Développement web 2

Web dynamique

Enseignante :

Ibtissem BARHOUMI

Objectif du cours : Conception d'applications Web

Objectifs du Cours

1. Apprendre les bases avancées de PHP et JavaScript.
2. Construire une CRUD complet.

Sommaire

1. Rappel HTML/CSS/Bootstrap
2. Architecture web avancée
3. Introduction à PHP
4. Introduction à JavaScript
5. Gestion des Données Dynamiques (PHP, JS)
6. Traitement des Formulaires (côté serveur et client)
7. Frameworks JavaScript modernes
8. Projet final

Chapitre 1 : Rappel HTML/CSS/Bootstrap

Introduction à HTML

- HTML (HyperText Markup Language) est le langage de base pour structurer une page web.
- Chaque page HTML est composée d'éléments appelés balises.

Structure de Base d'une Page HTML

```
<!DOCTYPE html>
    <html>
     <head>
         <title>Titre de la page</title>
     </head>
     <body>
        Contenu de la page
     </body>
    </html>
```

Éléments HTML Courants

- 1. Titres : <h1> à <h6>
- 2. Paragraphes : <p>
- 3. Liens : <a href="">
- 4. Images : <img src="">
- 5. Formulaires : <form>, <input>, <select> , <button>, etc.

Formulaires en HTML

- Un formulaire permet de collecter des données utilisateur :

```
<form>
  <label for='name'>Nom et Prénom  :</label>
  <input type='text' id='name' name='name'>
<label for=‘email’>email:</label>
  <input type=‘email' id=‘email' name=‘email'>

  <button type='submit'>Envoyer</button>
</form>
```

Introduction à CSS

- CSS (Cascading Style Sheets) permet de styliser une page HTML.
- Avec CSS, vous pouvez définir des couleurs, des polices, des marges, et plus.

Syntaxe CSS : Sélecteurs et Propriétés

- Un style CSS est défini comme suit :
- sélecteur { propriété: valeur; }
- Exemple :
- body { font-family: Arial; color: blue; }

Exemple de CSS Pratique

- 1. Modifier la couleur du texte :
  - h1 { color: red; }
- 2. Centrer une image :
  - img { display: block; margin: auto; }
- Etc…..

Introduction à Bootstrap

- Bootstrap est un framework CSS qui facilite la création de designs responsives.
- Avantages :
- 1. Système de grilles pour la mise en page.
- 2. Composants prêts à l’emploi.

Utilisation du CDN Bootstrap

- Pour intégrer Bootstrap :
- <link href='https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css' rel='stylesheet'>

Grilles et Conteneurs Bootstrap

- Bootstrap utilise un système de grilles :
  - 1. Conteneurs : .container, .container-fluid.
  - 2. Colonnes : .col-1 à .col-12.

Composants Bootstrap

- Exemples de composants :
  - 1. Boutons : <button class='btn btn-primary'>Bouton</button>
  - 2. Cartes : ...
  - 3. Barre de navigation : <nav class='navbar'>...</nav>
  - 4. Caroussel :

Exercice Pratique

- Créer une page web avec Bootstrap :
- 1. Inclure un conteneur responsive.
- 2. Styliser un formulaire avec des boutons Bootstrap.
- 3. Ajouter tableau listes des users avec 2 boutons
- 4. Ajouter bouton ajouter

Chapitre 2 :

Architecture web avancée : mise en œuvre client/serveur/base de données

Introduction aux architectures d’application web

- **Importance de l’architecture d’application web**
  - Assure stabilité, scalabilité et maintenabilité des applications web.
  - Améliore les performances et l’expérience utilisateur.
  - Simplifie le processus de développement.

- **Composants clés des architectures d’application**
  - Frontend
  - Backend
  - Bases de données
  - APIs

Modèles de conception d’applications web

- **Modèle-Vue-Contrôleur (MVC)**
  - Sépare l’application en trois composants : Modèle, Vue, Contrôleur.
  - Facilite la gestion et la maintenance du code.
- **Modèle-Vue-ViewModel (MVVM)**
  - Simplifie le développement d’interfaces utilisateur complexes.
  - Met l’accent sur la liaison de données entre la Vue et le ViewModel.
- **Modèle-Vue-Présentateur (MVP)**
  - Le Présentateur agit comme intermédiaire entre la Vue et le Modèle.
  - Facilite le développement piloté par les tests.

Architectures d’applications modernes

- **Microservices**
  - Divise l’application en petits services indépendants.
  - Chaque service est responsable d’une fonctionnalité spécifique.
  - Favorise l’agilité, la scalabilité et la résilience.
- **Informatique sans serveur (Serverless)**
  - Permet de construire et déployer des applications sans gérer l’infrastructure.
  - Les ressources sont allouées à la demande.
  - Réduit les coûts et la complexité opérationnelle.

Sélection de la bonne architecture d'application web

- **Considérations clés**
  - Exigences fonctionnelles et non fonctionnelles.
  - Échelle et complexité du projet.
  - Compétences de l'équipe de développement.
  - Contraintes budgétaires et temporelles.

- **Étapes pour choisir la bonne architecture**
  1. Analyser les besoins du projet.
  2. Évaluer les options architecturales disponibles.
  3. Considérer les avantages et inconvénients de chaque option.
  4. Sélectionner l'architecture la mieux adaptée aux objectifs du projet.

Scalabilité des applications web

- **Importance de la scalabilité**
  - Assure que l'application peut gérer une augmentation du nombre d'utilisateurs.
  - Maintient des performances optimales sous une charge accrue.
- **Stratégies de scalabilité**
  - **Mise à l'échelle verticale**
    - Ajouter des ressources à un serveur existant (CPU, RAM).
  - **Mise à l'échelle horizontale**
    - Ajouter plus de serveurs pour répartir la charge.
- **Techniques d'équilibrage de charge**
  - Distribuer le trafic réseau de manière égale entre plusieurs serveurs.
  - Améliore les performances et la fiabilité de l'application.

L’architecture du web

- L’architecture du web se base sur les modèles de **Client/Serveur.**

- Le client envoie des requêtes au serveur, comme:
  - Transfert de fichiers
  - Exécution de programmes sur le serveur
  - Mise à jour de fichiers

- Les objets manipulés sont repérés par leur **URL**.
- Le transfert se fait en utilisant le protocole **http**. Il définit le langage utilisé pour les échanges entre client et serveur Web. Ce protocole n’exige pas de session permanente entre client/serveur

L’architecture du web : un serveur web

- Un serveur informatique qui répond à des requêtes du World Wide Web, en utilisant principalement le protocole HTTP.

- Un ordinateur qui stocke les fichiers qui composent un site web (par exemple les documents HTML, les images, les feuilles de style CSS, les fichiers JavaScript) et qui les envoie à l'appareil de l'utilisateur qui visite le site.
- Cet ordinateur est connecté à Internet et est généralement accessible via un nom de domaine.

L’architecture du web : déroulement

DEROULEMENT D’UNE REQUETE :

- 1. Demande d’une connexion
- 2. Attente de la réponse du serveur
- 3. Etablissement de la connexion
- 4. Envoi d’une requête URL
- 5. Réponse du serveur
- 6. Affichage de la réponse
- 7. Fermeture de la connexion

L’architecture du web : Serveur DNS (domain name system)

- C’est un protocole permettant d’associer à des noms de domaine (par exemple www.test.tn) une adresse IP (par exemple 100.78.132.45). Les machines se connectent entre-elles à l’aide d’adresses IP, mais les noms de domaine servent à faciliter la mémorisation et l’utilisation pour les humains.

- Le serveur DNS est utilisé pour que l’ordinateur récupère l’adresse du serveur à joindre, ensuite il n’a plus besoin du serveur DNS.

L’architecture du web : Nom du domaine

- Une adresse Internet ou nom de domaine est la manière dont vos contacts et clients vont trouver votre site Internet sur le web. Un nom de domaine est donc indispensable lors de la création de votre site web
- Une adresse Internet se compose d'un préfixe "www" (world wideweb) et d'un nom de domaine.
- Ce nom de domaine est lui-même composé d'une chaîne de caractères et d'une extension.
- Dans l'exemple ci-dessous, l'extension utilisée est relative à la France : le **.fr**

Les différents types de sites web

Il existe différents types de sites web, chaque type correspond à un objectif.

On peut citer:

- **Les sites e-commerce ou sites marchands**: leur objectif principal est la vente en ligne.
- Les sites vitrines :utilisés généralement pour présenter et exposer les services d'une organisation ou une marque.
- **Les sites institutionnels** :qui présentent une organisation et ses valeurs, à travers la description de son activité, de ses chiffres clés et des informations nécessaires au public ciblé.
- **Les sites portails**: qui proposent des services de messagerie, d'actualités.
- **Les sites personnels** : réalisés au profit des particuliers, qui désirent partager en ligne leur passion pour un sujet précis.
- Cette liste est loin d'être exhaustive, on peut trouver d'autres sites.

Application web

- Une application web désigne un logiciel applicatif hébergé sur un serveur et accessible via un navigateur web.
- Contrairement à un logiciel traditionnel, l'utilisateur d'une application web n'a pas besoin de l'installer sur son ordinateur.
- Il lui suffit de se connecter à l'application à l'aide de son navigateur.

Application web

Avantages d'une application web:

- 1. Accès universel depuis n'importe quel type de poste : PC, portables, téléphone mobile, tablette.
- 2. Aucune incompatibilité de système d'exploitation(il suffit d'avoir un navigateur);
- 3. Travailler de puis n'importe quel endroit de la planète;
- 4. Les données sont centralisées;
- 5. Les données sont disponibles 24h sur 24 et 7j sur 7;
- 6. Aucun risque de perte de données.

Application web

Application lourde vs application légère

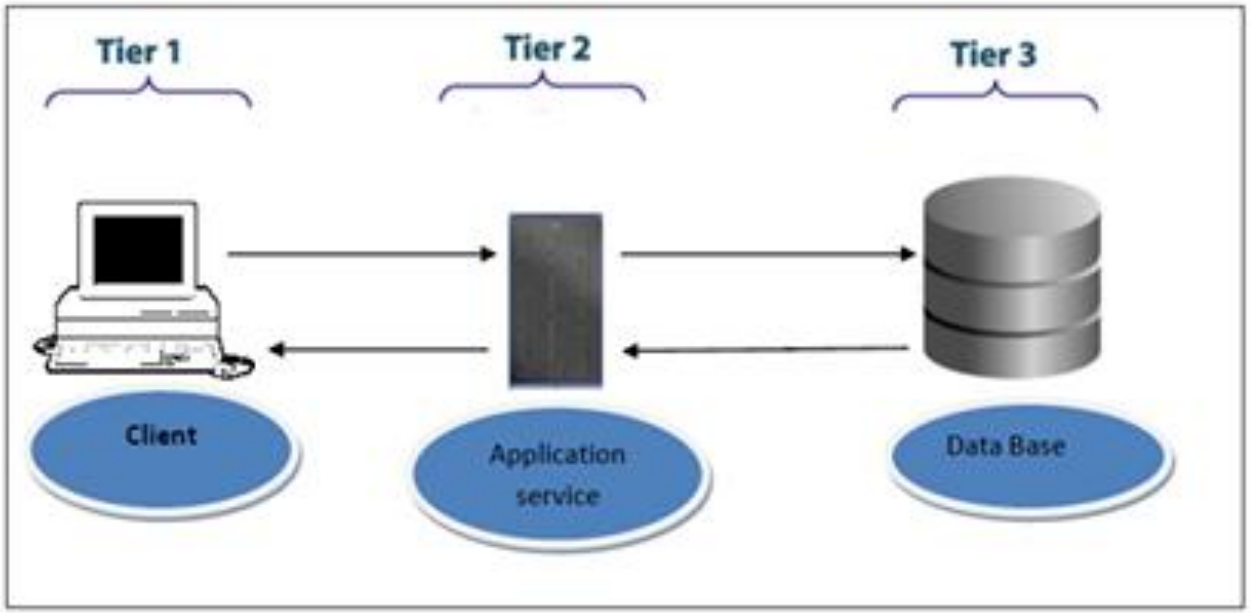
- Une application lourde, est une application réalisée pour s'exécuter sur une machine mono utilisateur.
- Une application légère est une application client/serveur dans laquelle la partie interface utilisateur est visible dans un navigateur Web, alors que la partie serveur (traitements + données) est hébergée sur un serveur(web + de base de données).
- L'avantage de l'application légère est qu'elle ne nécessite aucune installation sur les machines des utilisateurs, leur mise à jour se fait en un seul endroit(le serveur), et que l'utilisateur peut y accéder à partir de différents système d'exploitation/terminaux sans effort, ce qui n'est pas le cas des applications lourdes.

Application web

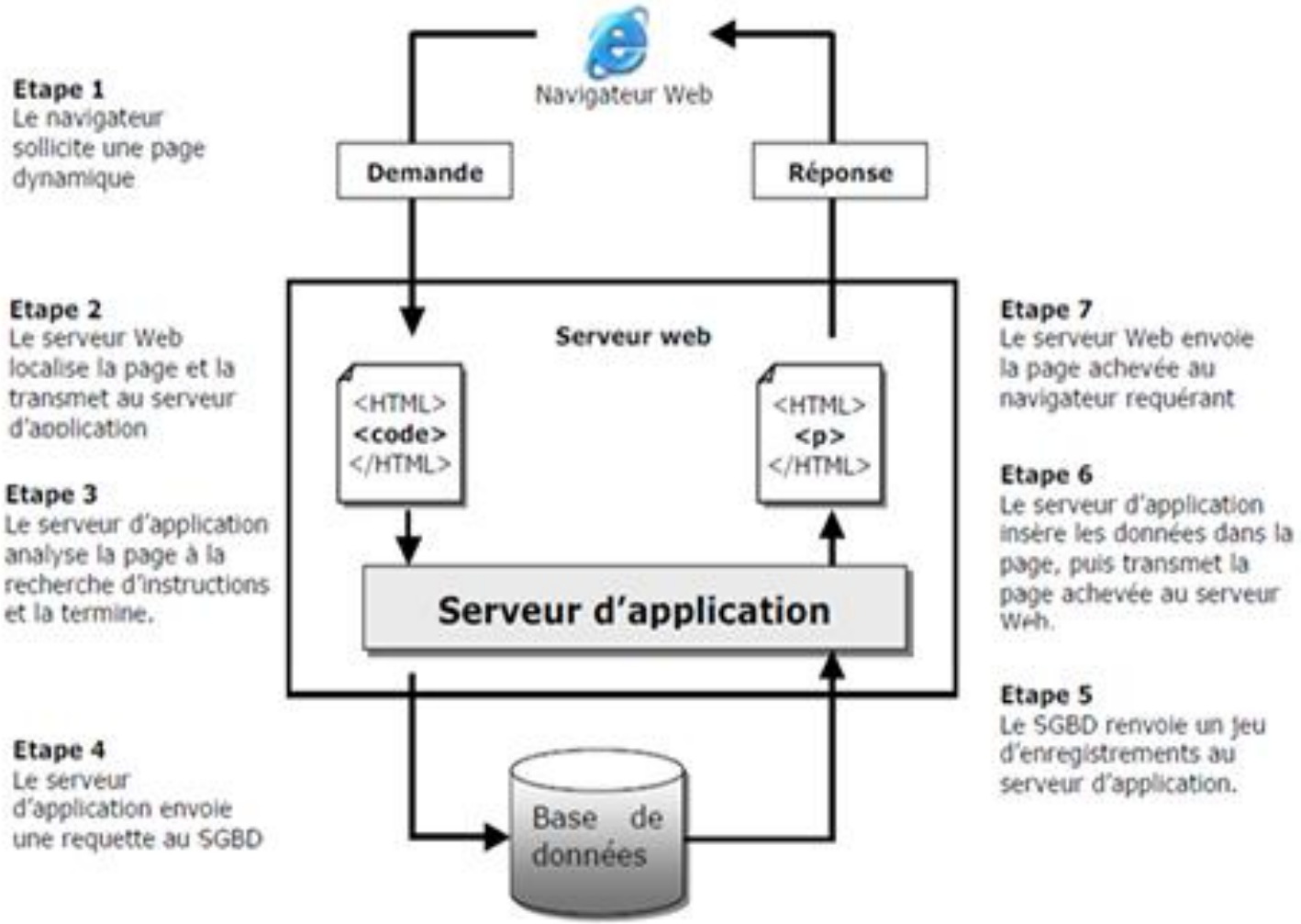
- L'architecture des applications web: En règle générale, une application est découpée en 3 niveaux (couches) d'abstraction :
- **1. La couche présentation :** c'est la partie de l'application visible par les utilisateurs (nous parlerons d'interface utilisateur). Dans notre cas, cette couche est un navigateur web, qui se présente sous forme de pages HTML, composée de formulaire et de bouton.
- **2. La couche métier**: correspond à la partie fonctionnelle de l'application, celle qui décrit les opérations exécutées sur les données, en fonction des requêtes d'un utilisateur effectuées à travers la couche présentation.
- **3. La couche accès aux données** : c'est la partie gérant l'accès à la base de données du l'application.

Application web

- L'architecture des applications web: Il existe différentes architectures pour une application web :
  - Architecture 1-tiers
  - Architecture 2-tiers
  - Architecture 3-tiers
  - Architecture n-tiers



Application web



Les langages utilisés pour la création des applications web

- Le développement d’une applications web nécessite la connaissance des différents langages utilisés dans les technologies web:
  - HTML pour la présentation des pages , CSS(Cascading Style Sheets) pour la charte graphique, JavaScript pour les scripts exécutés par le client
  - ainsi qu'un langage telque Java, le PHP , le NodeJS et d'autre.

Chapitre 3 : Introduction PHP

Introduction à PHP

- Objectifs :
    1. Comprendre ce qu'est PHP et son rôle.
    2. Apprendre la syntaxe de base de PHP.
    3. Intégrer PHP dans une page HTML.

PHP: Introduction et Installation (2 heures)

Objectifs :

- Comprendre le rôle de PHP et configurer un environnement de travail.

Qu'est-ce que PHP ?

- PHP (Hypertext Preprocessor) est un langage côté serveur.
- Permet de générer dynamiquement des pages web.

Installation de PHP

- Installez XAMPP, WAMP, ou MAMP.
- Placez vos fichiers PHP dans le dossier 'htdocs’./ www
- Accédez à vos scripts via http://localhost.

Installation et configuration

- Téléchargement et installation de **XAMPP**.
  - https://www.apachefriends.org/fr/download.html
- Structure des fichiers dans **htdocs**.
- Accéder à un script via **http://localhost**.

Premier Script PHP

```
<?php
  echo 'Bienvenue dans PHP !';
?>
```

Syntaxe de Base : Variables

Déclaration :

```
$variable = valeur;
```

Exemple :

```
$nom = 'Alice';
$age = 25;
```

Variables en PHP

- Déclarées avec le symbole $:
  - $nom = 'Alice';
  - $age = 25;

Exemple :

```
<?php
	$nom = "Alice";
	$age = 25;
	echo "Bonjour, $nom. Vous avez $age ans.";
?>
```

Conditions

Exemple :

```
if ($age >= 18) {
        echo 'Adulte';
      } else {
        echo 'Mineur';
}
```

Boucles

For :

```
for ($i = 0; $i < 5; $i++) {
        echo $i;
      }
```

While :

$x=2;

```
while ($x < 5) {
        echo $x++;
}
```

Boucles et conditions dans HTML :

- **Exemple : Génération d'une liste dynamique**

```
<?php
$articles = ["Article 1", "Article 2", "Article 3"];
?>
<ul>
    <?php foreach ($articles as $article): ?>
        <li><?php echo $article; ?></li>
    <?php endforeach; ?>
</ul>
```

PHP dans HTML

```
<!DOCTYPE html>
<html>
<body>
        <h1> <?php echo 'Titre'; ?> </h1>
</body>
</html>
```

Exercice pratique :

- Créez une page qui affiche dynamiquement la liste des produits dans un tableau .

- Déclaration :

```
$produits = [
    ["nom" => "Ordinateur", "prix" => 1200],
    ["nom" => "Téléphone", "prix" => 800],
["nom" => « Tableette", "prix" => 1000]
];
```

```
<td><?php echo htmlspecialchars($produit['nom']); ?></td>
<td><?php echo htmlspecialchars(number_format($produit['prix'], 2)); ?></td>
```

Tableaux (suite)

- Tableaux associatifs - parcours avec boucle foreach :

```php
<?php
//Tableaux associatifs- parcours avec boucle foreach :
 $jours=array("Lu"=>"Lundi","Ma"=>"Mardi",
"Me"=>"Mercredi","Je"=>"Jeudi","Ve"=>"Vendredi", "Sa"=>"Samedi","Di"=>"Dimanche" );
 foreach($jours as $key=>$val) echo $key. " - ".$val."<br>\n";
?>
```

Récupération Data – formulaire

- Form.html
  - Formululaire avec champs nom et bouton submit
- Reponse.php:
  - <?php echo $_GET['nom'] ?></h1>

Formulaires HTML avec PHP

```
<form method='POST'>
  <input type='text' name='nom'>
  <button type='submit'>Envoyer</button>
</form>
<?php
if ($_POST) {
  echo 'Bonjour, '.$_POST['nom'];
}
?>
```

Fonctions

Exemple :

```php
<?php
   function salut($nom) {
     return 'Bonjour, '.$nom;
   }
   echo salut('Alice');
?>
```

Exercice : Function

Enoncé : Exercice : Créer un script PHP pour des opérations mathématiques

1.Créez quatre fonctions : addition, soustraction, multiplication, et division.
2.Chaque fonction doit prendre deux paramètres et retourner le résultat correspondant.
3.Testez chaque fonction avec différents arguments et affichez les résultats dans un navigateur.

Cours : Génération de Contenu JSON ou CSV en PHP

- **Objectifs :**
- Comprendre les formats JSON et CSV.
- Apprendre à générer et manipuler ces formats en PHP.
- Utiliser JSON pour les échanges de données et CSV pour l'exportation tabulaire.

JSON (JavaScript Object Notation)

- Format léger pour l’échange de données.
- Lisible par les humains et facilement interprétable par les machines.
- Structure : Clé-valeur (similaire à un objet JavaScript).

Exemple de JSON :

{ "nom": "Alice", "age": 25,   "email": alice@example.com}

```
{
    "nom": "Alice",
     "age": 25,
     "email": alice@example.com
    }
```

CSV (Comma-Separated Values)

- Format simple pour représenter des données tabulaires.
- Les colonnes sont séparées par des virgules (ou d'autres délimiteurs comme ";").

Exemple de CSV :

```
Nom,Email,Age
Alice,alice@example.com,25
Bob,bob@example.com,30
```

Génération de JSON avec PHP

Utilisation de la fonction json_encode()

•Convertit un tableau PHP ou un objet en JSON.

Exemple

```php
<?php

// Définir les données à convertir en JSON

$data = [

    "nom" => "Alice",

    "age" => 25,

    "email" => "alice@example.com"

];
// Convertir le tableau en JSON avec un formatage lisible

$json = json_encode($data, JSON_PRETTY_PRINT);


// Définir l'en-tête pour indiquer que le contenu est au format JSON

header("Content-Type: application/json");


// Afficher le contenu JSON

echo $json;

?>
```

Génération de CSV avec PHP

Utilisation de la fonction fputcsv()

- Écrit une ligne dans un fichier CSV.

Exemple :

```php
<?php

$utilisateurs = [

    ["Nom", "Email", "Age"],

    ["Alice", "alice@example.com", 25],

    ["Bob", "bob@example.com", 30]

];
// Ouvrir un fichier en mode écriture

$fichier = fopen("utilisateurs.csv", "w");
// Parcourir les données et écrire chaque ligne dans le fichier

foreach ($utilisateurs as $utilisateur) {

    fputcsv($fichier, $utilisateur);

}
// Fermer le fichier

fclose($fichier);
echo "Fichier CSV généré avec succès.";

?>
```

Génération de contenu JSON ou CSV

Génération d'un fichier JSON :

```
<?
php $data = ["nom" => "Alice", "age" => 25];
header("Content-Type: application/json");
echo json_encode($data);
?>
```

Génération d'un fichier CSV :

```
<?php
    $file = fopen("export.csv", "w");
    fputcsv($file, ["Nom", "Email"]);
    fputcsv($file, ["Alice", "alice@example.com"]);
    fclose($file);
    echo "CSV généré.";
?>
```

Exercice pratique : 30 min

- Créez un fichier JSON contenant une liste de tâches, et un fichier CSV contenant une liste d’utilisateurs.

Ajout dynamique de contenu dans un fichier CSV (15 min)

- Écrire des données utilisateurs envoyés par formulaire.

- Indication :
  - fopen(nomfichier, « a »);

PHP: Connexion à une Base de Données /avec PDO

- **Objectifs :**
  - 1. Connecter PHP à une base de données MySQL.
  - 2.Apprenez à connecter PHP à une base de données MySQL de manière sécurisée et efficace.
  - 3. Effectuer des opérations CRUD.

Qu'est-ce que PDO ?

- PDO (PHP Data Objects) est une extension PHP pour interagir avec des bases de données.
- Supporte plusieurs types de bases de données (MySQL, PostgreSQL, SQLite, etc.).
- Avantages : Sécurité, Flexibilité, et Requêtes Préparées.

Connexion à MySQL avec PDO

- <?php
- $pdo = new PDO('mysql:host=localhost;dbname=test_db', 'root', '');
- echo 'Connexion réussie !';
- ?>

Connexion à une Base MySQL avec PDO

- **Exemple de connexion :**

```
<?php
$host = 'localhost';
$dbname = 'test_db';
$username = 'root';
$password = '';
try {
    $pdo = new PDO("mysql:host=$host;dbname=$dbname", $username, $password);
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    echo "Connexion réussie !";
} catch (PDOException $e) {
    echo "Erreur : " . $e->getMessage();
}
?>
```

Encodage JSON à partir d'une BD

- Si on récupère des données de la table CARNET pour les exporter en JSON

Génération de contenu dynamique depuis une base de données

- **Objectif :** Générer des pages dynamiques en récupérant des données depuis MySQL.
  - Connexion à la base de données avec PDO

```
<?php
    $pdo = new PDO("mysql:host=localhost;dbname=test_db", "root", "");
    $stmt = $pdo->query("SELECT nom, email FROM utilisateurs");
?>
```

  - Afficher les données dans un tableau HTML

Opérations CRUD avec PDO

- 1. Create : Ajouter des données
- 2. Read : Lire des données
- 3. Update : Mettre à jour des données
- 4. Delete : Supprimer des données

Requêtes CRUD avec PDO

- Create : INSERT INTO utilisateurs (nom, email) VALUES (:nom, :email)
- Read : SELECT * FROM utilisateurs
- Update : UPDATE utilisateurs SET email = :email WHERE nom = :nom
- Delete : DELETE FROM utilisateurs WHERE nom = :nom

Create : Ajouter des Données

- **Exemple** :

```
<?php
$sql = "INSERT INTO utilisateurs (nom, email) VALUES (:nom, :email)";
$stmt = $pdo->prepare($sql);
$stmt->execute(['nom' => 'Alice', 'email' => 'alice@example.com']);
echo "Utilisateur ajouté.";
?>
```

Read : Lire des Données

- Exemple :

```
<?php
$sql = "SELECT * FROM utilisateurs";
$stmt = $pdo->query($sql);

while ($row = $stmt->fetch()) {
    echo $row['nom'] . " - " . $row['email'] . "<br>";
}
?>
```

Delete : Supprimer des Données

- **Exemple** :

```
<?php
$sql = "DELETE FROM utilisateurs WHERE nom = :nom";
$stmt = $pdo->prepare($sql);
$stmt->execute(['nom' => 'Alice']);
echo "Utilisateur supprimé.";
?>
```

Update : Mettre à Jour des Données

- Exemple :

```
<?php
$sql = "UPDATE utilisateurs SET email = :email WHERE nom = :nom";
$stmt = $pdo->prepare($sql);
$stmt->execute(['email' => 'new_email@example.com', 'nom' => 'Alice']);
echo "Utilisateur mis à jour.";
?>
```

Fonctionnalités : **htmlspecialchars et trim**

- **htmlspecialchars**
  - est utilisé pour **sécuriser** les données en les rendant inoffensives lorsqu'elles sont affichées dans une page web.

Purquoi utiliser htmlspecialchars ?

- Sécuriser les champs de formulaire ou données affichées à l'utilisateur.
- Éviter que des scripts malveillants ne soient exécutés par le navigateur.

- **Exemple**

```
<td><?= htmlspecialchars($user['name']) ?></td>
```

- **trim**
  - est utilisé pour nettoyer les données en supprimant les espaces ou caractères inutiles avant de les traiter ou valider.
  - Exemple :

```
$name = trim($_POST['name']);
```

Avantages de PDO

- Sécurité : Requêtes préparées pour éviter les injections SQL.
- Portabilité : Compatible avec plusieurs bases de données.
- Flexibilité : Gestion simplifiée des erreurs.

Conclusion

- PDO est un outil puissant pour gérer les bases de données en PHP.
- Il offre une méthode sécurisée et flexible pour interagir avec des données dynamiques.

Exercice

- Créer une application :
- 1. Ajouter un utilisateur via un formulaire.
- 2. Afficher les utilisateurs dans un tableau.
- 3. Supprimer un utilisateur avec un bouton.

Notion des sessions

session_start() :

- La fonction **session_start()** en PHP est utilisée pour démarrer une nouvelle session ou reprendre une session existante. Une session permet de **conserver des données utilisateur entre plusieurs requêtes HTTP** (comme des informations de connexion, un panier d'achat, etc.).

session_destroy()

- Déconnexion de l'utilisateur **:**
- Lorsqu'un utilisateur se déconnecte, toutes ses données de session (comme l'ID utilisateur ou son panier) doivent être supprimées.

- $_SESSION

Pourquoi utiliser une session ?

•Le protocole HTTP est **stateless** (sans état), ce qui signifie qu'il ne peut pas "se souvenir" d'un utilisateur ou d'une action entre deux requêtes.

•Les sessions permettent de stocker des données sur le serveur, et un identifiant de session est envoyé au client (généralement via un cookie).

•Les données d'une session sont accessibles sur toutes les pages où session_start() est appelé.

Exemple :

- session_start(); // Reprendre ou démarrer la session active
- session_destroy(); // Détruire toutes les données associées à la session

Exemple :

```
<?php
session_start();
session_destroy();
echo "Vous avez été déconnecté avec succès.";
?>
```

Chapitre 4 : Introduction JS

Introduction à JavaScript pour le Web Dynamique

- **Objectifs pédagogiques :**

1. Comprendre le rôle de JavaScript dans la création de pages web interactives et dynamiques.
2. Apprendre à manipuler le DOM (Document Object Model).
3. Maîtriser les événements pour interagir avec les utilisateurs.
4. Découvrir les bases de la communication avec un serveur via AJAX.

Programmation coté client

- Ceci suppose d'avoir un langage de programmation *généraliste* (≠ HTML/CSS) compris par tous les navigateurs.

- Actuellement, ce langage est Javascript.

Note

Javascript a beaucoup évolué au cours de son histoire.

La version que nous présentons ici est ES6, une version relativement récente (2015) qui a apporté beaucoup de nouveautés au langage, mais également des incompatibilités avec les versions précédentes.

Gardez cela en tête lorsque vous trouverez des exemples en ligne.

Syntaxe

- Comme son nom l'indique, la syntaxe de Javascript est (librement) inspirirée de celle de Java (ou du C).
- La similitude s'arrête là : Javascript n'est pas basé sur Java.

Qu'est-ce que JavaScript ?

•JavaScript est un langage de programmation qui permet de rendre les pages web dynamiques et interactives. Contrairement au HTML et CSS qui se concentrent sur la structure et le style, JavaScript ajoute des fonctionnalités comme des formulaires interactifs, des animations, ou des réponses immédiates aux actions des utilisateurs.

•C’est un langage interprété directement dans le navigateur et pris en charge par tous les principaux navigateurs web.

•Par exemple, quand vous voyez une page qui met à jour son contenu sans se recharger, c'est souvent grâce à JavaScript.

Pourquoi l'utiliser côté client ?

•L’ 'utilisation de JavaScript côté client permet de réduire le temps de réponse car les interactions sont traitées directement dans le navigateur sans dépendre du serveur.

•Il offre une meilleure expérience utilisateur en rendant les pages web plus fluides et interactives.

•Il est également très flexible, permettant de créer tout type d’interaction, comme des menus déroulants, des carrousels d'images ou des validations de formulaire.

•JavaScript côté client réduit également la charge sur le serveur, car une partie du traitement est déportée chez l'utilisateur.

Intégration de JavaScript dans HTML

- Utilisation de la balise <script>.
- Inclusion interne (dans le fichier HTML).
- Inclusion externe (dans un fichier .js).

Exercice : Créez une page HTML contenant un script qui affiche "Bonjour, Monde !" dans la console.

Variables et Types de Données

- Variables : let, const, var.
- Types de données : string, number, boolean, array, object.
- Exemple : let nom = "Ibtissem"; const age = 25;

Exercice : Créez un script pour afficher un message de bienvenue avec une variable contenant le prénom.

Exemple : Déclaration de Variables

- ```javascript
- let nom = 'Alice';
- const age = 25;
- var ville = 'Paris';
- console.log(nom, age, ville);
- ```

Liste syntaxe

Condition

```javascript
if (i < 10) {
    j = j+1;
    k += i;
} else {
    j = 0;
}
```

Boucles

```javascript
while (i < 10) {
    j = j*i;
    i += 1;
}
```

```javascript
for(let i of [1,1,2,3,5,8]) {
    j = j*i;
}
```

```javascript
for(let i=2; i<1000; i=i*i) {
    console.log(i);
}
```

Fonctions

```javascript
function fact(n) {
  let f = 1;
  while (n>1) {
    f = f*n;
    n -= 1;
  }
  return f;
}
```

Exceptions

```javascript
if (i < 0) {
    throw new Error(
      "negative value");
}
```

```javascript
try {
    i = riskyFunction();
}
catch (err) {
    i = -1;
}
```

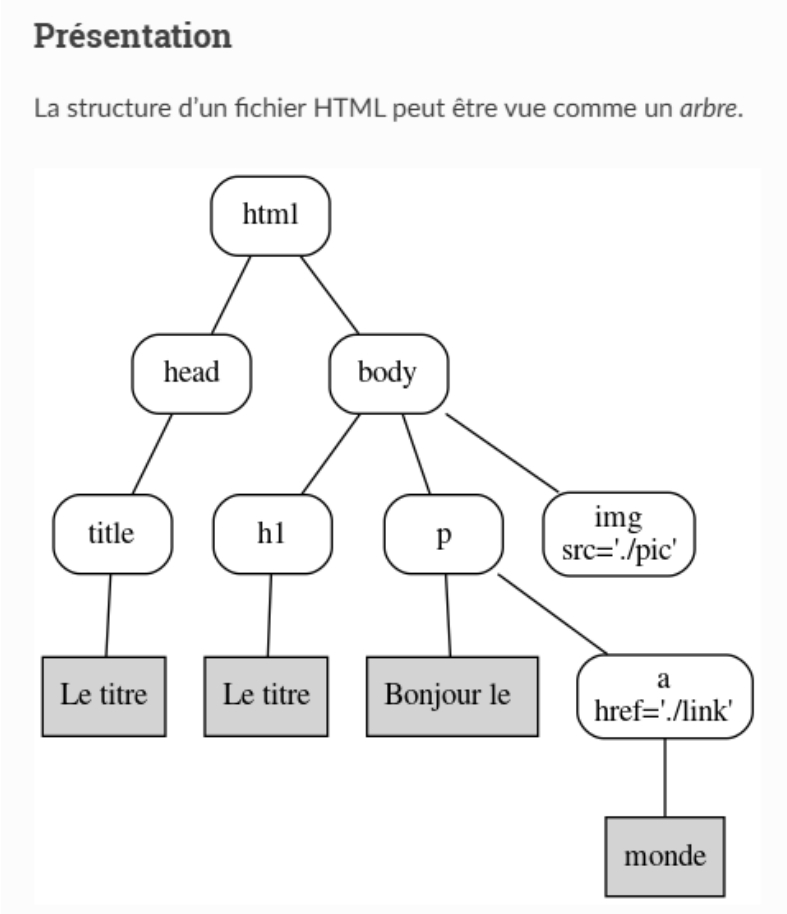
Tableaux

```javascript
let a = [4,1,3,6,4];
let i = 1;
while (i<a.length) {
  a[i] = a[i]+a[i-1];
  i += 1;
}
```

Manipulation du DOM

- Qu'est-ce que le DOM (Document Object Model) ?
  - Le DOM est une représentation en arbre de votre page HTML. Chaque élément HTML est un noeud de cet arbre.
  - JavaScript permet de naviguer dans cet arbre pour accéder, modifier ou supprimer les éléments HTML.

L'arbre DOM



Accéder à un élément :

•*document.getElementById:* retourne un élément avec un ID spécifique.

•*document.querySelector*: retourne le premier élément correspondant à un sélecteur CSS.

Modifier le contenu et les styles :

•*element.textContent:* modifie le texte contenu dans un élément.

•*element.style:* permet de modifier les styles CSS directement en JavaScript

Manipulation du DOM

Code Exemple :

```
const titre = document.getElementById('monTitre');
titre.textContent = "Bonjour JavaScript !";
titre.style.color = "blue";
```

Exercice : Ajoutez un bouton qui change le texte et la couleur d'un paragraphe.

Gestion des événements

- **Qu'est-ce qu'un événement ?**
  - Un événement est une action ou occurrence reconnue par JavaScript, comme un clic de souris, une touche appuyée, ou un survol de l'utilisateur.
  - Les événements permettent d'interagir avec l'utilisateur pour réagir à ses actions.

- **Les principaux événements :**
  - **click** : Se produit lorsque l'utilisateur clique sur un élément.
  - **mouseover** : Se produit lorsque l'utilisateur passe le curseur sur un élément.
  - **keydown** : Se produit lorsqu'une touche du clavier est enfoncée.

Événement addEventListener

- **Ajouter un écouteur d'événement avec addEventListener:**
  - Utilisez addEventListener pour attacher un événement à un élément HTML.

- Syntaxe :

```
element.addEventListener('event', function);
```

- Exemple :

```
const bouton = document.getElementById('monBouton');
bouton.addEventListener('click', () => {
    alert("Bouton cliqué !");
});
```

Exercice : Créez un champ texte et un bouton. Lorsque l'utilisateur clique sur le bouton, affichez la valeur du champ dans une alerte.

Création d'éléments Dynamiques

- Ajouter ou supprimer des éléments HTML.

1.Ajouter un élément HTML :

- Utilisez document.createElement pour créer un nouvel élément HTML dynamiquement.
- Modifiez ses propriétés ou son contenu avec des méthodes comme textContent ou setAttribute.
- Ajoutez-le à un parent existant avec appendChild ou insertBefore.

2.Supprimer un élément HTML :

- Utilisez parentElement.removeChild(element) pour retirer un enfant d'un élément parent.
- Vous pouvez cibler le dernier enfant avec lastChild ou un enfant spécifique avec querySelector.

Exemple : Liste dynamique

- Ajouter des éléments à une liste avec un bouton "Ajouter".
- Supprimer le dernier élément avec un bouton "Supprimer"

Code Exemple :

```
const liste = document.getElementById('maListe');
const boutonAjouter = document.getElementById('ajouter');
const boutonSupprimer = document.getElementById('supprimer');

boutonAjouter.addEventListener('click', () => {
    const nouvelElement = document.createElement('li');
    nouvelElement.textContent = "Nouvel élément";
    liste.appendChild(nouvelElement);
});

boutonSupprimer.addEventListener('click', () => {
    if (liste.lastChild) {
        liste.removeChild(liste.lastChild);
    }
});
```

Exercice

- **Exercice :** Créez une liste dynamique avec des boutons pour ajouter et supprimer des éléments.

Bonnes Pratiques et Debugging

1.console.log : Sert à afficher des messages dans la console pour suivre l'exécution d'un script ou vérifier des valeurs.

2.console.error : Utilisé pour signaler des erreurs dans le code.

3.console.table : Permet d'afficher des tableaux ou des objets sous forme de tableau pour une meilleure lisibilité.

Utilisation des outils de développement du navigateur :

- Accéder à la console via le menu "Inspecter" du navigateur.
- Utiliser l'onglet "Console" pour voir les messages et erreurs.
- Profitez des outils comme le débogueur pour mettre en pause et inspecter votre code en éxécution.

Exemple : Affichage des étapes dans la console pour suivre un script.

Exercice Final

- **Exercice :** Ajoutez des **console**.log à différentes étapes d'un script pour déboguer.

Récapitulatif et Questions

- Récapitulatif des concepts abordés :
  - Intégration de JavaScript.
  - Manipulation du DOM.
  - Gestion des événements.
  - Debugging.

Chapitre 7 : Projet Final

Présentation du Projet

- **Contexte**

  - Vous avez déjà étudié les bases du HTML, CSS, Bootstrap, et PHP, ainsi que des notions de CRUD et de sessions. Vous avez également suivi un cours sur le JavaScript côté client. Ce projet vous permettra de combiner toutes ces compétences pour créer une application web dynamique complète.

- Technologies utilisées : HTML, CSS, Bootstrap, JavaScript, PHP, MySQL.

Fonctionnalités Clés

- Ajouter, modifier, et supprimer des tâches/produits.
- Afficher une liste dynamique des tâches/ produits.
- Stocker les données dans une base de données.
- Notion de session
- Js Coté Client :recherche
- Export

Étapes du Projet

- 1. Créer la base de données (table tasks).
- 2. Développer l'interface utilisateur avec HTML/Bootstrap.
- 3. Ajouter des interactions dynamiques avec JavaScript.
- 4. Gérer le back-end avec PHP et PDO.

Exemple

- **Mini-Projet : Gestion d’un Carnet d’Adresses Dynamique**
- **Objectif :** Créer une application complète permettant de gérer un carnet d’adresses en utilisant HTML, CSS, Bootstrap, JavaScript (côté client), et PHP (backend). Ce projet met l'accent sur l'intégration de fonctionnalités JavaScript pour dynamiser l'interface utilisateur.

Critères d'évaluation

1. Fonctionnalités implémentées correctement (authentification, gestion des contacts, Export).
2. Fonctionalité recherche
3. Utilisation propre de Bootstrap pour une interface responsive.
4. Bonne organisation des fichiers (structure claire du projet).
5. Respect des consignes de sécurité (hashage des mots de passe, protection des sessions).
6. Code bien commenté , structuré et lisible.

Livrables

- Un fichier ZIP contenant votre projet +presentation dans la classe
- Une documentation simple expliquant :
  - Les prérequis pour exécuter le projet (par exemple, installation de XAMPP).
  - Comment lancer le projet et les fonctionnalités incluses.
  - La base de données

Exercice noté : Gestion des Étudiants avec Bootstrap

🎯 Objectifs :

- Vous devez développer une application web permettant de **gérer une liste d'étudiants** dans une base de données. L'application doit offrir les fonctionnalités suivantes :

1. **Ajouter un étudiant** avec les champs suivants :
    1. **Nom**
    2. **Email**
    3. **Age**
    4. **classe**
2. **Afficher la liste des étudiants** avec un tableau responsive utilisant **Bootstrap**.
3. **Modifier les informations d'un étudiant**.
4. **Supprimer un étudiant**.
5. **Rechercher un étudiant en temps réel** à l'aide d'une **barre de recherche en JavaScript**

Etape 2 d’exercice

Ajout d'un champ "Age" et "Classe« :
ALTER TABLE etudiants ADD age INT NOT NULL, ADD classe VARCHAR(50) NOT NULL;