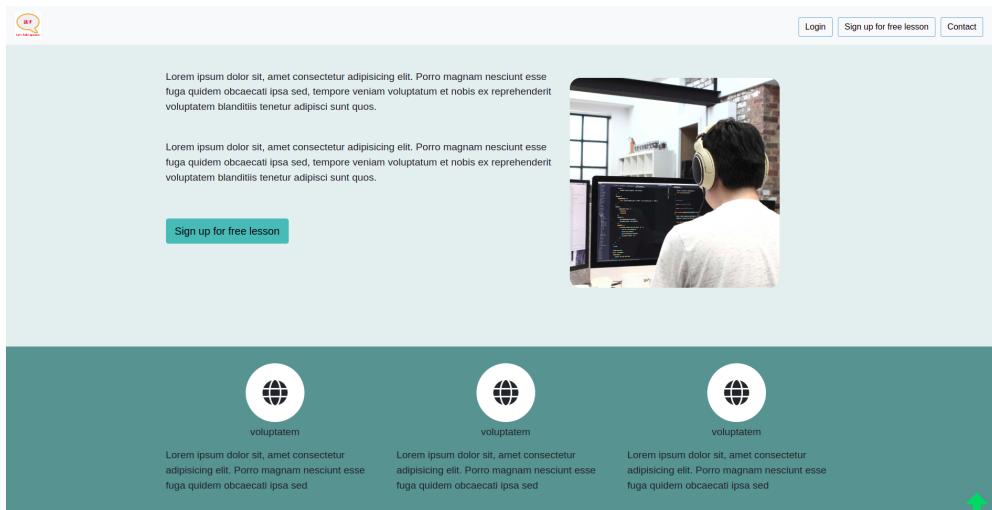
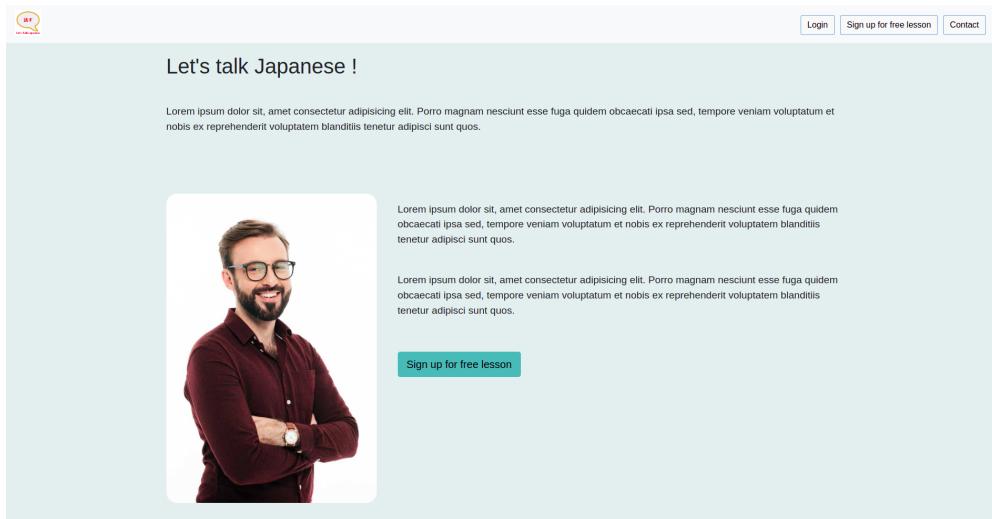
Fonctionnalité principale

Cette application se compose de **trois parties**.

1. Première partie

La première partie est le **site vitrine**, où l'administrateur du site communique leur profil, la forme et le contenu de leurs cours au public afin d'attirer de nouveaux clients.

Cette partie se compose de simples HTML et CSS. Avec un peu de connaissances de ces technologies, l'administrateur peut modifier les pages à volonté. En utilisant une variété de modèles accessibles au public, **il est possible d'adapter la conception à ses propres besoins**.



2. Seconde partie

La seconde partie est un **back-office pour l'administrateur** du site. Le back-office se compose de neuf éléments de menu.

- Settings
- Tickets
- Price
- Schedule
- Lesson room
- Reservation
- Lesson history
- Students
- Ticket purchase history

a. Settings

Tout d'abord, pour lancer le service cours en ligne, l'administrateur doit personnaliser le site. Les éléments suivants peuvent être configurés librement dans la page settings.

- Nom du site web
- Logo du site web
- Méta description (250-300 caractères de texte décrivant le contenu du site web)
- Email de l'administrateur
- Devise de paiement
- Symbole de devise
- Nombre de tickets gratuits offerts aux nouveaux clients
- Durée d'un cours : 25 ou 50 minutes
- Combien de minutes avant le début du cours les clients peuvent entrer dans la salle de cours
- Combien de minutes après l'heure de début du cours les élèves sont autorisés à entrer dans la salle
- Charte graphique
- Stripe secret key
- Agora app id
- Agora app certificate
- Timezone

b. Tickets & Price

L'administrateur fixe le prix de vente des tickets et le nombre de tickets vendus.

ID	Name	Ticket number	Price	Currency	Symbol	actions
1	1 lesson	1	5.00	EUR	€	show edit
2	5 lessons	5	24.00	EUR	€	show edit
3	10 lessons	10	40.00	EUR	€	show edit
4	20 lessons	20	70.00	EUR	€	show edit

c. Schedule

L'administrateur saisit le programme des leçons pour la semaine à venir. Sur un tableau englobant les 24 heures d'une journée et les 7 jours de la semaine, il peut ouvrir des créneaux en fonction de son temps disponible.



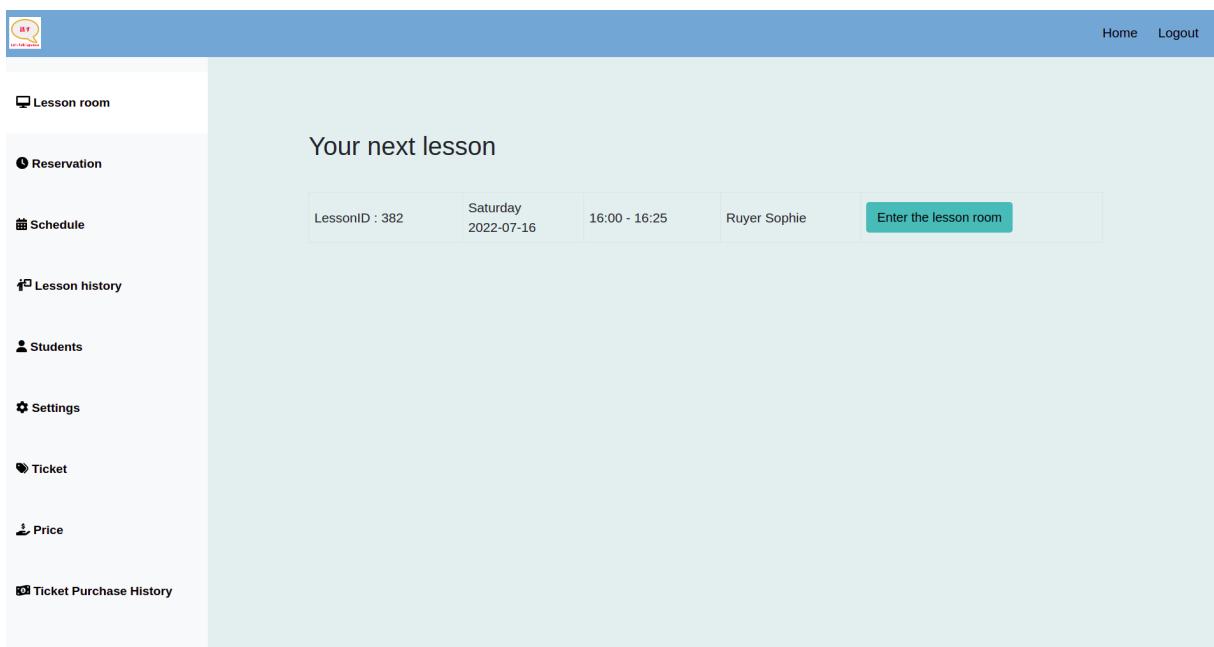
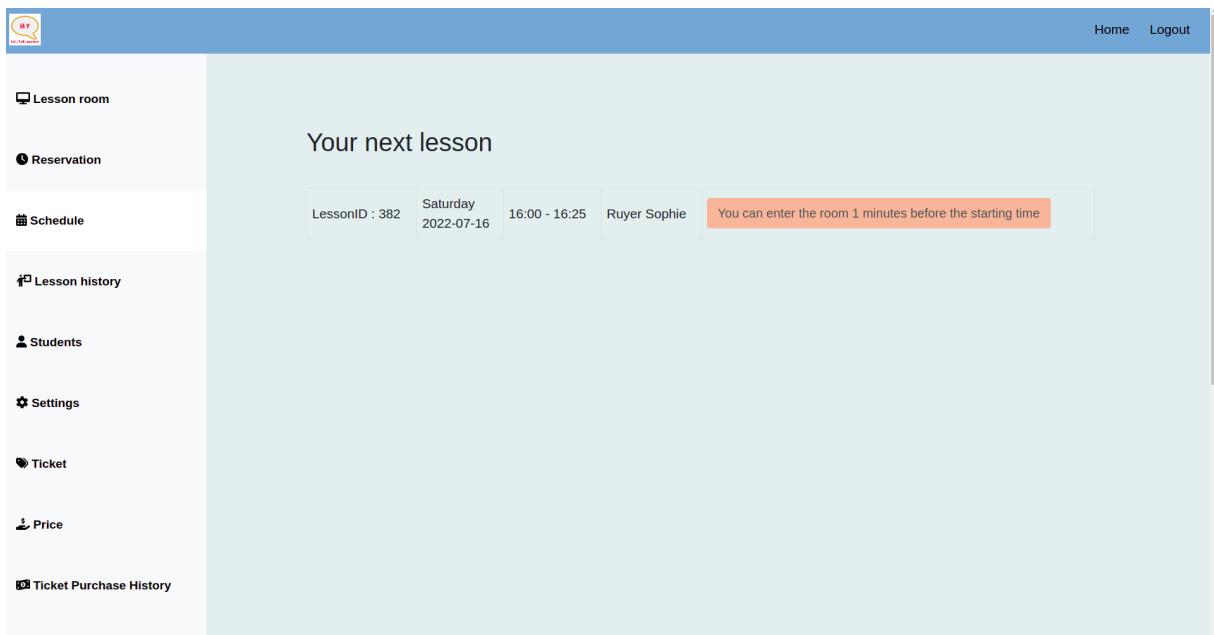
Home Logout

		Schedule						
		Saturday 2022-07-16	Sunday 2022-07-17	Monday 2022-07-18	Tuesday 2022-07-19	Wednesday 2022-07-20	Thursday 2022-07-21	Friday 2022-07-22
	Lesson room	Closed	Off	Off	Off	Off	Off	Off
	Reservation	15:00:00	Closed	Off	Off	Off	Off	Off
	Schedule	15:30:00	Closed	Off	Off	Off	Off	Off
	Lesson history	16:00:00	Off	Off	Off	Off	Off	Off
	Students	16:30:00	Off	Off	Off	Off	Off	Off
	Settings	17:00:00	Available	Available	Off	Off	Off	Off
	Ticket	17:30:00	Available	Available	Off	Off	Off	Off
	Price	18:00:00	Off	Off	Off	Off	Off	Off
	Ticket Purchase History	18:30:00	Off	Off	Off	Off	Off	Off
		19:00:00	Off	Off	Off	Available	Off	Off
		19:30:00	Off	Off	Off	Available	Off	Off
		20:00:00	Off	Off	Available	Available	Off	Off
		20:30:00	Off	Off	Available	Available	Off	Off
		21:00:00	Off	Off	Available	Available	Off	Off
		21:30:00	Off	Off	Off	Off	Off	Off
		22:00:00	Off	Off	Off	Off	Off	Off
		22:30:00	Off	Off	Off	Off	Off	Off
		23:00:00	Off	Off	Off	Off	Off	Off

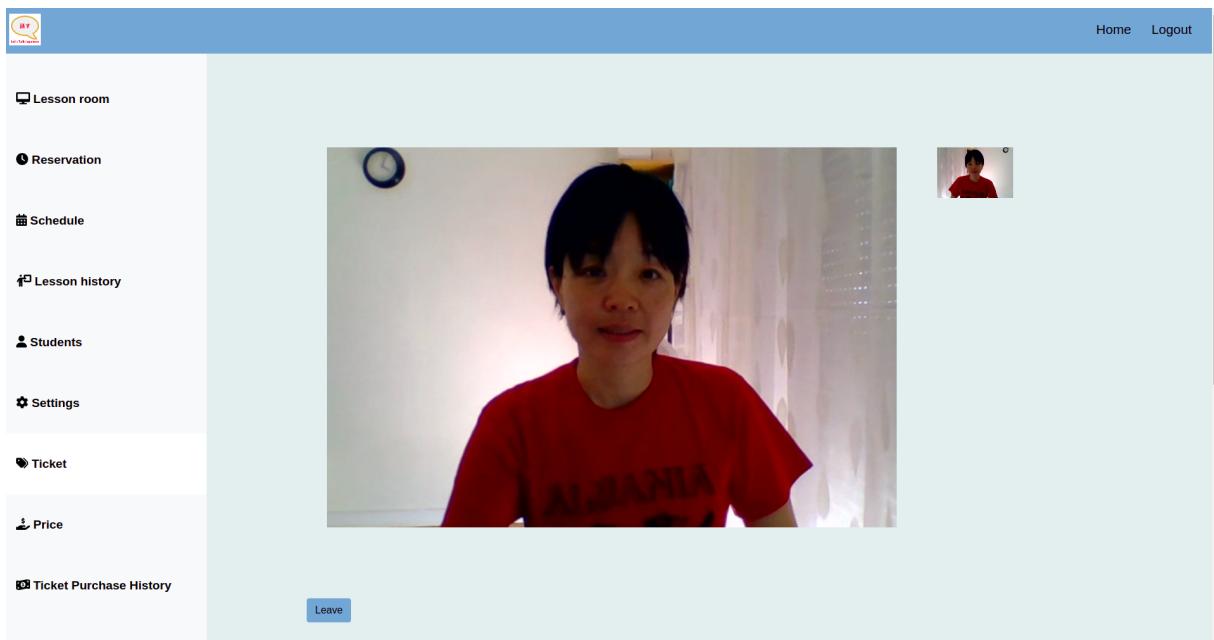
d. Lesson room

L'administrateur attend que les rendez-vous soient pris. Lorsqu'une réservation est effectuée par un client, la date et l'heure de début de la leçon réservée sont affichées sur la page d'accueil ou sur la page de la liste des réservations.

Lorsqu'il est temps de commencer la leçon, l'administrateur clique sur un bouton pour entrer dans la salle de cours. Ce bouton se transforme en bouton cliquable à un moment défini par l'administrateur, qui correspond au nombre de minutes avant que l'élève ne soit autorisé à entrer dans la salle.



Lorsque l'administrateur appuie sur le bouton pour entrer dans la leçon, l'appel vidéo commence. Le système fait en sorte que seul le client ayant réservé à l'heure de la leçon puisse entrer dans l'appel vidéo.



Lorsque la leçon est terminée, les administrateurs appuient sur le bouton "Quitter". Il sera alors ramené à la page d'accueil initiale.

e. Reservation

L'administrateur peut voir toutes les réservations sur la page de la liste des réservations.

The screenshot shows the "My Reservation" page. The sidebar on the left includes the following menu items:

- Lesson room
- Reservation**
- Schedule
- Lesson history
- Students
- Settings
- Ticket
- Price
- Ticket Purchase History

The main content area is titled "My Reservation" and contains the following text: "You can enter the classroom 15 minutes before the lesson start. Please click the button and proceed to the lesson." Below this, there is a table listing three upcoming lessons:

Lesson ID	Date	Time	
382	Saturday 2022-07-16	16:00 - 16:25	Ruyer Sophie
367	Saturday 2022-07-16	17:00 - 17:25	Pesquet Thomas
369	Saturday 2022-07-16	17:30 - 17:55	Ito Kota1

A blue button labeled "Begin class" is located to the right of the third lesson entry.

f. Lesson history

L'administrateur peut voir les leçons qu'il a terminées dans son historique de leçons.



Home Logout

 Lesson room Reservation Schedule Lesson history Students Settings Ticket Price Ticket Purchase History	<h3>Lesson history</h3> <p>Number of Lessons : 108 times. Total lesson time : 2700 min.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Lesson Id</th> <th>Lesson date</th> <th>Lesson time</th> <th></th> </tr> </thead> <tbody> <tr><td>301</td><td>2022-07-13 (Wednesday)</td><td>13:30 - 13:55</td><td>25 min. Ito Kota1</td></tr> <tr><td>347</td><td>2022-07-11 (Monday)</td><td>08:30 - 08:55</td><td>25 min. Ito Kota2</td></tr> <tr><td>346</td><td>2022-07-11 (Monday)</td><td>08:00 - 08:25</td><td>25 min. Pesquet Thomas</td></tr> <tr><td>365</td><td>2022-07-10 (Sunday)</td><td>17:00 - 17:25</td><td>25 min. Ruyer Sophie</td></tr> <tr><td>361</td><td>2022-07-10 (Sunday)</td><td>14:30 - 14:55</td><td>Ito Kota5</td></tr> <tr><td>360</td><td>2022-07-10 (Sunday)</td><td>14:00 - 14:25</td><td>Ito Kota5</td></tr> <tr><td>359</td><td>2022-07-10 (Sunday)</td><td>13:00 - 13:25</td><td>Ito Kota5</td></tr> <tr><td>358</td><td>2022-07-10 (Sunday)</td><td>12:30 - 12:55</td><td>Ito Kota2</td></tr> <tr><td>357</td><td>2022-07-10 (Sunday)</td><td>12:00 - 12:25</td><td>Ito Kota2</td></tr> <tr><td>356</td><td>2022-07-10 (Sunday)</td><td>11:30 - 11:55</td><td>Ruyer Sophie</td></tr> <tr><td>355</td><td>2022-07-10 (Sunday)</td><td>10:30 - 10:55</td><td>Pesquet Thomas</td></tr> <tr><td>354</td><td>2022-07-10 (Sunday)</td><td>10:00 - 10:25</td><td>Ito Kota1</td></tr> <tr><td>352</td><td>2022-07-10 (Sunday)</td><td>09:00 - 09:25</td><td>Pesquet Thomas</td></tr> <tr><td>345</td><td>2022-07-10 (Sunday)</td><td>08:30 - 08:55</td><td>Pesquet Thomas</td></tr> <tr><td>344</td><td>2022-07-10 (Sunday)</td><td>08:00 - 08:25</td><td>Pesquet Thomas</td></tr> <tr><td>312</td><td>2022-07-10 (Sunday)</td><td>00:30 - 00:55</td><td>25 min. Taglioni Alice</td></tr> <tr><td>300</td><td>2022-07-10 (Sunday)</td><td>00:00 - 00:25</td><td>25 min. Ito Kota1</td></tr> </tbody> </table>	Lesson Id	Lesson date	Lesson time		301	2022-07-13 (Wednesday)	13:30 - 13:55	25 min. Ito Kota1	347	2022-07-11 (Monday)	08:30 - 08:55	25 min. Ito Kota2	346	2022-07-11 (Monday)	08:00 - 08:25	25 min. Pesquet Thomas	365	2022-07-10 (Sunday)	17:00 - 17:25	25 min. Ruyer Sophie	361	2022-07-10 (Sunday)	14:30 - 14:55	Ito Kota5	360	2022-07-10 (Sunday)	14:00 - 14:25	Ito Kota5	359	2022-07-10 (Sunday)	13:00 - 13:25	Ito Kota5	358	2022-07-10 (Sunday)	12:30 - 12:55	Ito Kota2	357	2022-07-10 (Sunday)	12:00 - 12:25	Ito Kota2	356	2022-07-10 (Sunday)	11:30 - 11:55	Ruyer Sophie	355	2022-07-10 (Sunday)	10:30 - 10:55	Pesquet Thomas	354	2022-07-10 (Sunday)	10:00 - 10:25	Ito Kota1	352	2022-07-10 (Sunday)	09:00 - 09:25	Pesquet Thomas	345	2022-07-10 (Sunday)	08:30 - 08:55	Pesquet Thomas	344	2022-07-10 (Sunday)	08:00 - 08:25	Pesquet Thomas	312	2022-07-10 (Sunday)	00:30 - 00:55	25 min. Taglioni Alice	300	2022-07-10 (Sunday)	00:00 - 00:25	25 min. Ito Kota1
Lesson Id	Lesson date	Lesson time																																																																							
301	2022-07-13 (Wednesday)	13:30 - 13:55	25 min. Ito Kota1																																																																						
347	2022-07-11 (Monday)	08:30 - 08:55	25 min. Ito Kota2																																																																						
346	2022-07-11 (Monday)	08:00 - 08:25	25 min. Pesquet Thomas																																																																						
365	2022-07-10 (Sunday)	17:00 - 17:25	25 min. Ruyer Sophie																																																																						
361	2022-07-10 (Sunday)	14:30 - 14:55	Ito Kota5																																																																						
360	2022-07-10 (Sunday)	14:00 - 14:25	Ito Kota5																																																																						
359	2022-07-10 (Sunday)	13:00 - 13:25	Ito Kota5																																																																						
358	2022-07-10 (Sunday)	12:30 - 12:55	Ito Kota2																																																																						
357	2022-07-10 (Sunday)	12:00 - 12:25	Ito Kota2																																																																						
356	2022-07-10 (Sunday)	11:30 - 11:55	Ruyer Sophie																																																																						
355	2022-07-10 (Sunday)	10:30 - 10:55	Pesquet Thomas																																																																						
354	2022-07-10 (Sunday)	10:00 - 10:25	Ito Kota1																																																																						
352	2022-07-10 (Sunday)	09:00 - 09:25	Pesquet Thomas																																																																						
345	2022-07-10 (Sunday)	08:30 - 08:55	Pesquet Thomas																																																																						
344	2022-07-10 (Sunday)	08:00 - 08:25	Pesquet Thomas																																																																						
312	2022-07-10 (Sunday)	00:30 - 00:55	25 min. Taglioni Alice																																																																						
300	2022-07-10 (Sunday)	00:00 - 00:25	25 min. Ito Kota1																																																																						

g. Students

L'administrateur peut consulter, modifier et supprimer les données personnelles des clients.

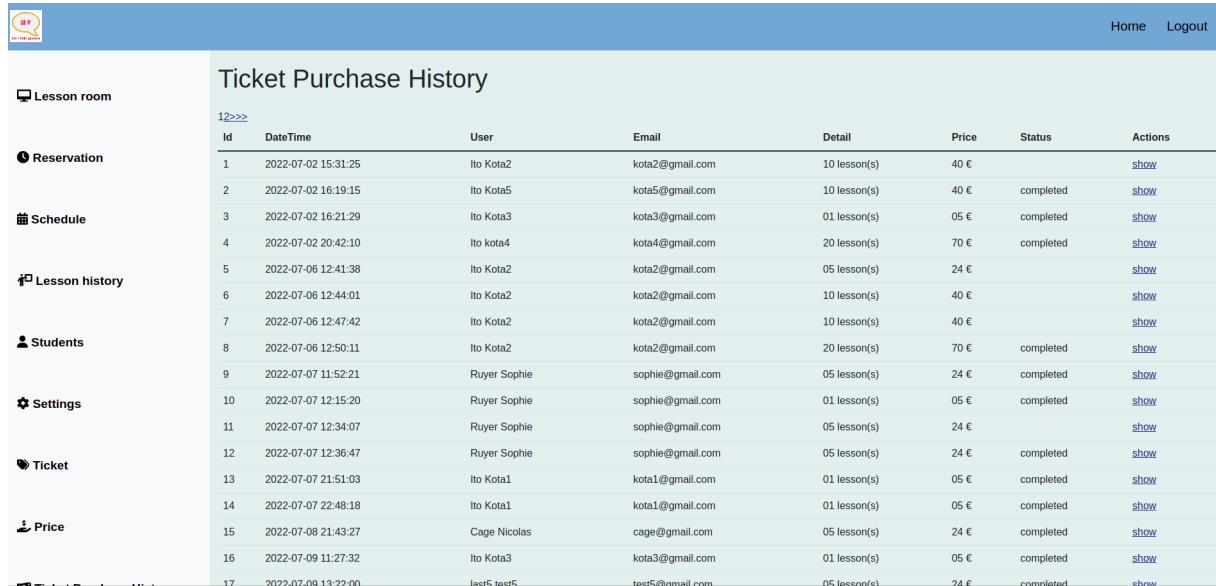


Home Logout

 Lesson room Reservation Schedule Lesson history Students Settings Ticket Price Ticket Purchase History	<h3>Students</h3> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th colspan="7">12>></th></tr> <tr> <th>ID</th><th>Last Name</th><th>First Name</th><th>Email</th><th>Ticket</th><th>Timezone</th><th>actions</th></tr> </thead> <tbody> <tr><td>7</td><td>Ito</td><td>Kota1</td><td>kota1@gmail.com</td><td>0</td><td>Europe/London</td><td>show edit</td></tr> <tr><td>8</td><td>Ito</td><td>Kota2</td><td>kota2@gmail.com</td><td>2</td><td>Europe/Paris</td><td>show edit</td></tr> <tr><td>9</td><td>Ito</td><td>Kota3</td><td>kota3@gmail.com</td><td>0</td><td>Europe/Paris</td><td>show edit</td></tr> <tr><td>10</td><td>Ito</td><td>Kota4</td><td>kota4@gmail.com</td><td>17</td><td>Asia/Kathmandu</td><td>show edit</td></tr> <tr><td>11</td><td>Ito</td><td>Mika</td><td>mika@gmail.com</td><td></td><td>Europe/Paris</td><td>show edit</td></tr> <tr><td>13</td><td>Ito</td><td>Kota5</td><td>kota5@gmail.com</td><td>4</td><td>Europe/Paris</td><td>show edit</td></tr> <tr><td>14</td><td>Ruyer</td><td>Sophie</td><td>sophie@gmail.com</td><td>4</td><td>Europe/Paris</td><td>show edit</td></tr> <tr><td>15</td><td>Cage</td><td>Nicolas</td><td>cage@gmail.com</td><td>3</td><td>Europe/Paris</td><td>show edit</td></tr> <tr><td>16</td><td>Taglioni</td><td>Alice</td><td>alice@gmail.com</td><td>36</td><td>Europe/Paris</td><td>show edit</td></tr> <tr><td>17</td><td>Pesquet</td><td>Thomas</td><td>thomas@gmail.com</td><td>2</td><td>Europe/Paris</td><td>show edit</td></tr> <tr><td>18</td><td>Mokran</td><td>Marian</td><td>marian@gmail.com</td><td></td><td>Europe/Paris</td><td>show edit</td></tr> <tr><td>19</td><td>Sut</td><td>Frederic</td><td>frederic@gmail.com</td><td></td><td>Europe/Paris</td><td>show edit</td></tr> <tr><td>20</td><td>Marc</td><td>John</td><td>john@gmail.com</td><td></td><td>Europe/Paris</td><td>show edit</td></tr> <tr><td>21</td><td>Maro</td><td>Sammuel</td><td>sammuel@gmail.com</td><td></td><td>Europe/Paris</td><td>show edit</td></tr> <tr><td>22</td><td>Carey</td><td>Mariah</td><td>mariah@gmail.com</td><td></td><td>Europe/Paris</td><td>show edit</td></tr> <tr><td>23</td><td>Ri</td><td>Jeong hyeok</td><td>ri@gmail.com</td><td></td><td>Europe/Paris</td><td>show edit</td></tr> <tr><td>24</td><td>last1</td><td>test1</td><td>test1@gmail.com</td><td></td><td>Europe/Paris</td><td>show edit</td></tr> <tr><td>25</td><td>last2</td><td>test2</td><td>test2@gmail.com</td><td></td><td>Europe/Paris</td><td>show edit</td></tr> <tr><td>26</td><td>last3</td><td>test3</td><td>test3@gmail.com</td><td></td><td>Europe/Paris</td><td>show edit</td></tr> </tbody> </table>	12>>							ID	Last Name	First Name	Email	Ticket	Timezone	actions	7	Ito	Kota1	kota1@gmail.com	0	Europe/London	show edit	8	Ito	Kota2	kota2@gmail.com	2	Europe/Paris	show edit	9	Ito	Kota3	kota3@gmail.com	0	Europe/Paris	show edit	10	Ito	Kota4	kota4@gmail.com	17	Asia/Kathmandu	show edit	11	Ito	Mika	mika@gmail.com		Europe/Paris	show edit	13	Ito	Kota5	kota5@gmail.com	4	Europe/Paris	show edit	14	Ruyer	Sophie	sophie@gmail.com	4	Europe/Paris	show edit	15	Cage	Nicolas	cage@gmail.com	3	Europe/Paris	show edit	16	Taglioni	Alice	alice@gmail.com	36	Europe/Paris	show edit	17	Pesquet	Thomas	thomas@gmail.com	2	Europe/Paris	show edit	18	Mokran	Marian	marian@gmail.com		Europe/Paris	show edit	19	Sut	Frederic	frederic@gmail.com		Europe/Paris	show edit	20	Marc	John	john@gmail.com		Europe/Paris	show edit	21	Maro	Sammuel	sammuel@gmail.com		Europe/Paris	show edit	22	Carey	Mariah	mariah@gmail.com		Europe/Paris	show edit	23	Ri	Jeong hyeok	ri@gmail.com		Europe/Paris	show edit	24	last1	test1	test1@gmail.com		Europe/Paris	show edit	25	last2	test2	test2@gmail.com		Europe/Paris	show edit	26	last3	test3	test3@gmail.com		Europe/Paris	show edit
12>>																																																																																																																																																				
ID	Last Name	First Name	Email	Ticket	Timezone	actions																																																																																																																																														
7	Ito	Kota1	kota1@gmail.com	0	Europe/London	show edit																																																																																																																																														
8	Ito	Kota2	kota2@gmail.com	2	Europe/Paris	show edit																																																																																																																																														
9	Ito	Kota3	kota3@gmail.com	0	Europe/Paris	show edit																																																																																																																																														
10	Ito	Kota4	kota4@gmail.com	17	Asia/Kathmandu	show edit																																																																																																																																														
11	Ito	Mika	mika@gmail.com		Europe/Paris	show edit																																																																																																																																														
13	Ito	Kota5	kota5@gmail.com	4	Europe/Paris	show edit																																																																																																																																														
14	Ruyer	Sophie	sophie@gmail.com	4	Europe/Paris	show edit																																																																																																																																														
15	Cage	Nicolas	cage@gmail.com	3	Europe/Paris	show edit																																																																																																																																														
16	Taglioni	Alice	alice@gmail.com	36	Europe/Paris	show edit																																																																																																																																														
17	Pesquet	Thomas	thomas@gmail.com	2	Europe/Paris	show edit																																																																																																																																														
18	Mokran	Marian	marian@gmail.com		Europe/Paris	show edit																																																																																																																																														
19	Sut	Frederic	frederic@gmail.com		Europe/Paris	show edit																																																																																																																																														
20	Marc	John	john@gmail.com		Europe/Paris	show edit																																																																																																																																														
21	Maro	Sammuel	sammuel@gmail.com		Europe/Paris	show edit																																																																																																																																														
22	Carey	Mariah	mariah@gmail.com		Europe/Paris	show edit																																																																																																																																														
23	Ri	Jeong hyeok	ri@gmail.com		Europe/Paris	show edit																																																																																																																																														
24	last1	test1	test1@gmail.com		Europe/Paris	show edit																																																																																																																																														
25	last2	test2	test2@gmail.com		Europe/Paris	show edit																																																																																																																																														
26	last3	test3	test3@gmail.com		Europe/Paris	show edit																																																																																																																																														

h. Ticket purchase history

L'administrateur peut consulter l'historique des achats de tickets du client.



Ticket Purchase History								
	Id	Date/Time	User	Email	Detail	Price	Status	Actions
● Reservation	1	2022-07-02 15:31:25	Ito Kota2	kota2@gmail.com	10 lesson(s)	40 €	40 €	show
■ Schedule	2	2022-07-02 16:19:15	Ito Kota5	kota5@gmail.com	10 lesson(s)	40 €	completed	show
▢ Lesson history	3	2022-07-02 16:21:29	Ito Kota3	kota3@gmail.com	01 lesson(s)	05 €	completed	show
▢ Students	4	2022-07-02 20:42:10	Ito Kota4	kota4@gmail.com	20 lesson(s)	70 €	completed	show
▢ Settings	5	2022-07-06 12:41:38	Ito Kota2	kota2@gmail.com	05 lesson(s)	24 €	24 €	show
▢ Ticket	6	2022-07-06 12:44:01	Ito Kota2	kota2@gmail.com	10 lesson(s)	40 €	40 €	show
▢ Price	7	2022-07-06 12:47:42	Ito Kota2	kota2@gmail.com	10 lesson(s)	40 €	40 €	show
	8	2022-07-06 12:50:11	Ito Kota2	kota2@gmail.com	20 lesson(s)	70 €	completed	show
	9	2022-07-07 11:52:21	Ruyer Sophie	sophie@gmail.com	05 lesson(s)	24 €	completed	show
	10	2022-07-07 12:15:20	Ruyer Sophie	sophie@gmail.com	01 lesson(s)	05 €	completed	show
	11	2022-07-07 12:34:07	Ruyer Sophie	sophie@gmail.com	05 lesson(s)	24 €	24 €	show
	12	2022-07-07 12:36:47	Ruyer Sophie	sophie@gmail.com	05 lesson(s)	24 €	completed	show
	13	2022-07-07 21:51:03	Ito Kota1	kota1@gmail.com	01 lesson(s)	05 €	completed	show
	14	2022-07-07 22:48:18	Ito Kota1	kota1@gmail.com	01 lesson(s)	05 €	completed	show
	15	2022-07-08 21:43:27	Cage Nicolas	cage@gmail.com	05 lesson(s)	24 €	completed	show
	16	2022-07-09 11:27:32	Ito Kota3	kota3@gmail.com	01 lesson(s)	05 €	completed	show
	17	2022-07-09 13:22:00	Ito Kota5	test5@gmail.com	05 lesson(s)	24 €	completed	show

3. Troisième partie

La troisième partie est réservée aux clients enregistrés. En se connectant, les clients peuvent réserver des cours, assister à des cours, acheter des billets, etc.

Elle se compose des pages suivantes.

- S'inscrire
- Page d'accueil
- Mon compte
- Historique des leçons
- Faire une réservation
- Voir ma réservation
- Acheter des tickets
- Historique des achats de ticket

a. S'inscrire

Pour prendre des cours, les clients doivent d'abord s'inscrire.

First name:

Last name:

Email:

Password:

Timezone:

By checking out you agree with our Terms of Service. We will process your personal data for the fulfillment of your order and other purposes as per our Privacy Policy

b. Page d'accueil

Une fois que le client s'est inscrit et connecté, il accède à la page d'accueil.

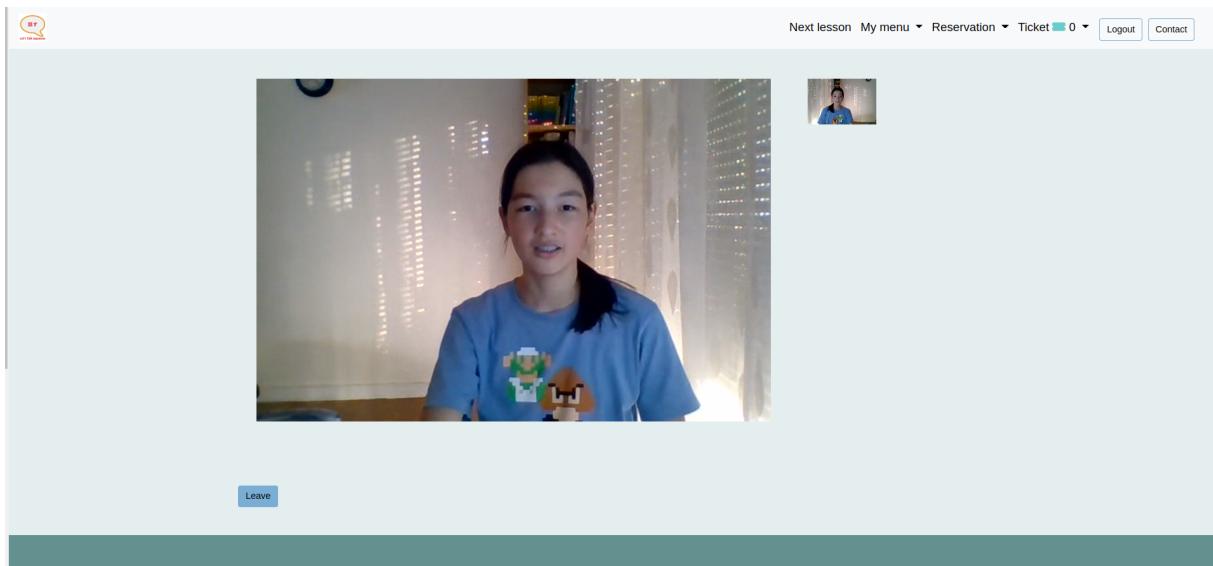
La barre de menu permet de naviguer facilement vers toutes les pages.

La page d'accueil est très simple et affiche des informations sur la leçon réservée. Si la leçon est sur le point de commencer, un bouton d'entrée apparaît et le client peut se rendre directement dans la salle de cours.

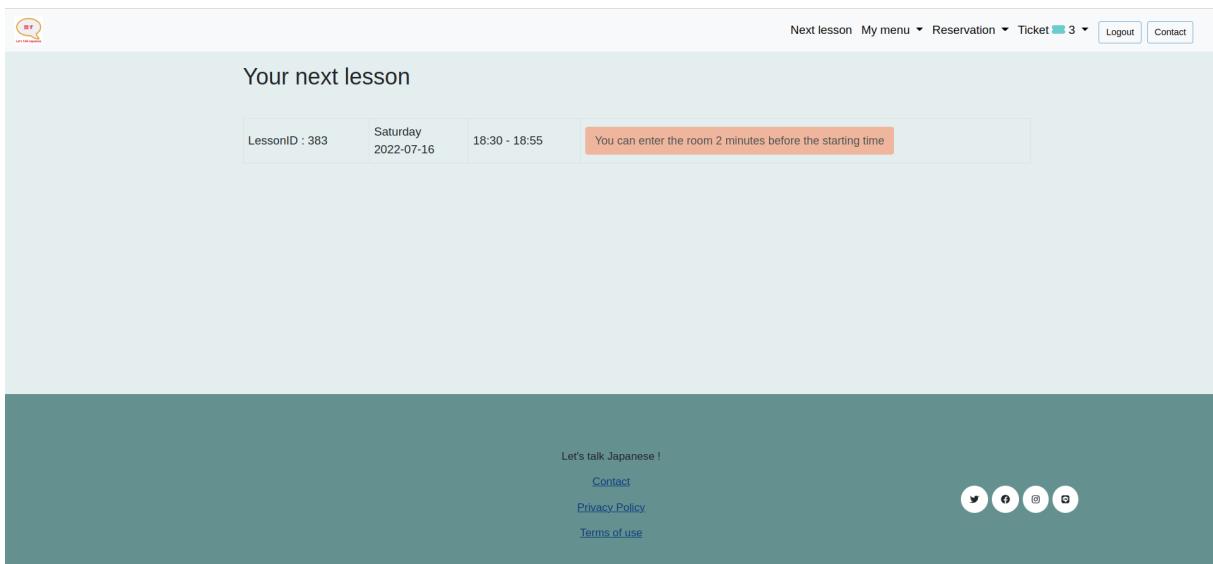
LessonID : 383 Saturday
2022-07-16 18:30 - 18:55

Let's talk Japanese !
[Contact](#)
[Privacy Policy](#)
[Terms of use](#)

Dans la salle de cours, les clients peuvent quitter la leçon et y revenir à tout moment pendant la durée du cours.

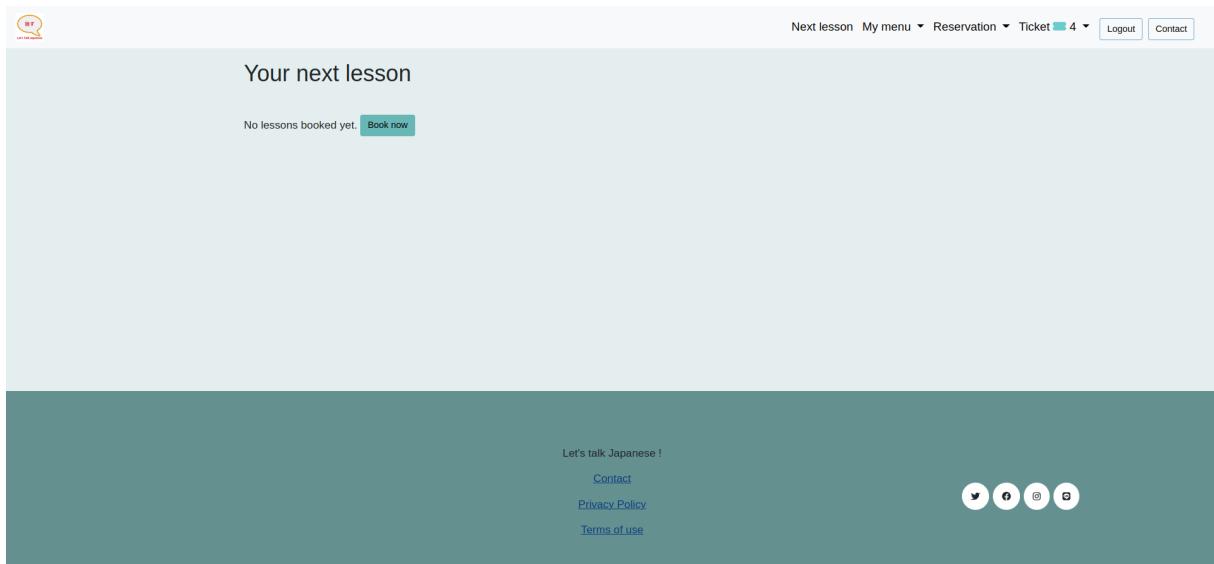


S'il reste du temps avant le début de la leçon, le bouton "Entrer" s'affiche, mais le bouton ne peut être cliqué.



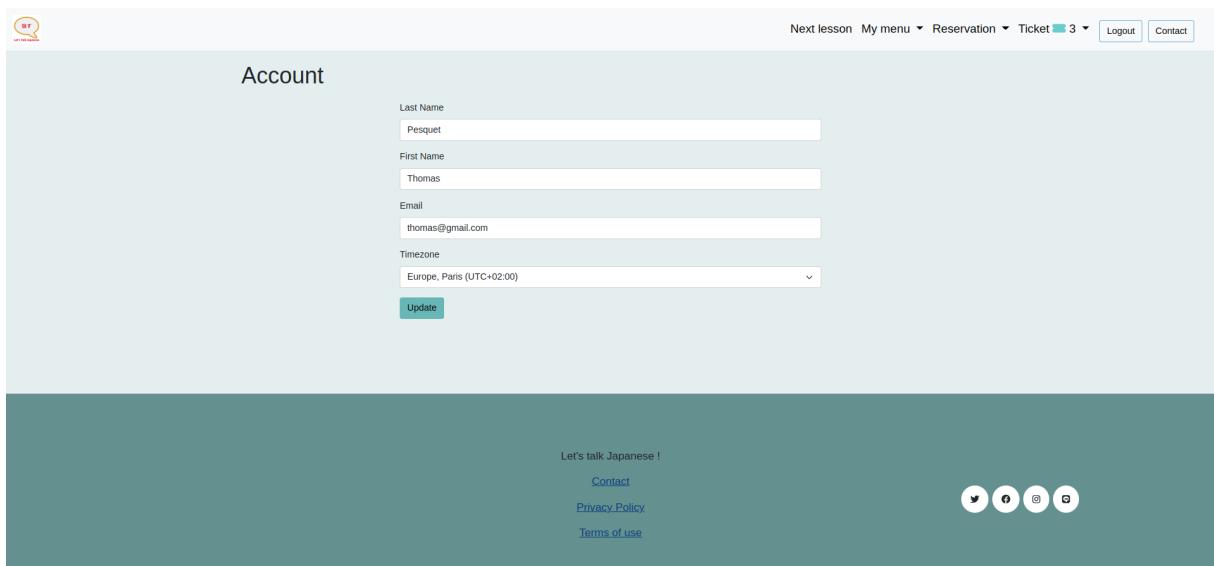
Les administrateurs peuvent définir depuis le back-office combien de minutes avant le début de la leçon les clients sont autorisés à entrer dans la salle.

Si la leçon n'a pas encore été réservée, un lien bouton vers la page de réservation s'affiche.



c. Mon compte

La page du compte permet aux utilisateurs de mettre à jour leurs informations.



d. Historique des leçons

Sur la page Historique des leçons, le client peut vérifier son historiques de leçons, le nombre total de leçons et la durée des leçons.

Lesson history

Number of Lessons : 40 times.
Total lesson time : 1000 min.

Lesson Id	Lesson date	Lesson time
369	2022-07-16 (Saturday)	17:30 - 17:55
301	2022-07-13 (Wednesday)	13:30 - 13:55
354	2022-07-10 (Sunday)	10:00 - 10:25
309	2022-07-10 (Sunday)	00:00 - 00:25
318	2022-07-09 (Saturday)	22:00 - 22:25
313	2022-07-09 (Saturday)	16:00 - 16:25
273	2022-07-09 (Saturday)	12:30 - 12:55
272	2022-07-09 (Saturday)	12:00 - 12:25
285	2022-07-09 (Saturday)	10:30 - 10:55
266	2022-07-09 (Saturday)	08:30 - 08:55
284	2022-07-08 (Friday)	22:00 - 22:25
281	2022-07-08 (Friday)	18:30 - 18:55
265	2022-07-08 (Friday)	18:00 - 18:25
262	2022-07-08 (Friday)	15:30 - 15:55
261	2022-07-08 (Friday)	15:00 - 15:25

e. Faire une réservation

La page de réservation indique le calendrier de disponibilité de l'administrateur pour la semaine à venir, et les clients peuvent choisir la date et l'heure qu'ils souhaitent réserver.

Si le client possède un ticket, il peut effectuer une réservation, mais s'il n'a pas de ticket, il ne peut pas effectuer de réservation.

Lesson Reservation

Click the button for the date and time you wish to reserve a lesson.
Blue is the available time. Your reservation will be shown in orange.
Lesson tickets are required for reservations

Saturday 2022-07-16	21:00
Sunday 2022-07-17	17:00 17:30
Monday 2022-07-18	20:00 20:30 21:00
Tuesday 2022-07-19	20:00 20:30 21:00
Wednesday 2022-07-20	19:00 19:30 20:30 21:00

The screenshot shows a 'Lesson Reservation' interface. On the left, a vertical calendar lists days from Saturday, 2022-07-16 to Wednesday, 2022-07-20. Each day has a list of available times. A modal window titled 'Reservation Confirmation' is open, displaying the message: 'There are not enough tickets, please purchase them'. It contains fields for 'Date and time' (2022-07-16 21:00), 'Number of tickets required' (1), and 'Number of tickets you currently have' (0). At the bottom of the modal are three buttons: 'Cancel' (red), 'Buy Tickets' (yellow), and 'Confirm Reservation' (green).

f. Voir mes réservations

Le client peut consulter toutes les réservations à venir sur la page "Ma réservation". Pour les réservations de leçons qui sont sur le point de commencer ou qui ont déjà commencé, un bouton apparaîtra pour le diriger vers la salle de cours.

The screenshot shows a 'My Reservation' page. It displays a table of upcoming lessons:

Lesson ID	Date	Time	
385	Saturday 2022-07-16	20:30 - 20:55	Take Lesson
377	Wednesday 2022-07-20	20:00 - 20:25	

At the bottom of the page, there is footer text: 'Let's talk Japanese !' followed by links to 'Contact', 'Privacy Policy', and 'Terms of use'. There are also social media icons for Twitter, Facebook, Instagram, and YouTube.

g. Acheter des tickets

Le client peut acheter des tickets sur la page "Acheter des tickets". Le client sélectionne le nombre de tickets qu'il souhaite acheter et clique sur le bouton, ce qui l'amène à l'écran de paiement par Stripe. Si le paiement Stripe est réussi, il reviendra à l'écran original d'achat de tickets et un message s'affichera pour lui indiquer que le processus s'est déroulé avec succès.

Buy Tickets

Payment successfully completed.

You have actually **10** tickets [Book now](#)

Select the number of tickets

- 1 lesson 5.00 €
- 5 lessons 24.00 €
- 10 lessons 40.00 €
- 20 lessons 70.00 €

[Buy now](#)

Let's talk Japanese !

[Contact](#) [Privacy Policy](#) [Terms of use](#)

h. Historique des achats de ticket

Le client peut vérifier l'historique de leurs achats de billets.

Ticket Purchase History

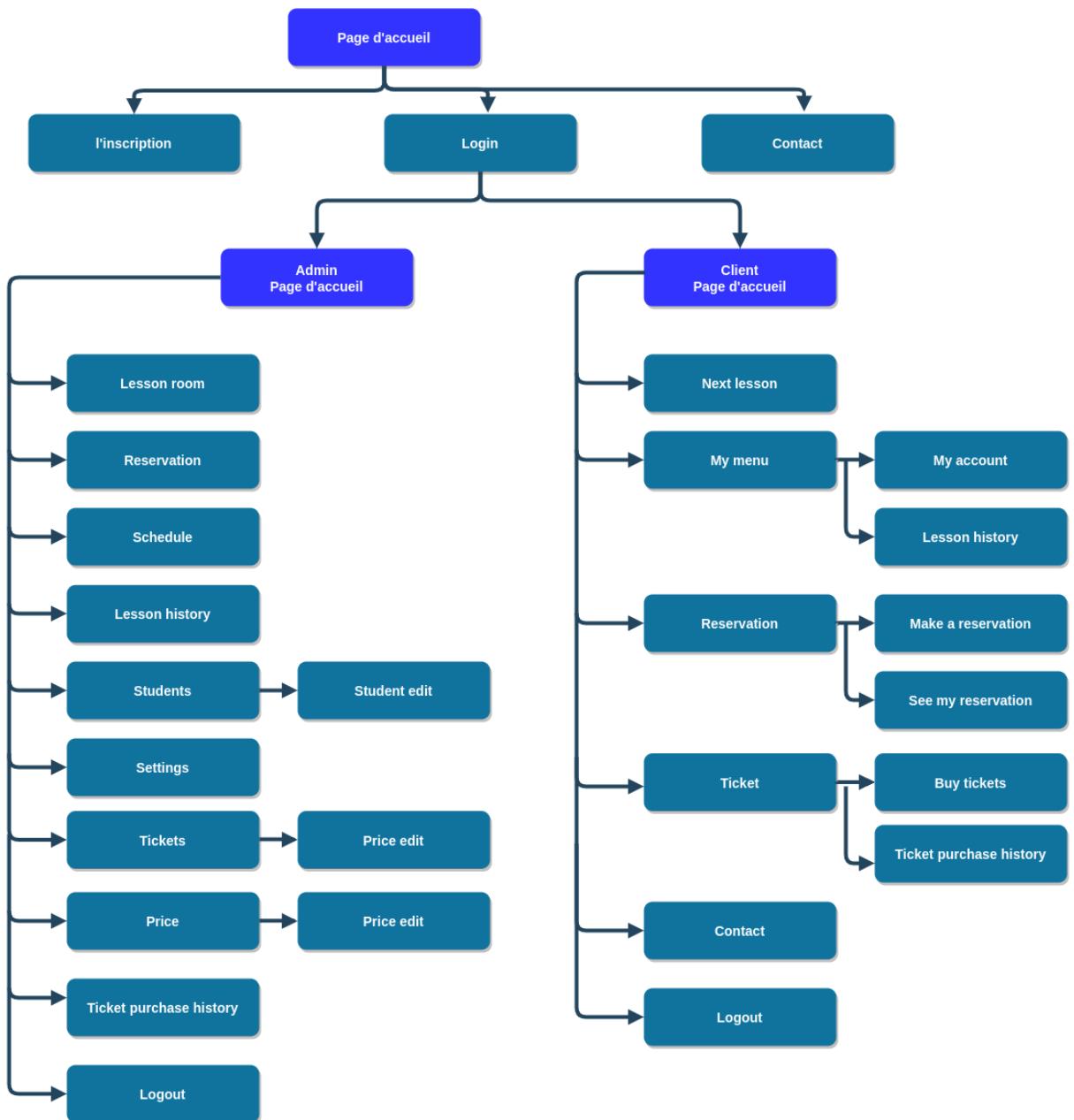
Purchase Id	Purchase Date	Purchase Item	Price
23	2022-07-10 11:09	05 lesson(s)	24 €
12	2022-07-07 12:36	05 lesson(s)	24 €
10	2022-07-07 12:15	01 lesson(s)	05 €
9	2022-07-07 11:52	05 lesson(s)	24 €

Let's talk Japanese !

[Contact](#) [Privacy Policy](#) [Terms of use](#)

SPÉCIFICATIONS TECHNIQUES

Structure des pages



Choix des technologies

1. Language de programmation

- a. PHP
- b. Javascript
- c. Sass

2. Framework

- a. Symfony
- b. Bootstrap

3. Plugin

- a. File upload (vich/uploader-bundle)
- b. Mail (symfony/mailer, twig / extra-bundle twig/cssinliner-extra)
- c. Paginator (knplabs/knp-paginator-bundle)

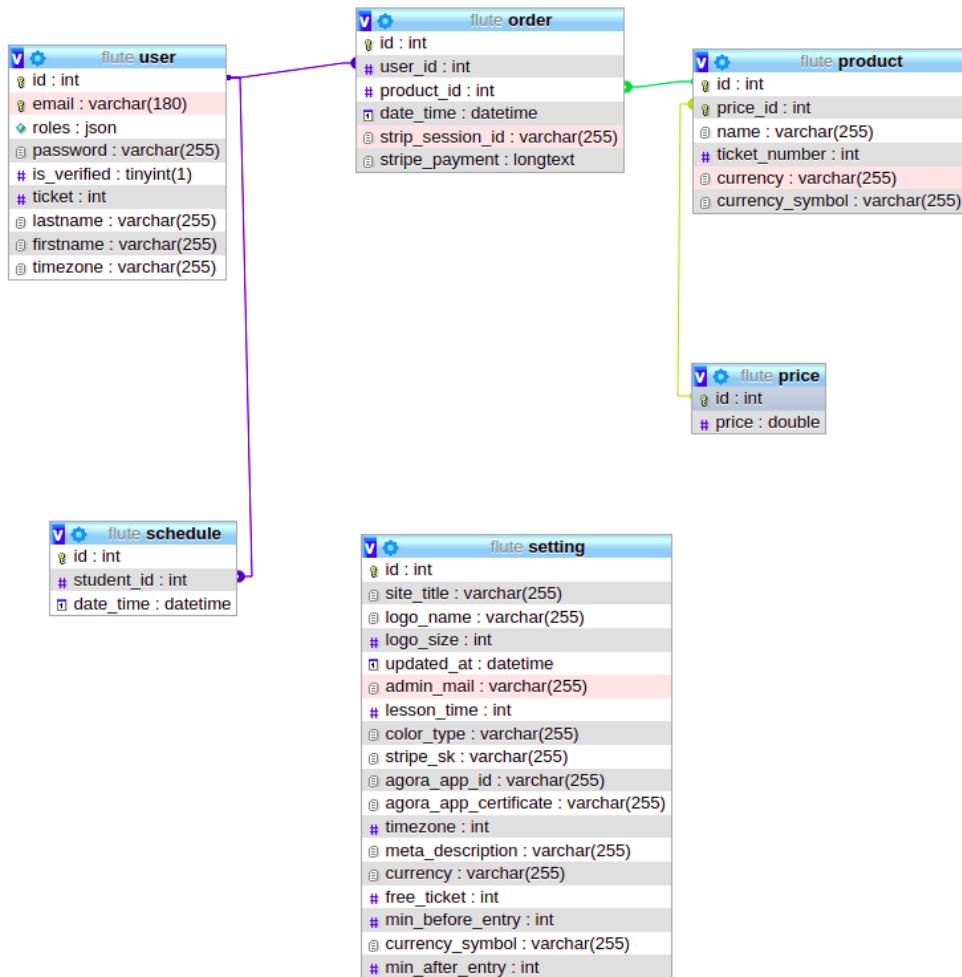
4. API

- a. Traitement des paiements (Stripe)
- b. Vidéo call (Agora)

5. Outil de développement

- a. ngrok(application multiplateforme qui expose les ports des serveurs locaux à Internet.)

Schéma de données

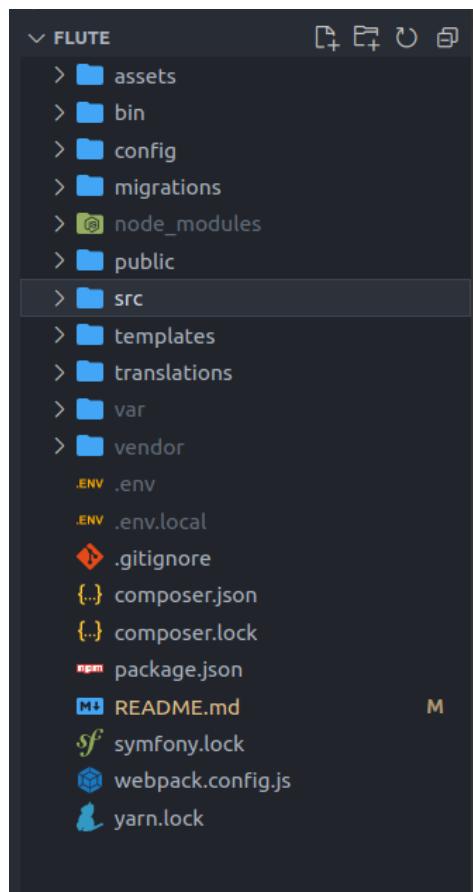


Architecture

1. Structure globale du projet

Le projet utilise le framework Symfony et est conçu selon le modèle MVC.

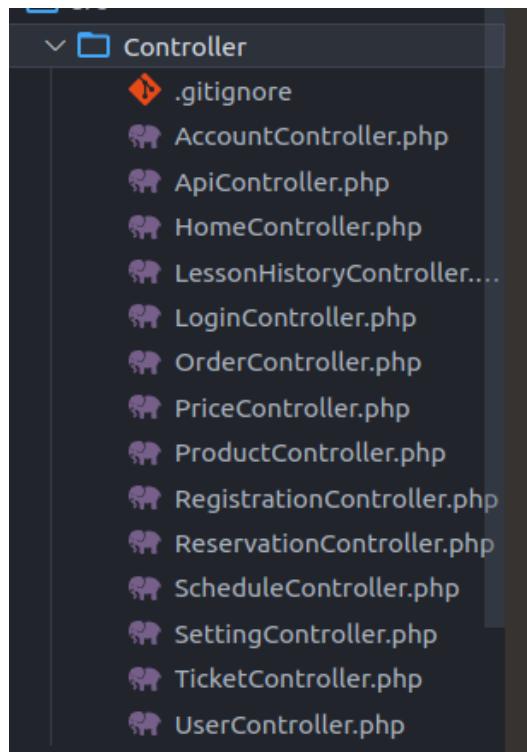
L'architecture MVC (Model-View-Controller), est l'une des architectures logicielles les plus utilisées pour les applications Web. Elle permet de créer une application web pour bien gérer la structuration d'un projet en trois parties. [La base de données \(modèles\)](#), [la page html \(Vue\)](#) et tout [le traitement \(Contrôleur\)](#).



2. Controller

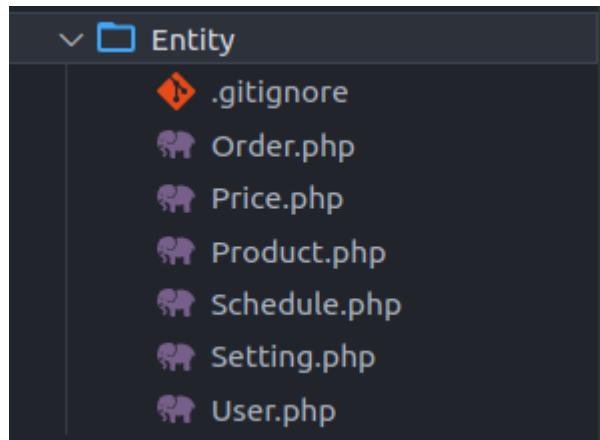
Controller s'occupe le traitement des données. J'ai divisé les contrôleurs en fonction du type de contrôle.

Le système utilise l'API pour activer/désactiver les créneaux horaires des administrateurs, obtenir le nombre de tickets détenus par les utilisateurs, et les webhooks stripe. Tous les processus qui acceptent les requêtes API ont été regroupés dans le ApiController.



3. Entity

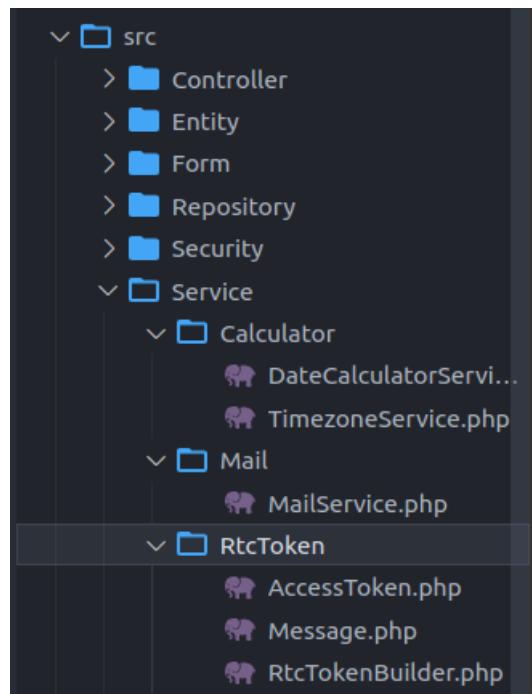
Une entité est un type d'objet qui sera manipulé par l'application Web Symfony. Les entités sont converties en tables de la base de données, à partir du schéma.



4. Service

Si un contrôleur contient trop de logique, la lisibilité en pâtit. J'ai donc créé un service, qui est une collection de logique spécifique, qui peut être utilisé par plusieurs

contrôleurs.



Exemples de codes

1. Contrôle du fuseau horaire

Si les administrateurs et les utilisateurs vivent dans des fuseaux horaires différents, la date et l'heure seront affichées différemment. Pour contrôler cela, j'ai implémenté les processus suivants.

a. TimezoneService.php (Service)

La logique calcule le décalage horaire entre le fuseau horaire de l'utilisateur et le fuseau horaire GMT.

```
8 class TimezoneService
9 {
10     protected $settingRepository;
11
12     // List of seconds difference from standard time(GMT)
13     // Associative array ["Europe/Paris" => 7200]
14     public $timezoneTable;
15
16     // List of timezone
17     // Associative array ["Europe/Paris" => "Europe/Paris"]
18     public $timezoneList;
19
20     public function __construct(SettingRepository $settingRepository)
21     {
22         $this->settingRepository = $settingRepository;
23         $this->timezoneTable = [];
24         $this->timezoneList = [];
25         $timezone = DateTimeZone::listIdentifiers();
26
27         foreach ($timezone as $val) {
28
29             date_default_timezone_set($val);
30             date_default_timezone_get();
31             $localTime = strtotime(date("Y-m-d H:i:s"));
32             $gmtTime = strtotime(gmdate("Y-m-d H:i:s"));
33             $diff = $localTime - $gmtTime;
34             $this->timezoneList[$val] = $val;
35             $this->timezoneTable[$val] = $diff;
36         }
37     }
38 }
```

```
39     public function getGmtTimeDifference($timezone)
40     {
41         $timeDifference = $this->timezoneTable[$timezone];
42
43         if ($timeDifference > -1) {
44             $strTimeDifference = '+' . strval($timeDifference) . ' second';
45         } else {
46             $strTimeDifference = strval($timeDifference) . ' second';
47         }
48         return $strTimeDifference;
49     }
50
51     public function getGmtTime($time, $timezone)
52     {
53         if ($this->timezoneTable[$timezone] > -1) {
54             $timeDiff = (strval($this->timezoneTable[$timezone]));
55             $modifyDiff = "-$timeDiff second";
56         } else {
57             $timeDiff = (strval(($this->timezoneTable[$timezone]) * -1));
58             $modifyDiff = "+$timeDiff second";
59         }
60         $gmtTime = $time->modify($modifyDiff);
61         return $gmtTime;
62     }
63 }
64
```

b. ReservationController.php (Controller)

Le programme affiche une liste des réservations des utilisateurs. Il récupère les données de réservation pour la période concernée (la semaine à venir) dans la schedule table. Les données sont ensuite transmises à Vue.

```
13 class ReservationController extends AbstractController
14 {
15     /**
16      * Displays a list of user reservations
17      * @Route("/reservation", name="app_reservation")
18      */
19     public function index(
20         TimezoneService $timezoneService,
21         DateCalculatorService $dateCalculatorService,
22         SettingRepository $settingRepository,
23         ScheduleRepository $scheduleRepository
24     ): Response {
25         $this->denyAccessUnlessGranted('IS_AUTHENTICATED_FULLY');
26
27         // Calculate the difference between user time and GMT time
28         $timeDifference = $timezoneService->getGmtTimeDifference($this->getUser()->getTimezone());
29
30         // Get GMT dates for 1 week from today
31         $list[$from, $to] = $dateCalculatorService->getGmtPeriod(true);
32
33         // Search the schedule repository for a user ID and the one-week time period just retrieved
34         $myReservation = $scheduleRepository->findByWeekAndUser($from, $to, $this->getUser());
35
36         return $this->render('home/reservation_list_user.html.twig', [
37             'now' => $from,
38             'myReservation' => $myReservation,
39             'setting' => $settingRepository->find(1),
40             'timeDifference' => $timeDifference
41         ]);
42     }
}
```

c. reservation_list.html.twig (templates)

Ce programme affiche les réservations de l'utilisateur pour la semaine en tenant compte des décalages horaires.

```

1 <div class="title">
2   <h1>My Reservation</h1>
3 </div>
4 {%- if myReservation %}
5   <p class="mt-5">You can enter the classroom
6     {{setting.minBeforeEntry}}
7     minutes before the lesson start. Please click the button and proceed to the lesson.</p>
8 {%- endif %}
9 {%- set lessonTime = '+' ~ setting.lessonTime ~ ' minute' %}
10 {%- set minBeforeEntry = '+' ~ setting.minBeforeEntry ~ ' minute' %}
11
12 <table class="table">
13   <thead>
14     <tr>
15       <th>Lesson ID</th>
16       <th>Date</th>
17       <th>Time</th>
18       <th></th>
19       <th></th>
20     </tr>
21   </thead>
22   <tbody>
23     {% for eachReservation in myReservation %}
24       <tr>
25         <td>
26           {{eachReservation.id}}
27         </td>
28         <td>
29           {{eachReservation.dateTime|date_modify(timeDifference)|date('l')}}<br>
30           {{eachReservation.dateTime|date_modify(timeDifference)|date('Y-m-d')}}<br>
31         </td>
32         <td>
33           {{eachReservation.dateTime|date_modify(timeDifference)|date('H:i')}}<br>
34           -
35           {{eachReservation.dateTime|date_modify(timeDifference)|date_modify(lessonTime)|date('H:i')}}<br>
36         </td>
37         {% if is_granted('ROLE_ADMIN') %}
38           <td class="table-row">
39             {{eachReservation.student.lastname}}
40             {{eachReservation.student.firstname}}
41           </td>
42         {% else %}
43           <td></td>
44         {% endif %}
45         <td>
46           {% if eachReservation.dateTime|date_modify(timeDifference) <= "now"|date_modify(timeDifference)|date_modify(minBeforeEntry) %}
47             {% if is_granted('ROLE_ADMIN') %}
48               <a class="btn btn-primary" href="{{ path('app_home') }}">Begin class</a>
49             {% else %}
50               <a class="btn btn-primary" href="{{ path('app_home') }}">Take Lesson</a>
51             {% endif %}
52           {% endif %}
53         </td>
54       </tr>
55     {% else %}
56       <tr>
57         <td>
58           <p>No reservations yet</p>
59           {% if not is_granted('ROLE_ADMIN') %}
60             <a href="{{path('app_reservation_new')}}" class="btn btn-info btn_flat mt-5">Book now</a>
61           {% endif %}
62         </td>
63       </tr>
64     {% endfor %}
65   </tbody>
66 </table>
67

```

2. Contrôle l'activation et la désactivation des créneaux horaires.

L'interface où les administrateurs saisissent le calendrier des disponibilités est représentée par un tableau de 24 heures sur 1 semaine. Comme il n'est pas pratique de recharger la page chaque fois que les administrateurs ouvrent ou ferment un créneau, Ajax a été utilisé pour contrôler le rafraîchissement de la page.

a. Schedule.js

Modifier l'état des boutons d'affichage après la mise à jour du schedule table.

```
● ● ●  
1 const location = window.location.hostname;  
2  
3 let scheduleOnOffApiurl = `https://${location}/admin/api/schedule/update`;  
4 let reservationApiurl = `https://${location}/api/reservation/update`;  
5 let ticketCountApiurl = `https://${location}/api/ticket/count`;  
6  
7 if (location == "localhost") {  
8   scheduleOnOffApiurl = `http://${location}:3000/admin/api/schedule/update`;  
9   reservationApiurl = `http://${location}:3000/api/reservation/update`;  
10  ticketCountApiurl = `http://${location}:3000/api/ticket/count`;  
11 }
```

```
54 updateSchedule = async (date, clickElement) => {
55   const data = {
56     "date-time": date,
57   };
58   const settings = {
59     method: "POST",
60     headers: {
61       Accept: "application/json",
62       "Content-Type": "application/json",
63     },
64     body: JSON.stringify(data),
65   };
66   try {
67     const fetchResponse = await fetch(scheduleOnOffApiurl, settings);
68     const data = await fetchResponse.json();
69
70     clickElement.innerHTML = data.status;
71
72     if (data.status == "Available") {
73       clickElement.classList.remove("btn-secondary");
74       clickElement.classList.add("btn-warning");
75     } else {
76       clickElement.classList.remove("btn-warning");
77       clickElement.classList.add("btn-secondary");
78     }
79     return;
80   } catch (e) {
81     return e;
82   }
83 };
```

b. ApiController.php(public function scheduleUpdate)

Si la date sélectionnée par les admins n'existe pas dans le schedule table, une nouvelle entrée est ajoutée. Sinon, elle met à jour le statut de l'entrée. Si le statut d'un créneau horaire a déjà été réservé par le client, il ne peut être désactivé.

```
141  /**
142   * @Route("/admin/api/schedule/update", name="api_schedule_update", methods={"POST"})
143  */
144 public function scheduleUpdate(
145     Request $request,
146     ManagerRegistry $doctrine,
147     ScheduleRepository $scheduleRepository
148 ): JsonResponse {
149     $data = json_decode($request->getContent(), true);
150
151     $entityManager = $doctrine->getManager();
152     $schedule =
153     $scheduleRepository->findOneByDateTime(\DateTime::createFromFormat('Y-m-d H:i:s', $data['date-time']));
154     if ($schedule) {
155         // This slot has already been reserved by a student and its status cannot be changed
156         if ($schedule->getStudent()) {
157             $status = 'Available';
158         } else {
159             $entityManager->remove($schedule);
160             $entityManager->flush();
161             $status = 'Off';
162         }
163     } else {
164         $schedule = new Schedule();
165         $schedule->setDateTime(\DateTime::createFromFormat('Y-m-d H:i:s', $data['date-time']));
166         $entityManager->persist($schedule);
167         $entityManager->flush();
168         $status = 'Available';
169     }
170
171     return new JsonResponse([
172         'status' => $status,
173         'dateTime' => $data['date-time'],
174     ]);
175 }
```

3. Vidéo call

La fonctionnalité d'appel vidéo a été mise en œuvre à l'aide de l'API Agora.

a. basicVideoCall.js

Tout d'abord, un token pour la connexion est obtenu du serveur. Avec le token obtenu, l'utilisateur peut se connecter au canal et attendre que remote user se connecte.

```
25 const getRtcToken = async (lessonId, dataId) => {
26   const data = {
27     lesson: lessonId,
28     data: dataId,
29   };
30   const settings = {
31     method: "POST",
32     headers: {
33       Accept: "application/json",
34       "Content-Type": "application/json",
35     },
36     body: JSON.stringify(data),
37   };
38   try {
39     const fetchResponse = await fetch(getRtcTokenApiurl, settings);
40     const data = await fetchResponse.json();
41     return data;
42   } catch (e) {
43     return e;
44   }
45 };
```

```
98 window.onload = function () {
99     if (!joinBtn) return;
100    joinBtn.onclick = async function () {
101        // Get token
102        const lessonId = joinBtn.getAttribute("lesson-id");
103        const dataId = joinBtn.getAttribute("data-id");
104        const data = getRtcToken(lessonId, dataId);
105        options = await data;
106
107        // Join an RTC channel.
108        await rtc.client.join(
109            options.appId, //Agora App ID
110            options.channel, // 'lesson'+ userId
111            options.token, // Agora Token
112            options.uid // userId
113        );
114        // Create a local audio track from the audio sampled by a microphone.
115        rtc.localAudioTrack = await AgoraRTC.createMicrophoneAudioTrack();
116        // Create a local video track from the video captured by a camera.
117        rtc.localVideoTrack = await AgoraRTC.createCameraVideoTrack();
118        // Publish the local audio and video tracks to the RTC channel.
119        await rtc.client.publish([rtc.localAudioTrack, rtc.localVideoTrack]);
120        // Dynamically create a container in the form of a DIV element for playing the local video track.
121        const localPlayerContainer = document.createElement("div");
122        // Specify the ID of the DIV container. You can use the uid of the local user.
123        localPlayerContainer.id = options.uid;
124        // localPlayerContainer.textContent = "Local user " + options.uid;
125        localPlayerContainer.style.width = "120px";
126        localPlayerContainer.style.height = "80px";
127        localVideo.append(localPlayerContainer);
128        reservationWrapper.style.display = "none";
129        leaveBtn.style.display = "block";
130        // Play the local video track.
131        // Pass the DIV container and the SDK dynamically creates a player in the container for playing the local video track.
132        rtc.localVideoTrack.play(localPlayerContainer);
133        console.log("publish success!");
134    };
}
```

```
47  async function startBasicCall() {
48    // Create an AgoraRTCClient object.
49    rtc.client = AgoraRTC.createClient({ mode: "rtc", codec: "vp8" });
50
51    // Listen for the "user-published" event, from which you can get an AgoraRTCRemoteUser object.
52    rtc.client.on("user-published", async (user, mediaType) => {
53      // Subscribe to the remote user when the SDK triggers the "user-published" event
54      await rtc.client.subscribe(user, mediaType);
55      console.log("subscribe success");
56
57      // If the remote user publishes a video track.
58      if (mediaType === "video") {
59        // Get the RemoteVideoTrack object in the AgoraRTCRemoteUser object.
60        const remoteVideoTrack = user.videoTrack;
61        // Dynamically create a container in the form of a DIV element for playing the remote video track.
62        const remotePlayerContainer = document.createElement("div");
63        // Specify the ID of the DIV container. You can use the uid of the remote user.
64        remotePlayerContainer.id = user.uid.toString();
65        // remotePlayerContainer.textContent = "Remote user " + user.uid.toString();
66        if(window.innerWidth < 800){
67          remotePlayerContainer.style.width = "500px";
68          remotePlayerContainer.style.height = "333px";
69        }else{
70          remotePlayerContainer.style.width = "900px";
71          remotePlayerContainer.style.height = "600px";
72        }
73        remoteVideo.append(remotePlayerContainer);
74
75        // Play the remote video track.
76        // Pass the DIV container and the SDK dynamically creates a player in the container for playing the remote video track.
77        remoteVideoTrack.play(remotePlayerContainer);
78      }
79
80      // If the remote user publishes an audio track.
81      if (mediaType === "audio") {
82        // Get the RemoteAudioTrack object in the AgoraRTCRemoteUser object.
83        const remoteAudioTrack = user.audioTrack;
84        // Play the remote audio track. No need to pass any DOM element.
85        remoteAudioTrack.play();
86      }
87
88      // Listen for the "user-unpublished" event
89      rtc.client.on("user-unpublished", (user) => {
90        // Get the dynamically created DIV container.
91        const remotePlayerContainer = document.getElementById(user.uid);
92        console.log(remotePlayerContainer);
93        // Destroy the container.
94        remotePlayerContainer.remove();
95      });
96    });
97  }
```

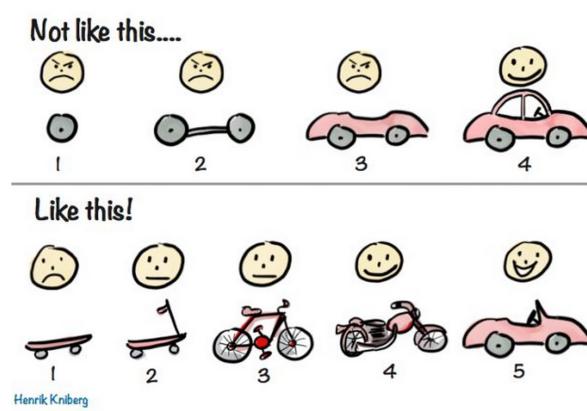
RÉALISATIONS

Organisation du travail

1. Commencer par l'inconnu

En développant ce projet, je me suis assuré commencer par les composants que je n'ai jamais implémentés au par avant. La raison est qu'il est difficile de prévoir le temps nécessaire. Cela inclut le traitement des paiements avec Stripe webhook, et vidéo call avec Agora API. Grâce à la priorité accordée à ces développements, j'ai pu achever le projet à temps.

2. Méthode agile



En développant ce projet, j'ai essayé de suivre une approche de développement agile. Petit à petit, j'ai créé un système qui fonctionne à petite échelle. Cette image est une métaphore du développement en waterfall et du développement agile. Plutôt que de fournir des pneus qui ne servent rien, j'ai essayé de fournir un skateboard qui marche. (voir illustration ci-dessus).

3. Utilisation de commande “make” de Symfony

De nombreuses commandes make ont été utilisées pour réduire le temps de développement.

- a. make:user
- b. make:auth
- c. make:crud
- d. make:entity

Difficultés rencontrées

1. Vérifier mail fonction

J'ai installé verify-email-bundle pour gagner du temps. Il s'agit d'un module qui vérifie les adresses électroniques des utilisateurs nouvellement enregistrés avant qu'ils ne s'inscrivent au processus d'enregistrement définitif.

Le problème est que le concept de ce module est étrange et ne correspond pas au processus d'enregistrement tel que je l'envisage.

Lorsque l'utilisateur remplit les champs obligatoires du formulaire d'inscription, il reçoit un mail de confirmation contenant une URL. Lorsque l'utilisateur clique sur l'URL, l'enregistrement est terminé et l'utilisateur peut se connecter à partir de ce moment-là.

Cependant, avec ce bundle, l'utilisateur doit se connecter avant de cliquer sur l'URL.

Permettre aux utilisateurs sans adresse électronique vérifiée de se connecter est une erreur de principe.

J'ai passé beaucoup de temps avant de me rendre compte de ce problème.

Nous utilisons une variété de modules, à la fois "open source" et non "open source", dans le développement de nos applications. Cependant, j'ai remarqué qu'il faut être prudent lors de l'utilisation de ces modules.

2. Stripe webhook

J'avais déjà créé un processus de paiement à l'aide de Stripe sur un site e-commerce, mais à l'époque, je n'avais pas suffisamment appris sur le traitement des paiements en ligne et j'avais créé un processus de paiement qui n'utilisait pas de webhook.

Il existe différents types d'utilisateurs du web, par exemple les utilisateurs qui cliquent sur un bouton de paiement puis ferment instantanément leur navigateur. En outre, tous les terminaux web ne disposent pas de connexions stables à tout moment.

J'ai utilisé stripe webhook pour compléter le processus de paiement dans diverses situations, mais il a fallu un certain temps pour comprendre son fonctionnement.

3. Agora vidéo call

C'est la première fois que j'utilise Agora Api. Il est facile à utiliser et bien documenté. Mais c'était quand même ma première fois, alors j'ai eu un peu de mal à m'y faire.

Ce qu'il me reste à faire

1. Vérifier mail fonction
2. Ajoutez des fonctions de partage d'écran et de chat pendant la leçon
3. Ajouter un processus pour le cas où un utilisateur oublierait son mot de passe.
4. Support multilingue
5. Color type(Permet au administrateur de personnaliser le site en fonction de sa couleur préférée.)
6. Durée de la leçon : 50 minutes.
7. Si je pouvais trouver un graphiste pour améliorer le design du site...

CONCLUSION

J'ai beaucoup appris grâce à ce projet. J'ai acquis une meilleure compréhension de PHP et de Symfony. J'ai aussi appris à rédiger le dossier projet.

Cependant, j'ai également ressenti les limites du développement d'applications par mes propres moyens. En particulier, la conception UI et l'intégration front-end sont mes points faibles. J'espère donc surmonter mes faiblesses et trouver des bons partenaires.