🔷 FRONT-END : HTML / CSS / ACCESSIBILITÉ / UX-UI

✅ HTML

- Qu'est-ce qu'un attribut `href` ? Donne un exemple.
- Quelle est la différence entre les balises `<section>`, `<article>`, `<div>`, `<main>`, et `<aside>` ?
- Quelle est l'utilité des balises `<header>` et `<footer>` ?
- À quoi sert la balise `<label>` ?
- Comment intégrer une image en HTML de façon accessible ?

✅ Accessibilité

- Qu'est-ce qu'un texte alternatif (`alt`) ? Pourquoi est-il important ?
- Quelle est la différence entre `aria-label`, `aria-hidden`, et `role` ?
- Que fait l'attribut `title` ?
- Donne des exemples de bonnes pratiques d'accessibilité (contraste, navigation clavier, etc.)

✅ CSS

- Qu'est-ce qu'un sélecteur descendant ? enfant ? adjacent ?
- Qu'est-ce que le modèle de boîte (box model) ?
- C'est quoi le `z-index` ?
- Que signifient les préfixes vendeurs (`-webkit-`, `-moz-`...) ?
- Quelle est la différence entre `em`, `rem`, `%`, `px` ?

✅ Responsive / Mobile First

- Qu'est-ce que la méthodologie mobile first ?
- Comment fonctionnent les media queries ? Syntaxe d'un exemple ?
- Quelle est la différence entre `min-width` et `max-width` ?
- Quelle différence entre `vh`, `vw`, `%` et `px` ?

✅ UX / UI / Maquettage

- Quelle est la différence entre un wireframe et une maquette ?
- Donne des bonnes pratiques UX : contraste, hiérarchie visuelle, taille des boutons, etc.
- Que signifie RWD (Responsive Web Design) ?
- Qu'est-ce que le format d'image WebP ? Quels avantages ?

🔷 BACK-END : PHP / SÉCURITÉ / LOGIQUE MÉTIER

✅ PHP / Sécurité

- Quelle est la différence entre le hachage et le chiffrement ?
- Pourquoi utilise-t-on `PASSWORD_BCRYPT` plutôt que `MD5()` ?
- Que signifie XSS ? Comment s'en prémunir ?
- Qu'est-ce qu'une injection SQL ? Comment la prévenir ?
- Qu'est-ce qu'un certificat SSL et à quoi sert le HTTPS ?

✅ Composants métiers et accès aux données

- Que signifie "composant métier" ?
- Quelle est la différence entre un composant de présentation et un composant métier ?
- Comment accéder à une base de données en PHP ? Donne un exemple de requête avec PDO.
- Qu'est-ce qu'une requête préparée ? Pourquoi l'utiliser ?

🔷 BASES DE DONNÉES : SQL / MERISE / SGBD

✅ SQL & MySQL

- Qu'est-ce qu'un SGBD ?
- Quelle est la différence entre MySQL et SQLite ?
- Que permet l'indexation d'une table ?
- Pourquoi utiliser `INNER JOIN` plutôt qu'une jointure implicite (`WHERE`) ?
- Que signifie une jointure croisée accidentelle ?
- Quels sont les différents types de jointures ?
- Expliquer les clauses : `SELECT`, `WHERE`, `GROUP BY`, `HAVING`, `ORDER BY`, `LIMIT`.

✅ Conception & modélisation

- Qu'est-ce qu'un MCD ?
- Qu'est-ce qu'un MPD ?
- Quelle est la méthode Merise ?
- Comment représenter une relation plusieurs-à-plusieurs ?
- Qu'est-ce qu'une table de liaison ? Pourquoi l'utilise-t-on ?
- Expliquer les cardinalités 0,1 ; 1,1 ; 0,N ; 1,N.
- Expliquer les contraintes `CASCADE`, `RESTRICT`, `SET NULL`.
- Différence entre moteur `InnoDB` et `MyISAM`.

✅ Règles & bonnes pratiques

- Quelles sont les conventions de nommage pour les tables et colonnes ?
- Pourquoi faut-il normaliser une base de données ?
- Comment éviter les redondances dans une BDD ?
- Qu'est-ce qu'une clé primaire ? une clé étrangère ?

🔷 DÉPLOIEMENT & PRODUCTION

- Quelles sont les étapes pour déployer une application web ?
- Quelle est la différence entre un environnement de développement et un environnement de production ?
- Qu'est-ce que le `.env` ? Pourquoi est-il important ?
- Pourquoi ne doit-on jamais exposer ses identifiants de BDD ?
- Qu'est-ce qu'un serveur Apache ? Nginx ?
- Qu'est-ce qu'un hébergement mutualisé vs dédié ?

🔷 BONUS – QUESTIONS TRANSVERSALES

- Qu'est-ce que le SEO ? Donne des bonnes pratiques (balises title, meta description, liens internes…).
- Quelle est la différence entre un site statique et un site dynamique ?
- Donne des exemples de frameworks front ou back.

- À quoi sert un gestionnaire de versions comme Git ?
- Quelle est la structure de base d’un projet web moderne ?

🔷 SQL

🔷 1. Notions de base SQL

💬 Syntaxe & sélection

1. Quelle est la syntaxe d'une requête `SELECT` simple ?
2. Comment renommer une colonne dans le résultat d'une requête ?
3. Quelle est la différence entre `SELECT *` et `SELECT colonne1, colonne2` ?
4. Comment supprimer les doublons d'un résultat ?
5. Que fait la clause `LIMIT` ?

💬 Filtres

6. Quelle est la différence entre `WHERE` et `HAVING` ?
7. Donne un exemple d'utilisation de `BETWEEN` et `IN`.
8. Quelle est la différence entre `IS NULL` et `= NULL` ?

🔷 2. Manipulation de données

💬 Insertion / Mise à jour / Suppression

9. Quelle est la syntaxe pour insérer une ligne dans une table ?
10. Quelle est la différence entre `DELETE` et `TRUNCATE` ?
11. Comment mettre à jour plusieurs lignes à la fois ?
12. Peut-on insérer plusieurs lignes d'un coup ? Comment ?

🔷 3. Fonctions et agrégats

13. Donne des exemples de fonctions d'agrégation SQL (au moins 3).
14. Que retourne la fonction `COUNT(*)` ?
15. Quelle est la différence entre `AVG(col)` et `SUM(col)/COUNT(col)` ?
16. Pourquoi utiliser `GROUP BY` ? Donne un exemple.

🔷 4. Clauses avancées

17. Que permet la clause `ORDER BY` ? Donne un exemple avec `DESC`.
18. Que permet la clause `GROUP BY` ?

19. Quelle est la différence entre `WHERE` et `HAVING` dans une requête avec `GROUP BY` ?

🔷 5. Les jointures

💬 Types de jointures

20. Qu'est-ce qu'une jointure ?
21. Quelle est la différence entre `INNER JOIN`, `LEFT JOIN`, et `RIGHT JOIN` ?
22. Qu'est-ce qu'une jointure croisée ?
23. Pourquoi vaut-il mieux utiliser une jointure explicite plutôt qu'implicite (`FROM A, B WHERE...`) ?
24. Donne un exemple de jointure entre 2 tables `utilisateurs` et `commandes`.

🔷 6. Contraintes, clés et intégrité

25. Qu'est-ce qu'une clé primaire ? une clé étrangère ?
26. Peut-on avoir plusieurs clés étrangères dans une table ?
27. Qu'est-ce qu'une contrainte `UNIQUE` ? `NOT NULL` ?
28. Explique les options `ON DELETE CASCADE` et `ON DELETE RESTRICT`.
29. Que fait la contrainte `DEFAULT` ?

🔷 7. Indexation et performances

30. À quoi sert un index ?
31. Comment créer un index sur une colonne ?
32. Quels sont les avantages et les inconvénients d'utiliser des index ?
33. Peut-on avoir plusieurs index sur une même table ? Pourquoi ?

🔷 8. Modélisation / Conception de base de données

34. Qu'est-ce qu'une table de liaison ? Donne un exemple.
35. Quelle est la différence entre relation 1-1, 1-N et N-N ?
36. Donne un exemple de MCD et son équivalent MPD (avec 2 ou 3 entités).
37. Pourquoi normaliser une base de données ?
38. Quels sont les avantages d'un moteur `InnoDB` par rapport à `MyISAM` ?

39. Qu’est-ce qu’un SGBD ? Cite-en 3.
40. Pourquoi utiliser un schéma de base de données (modèle logique) avant de coder les tables ?

🔷 9. Sécurité & bonnes pratiques

41. Qu’est-ce qu’une injection SQL ? Donne un exemple.
42. Comment se protéger contre les injections SQL ?
43. Pourquoi faut-il éviter d’écrire directement les valeurs dans une requête SQL ?
44. Que sont les requêtes préparées ? Pourquoi sont-elles plus sûres ?
45. Quelle est la structure d’une requête préparée avec PDO ?

🔷 JAVASCRIPT

🔷 1. Fondamentaux JavaScript

💬 Variables & types

1. Quelle est la différence entre `var`, `let` et `const` ?
2. Quels sont les types de données primitifs en JavaScript ?
3. Comment tester le type d'une variable ?
4. Quelle est la différence entre == et === ?
5. Qu'est-ce qu'une variable hoistée ?

💬 Structures de contrôle

6. Quelle est la différence entre une boucle `for`, `for...of`, `for...in` et `while` ?
7. Quand utiliser un `switch` plutôt qu'une structure `if/else` ?
8. Que fait la structure `try / catch` ?

💬 Fonctions

9. Quelle est la différence entre une fonction déclarée et une fonction fléchée (=>) ?
10. Qu'est-ce qu'une fonction de rappel (callback) ?
11. C'est quoi une fonction anonyme ?
12. À quoi sert `return` dans une fonction ?
13. Peut-on passer une fonction en paramètre d'une autre ? Dans quel cas ?

🔷 2. Tableaux, objets, méthodes

💬 Tableaux

14. Comment créer un tableau ? Comment y accéder ?
15. Quelle est la différence entre `.map()`, `.forEach()`, `.filter()` et `.reduce()` ?
16. Que renvoie `array.length` ? Comment ajouter ou retirer un élément ?
17. Comment trier un tableau ?

💬 Objets

18. Qu'est-ce qu'un objet en JavaScript ?
19. Comment accéder à une propriété d'un objet ?
20. Quelle est la différence entre un tableau et un objet ?
21. Comment parcourir un objet avec `for...in` ?

🔷 3. DOM et événements

22. Qu'est-ce que le DOM ?

23. Comment sélectionner un élément avec JavaScript ?
24. Quelle est la différence entre `getElementById`, `querySelector` et `getElementsByClassName` ?
25. Comment modifier le contenu HTML d'un élément ?
26. Comment modifier une classe CSS d'un élément en JS ?
27. Comment ajouter un événement `click` sur un bouton ?
28. Qu'est-ce que le `preventDefault()` ?
29. Quelle est la différence entre `addEventListener` et les attributs HTML `onclick`, etc. ?

🔷 4. Asynchrone, fetch, JSON

30. Qu'est-ce qu'une fonction asynchrone ?
31. Quelle est la différence entre `setTimeout`, `setInterval` et `requestAnimationFrame` ?
32. Qu'est-ce que `fetch` ? Donne un exemple simple pour appeler une API.
33. Qu'est-ce que `await` ? Peut-on l'utiliser sans `async` ?
34. Quelle est la différence entre `then()` et `async/await` ?
35. Comment parser une chaîne JSON en objet JS ? Et inversement ?

🔷 5. Stockage local, modules, sécurité

36. À quoi sert `localStorage` ? Quelle est la différence avec `sessionStorage` ?
37. Comment stocker un objet dans le `localStorage` ?
38. Qu'est-ce qu'un module ES6 (`import` / `export`) ?
39. Qu'est-ce qu'une IIFE ? (Immediately Invoked Function Expression)
40. Qu'est-ce que le CORS ? Pourquoi il peut poser problème ?
41. Quelles sont les failles de sécurité possibles côté JS (XSS, etc.) ?
42. Peut-on vraiment protéger des données sensibles avec JS côté client ?

🔷 6. Bonus – Algorithmie & logique

43. Comment inverser une chaîne de caractères ?
44. Comment vérifier si un mot est un palindrome ?
45. Comment calculer le factoriel d'un nombre ?
46. Comment supprimer les doublons d'un tableau ?
47. Comment vérifier si une valeur existe dans un tableau ?
48. Comment fusionner deux tableaux sans doublons ?

🔷 7. Bonus – Notions avancées (si jury curieux)

49. Qu'est-ce que le `this` ? Que vaut-il dans une fonction fléchée ?
50. Qu'est-ce qu'un closure (fermeture) en JavaScript ?
51. Qu'est-ce que l'event bubbling et event delegation ?
52. Qu'est-ce que le destructuring ?
53. Qu'est-ce que l'opérateur `...` (rest/spread) ?

🔷 PHP

🔷 1. Bases du langage PHP

💬 Syntaxe & types

1. Quelle est la différence entre `echo` et `print` ?
2. Quelles sont les règles de base pour nommer une variable en PHP ?
3. Quels sont les types de données principaux en PHP ?
4. Quelle est la différence entre == et === ?
5. À quoi sert `isset()` ? et `empty()` ?

💬 Opérations

6. Comment concaténer deux chaînes en PHP ?
7. Que renvoie l'opérateur `.` ?
8. Quelle est la différence entre `include`, `require`, `include_once`, `require_once` ?
9. Comment définir une constante en PHP ?

🔷 2. Structures de contrôle & fonctions

10. Quelles sont les structures conditionnelles en PHP ?
11. Donne un exemple d'une boucle `for`, `while` et `foreach`.
12. Comment créer une fonction en PHP ?
13. Peut-on définir des valeurs par défaut dans les paramètres d'une fonction ?
14. Qu'est-ce qu'une fonction anonyme ? Donne un exemple.

🔷 3. Tableaux & manipulation

15. Comment créer un tableau associatif ?
16. Quelle est la différence entre `array_push` et `$array[] = valeur` ?
17. Que fait la fonction `array_merge()` ?
18. Quelles fonctions permettent de trier un tableau ?
19. Quelle est la différence entre `count()` et `sizeof()` ?

🔷 4. Formulaires & superglobales

20. Qu'est-ce que la superglobale `$_POST` ? Et `$_GET` ?
21. Quelle est la différence entre méthode `GET` et `POST` ?
22. Comment sécuriser les données d'un formulaire avant traitement ?
23. Pourquoi utiliser `htmlspecialchars()` ou `strip_tags()` ?
24. Comment vérifier si un champ de formulaire est vide ?

🔷 5. Sessions & cookies

25. Comment démarrer une session PHP ?
26. Quelle est la différence entre `$_SESSION` et `$_COOKIE` ?
27. Comment enregistrer des données en session ?
28. Comment détruire une session ?
29. Comment définir un cookie avec une durée de vie personnalisée ?

🔷 6. Programmation orientée objet (POO)

30. Qu'est-ce qu'une classe ? un objet ?
31. Quelle est la différence entre `public`, `private` et `protected` ?
32. À quoi sert un constructeur (`__construct`) ?
33. Qu'est-ce que l'héritage en PHP ?
34. Qu'est-ce qu'une interface ? une classe abstraite ?
35. Quelle est la différence entre une méthode statique et une méthode d'instance ?

🔷 7. Accès à la base de données (MySQL avec PDO)

36. Comment se connecter à une base de données avec PDO ?
37. Quelle est la différence entre requête simple et requête préparée ?
38. Pourquoi faut-il toujours utiliser des requêtes préparées ?
39. Comment récupérer tous les résultats d'une requête ?

40. Que signifie le mode `PDO::FETCH_ASSOC` ?

🔷 8. Sécurité

41. Qu'est-ce qu'une injection SQL ?
42. Comment se protéger contre les injections SQL ?
43. Pourquoi `MD5()` n'est-il plus recommandé pour hacher un mot de passe ?
44. Que fait `password_hash()` ? Et `password_verify()` ?
45. Quelle est la différence entre hachage et chiffrement ?

🔷 9. Bonnes pratiques & outils

46. À quoi sert Composer ?
47. Que contient généralement le fichier `composer.json` ?
48. Que fait l'autoloading en PHP ?
49. Qu'est-ce qu'un namespace ?
50. Qu'est-ce qu'un framework PHP ? Cites-en 2.

🔷 10. Bonus – Scénarios réels

51. Comment valider un formulaire de connexion sécurisé ?
52. Comment faire une redirection en PHP ?
53. Comment envoyer un email en PHP ?
54. Qu'est-ce que le MVC ?
55. Comment structurer un petit projet PHP sans framework ?

SYMFONY

1. Base du framework Symfony

1. Qu'est-ce que Symfony ? Pourquoi l'utilise-t-on ?
2. Quelle est la structure de base d'un projet Symfony ?
3. Quelle est la différence entre un composant Symfony et un bundle ?
4. Comment créer un nouveau projet Symfony ?
5. Que contient le fichier `.env` ?

2. Routing (routes)

6. Qu'est-ce qu'une route dans Symfony ?
7. Où et comment définit-on une route ?
8. Quelle est la différence entre annotation (`#[Route]`), YAML et PHP pour définir les routes ?
9. À quoi sert la commande `php bin/console debug:router` ?
10. Comment passer des paramètres dynamiques dans une route (`/article/{id}`) ?

3. Contrôleurs

11. Qu'est-ce qu'un contrôleur dans Symfony ?
12. Quelle est la signature classique d'une méthode de contrôleur ?
13. Que renvoie un contrôleur ?
14. Comment injecter un service dans un contrôleur ?
15. Comment renvoyer un JSON depuis un contrôleur ?

4. Twig (moteur de templates)

16. Qu'est-ce que Twig ?
17. Comment afficher une variable dans Twig ?
18. Quelle est la différence entre `{{ }}` et `{% %}` dans Twig ?
19. Comment inclure un template dans un autre ?

20. Comment créer une condition `if` et une boucle `for` en Twig ?

🔷 5. Doctrine & base de données

21. Qu'est-ce que Doctrine ?
22. Comment créer une entité dans Symfony ?
23. À quoi sert la commande `make:migration` ?
24. Que fait `php bin/console doctrine:migrations:migrate` ?
25. Comment faire une relation ManyToOne ou ManyToMany ?
26. Quelle est la différence entre `persist()` et `flush()` ?
27. Que fait `find()`, `findBy()`, `findOneBy()` ?
28. Comment injecter le `EntityManagerInterface` dans un service ou un contrôleur ?

🔷 6. Formulaires

29. Comment créer un formulaire avec Symfony ?
30. Qu'est-ce qu'un `FormType` ?
31. Comment lier un formulaire à une entité ?
32. Quelle est la méthode pour gérer la soumission et la validation d'un formulaire ?
33. Quelle est la différence entre `handleRequest()` et `isSubmitted()` ?
34. Comment afficher un formulaire dans Twig ?

🔷 7. Validation & erreurs

35. Comment ajouter une contrainte de validation sur une propriété d'entité ?
36. Cites-en 3 contraintes courantes (`NotBlank`, `Email`, `Length`, etc.)
37. Où place-t-on les messages d'erreurs de validation dans Twig ?
38. Comment afficher un message flash en cas de succès ou d'échec ?

🔷 8. Sécurité & utilisateurs

39. Comment fonctionne le système d'authentification dans Symfony ?
40. Quelle est la commande pour générer un système de login complet (`make:auth`) ?
41. Quelle est la différence entre `IS_AUTHENTICATED_FULLY`, `IS_GRANTED`, et `ROLE_USER` ?
42. Comment protéger une route par rôle ?
43. Que fait le fichier `security.yaml` ?
44. Quelle est la différence entre le hashage `bcrypt` et `argon2i` ?
45. Qu'est-ce que le `UserInterface` ? Pourquoi faut-il l'implémenter ?

🔷 9. Services, injections et évènements

46. Qu'est-ce qu'un service dans Symfony ?
47. Comment créer un service personnalisé ?
48. Qu'est-ce que l'auto-wiring ?
49. Comment injecter un service Symfony dans un contrôleur ou un autre service ?
50. Qu'est-ce qu'un évènement ? À quoi sert un EventSubscriber ou un Listener ?

🔷 10. Console, debug, outils développeurs

51. Que fait la commande `debug:container` ?
52. Que permet `debug:autowiring` ?
53. À quoi sert le Web Profiler ?
54. Que fait `bin/console cache:clear` ?
55. Qu'est-ce que `MakerBundle` et que permet `make:entity`, `make:controller`, etc. ?

🔷 11. API / JSON / AJAX

56. Comment retourner une réponse JSON dans un contrôleur ?
57. Quelle est la différence entre `JsonResponse` et `Response` ?
58. Comment gérer une requête AJAX dans un formulaire Symfony ?

59. À quoi sert l'annotation `@IsGranted` ?
60. Qu'est-ce que le composant HttpClient de Symfony ?

🔷 12. Déploiement et environnement

61. Quelle est la différence entre les environnements `dev`, `prod`, `test` ?
62. À quoi sert le fichier `.env.local` ?
63. Comment configurer une base de données différente pour la production ?
64. Que fait la commande `composer install --no-dev` ?
65. Pourquoi vider le cache en production ?

🔷 13. Bonus – Notions avancées

66. Qu'est-ce qu'un `DataTransformer` ?
67. Qu'est-ce qu'un `Voter` ?
68. Qu'est-ce qu'un `ParamConverter` ?
69. Comment ajouter un champ personnalisé à un formulaire généré automatiquement ?
70. Quelles sont les nouveautés ou changements dans Symfony 6 par rapport à Symfony 5 ? (attributs PHP 8, annotations remplacées…)

🔷 NO SQL & MONGODB

🔷 10 Questions sur NoSQL & MongoDB

1. **Qu'est-ce qu'une base de données NoSQL ?**
   ➤ En quoi diffère-t-elle d'une base de données relationnelle (SQL) ?

2. **Cite 3 types de bases NoSQL et donne un exemple pour chacune.**
   ➤ (Clé-valeur, orientée document, orientée graphes, colonnes...)

3. **Qu'est-ce que MongoDB ?**
   ➤ Quel type de base NoSQL est-ce ? Quelle est sa particularité principale ?

4. **Quelle est la différence entre un document et une collection dans MongoDB ?**
   ➤ (Compare à une ligne et une table SQL.)

5. **À quoi ressemble un document MongoDB ?**
   ➤ Donne un exemple de document (format JSON).

6. **Comment insérer un document dans une collection MongoDB ?**
   ➤ Donne la commande MongoDB équivalente à `INSERT`.

7. **Comment faire une requête pour récupérer tous les documents avec un champ précis (ex : age > 25) ?**
   ➤ Utilise la méthode `find()`.

8. **Qu'est-ce que le `_id` dans MongoDB ?**
   ➤ Peut-on le modifier ? À quoi sert-il ?

9. **Quelles sont les limites d'une base NoSQL comme MongoDB ?**
   ➤ (Transactions, relations, contraintes, etc.)

10. **Dans quel cas choisir MongoDB plutôt qu'un SGBDR comme MySQL ?**
    ➤ (Flexibilité du schéma, performance sur gros volumes non structurés, etc.)

🔷 JARGON

🔷 1. Web & Intégration

Terme	Définition
HTML	Langage de balisage pour structurer le contenu d'une page web.
CSS	Langage de style utilisé pour définir l'apparence d'une page HTML.
Responsive Web Design (RWD)	Conception de sites web qui s'adaptent à toutes tailles d'écran.
Mobile First	Approche de conception qui commence par les petits écrans avant d'adapter vers les grands.
Media Queries	Règles CSS permettant d'adapter le style selon la taille de l'écran.
WebP	Format d'image léger et optimisé pour le web.
Préfixes vendeurs	Préfixes CSS comme `-webkit-`, `-moz-` pour la compatibilité
UX (User Experience)	Expérience utilisateur globale sur un site ou application.
UI (User Interface)	Interface graphique visible par l'utilisateur.
Wireframe	Schéma fonctionnel basique représentant l'organisation d'une page.
Maquette	Représentation graphique finalisée d'une page ou application.

🔷 2. Accessibilité & Conformité

Terme	Définition
Accessibilité numérique	Capacité d'un site à être utilisable par tous, y compris les personnes en situation de handicap.
RGAA	Référentiel Général d'Amélioration de l'Accessibilité (norme française
Alt	Attribut HTML décrivant une image pour les lecteurs d'écran.
Aria-label / ARIA	Attributs HTML pour améliorer l'accessibilité des composants interactifs.
Contrast ratio	Rapport de contraste entre texte et arrière-plan pour assurer la lisibilité.

🔷 3. Développement Front-End

Terme	Définition
DOM (Document Object Model)	Représentation en arbre d'un document HTML manipulable en JS.
Évènement (Event)	Action utilisateur captée en JavaScript (`click`, `submit`,
API Fetch	Interface JS moderne pour effectuer des requêtes HTTP.

JSON	Format de données léger utilisé pour les échanges avec une API.
LocalStorage / SessionStorage	Mécanismes de stockage de données côté navigateur.

🔷 4. Développement Back-End

Terme	Définition
PHP	Langage de script côté serveur pour générer des pages dynamiques.
POO (Programmation Orientée Objet)	Paradigme basé sur des objets contenant données et comportements.
Encapsulation / Héritage / Abstraction	Principes fondamentaux de la POO.
MVC	Modèle de séparation des responsabilités (Modèle-Vue-Contrôleur).
Formulaire	Moyen de collecter des données utilisateur pour les traiter côté serveur.
Session	Mécanisme permettant de conserver des données utilisateur entre les pages.
Cookie	Donnée stockée côté client pour identifier ou suivre un utilisateur.

🔷 5. Base de Données (SQL / NoSQL)

Terme	Définition
SGBD	Système de Gestion de Base de Données (MySQL, PostgreSQL, etc.).
SQL	Langage pour manipuler des bases relationnelles.
Requête préparée	Requête SQL sécurisée contre les injections.
Jointure	Association de données entre plusieurs tables (`INNER JOIN`, etc.).
Clé primaire / étrangère	Identifiants uniques et liens entre les tables.
Index	Accélère les recherches sur les colonnes ciblées.
MCD / MLD / MPD	Étapes de modélisation des données (Merise).
MERISE	Méthode de conception de bases de données.
MongoDB	Base NoSQL orientée documents, alternative aux bases relationnelles.
NoSQL	Bases de données non relationnelles, adaptées à la scalabilité.

🔷 6. API & Architecture

Terme	Définition
API	Interface permettant à des applications de communiquer entre elles.
REST	Style d'architecture pour concevoir des APIs lisibles et standardisées.
CRUD	Opérations de base : Create, Read, Update, Delete.
Endpoint	URL d'accès à une ressource dans une API.
Token / JWT	Jeton d'authentification dans une API sécurisée.

🔷 7. Sécurité Web

Terme	Définition
XSS (Cross Site Scripting)	Insertion de script malveillant dans une page web.
CSRF	Attaque de type "cross-site request forgery".
Injection SQL	Insertion de code SQL dans un champ non sécurisé.
HTTPS / SSL	Chiffrement des échanges entre le navigateur et le serveur.
Hashage / Bcrypt / Argon2	Méthodes de sécurisation des mots de passe.

🔷 8. Outils, versioning et déploiement

Terme	Définition
Git / GitHub	Outil de versionnement de code.
Commit / Push / Pull	Actions Git pour gérer les versions.
.env	Fichier de configuration d'un projet (variables d'environnement).
Composer	Gestionnaire de dépendances PHP.
npm / yarn	Gestionnaires de dépendances front-end JS.
Webpack / Vite	Bundlers pour optimiser le code JS/CSS.
Docker	Conteneurisation d'applications.
CI/CD	Intégration et déploiement continus.

🔷 9. Frameworks et environnements

Terme	Définition

Symfony	Framework PHP MVC robuste et modulaire.
Twig	Moteur de templates de Symfony.
Doctrine	ORM (Object-Relational Mapper) utilisé avec Symfony.
MakerBundle	Bundle Symfony pour générer du code (make:entity, make:form, etc.).
Laravel	Framework PHP moderne orienté développeur.
React / Vue / Angular	Frameworks front-end JS.