Programmation Web

Presentation du Module

Anastasia.Bezerianos@universite-paris-saclay.fr

Objectifs

Apprendre ou consolider ses connaissances dans le domaine du Web

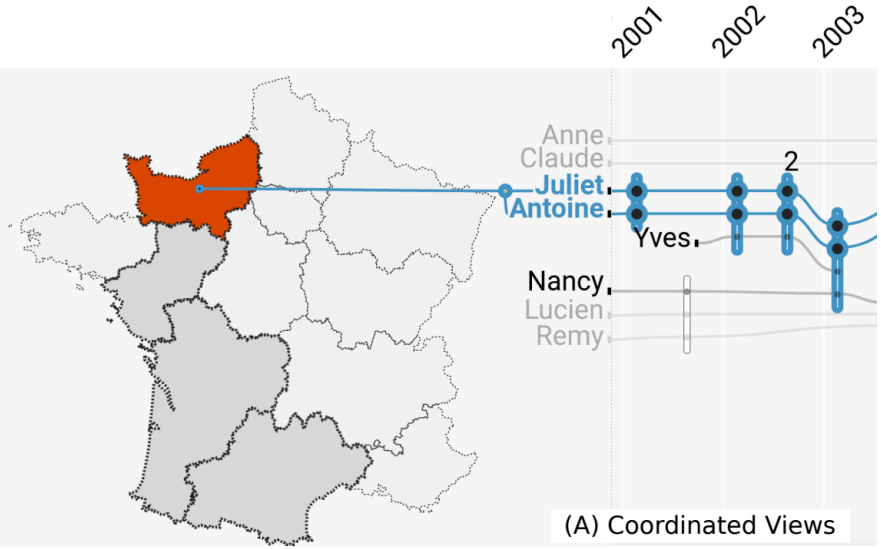
Apprentissage par la pratique, i.e., «c'est en forgeant…»

Construire ses compétences

Communication / Resources

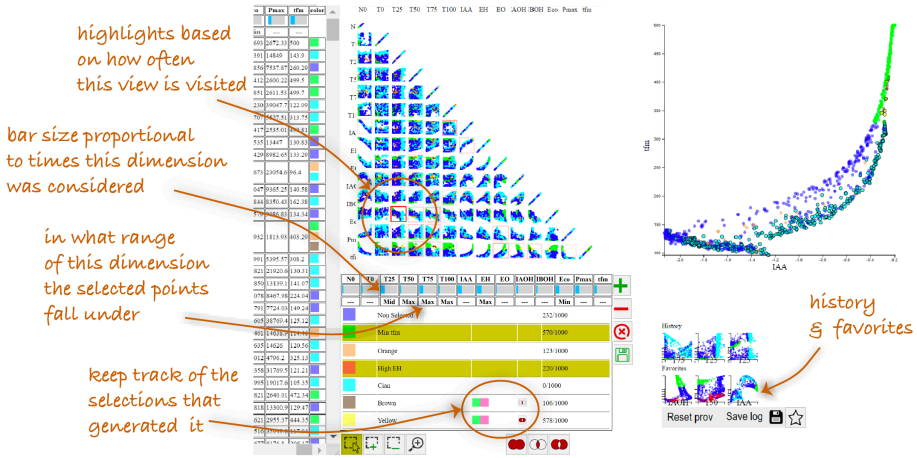
- eCampus
  - resources dans eCampus
    - nom du module : ProgWeb-App3-Info
    - https://ecampus.paris-saclay.fr/course/view.php?id=93552
  - communication
    - questions dans le Forum (vous pouvez aussi répondre !)
    - email si question privée

Anastasia



Visualisation spatio-temporelle pour les journalistes de Ouest-France

Visualisation des donnés multidimensionnelles issues des simulations pour les agronomes de INRAe



Et vous …?

Experience avec Web ?

- html / css
- JavaScript
- Php
- MySql
- Ajax

cible du cours,
mais …

- Node.js
- Framework : JQuery, Angular, Vue, React, …

Adapter le cours

Nous allons couvrir les bases

… mais adapter et customiser le cours selon

- votre projet professionnel
- vos besoins / votre curiosité

Cela nécessite une forte implication de votre coté :

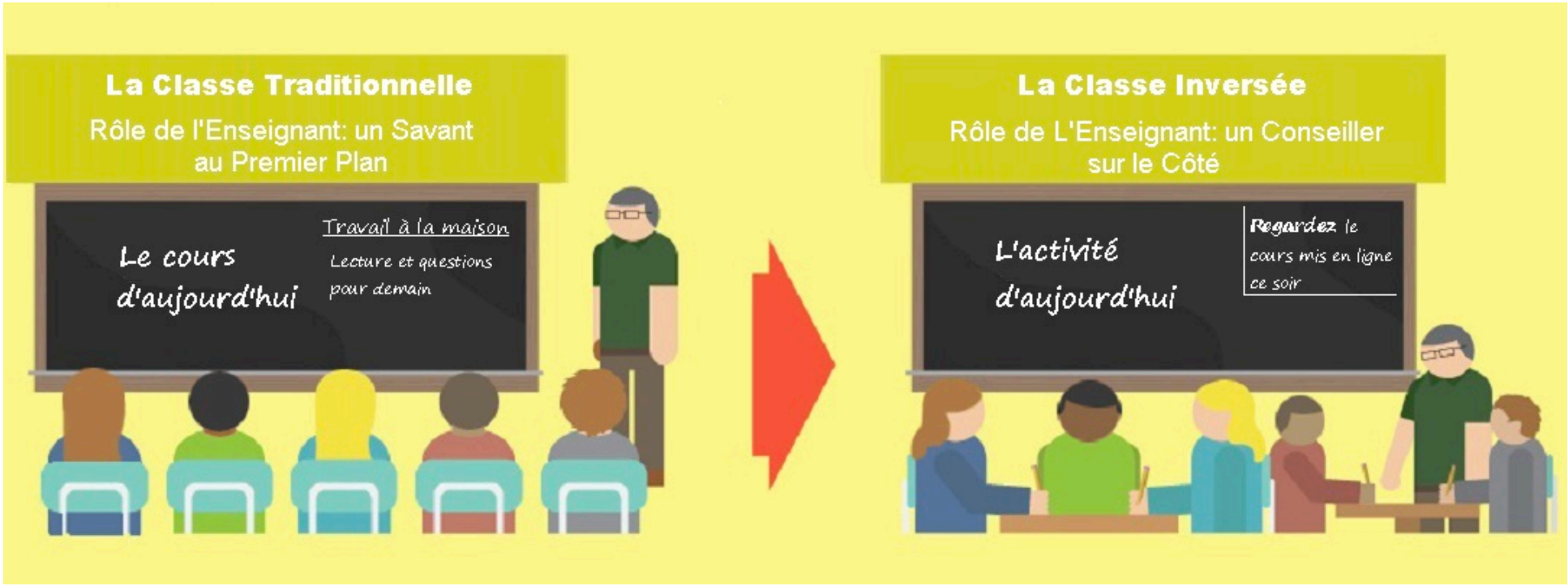
Choisir votre cible / choses à apprendre

Lire et suivre des tutos

… Mais comment ?

Planning des séances

On commence avec une classe normale, mais …
on continue avec une classe **inversée**.



Planning des séances

Une classe **inversée**.

Classe normale : le prof fait cours avec un support et les étudiants travaillent des exercices chez eux après en se référant au support.

Classe inversée : vous devez travailler le support de cours AVANT chez vous et en cours nous faisons tous ensemble des exercices.

… et si vous connaissez déjà le topic, vous définissez avec Anastasia votre propre “support" à lire (topic avancé).

Planning des séances

Vous devez donc avant chaque séance :

- relire vos support de cours en ligne, ou ..
- trouver des ressources libres sur le web qui vous conviennent (*).

Á la fin de chaque séance il faut :

- soumettre votre travail de la session

(*) Si vous êtes à l'aise avec parties du contenu, vous allez faire d'abord toutes les exercices que vous pouvez (soit en session, soit seul.e.s). Et discuter avec Anastasia pour passer au projet et étudier des topics avancés adaptés à votre projet d'évolution personnel.

Planning des séances

Á lire **avant** la prochaine fois (14/10) il faut lire AVANT la séance :

cours1-HTML-à-lire.pdf (section HTML)

cours2-CSS-à_lire.pdf (section CSS)

Planning des séances*

lundi 10 octobre	- intro module, intro web, projet
vendredi 14 octobre	- html/css
jeudi 20 octobre	- JS
vendredi 21 octobre	- Php / sql
vendredi 9 décembre	- projet
mercredi 14 décembre	- projet
jeudi 09 mars	- projet
mardi 14 mars	- TD noté (avec connection web)
▸ vendredi 17 mars	▸ soutenance de projet

* Si vous êtes à l'aise avec le contenue, faites les exercises et les quizzes et continuez au cours suivant

Évaluation

Évaluation du module sur 3 notes :

- **TD noté** (30%) en classe sur les technologiques dites «classiques» (HTML, CSS, JS, PHP et SQL)
- **Projet** (60%) en binôme (l'essentiel du module) :
  - Rapport
  - Soutenance / Rendu (démo)
  - Évaluation du projet en fonction de la progression et de l'apprentissage des nouvelles technologies.
- **Participation** (10%) quiz, soumission des travaux, … (travaux pas notés - c'est pour vous …)

Environnement de travail

Sublime-text (avec le/les modules dont on parle dans le support de cours) … ou un éditeur de texte au moins équivalent

Un/des navigateurs web. Il faut au moins firefox ou chrome installés et mis à jour

Serveur web local qui fasse tourner PHP (ex. MAMP mais il y a LAMP, XAMPP, WAMP et pleins d'autres)

Liens utiles

Pour les couleurs : https://color.adobe.com/

Pour les polices : https://www.fontsquirrel.com/

Pour la compatibilité avec les différents navigateurs : https://caniuse.com/

Pour la validation du HTML : http://validator.w3.org/

Pour la validation du CSS : https://jigsaw.w3.org/css-validator/

Intro Web

1. Un peu d’histoire

Internet

Années 1969-70s : création de l'ARPANET (réseau expérimental de communication indépendant du matériel utilisé) ; développement de la pile de protocoles TCP/IP (Transmission Control Protocol/Internet Protocol), utilisés sur les réseaux dérivés de l'ARPANET ;

Années 80 et 90 : de plus en plus de réseaux sont reliés entre-eux

Utilisation de protocoles communs (TCP/IP)

Ces liaisons entre ces différentes réseaux («inter-net») donnèrent naissance à un réseau plus grand : l'Internet

Une révolution était en marche. Les protocoles TCP/IP sont devenus une norme et seront utilisés pour transporter les protocoles de plus haut niveau utilisés par les applications utilisateur.

le World Wide Web (WWW) ou Web

Début des années 90 : des physiciens du CERN proposent un langage de création de documents électroniques

⇒ L'HTML (Hypertext Markup Language ou langage de balisage **< >** pour l'hypertexte) et les navigateurs étaient nés.

Utilisé pour partager de documents électroniques intégrés (textes, images et sons) sous la forme de liens.

```
1  <!doctype html>
2  <html lang="fr">
3  <head>
4    <meta charset="utf-8">
5    <title>Titre de la page</title>
6  </head>
7  <body>
8    Bonjour le monde
9    <div>
10     <span>
11       Bonjour
12     </div>
13   </span>
14 </body>
15 </html>
```

le World Wide Web (WWW) ou Web

L'HTML réalisait la fusion des éléments (textes, images, etc.). Le World Wide Web permettait la liaison hypertexte par laquelle des documents pointent automatiquement sur d'autres documents situés n'importe où dans le monde.

Navigateur Web

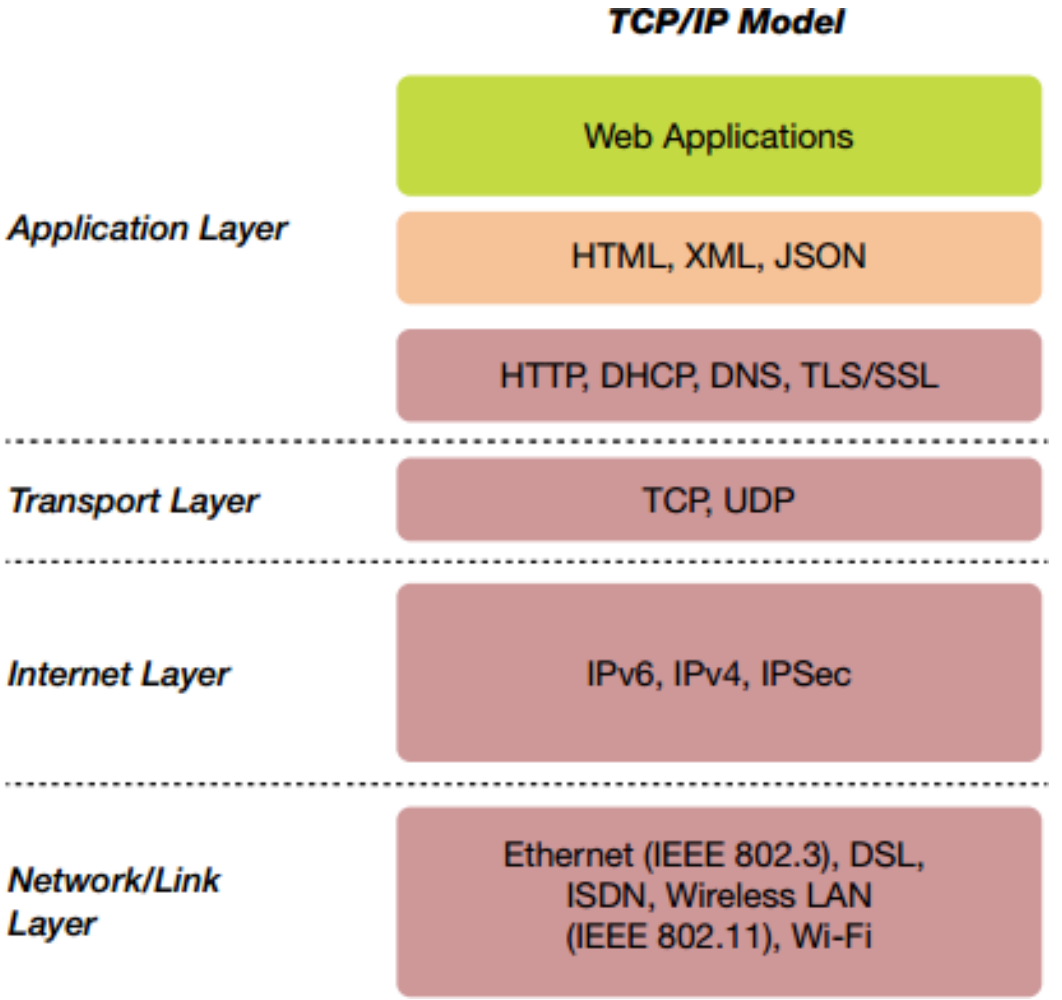
Un navigateur Web est un logiciel client conçu pour accéder (visualiser) aux ressources du World Wide Web ;

Il permets la consultation des documents HTML mis à disposition par les serveurs Web (ou serveurs HTTP) ;

Le dialogue entre navigateur et serveur Web se fait suivant le protocole HTTP (HyperText Transfer Protocol). Le but du protocole HTTP est le transfert de fichiers hypertextes au format HTML.

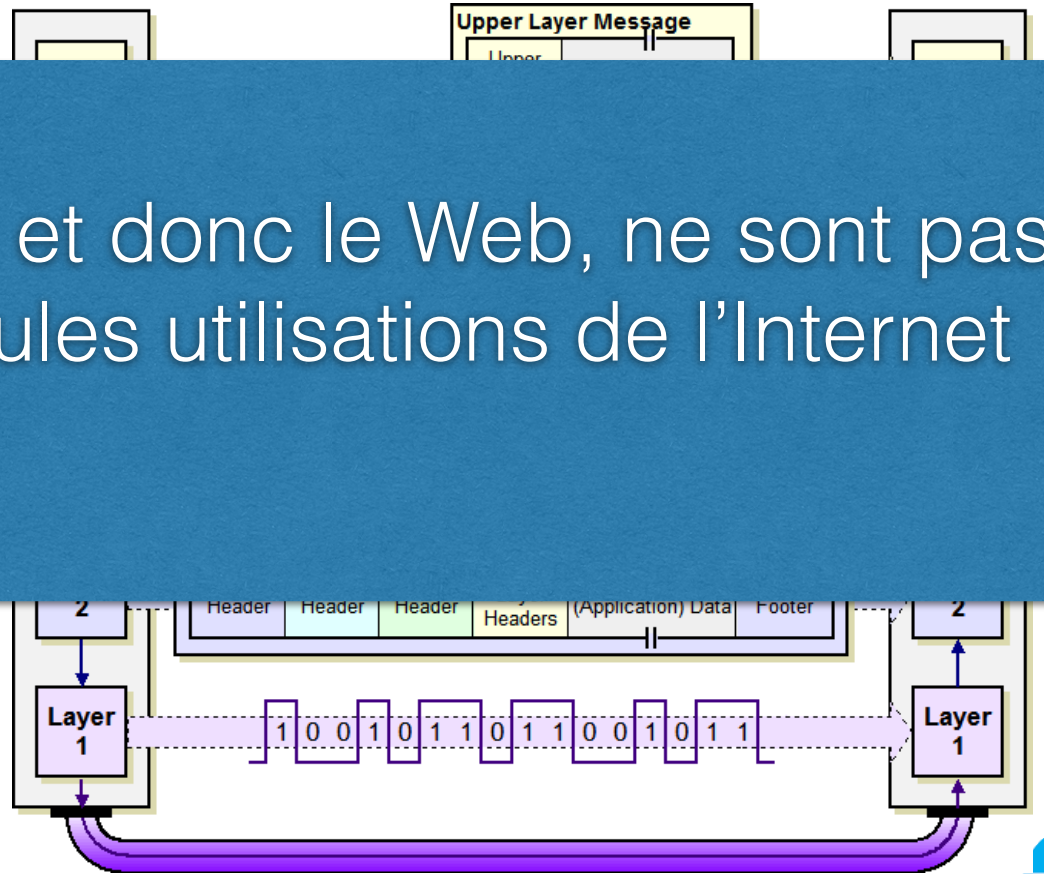
Pile TCP/IP et HTTP

⇒ Le protocole HTTP est transporté par les protocoles de la pile TCP/IP.



Principe d'encapsulation des protocoles

Le protocole HTTP sera encapsulé dans les différents protocoles qui vont assurer son transport de son expéditeur vers son destinataire.



L'HTTP et donc le Web, ne sont pas les seules utilisations de l'Internet

URL

L'accès à une donnée Web (page) se fait à l'aide d'une adresse Web ou URL qui permet au navigateur (rôle de client) d'atteindre le serveur Web hébergeant la donnée désirée.

URL : Uniform Ressource Locator
Les données sont appelées **ressources**.

URL

Exemple, pour `http://monsite.fr/index.html`

Trois parties :

- `http://` indique le protocole utilisé
- `monsite.fr` indique le nom qualifié du serveur
- `index.html` est le nom de la ressource (donnée) que l'on désire récupérer.

Adresse IP

Le nom **`monsite.fr`** est une étiquette utilisée en vue de rendre plus simple l'identification et la mémorisation par l'utilisateur du serveur contenant la ressource.
⇒ c'est un **nom de domaine**.

Un nom de domaine pointe sur l'adresse IP (sur 32 bits) identifiant le serveur.

Adresse IP

C’est le navigateur qui effectue la conversion nom de domaine vers adresse IP. Pour cela, il s’adresse à un serveur de noms de domaines qui contient une table de correspondance (Domain Name Server ou DNS).

Par exemple, dans son navigateur, si l’on tape 216.58.209.227, le navigateur ira afficher la page d’accueil de Google.

216.58.209.227 est une des adresses IP du site www.google.com.

Il est bien plus simple de se souvenir du nom du domaine que de son IP ! Mais c’est pourtant bien l’adresse IP qui est utilisée par le navigateur.

2. Quelques définitions

Statique vs. Dynamique

On distingue les pages :

- Statiques
- Dynamiques

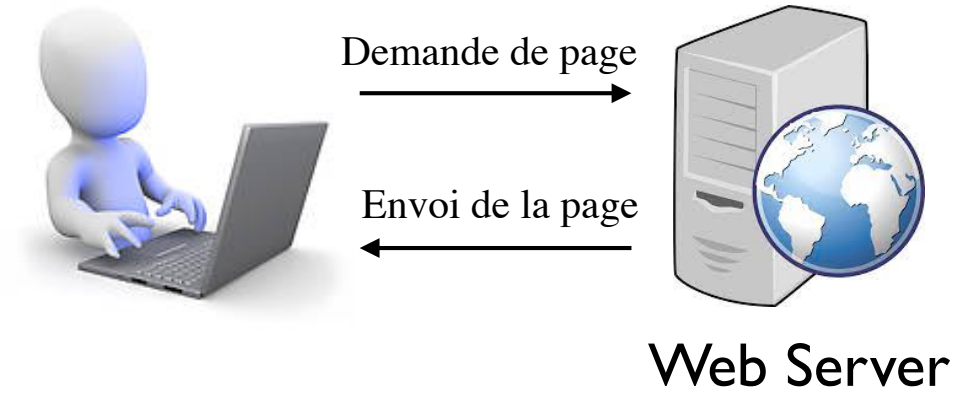
Web Server

Statique vs. Dynamique

On distingue les pages :

Statiques : le serveur renvoie le contenu du fichier la page
tout ceux qui demandent la page reçoivent le même contenu

Dynamiques

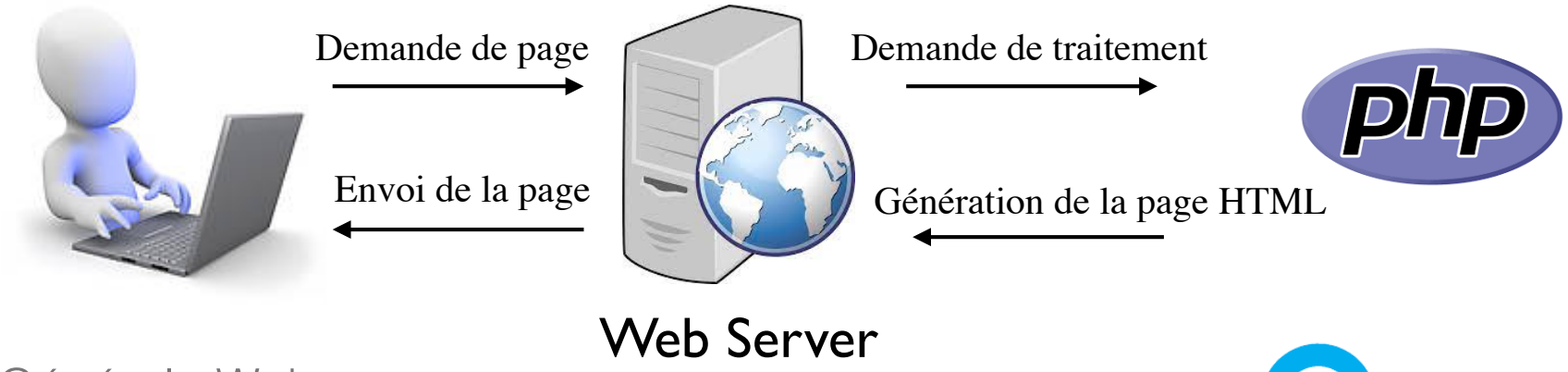


Statique vs. Dynamique

On distingue les pages :

Statiques : le serveur renvoie le contenu du fichier la page tout ceux qui demandent la page reçoivent le même contenu

Dynamiques : le serveur appelle un logiciel (souvent PHP) qui génère le contenu (page), le serveur renvoie cette page

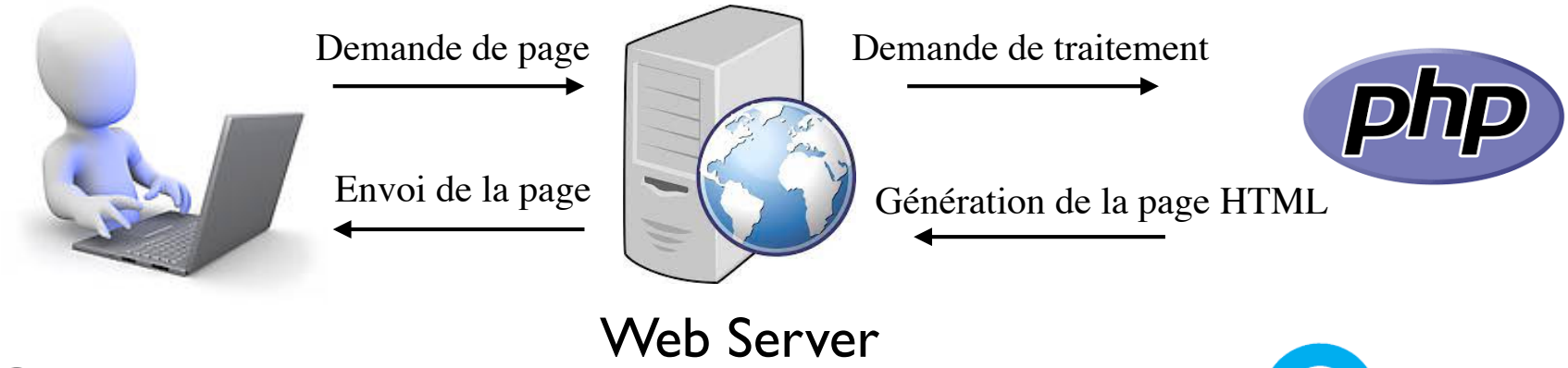


Statique vs. Dynamique

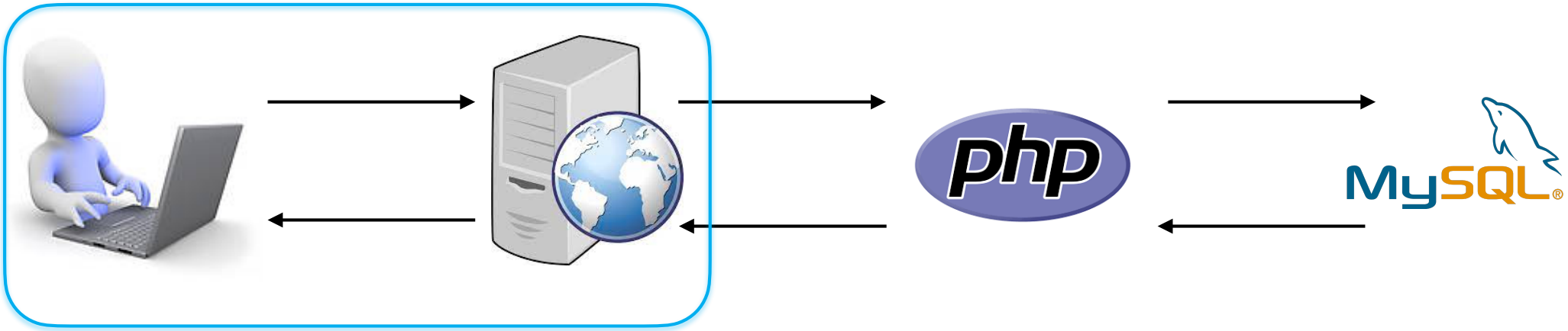
On distingue les pages :

Statiques : le serveur renvoie le contenu du fichier la page tout ceux qui demandent la page reçoivent le même contenu

Dynamiques : le serveur appelle un logiciel (souvent PHP) qui génère le contenu (page), le serveur renvoie cette page ; le contenu dépend de la demande (heure, adresse IP, formulaire, etc.) même s'il peut être le même (ex youtube) …



Front end & back end



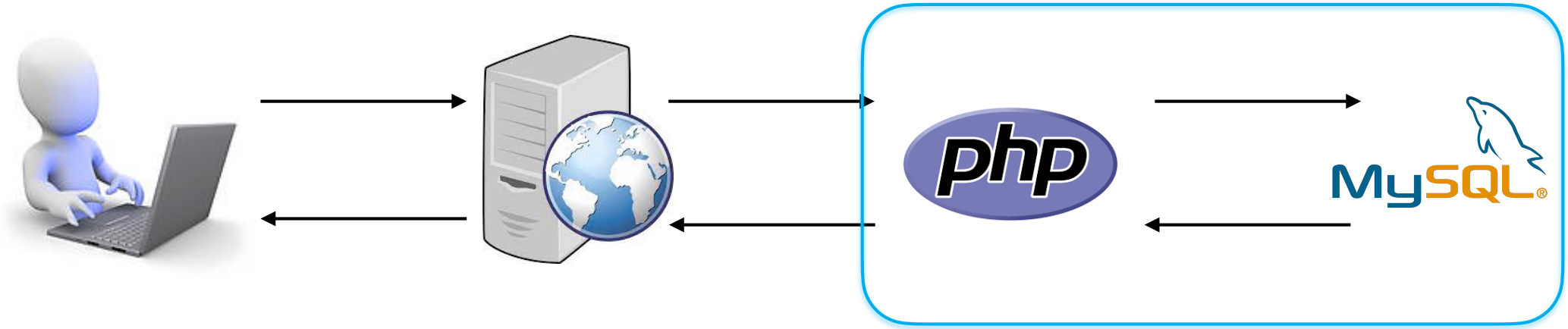
Client-side

Côté client

Front end

HTML, CSS, JS, etc.

Front end & back end



Client-side

Côté client

Front end

HTML, CSS, JS, etc.

Server-side

Côté serveur

Back

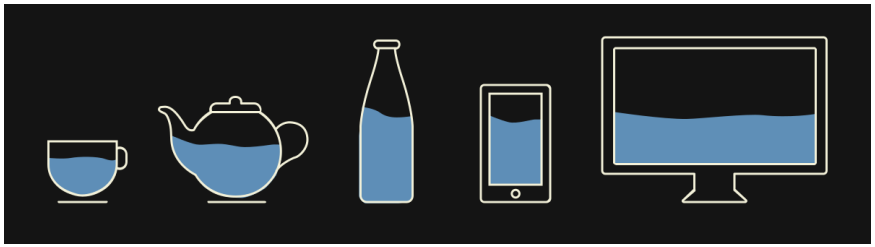
PHP, SQL, Node.js, etc.

Site réactif
Responsive Web Design

Site qui s'adapte pour offrir une consultation confortable sur des écrans de tailles très différentes.

Site réactif
Responsive Web Design

Site qui s'adapte pour offrir une consultation confortable sur des écrans de tailles très différentes.



Deux idées à garder :
- Notion de point de rupture, i.e., tailles provoquant des changements dans les affichages (ex : **Bootstrap**) ;
- Utiliser des pourcentages pour les éléments.

Plus de détails : Wikipédia - site web réactif

Modèle MVC

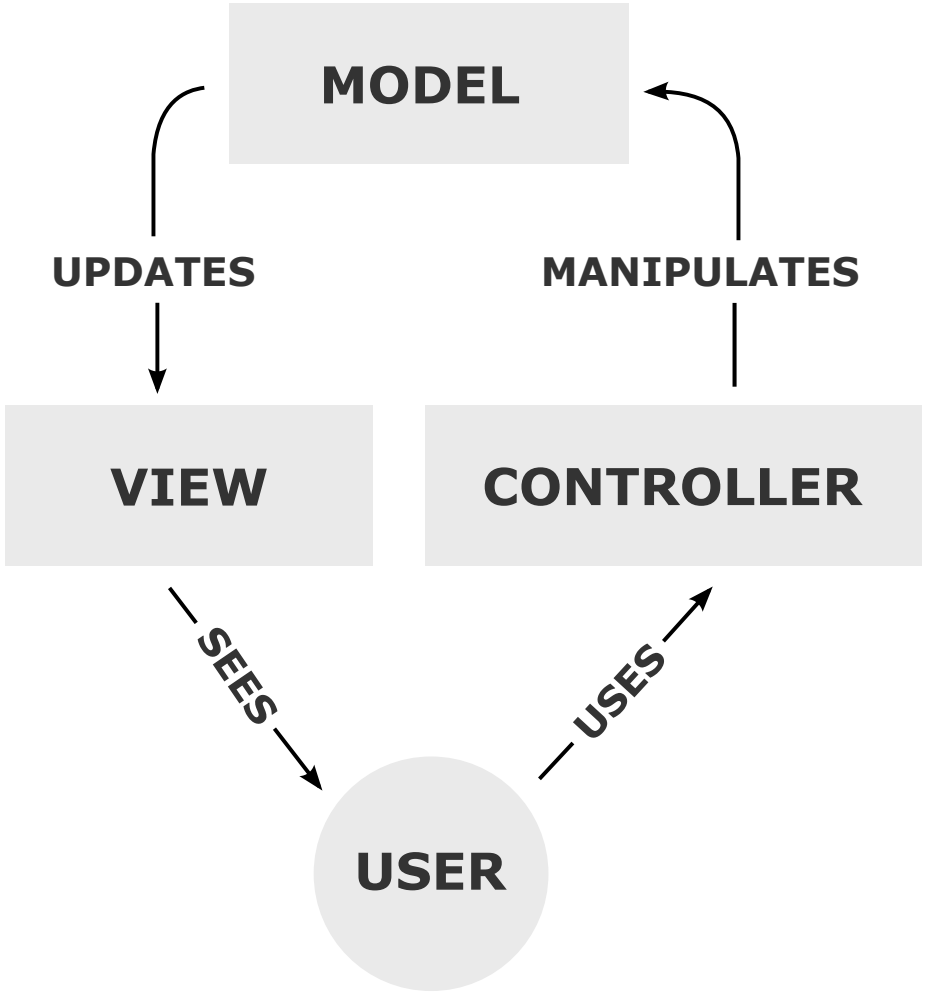
Motif d'architecture logicielle utilisée sur un site dynamique :

Un modèle (Model) contient les données à afficher, et gère leur accès, leur enregistrement, et les actions qu'on peut vouloir faire dessus ;

Une vue (View) contient la partie visible de l'interface graphique ;

Un contrôleur (Controller) contient la logique concernant les actions effectuées par l'utilisateur.

Modèle MVC



3. Technologies

Bibliothèque, Framework et API

Bibliothèque : boîte à outils (pour simplifier la vie du développeur)

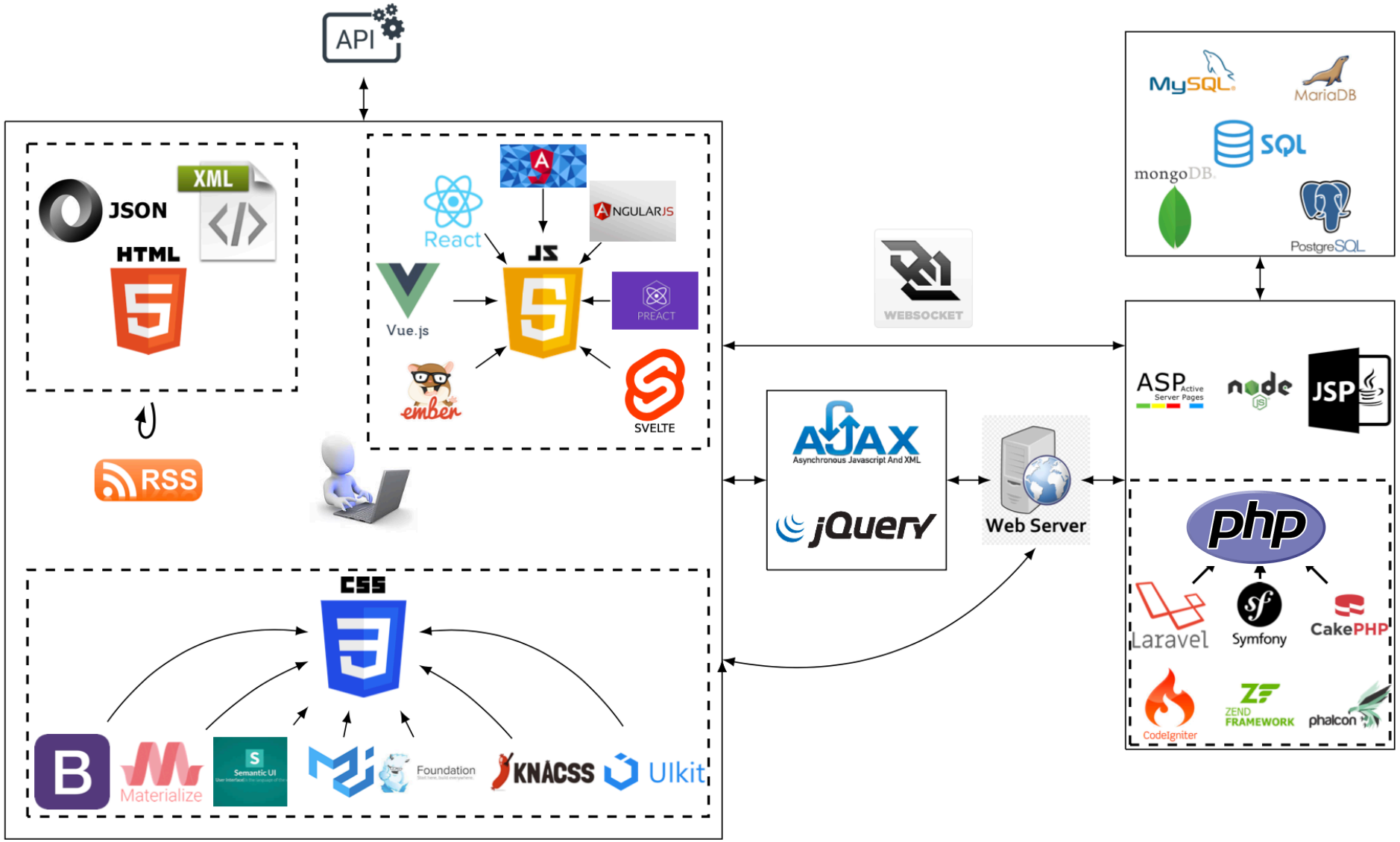
Framework :

Des variables, paramètres ;
Des boîtes à outils ;
Un formalisme, un cadre d'application (ex : MVC).

Philosophie : ⇒ on utilise une bibliothèque, mais le framework nous contraint dans notre façon d'écrire.

API (Application Programming Interface) : ensemble normalisé de classes, de méthodes, de fonctions, et de constantes qui sert d'interface par laquelle un logiciel offre des services à d'autres logiciels.
Plus simplement : c'est comme un service Web mais entre 2 applications (même si différent : Cf. ici).

Carte des technologies



Cinq langages de base

HTML : langage de balisage pour la structure des pages web

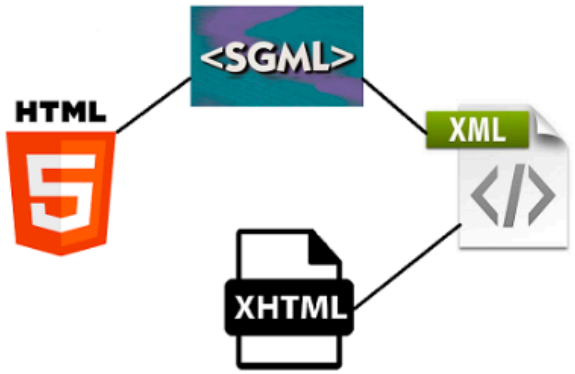
CSS : «feuilles de style en cascade», langage pour la présentation des documents HTML et XML

JavaScript : langage de programmation de scripts, utilisé pour les pages web interactives (mais aussi pour les serveurs avec Node.js )

PHP : langage de programmation server-side

SQL : langage pour exploiter les bases de données relationnelles

Format des pages



- HTML
- XML , eXtensible Markup Language, langage de balisage extensible
- JSon **JSON** : format de données textuelles dérivé de la notation des objets du langage `JavaScript`. Plus compact et proche des langages de programmation.

Flux RSS : un **agrégateur** peut lire ces flux depuis un autre site

Framework CSS

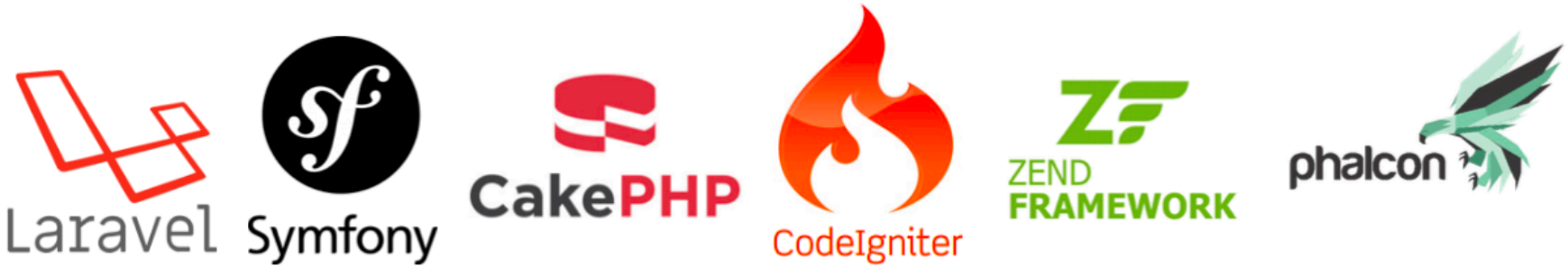
Les principaux :

- Bootstrap
- Materialize
- Material UI
- Foundation

Framework JS (coté client)

- `AngularJS vs. Angular` (Cf. ici)
- Trois plus utilisés :
  - `Angular`
  - `React`
  - `Vue.JS`

Framework PHP



Les plus connus :

- `Laravel` : de plus en plus utilisé
- `Symfony` : grande communauté
- `CodeIgniter` : rapide, léger, bonne communauté
- `CakePHP` : bien pour débuter

Mais aussi systèmes de gestion de contenu (SGC). Ex : WordPress
(CMS : *Content Management System*)

Framework JS (coté client)

, etc.

VS.

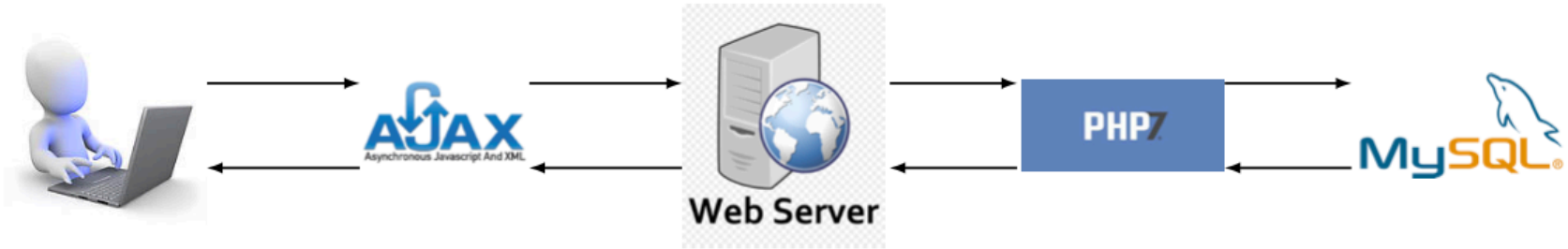
- Langage de programmation
- plus simple à apprendre et utiliser
- Langage SQL principalement
- Frameworks, grande communauté

- Environnement de JavaScript côté serveur
- Même langage que le front
- Langages noSQL
- asynchrone, plus rapide et léger
- Qq frameworks (METEOR express JS)

Choix dans ce cours de faire **apprendre au moins le PHP**

Rapidité et temps réel

AJAX : Asynchronous JavaScript and XML. Principe :

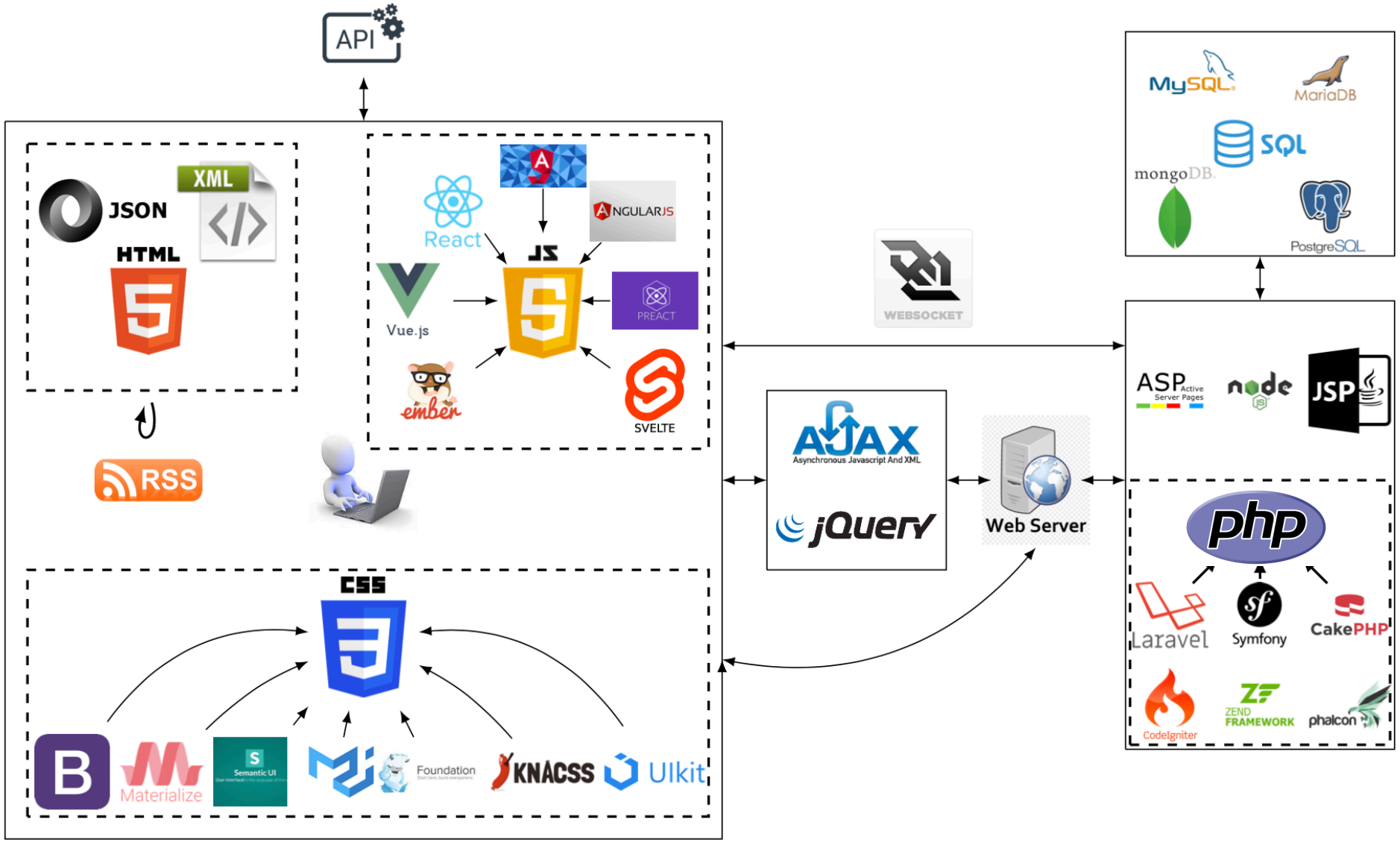


Utilité : éviter de recharger la page. Exemple d'applications :

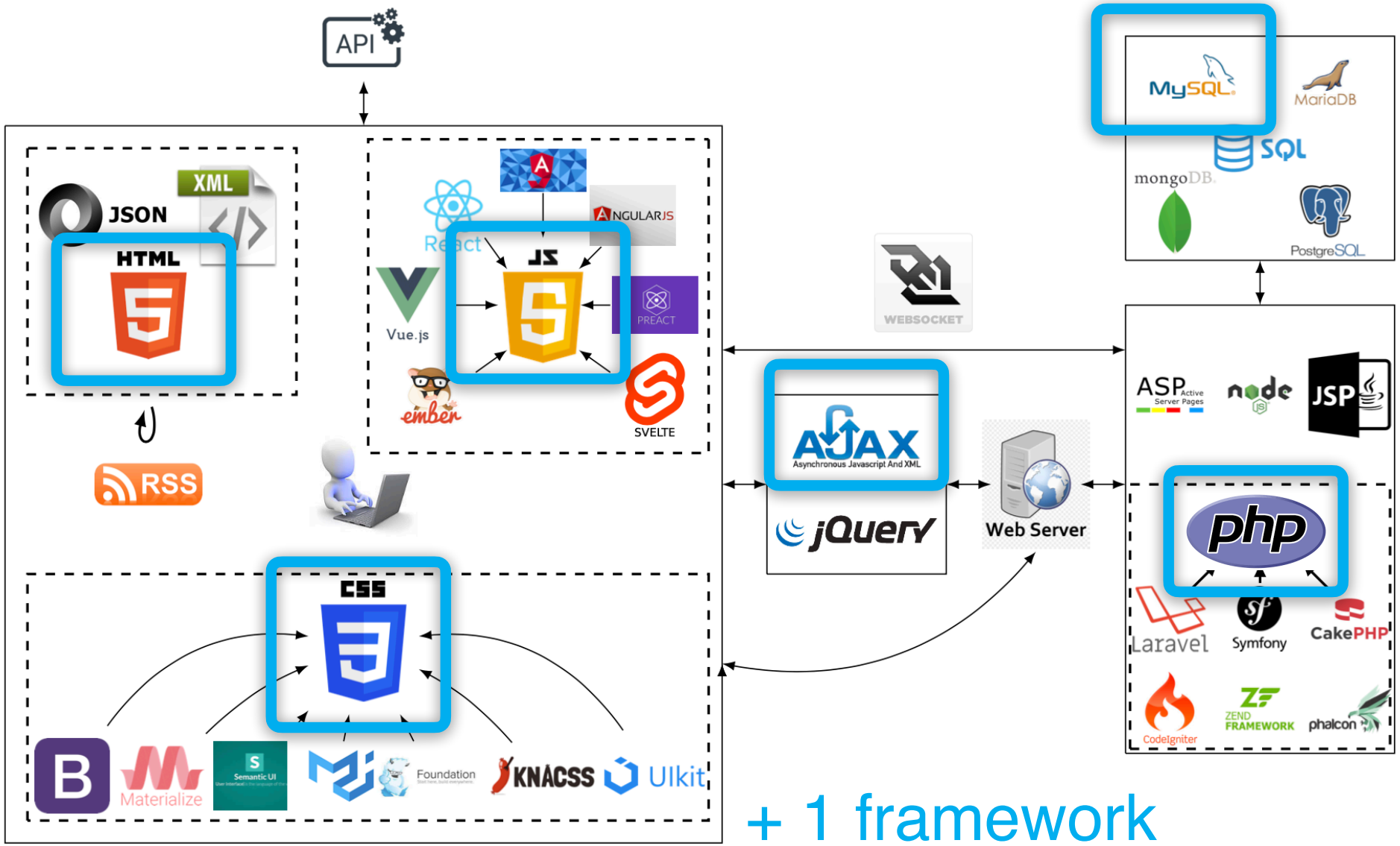
- Auto-complétion dans la recherche d'un nom d'une BD ;
- Sauvegarde automatique de textes ;
- Calculs déportés sur le serveur.

JQuery est une bibliothèque JS qui facilite (entre autres) l'utilisation d'AJAX.

Carte des technologies



Carte des technologies



+ 1 framework

Projet

Rappelle : Planning des séances

Vous devez donc **avant** chaque séance :

- si vous apprenez le contenue, il faut relire vos support de cours en ligne, ou ..

- si vous êtes à l'aise avec parties du contenu, vous allez faire d'abord toutes les exercices et les quizzes que vous pouvez soit seules soit dans la classe (ceci est pour vous, pour assurer que vous êtes à l'aise).

  Et puis discuter avec Anastasia pour passer au projet et étudier des topics avancés adaptés à votre projet d'évolution personnel.

Á la fin de chaque séance il faut :

- soumettre votre quiz et travail de la séance

Á lire pour la prochaine fois (14/10) il faut lire AVANT la séance :

cours1-HTML-à-lire.pdf (section HTML)

cours2-CSS-à_lire.pdf (section CSS)

Adapter le cours

Même si vous connaissez de nombreuses technologies web, c'est l'occasion d'en apprendre plus.

Vous êtes des professionnels, l'auto-apprentissage devrait toujours être votre objectif.

Nous apprenons mieux en groupe.

Fonctionnement du projet

Travail en binôme :

Créer un site web dynamique (front-end / back-end) ;

Utiliser au moins un framework ;
⇒ Philosophie : mener un projet de son début à sa fin

tout en pensant à sa maintenance par autrui, i.e., documentation, etc.

⇒ Rappel de l'objectif : **Apprendre !**

Attendus pour le rapport

Choix du sujet et des technologies utilisées ;

Design responsive (adaptation au support), sécurité, etc. ;

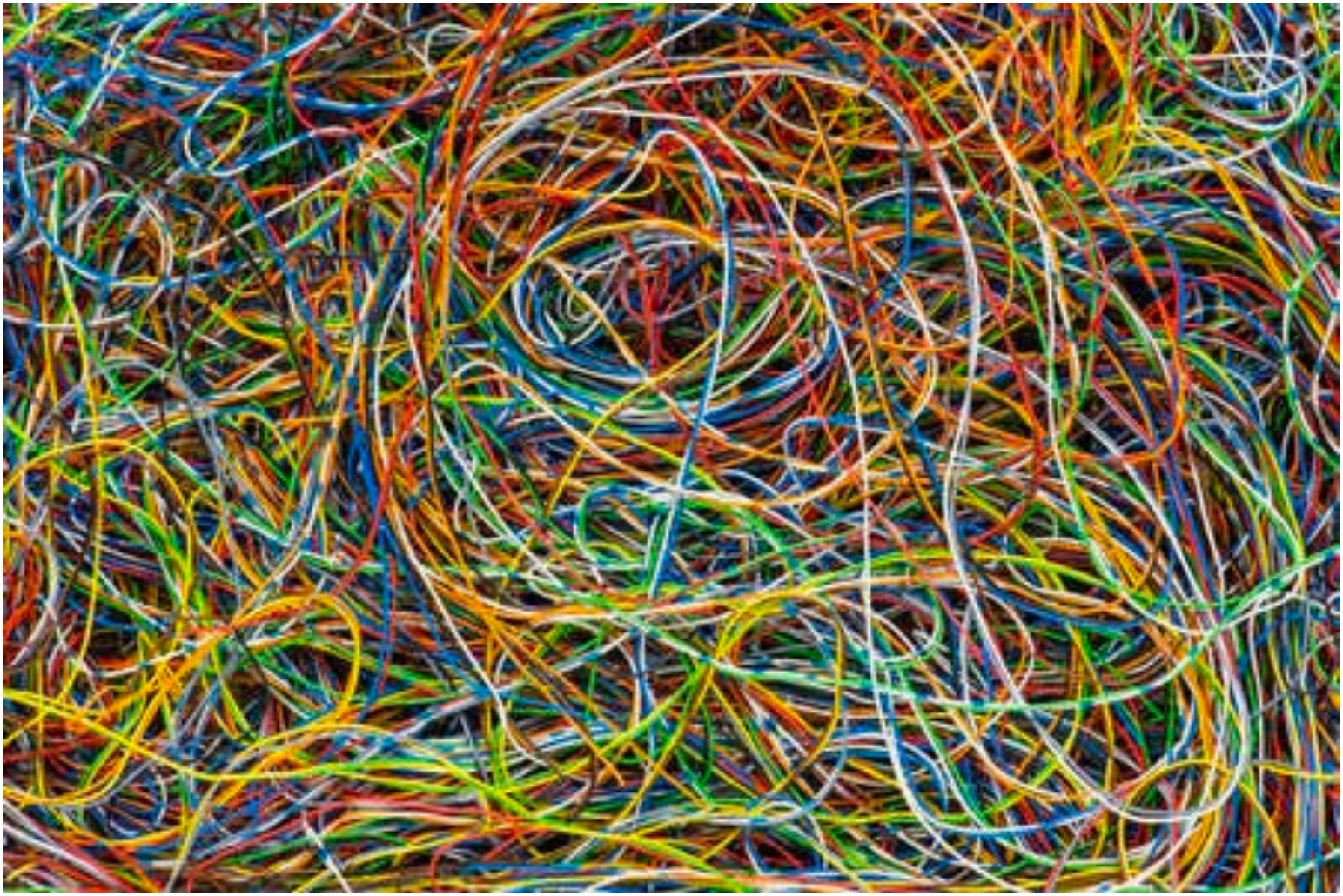
Répartition du travail, planification, calendrier, jalons ;

Cahier d'avancement au jour le jour ;

Erreurs, fausses routes avec analyse du pourquoi et solutions apportées ou envisagées ;

Recul sur les choix, auto-critique, etc.

Créer des groupes



Technos possibles

(pour créer les groupes)

HTML / CSS / JavaScript / SQL + PHP (Bootstrap)

JQuery / Ajax

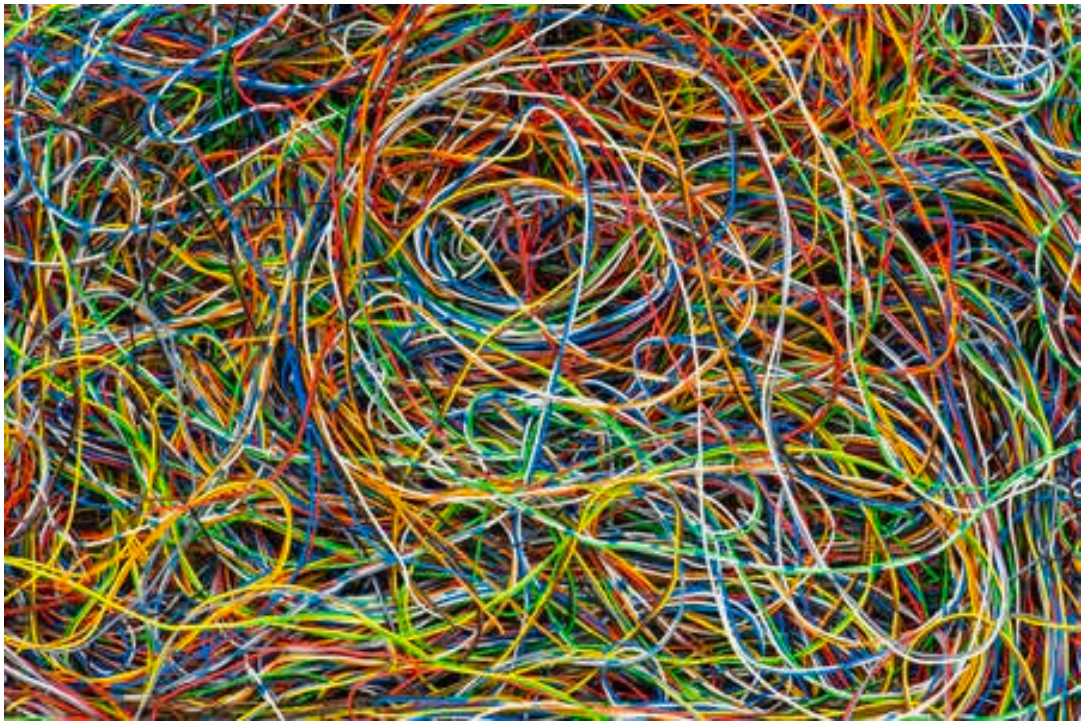
Node.js

Vue.js

React.js

Angular.js

Securité …



Et projet à définir

Créer un site web dynamique (front-end / back-end).

Utiliser au moins un framework.