

TP 5

EFREI 3A WEB

Louis Cherel - 08/2020

Préambule

Merci de lire **toutes les consignes**, elles sont écrites pour être lues.

Les TP sont tous ramassés

Tous les TP sont ramassés et doivent être rendus au plus tard à 23h55 le soir de la séance sur l'espace Moodle prévu à cet effet.

Chaque jour de retard vous retire 1 point sur votre note de TP, avec une limite à 3 jours de retard, ce qui, passé ce délai, vous vaudra une note de 0 sur votre TP.

Seul un des TP sera noté, mais ne pas rendre les autres vous vaudra quand même 0.

Les TP et le projet sont à réaliser en binôme ou trinômes, et doivent être enregistrés auprès de votre enseignant·e avant la fin de la première séance. Sans votre enregistrement d'équipe, vous ne pourrez pas rendre votre travail.

Les ressources conseillées

Vos armes seront donc le site officiel de Vue.js, Mozilla Developer Network (MDN), Stack Overflow, et les recherches en anglais. **Évitez au maximum** les forums (pas les cours) des sites comme OpenClassrooms ou Comment Ça Marche, qui proposent des solutions souvent peu fiables. W3Schools peut être une bonne ressource, mais les bonnes pratiques et les standards ne sont pas leur fort. Leur préférer le MDN lorsque possible.

Quantité de blogs sont très bons, notamment alsacreations (en français) ou css-tricks (en anglais).



Cette œuvre est mise à disposition par Louis Cherel selon les termes de la [Licence Creative Commons Attribution - Pas d'Utilisation Commerciale - Partage dans les Mêmes Conditions 4.0 International](#)

Installations préalables

Pour réaliser ce TP, vous aurez besoin d'installer sur votre ordinateur:

- [PostgreSQL](#): ce sera notre système de bases de données
 - Lors de l'installation, vous devez installer postgres, pgadmin et command-line tools; Stackdriver n'est pas requis



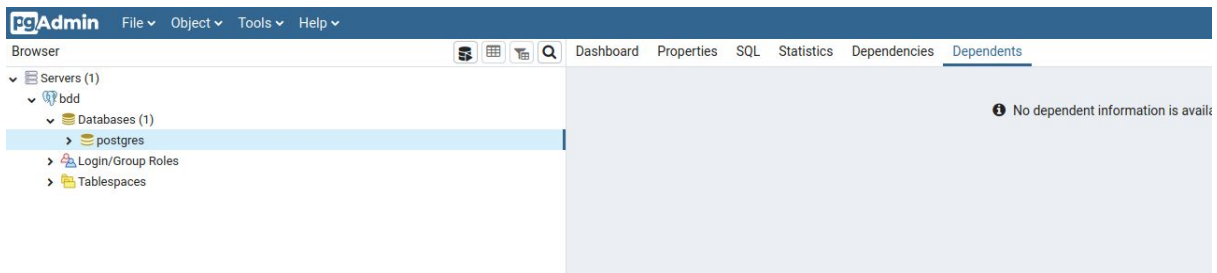
Cette œuvre est mise à disposition par Louis ChereI selon les termes de la [Licence Creative Commons Attribution - Pas d'Utilisation Commerciale - Partage dans les Mêmes Conditions 4.0 International](#)

Exercices

Exercice 1: Configurer Postgres (🕒 20 min pratique)

L'objectif de cet exercice est de créer des tables SQL grâce à pgadmin

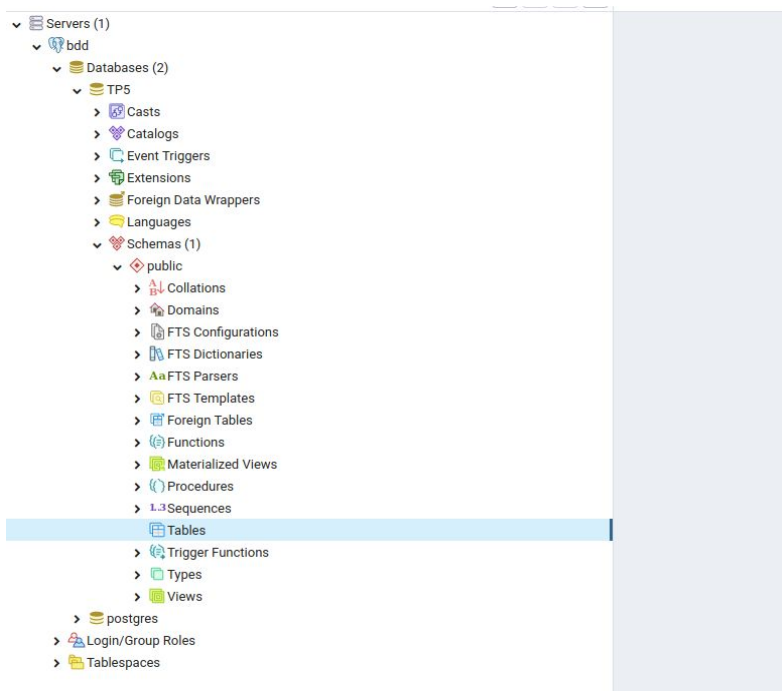
- 1) Ouvrez pgadmin, qui est normalement installé en même temps que postgres



- 2) Le nom "bdd" peut varier selon votre système, mais vous devriez avoir une arborescence similaire à celle ci-dessus. Créez une base de données en faisant clic droit sur "Databases" > Create... > database

a) Appelez votre database TP5, et cliquez sur Save

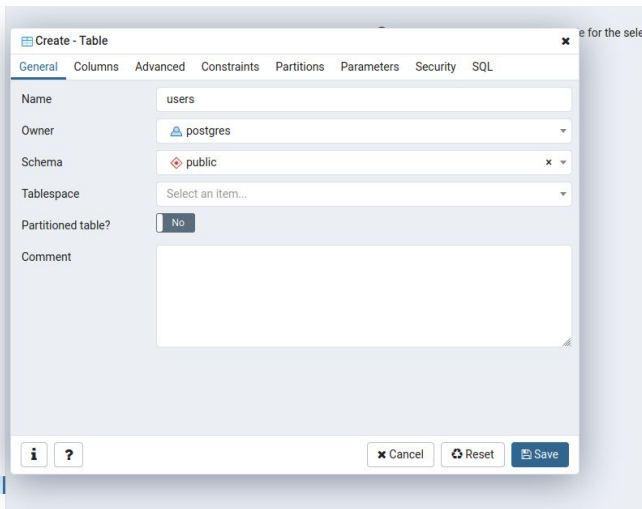
- 3) Dans l'arborescence de TP5, Suivez le chemin suivant: Schemas > public > Tables



Cette œuvre est mise à disposition par Louis ChereI selon les termes de la [Licence Creative Commons Attribution - Pas d'Utilisation Commerciale - Partage dans les Mêmes Conditions 4.0 International](https://creativecommons.org/licenses/by-nc-sa/4.0/)

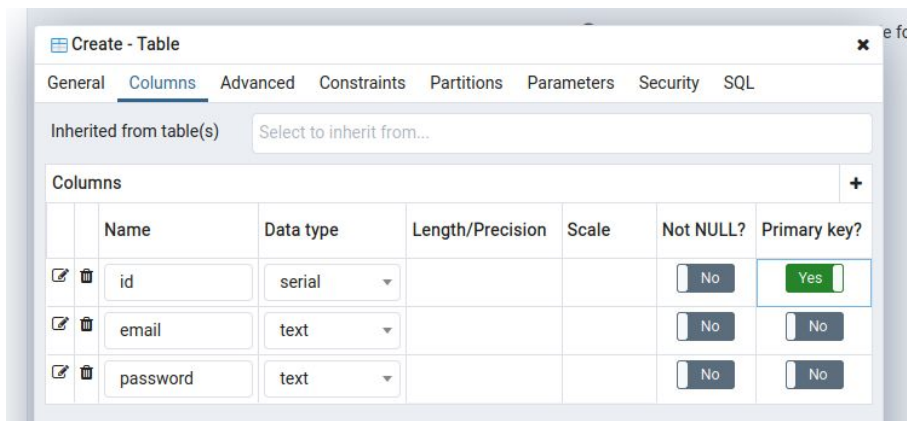
4) Faites clic droit sur Tables > Create... > Table

a) Appelez votre table users



b) Dans l'onglet columns, ajoutez (via le + à droite de l'interface) les colonnes suivantes:

- i) id; type: serial, primary key
- ii) email; type: text
- iii) password; type: text



	Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?
	id	serial			<input type="checkbox"/> No	<input checked="" type="checkbox"/> Yes
	email	text			<input type="checkbox"/> No	<input type="checkbox"/> No
	password	text			<input type="checkbox"/> No	<input type="checkbox"/> No

c) Sauvegardez la table

5) Félicitations, votre Postgres est configuré



Cette œuvre est mise à disposition par Louis Chere! selon les termes de la [Licence Creative Commons Attribution - Pas d'Utilisation Commerciale - Partage dans les Mêmes Conditions 4.0 International](https://creativecommons.org/licenses/by-nc-sa/4.0/)

Exercice 2: Inscription (🕒 1h pratique)

Lecture préliminaire: [Slides - 5. Authentification et Bases de données](#)

1) Repartez de votre TP4

2) Dans votre dossier de projet, installez les modules pg et bcrypt grâce à la commande suivante: `$npm install pg bcrypt`

3) En haut de votre fichier server/routes/api.js, initialisez la connexion à postgres avec ce code (mettez-le après les require de début de fichier, mais avant les routes):

```
const bcrypt = require('bcrypt')
const { Client } = require('pg')
```

```
const client = new Client({
  user: 'postgres',
  host: 'localhost',
  password: 'yourpassword',
  database: 'TP5'
})
```

```
client.connect()
```

a) Remplacez le champ "password" par votre propre mot de passe

4) Dans le fichier server/routes/api.js, ajoutez une nouvelle route POST dont l'url est /register . L'objectif de cette route est d'inscrire un utilisateur:

a) Cette route doit recevoir deux paramètres dans body: email et password

b) Dans la route, grâce à une requête sql (cf [les slides](#)), vérifier si un utilisateur avec cet email n'existe pas déjà dans la table `users`

i) Si l'utilisateur existe, retourner une erreur

c) Si tout est bon, hasher le mot de passe grâce à la fonction `bcrypt.hash`, exemple d'utilisation:

```
const hash = await bcrypt.hash(password, 10)
```

d) Stocker enfin l'utilisateur avec son mot de passe hashé grâce à une requête sql de type INSERT INTO

5) Vérifiez que votre route fonctionne grâce à Postman



Cette œuvre est mise à disposition par Louis Chere! selon les termes de la [Licence Creative Commons Attribution - Pas d'Utilisation Commerciale - Partage dans les Mêmes Conditions 4.0 International](#)

- 6) Côté Vue.js, créez un nouveau composant Register.vue, et associez-lui la route /register
 - a) Faites en sorte que ce composant crée l'utilisateur correctement en utilisant axios

Exercice 3: Connexion (🕒 1h pratique)

- 1) Implémentez la route POST /login
 - a) Vérifiez que l'utilisateur existe, que la forme hashée du mot de passe correspond à ce qui est base avec bcrypt.compare (cf les slides, toujours)
 - b) Lorsque l'utilisateur est validé, sauvez son champ id dans req.session.userId
 - c) Si l'utilisateur est déjà authentifié, retournez une erreur 401
 - d) Vérifiez que votre route fonctionne avec Postman
- 2) Côté Vue, créez un nouveau composant Login.vue
 - a) Faites en sorte que l'utilisateur puisse se connecter à l'application grâce à ce composant

Exercice 4: Who am I (🕒 20m pratique)

- 1) Dans server/routes/api.js, créez une route GET /me, qui retourne simplement l'utilisateur actuellement connecté
 - a) Si l'utilisateur n'est pas connecté, lui retourner une erreur 401
 - b) Bien penser à NE PAS ENVOYER LE MOT DE PASSE, même hashé
- 2) Vous pouvez maintenant vous servir de cette route dans Vue pour vérifier si l'utilisateur est déjà connecté



Cette œuvre est mise à disposition par Louis ChereI selon les termes de la [Licence Creative Commons Attribution - Pas d'Utilisation Commerciale - Partage dans les Mêmes Conditions 4.0 International](https://creativecommons.org/licenses/by-nc-sa/4.0/)

Exercice 5: Seuls les utilisateur connectés peuvent payer (🕒 20 min pratique)

- 1) Côté server/routes/api.js, faites en sorte que la route POST /panier/pay ne reçoive plus les identifiants de l'utilisateur, mais faites simplement en sorte que:
 - a) Si l'utilisateur est connecté, alors accepter
 - b) Si l'utilisateur n'est pas connecté, refuser
- 2) Côté Vue.js, gérer le cas d'erreur et renvoyer l'utilisateur sur la page de login si l'utilisateur n'est pas connecté
 - a) Proposez depuis la page de login un lien amenant vers la page d'inscription si l'utilisateur ne s'était pas encore inscrit

Conclusion

Durant tous ces TPs, vous avez vu les éléments essentiels qui permettent de constituer un serveur Web et une application Web avec un framework front-end.

Malgré la difficulté d'absorber toutes ces notions aussi rapidement, vous êtes maintenant capables de comprendre une architecture de site simple, **moderne**.

Pour aller plus loin

Exercice 6: Tout est SQL (🕒 2h pratique)

- 1) Créez une table articles dans pgadmin
- 2) Remplacez le tableau javascript "articles" par son équivalent en SQL dans tout le fichier server/routes/api.js



Cette œuvre est mise à disposition par Louis ChereI selon les termes de la [Licence Creative Commons Attribution - Pas d'Utilisation Commerciale - Partage dans les Mêmes Conditions 4.0 International](https://creativecommons.org/licenses/by-nc-sa/4.0/)