**Comprendre le web (www -> World Wide Web )**

Internet (arpanet 1969) avant le web

Internet -> réseau et communication entre ordinateurs pour l’armé et les universités

Services internet

Web (1991) -> Consulter des pages

Obtenir des infos

E-mails (1972) -> échanger des messages

Newsgroups -> communiquer dans des forums

FTP -> échanger des fichiers

Partez à la recherche des premiers sites web ! Votre mission, si vous l'acceptez, consistera à retrouver les premières versions des sites web suivants :

* la première version de Google ;
* la première version de W3C ;
* un moteur de recherche avant Google ;
* une boutique de vente en ligne comme Amazon ;
* le tout, tout premier site web créé par Tim lui-même.

Réponse :

* [Google](https://sdz-upload.s3.amazonaws.com/prod/upload/Cour0.11.PNG)
* [W3C](https://sdz-upload.s3.amazonaws.com/prod/upload/w3c14.png)
* [Moteur de recherche avant Google](https://sdz-upload.s3.amazonaws.com/prod/upload/Altavista%20%28Moteur%20de%20Recherche%20Avant%20Google%29.PNG), ici par exemple AltaVista
* [Une boutique de vente en ligne de livres](https://sdz-upload.s3.amazonaws.com/prod/upload/Amazon_1994.jpg), on pense forcément à Amazon même si d'autres sites sont acceptables.
* [Le tout premier site web](https://sdz-upload.s3.amazonaws.com/prod/upload/Cour0.72.PNG)

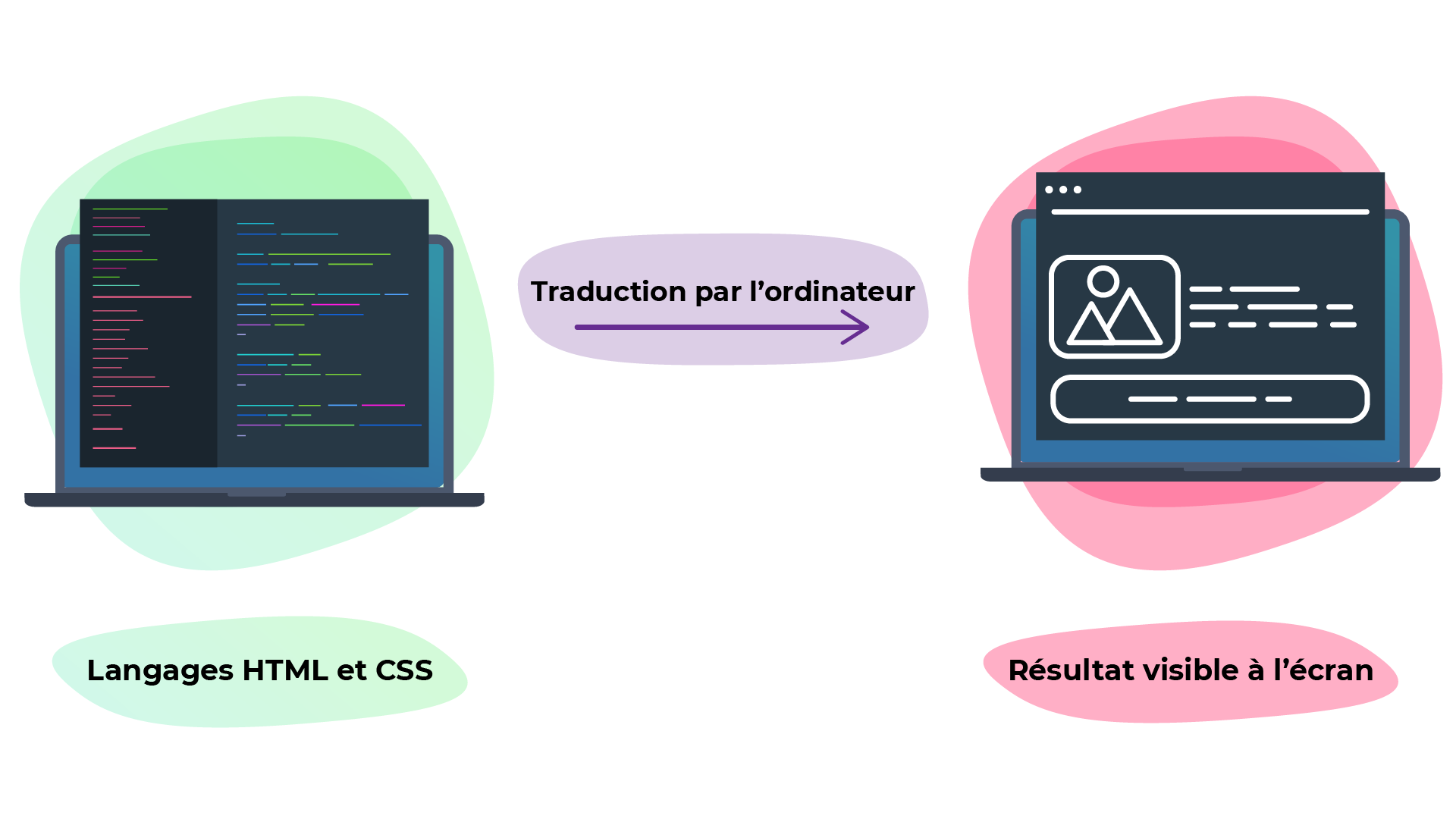
**les langages clients / serveurs**

* **Des clients :** c’est vous, votre ordinateur qui sert à aller consulter des sites web.
* **Des serveurs :** ce sont des ordinateurs spéciaux (souvent très puissants) qui envoient les sites web aux clients. Les serveurs "possèdent" les sites web et les distribuent à ceux qui veulent les visiter.
* Clients ou frontend :
  + HTML
  + CSS
  + JavaScript

### ****Les navigateurs :****

* Google Chrome
* Mozilla Firefox
* Internet Explorer, et son successeur appelé Edge
* Opera
* Safari

Le rôle des navigateurs est de traduire les langages HTML, CSS et JavaScript sous la forme de sites web utilisables par tout le monde.



<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<title>Ma page web</title>

</head>

<body>

<p>Bienvenue sur mon site web !</p>

</body>

</html>

Test ecriture HTML :

1. Modifiez le paragraphe de bienvenue pour indiquer votre nom (remplacez les XXXX)
2. Ajoutez un second paragraphe à la suite sur le même modèle, avec les balises <p> et </p> du code HTML qui délimitent un paragraphe.
3. Ajoutez des balises <strong> et </strong> pour délimiter le mot "Bienvenue" comme ceci:  <strong>Bienvenue</strong>  . Que se passe-t-il ?

## les langages serveur

Les langages serveur sont nombreux. En voici quelques exemples :

* PHP
* Java (rien à voir avec JavaScript, attention !)
* Python
* C#
* ...

### ****Les frameworks****

En plus de ces langages, on a inventé des frameworks pour faciliter la création de sites web dans ces langages.

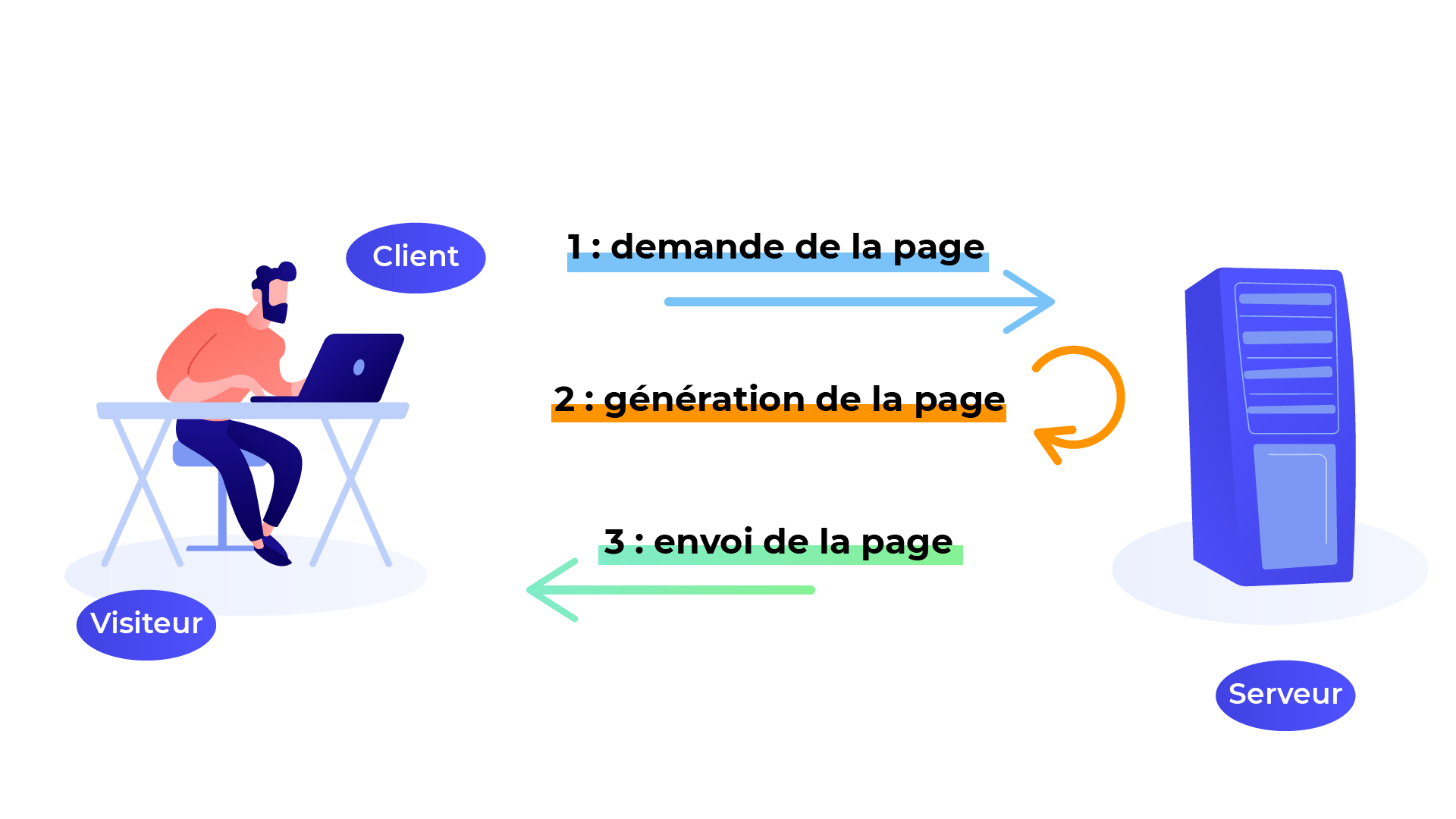
Les frameworks sont des boîtes à outils qui se révèlent de plus en plus indispensables aujourd’hui. De la même façon que l’on n’envisage plus de creuser un trou "à la main" (on préfère utiliser une pelleteuse !), on n’envisage plus trop de construire un site web à la main. On se fait aider un peu par un framework !

Les frameworks se rapportent à des langages. On peut citer :

* Pour PHP : Symfony, Zend...
* Pour Java : Java EE (ou J2EE)
* Pour Python : Django
* Pour C# : ASP .NET
* Pour JS : Vue JS

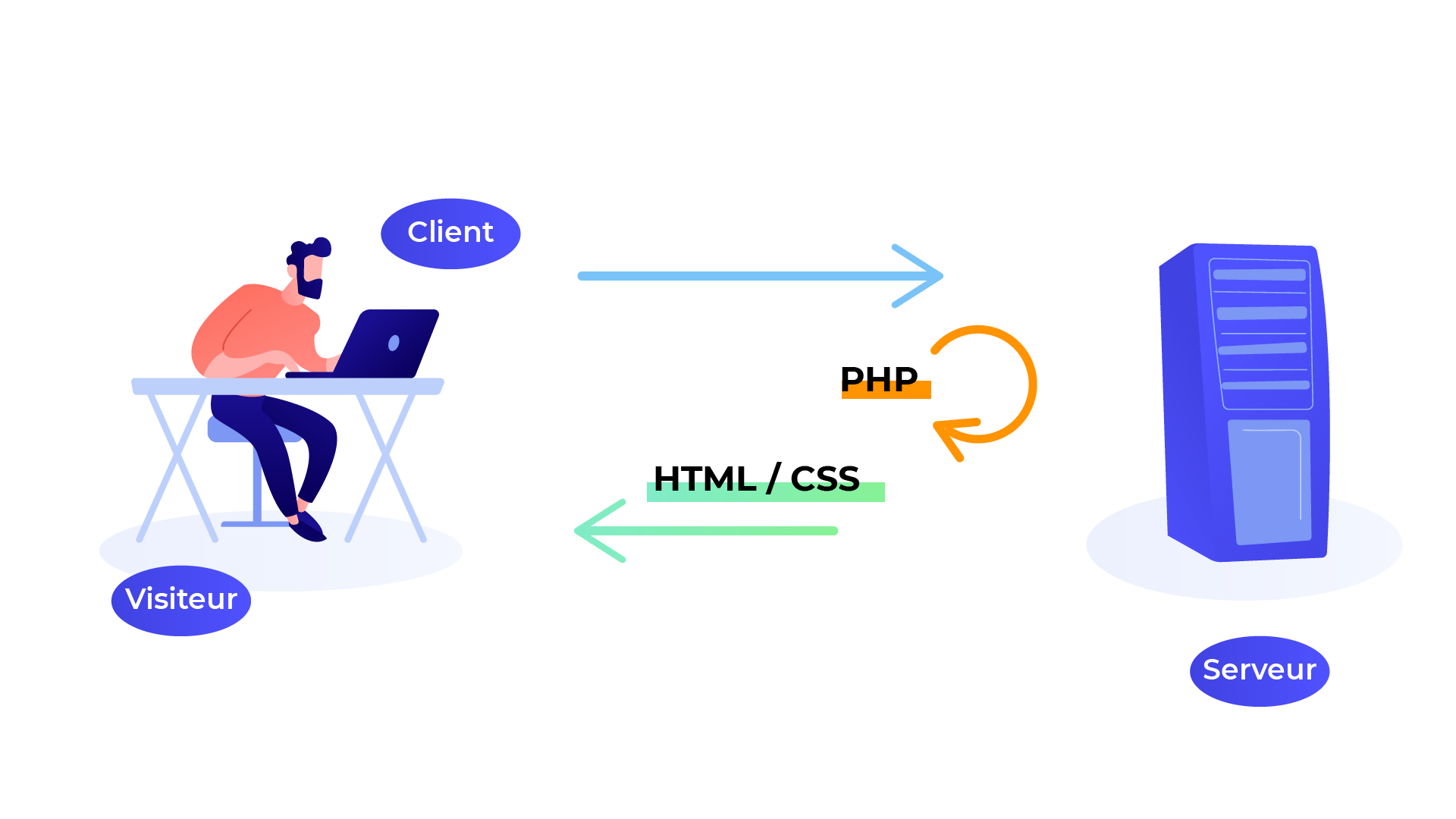
### ****Comment un langage serveur génère-t-il une page ?****

Le rôle d’un langage serveur est de générer une page web. On peut résumer l’interaction entre le client et le serveur comme ceci :



1. Le client demande une page au serveur.
2. Le serveur génère la page (à l’aide du langage serveur).
3. Le serveur envoie la page (sous forme de code HTML et CSS).

Par exemple, si vous utilisez PHP comme langage serveur, et HTML et CSS comme langages client, voici à quoi ressemblera le schéma :



Je vous ai listé ci-dessous des développements qu’on a dû effectuer pour créer cette fonctionnalité. Votre mission est de déterminer, pour chacun de ces développements, lesquels ont été programmés avec les langages serveur et lesquels avec des langages client. C’est parti !

* Créer un champ texte “Adresse mail ou nom d’utilisateur”
* Autoriser la connexion des utilisateurs ayant un mot de passe respectant les critères de sécurité
* Faire apparaître le bouton “SE CONNECTER” en violet
* Créer le bouton à cocher “Se souvenir de moi”
* Souligner en violet le lien “Mot de passe oublié ?”
* Autoriser les utilisateurs ayant une adresse mail correcte à se connecter
* Générer un tableau de bord personnalisé pour les utilisateurs se connectant
* Créer un lien vers la page “Inscrivez-vous gratuitement”

### ****Vérifiez votre travail****

Alors, vous êtes allé au bout ? Vérifiez votre travail !

Langage client :

* Créer un champ texte “Adresse mail ou nom d’utilisateur”
* Faire apparaître le bouton “SE CONNECTER” en violet
* Créer le bouton à cocher “Se souvenir de moi”
* Souligner en violet le lien “Mot de passe oublié ?”
* Créer un lien vers la page “Inscrivez-vous gratuitement”

Langage serveur :

* Autoriser la connexion des utilisateurs ayant un mot de passe respectant les critères de sécurité
* Autoriser les utilisateurs ayant une adresse mail correcte à se connecter
* Générer la page “Tableau de bord” pour les utilisateurs se connectant

Vous l’aurez compris : toute la partie apparente du site est codée avec les langages client, comme html, alors que tout ce qui décrit le comportement du site relève du langage serveur.

## les bases de données

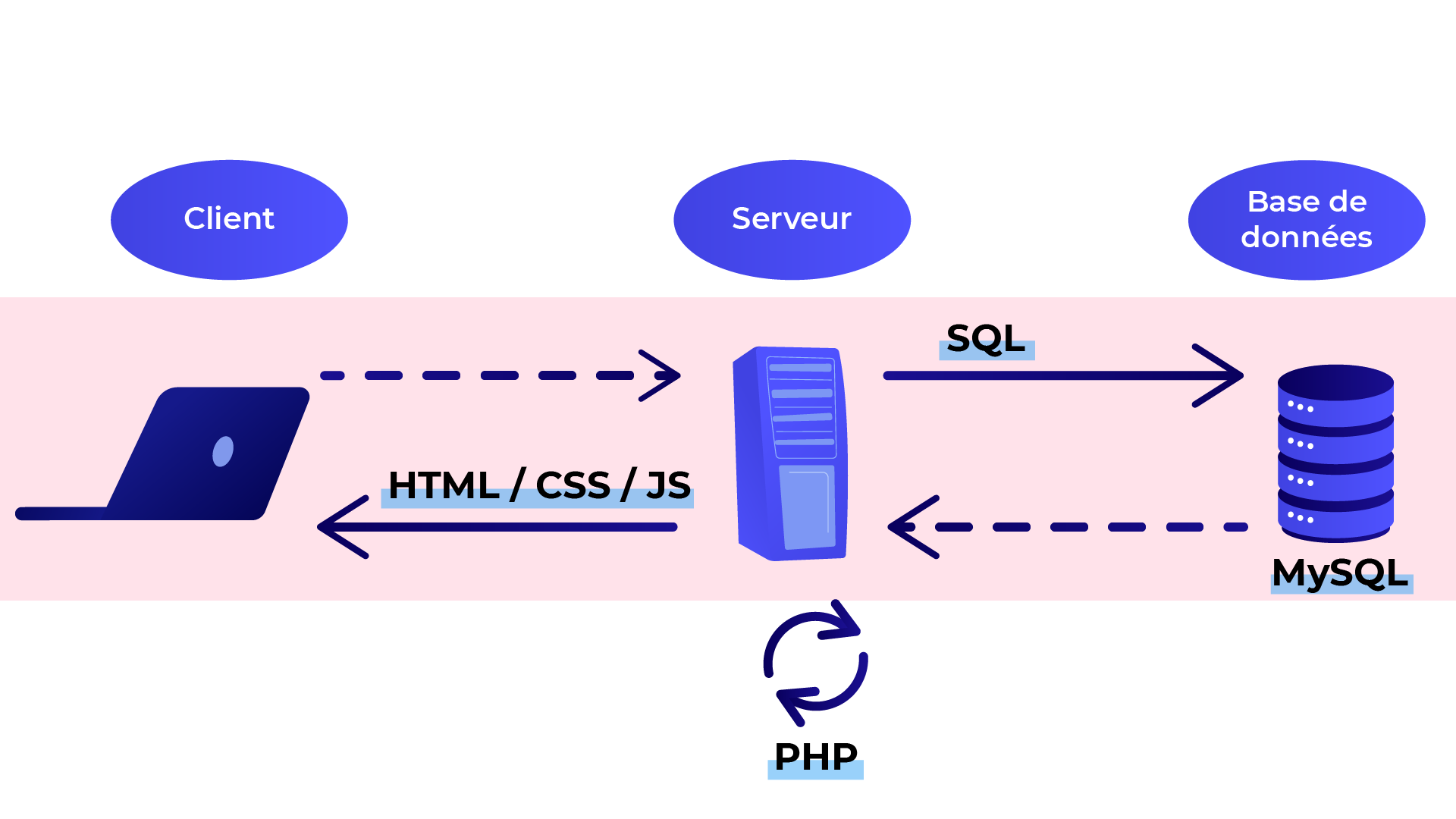
Parmi les logiciels de base de données les plus célèbres, on peut citer :

* MySQL
* PostgreSQL
* SQL Server
* Oracle
* SQLite
* Etc.

Voici à quoi ressemble une requête SQL pour vous donner une idée :

SELECT id, name, login FROM users ORDER BY id DESC

C’est généralement le serveur qui communique avec la base de données, par le biais du langage serveur. Résumons dans un schéma pour tenter d’y voir plus clair :



### ****À vous de jouer !****

Rendez-vous dans [cet exercice](https://www.db-fiddle.com/f/9y8dT6CBqtUcMdhax74HLD/818) ( <https://www.db-fiddle.com/f/9y8dT6CBqtUcMdhax74HLD/818> ). Vous allez pouvoir vous amuser avec une base de données déjà conçue qui contient les noms de tous les départements français. Comme vous le voyez, une requête est déjà prête :

SELECT departement\_nom FROM departement;

1. Lancer la requête en cliquant sur "Run". Vous devriez voir s’afficher en bas la liste des noms de départements !
2. Modifiez votre demande en changeant la requête dans la section de droite. On va récupérer tous les départements ainsi que leur code (numéro):   
   SELECT departement\_code, departement\_nom FROM departement;  
   Cliquez sur "Run" et regardez les résultats s'afficher en bas !
3. Vous voulez savoir quel département possède le code 30 ? Demandez-le à la base de données !   
   SELECT departement\_code, departement\_nom FROM departement WHERE departement\_code = '30';
4. Pour aller plus loin : on peut aussi demander la liste des départements qui commencent par la lettre C ! Ou n’importe quelle autre lettre d’ailleurs.   
   SELECT departement\_code, departement\_nom FROM departement WHERE departement\_nom LIKE 'C%';

* **Un site web responsive :** vous créez votre site web dès le départ en pensant aux smartphones. Vous faites en sorte que le design s’adapte automatiquement aux appareils de petite taille (on dit qu’il est *responsive*). C’est la solution la plus simple en général.
* **Une application native :** vous créez une application mobile (pour iOS, Android, etc.). C’est bien plus lourd, car il faut utiliser d’autres langages, mais l’expérience sera bien meilleure si vous avez un service complexe.

### ****L’adresse IP****

Chaque ordinateur possède une adresse. On parle d’adresse IP. Il s’agit d’une suite de nombres comme 205.89.177.26. Vous pouvez voir cela comme une sorte de numéro de téléphone.

En théorie donc, vous pouvez aller sur un site web en tapant directement l’adresse du serveur dans votre barre d’adresse.

D’accord, mais comment je connais l’adresse IP du serveur de Google, moi ?

C’est vrai, est-ce que vous connaissez les numéros de téléphone de tous vos contacts, vous ? Moi pas !

(En plus, Google a des milliers de serveurs...)

### ****Les noms d’hôtes et DNS****

DNS signifie 'Domaine Name Service' en anglais, en français on peut dire service de noms de domaine.

On a donc créé des noms d’hôte, comme "google.com", pour pouvoir se souvenir plus facilement du nom du service que l’on veut contacter.

Et l’on a ensuite créé un service d’annuaire, les DNS, pour faire le lien entre le nom d’hôte et l’adresse IP :

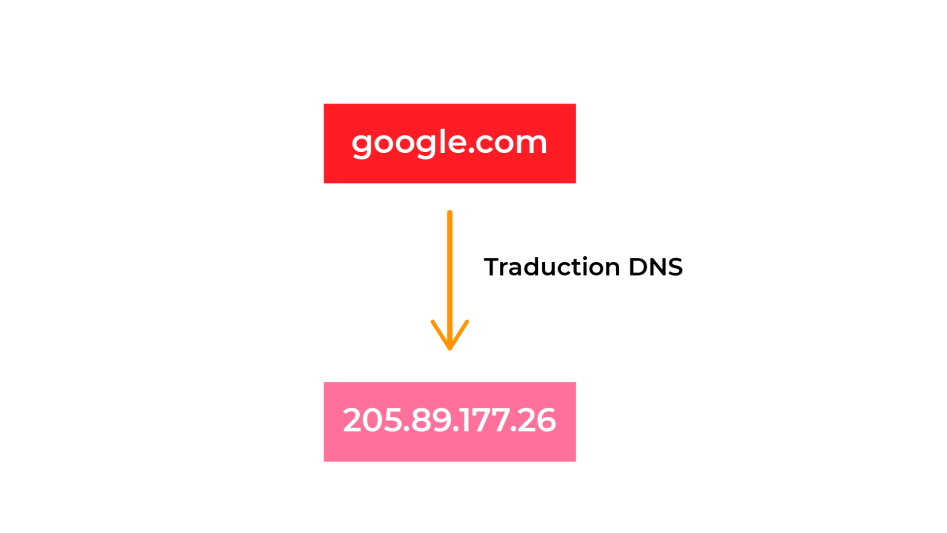
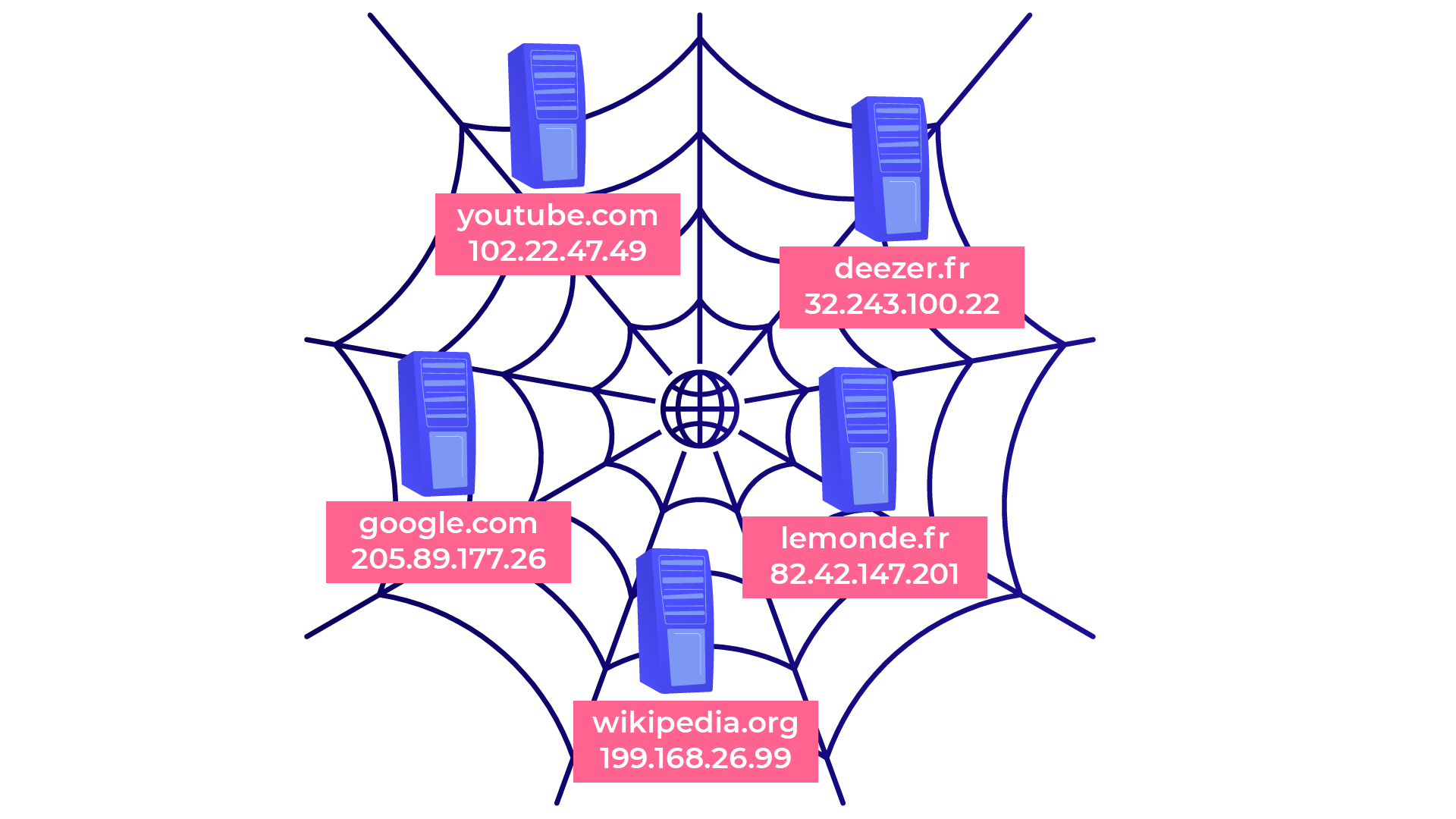


Schéma représentant la traduction DNS du nom d'hôte en adresse IP

Le DNS permet de traduire le nom d’hôte en adresse IP.

Tout ceci fonctionne heureusement sans que l’on ait besoin d’y penser. On tape juste "google.com" dans notre navigateur, et le site web s’affiche ! Cela dit, cela ne peut pas faire de mal de savoir que tous ces services sont à l’œuvre derrière. 🙂

Toile représentant les serveurs avec leur adresse IP et nom d'hôte

Chaque serveur sur la toile a une IP et un nom d’hôte (plus facile à retenir).

On a donc inventé des langages de communication pour que les machines se parlent entre elles. On les appelle les **protocoles**.

### ****Protocoles bas niveau****

À la base d’Internet, on a des protocoles de bas niveau. On les retrouve dans toutes les communications :

* TCP (Transmission Control Protocol)
* UDP (User Datagram Protocol)

**Protocoles haut niveau**

Les protocoles dits "haut niveau" sont généralement basés sur TCP (ou UDP). Ils forment une surcouche qui englobe TCP.

Les noms des protocoles haut niveau devraient vous être un peu plus familiers pour certains :

* **HTTP :** le protocole qui permet d’échanger des pages web entre le client et le serveur. En plus du langage HTML, Tim Berners-Lee a inventé les bases du protocole HTTP qui permet d’échanger les pages. C’est ce que signifie le "http://" que vous voyez au début des adresses web !
* **HTTPS :** identique à HTTP, avec le "S" en plus qui signifie "Secure". Les pages sont chiffrées, pour garantir que personne ne peut les lire (par exemple quelqu’un qui espionnerait les données qui transitent dans les câbles sous-marins !).
* **FTP :** permet d’échanger des fichiers.
* **SMTP :** permet d’envoyer des e-mails.
* Etc.

### À vous de jouer !

Je vous propose de vous entraîner à parler avec un serveur web. Vous allez utiliser le protocole HTTP en utilisant des commandes que je vais vous apprendre ! Nous allons utiliser un site qui permet d’afficher les retours HTTP d’un serveur. Nous allons littéralement voir le serveur nous parler ! Pour ça, rendez-vous sur <https://reqbin.com/curl> et suivez les étapes suivantes :

1. Essayez de contacter la page d'accueil de Google en tapant la commande suivante:  
   curl  https://google.com --head  (1).
2. Cliquez sur “Send” (2).
3. Au bout de quelques instants, le serveur répond. Cliquez sur “Raw” (3) à droite pour lire le résultat “brut”.
4. Essayez maintenant avec une adresse qui n’existe pas. Par exemple tapez la commande :

https://www.google.com/cudkedpqma --head

Cette fois, le serveur répond par “HTTP/1.1 404 Not Found” : la page demandée n’a pas été trouvée (la fameuse erreur 404 !).

### Les différentes spécialités

Voici une introduction à quelques spécialités possibles :

* **le développeur front-end** : le développeur front-end travaille sur toute la partie apparente du site, ce que nous voyons lorsque nous allons sur une page web. Il crée l’interface utilisateur de telle sorte qu’elle soit la plus adaptée au besoin, et la plus simple possible. Il utilise les langages client tels que HTML, CSS ou JavaScript ;
* **le développeur back-end** : le développeur back-end travaille au contraire sur toutes les composantes cachées d’un site. Il crée l’infrastructure adaptée, et coordonne toutes les relations entre le site et les bases de données. Il gère la “logique” du site web. Il connaît bien un ou plusieurs langages de programmation (PHP, Ruby, Python, Java) et le langage des bases de données (SQL), ainsi que les frameworks qui lui facilitent la tâche ;
* **le développeur mobile** : il développe des applications Android ou iOS, parfois même les deux, même si c’est plus rare ! Il travaille sur toutes les étapes de la conception : depuis le cahier des charges jusqu’à la correction des bugs. Quand on développe sur mobile, on travaille en général aussi bien sur l’interface (front-end) que sur la logique (back-end).