GUIDE D'ENTRETIEN TECHNIQUE POUR DÉVELOPPEURS

ANGULAR

Qu'est-ce qu'Angular et quels sont ses avantages?

Réponse: Angular est un framework front-end open-source développé par Google pour créer des applications web dynamiques. Ses avantages incluent une architecture basée sur les composants, l'utilisation de TypeScript, le two-way data binding, une structure modulaire, des outils de test intégrés, et un écosystème riche avec des bibliothèques prêtes à l'emploi.

Quelle est la différence entre AngularJS et Angular?

Réponse: AngularJS (Angular 1.x) est la première version basée sur JavaScript et utilise le modèle MVC. Angular (2+) est une réécriture complète en TypeScript, utilise une architecture basée sur les composants, est plus performant, emploie le binding unidirectionnel par défaut, et intègre un système de modules et d'injection de dépendances amélioré.

Expliquez le concept de composants dans Angular

Réponse: Les composants sont les blocs fondamentaux d'Angular. Chaque composant contrôle une portion de l'écran et contient:

- Un template HTML (la vue)
- Une classe TypeScript (la logique)
- Des métadonnées (décorateur @Component)
- Des styles CSS (optionnels)

Ces composants sont indépendants, réutilisables et peuvent être imbriqués.

Comment fonctionne le data binding dans Angular?

Réponse: Angular propose quatre types de data binding:

- Interpolation (`{{data}}`): affiche une propriété du composant dans le template
- Property binding (`[property]="data"`): liaison de données du composant vers le DOM
- Event binding (`(event)="function()"`): liaison d'événements du DOM vers le composant
- Two-way binding (`[(ngModel)]="data"`): combinaison des deux précédents, synchronisation bidirectionnelle

Qu'est-ce qu'un service dans Angular et comment l'injecter?

Réponse: Un service est une classe qui encapsule la logique métier, les fonctions utilitaires ou les interactions avec les API. Les services favorisent la réutilisation du code et suivent le principe de responsabilité unique.

Pour injecter un service:

1. Créer le service avec le décorateur `@Injectable()`
2. L'enregistrer dans un module (providers) ou avec `providedIn: 'root'`
3. L'injecter via le constructeur du composant

Expliquez les différents types de directives dans Angular

Réponse: Il existe trois types de directives:

- Directives de composants: avec template
- Directives structurelles: modifient la structure du DOM (*ngIf, *ngFor)
- Directives d'attributs: modifient l'apparence ou le comportement (ngClass, ngStyle)

Comment gérer le routage dans une application Angular?

Réponse: Le routage permet de naviguer entre les vues:

1. Importer RouterModule dans le module racine
2. Définir les routes dans un tableau
3. Enregistrer les routes avec RouterModule.forRoot(routes)
4. Ajouter <router-outlet> où les composants doivent s'afficher
5. Utiliser routerLink pour créer des liens ou router.navigate() en TypeScript

Qu'est-ce que RxJS et comment l'utilisez-vous dans Angular?

Réponse: RxJS est une bibliothèque pour la programmation réactive, centrale dans Angular. Elle permet de gérer les données asynchrones via des Observables.

Utilisations courantes:

- Services HTTP pour les requêtes API
- Gestion des événements utilisateur
- Communication entre composants via des services
- Gestion des formulaires réactifs

SPRING BOOT

Qu'est-ce que Spring Boot et quels problèmes résout-il?

Réponse: Spring Boot est un framework Java qui simplifie le développement d'applications Spring en fournissant:

- Une configuration automatique
- Des starters prédéfinis pour intégrer rapidement des fonctionnalités
- Un serveur embarqué (Tomcat, Jetty, Undertow)

- Des outils de production prêts à l'emploi

Il résout les problèmes de configuration complexe, de déploiement et d'intégration de Spring classique.

Expliquez le concept d'Inversion de Contrôle (IoC) et d'Injection de Dépendances

Réponse: IoC est un principe où le contrôle du flux d'exécution est inversé: le framework appelle notre code, pas l'inverse.

L'Injection de Dépendances est un pattern qui implémente l'IoC:

- Les objets reçoivent leurs dépendances au lieu de les créer
- Spring gère le cycle de vie des objets (beans)
- Les dépendances sont injectées via le constructeur, les setters ou les champs

Avantages: code modulaire, testable et découplé.

Comment configurer une application Spring Boot?

Réponse: La configuration se fait principalement via:

- application.properties ou application.yml pour les propriétés
- Annotations comme @Configuration, @EnableAutoConfiguration
- Classe principale avec @SpringBootApplication
- Profils pour différents environnements
- Starters dans le pom.xml (Maven) ou build.gradle (Gradle)

Qu'est-ce qu'un contrôleur REST et comment l'implémenter?

Réponse: Un contrôleur REST expose des endpoints HTTP pour interagir avec l'application.

Implémentation avec les annotations @RestController, @RequestMapping, @GetMapping, @PostMapping, etc. Le contrôleur reçoit les requêtes HTTP, les traite avec l'aide des services, et renvoie des réponses avec ResponseEntity.

Comment gérer la persistance des données avec Spring Boot?

Réponse: Spring Boot utilise Spring Data JPA pour la persistance:

1. Ajouter le starter spring-boot-starter-data-jpa
2. Configurer la source de données dans application.properties
3. Créer des entités avec @Entity
4. Définir des repositories qui étendent JpaRepository
5. Utiliser les méthodes CRUD prédéfinies ou créer des requêtes personnalisées avec @Query

Qu'est-ce que Spring Security et comment l'intégrer?

Réponse: Spring Security est un framework d'authentification et d'autorisation pour Spring.

Intégration:

1. Ajouter le starter spring-boot-starter-security
2. Créer une classe de configuration pour la sécurité
3. Configurer l'authentification (mémoire, JDBC, LDAP, OAuth)
4. Définir les règles d'autorisation
5. Personnaliser selon les besoins (JWT, formulaires personnalisés)

Expliquez le concept de Spring Boot Actuator

Réponse: Spring Boot Actuator fournit des fonctionnalités de surveillance et de gestion pour les applications en production:

- Endpoints pour la santé, les métriques, l'info, les traces
- Intégration avec des systèmes de monitoring
- Audit des événements
- Gestion à distance (JMX)

BASES DE DONNÉES

Quelle est la différence entre MySQL et Oracle?

Réponse:
MySQL:

- Open-source (bien que propriétaire Oracle)
- Léger, facile à installer
- Idéal pour les applications web de petite à moyenne taille
- Moins de fonctionnalités avancées

Oracle:

- Commercial, coûteux
- Robuste pour les grandes entreprises
- Fonctionnalités avancées (partitionnement, RAC, sécurité)
- Meilleures performances pour les charges de travail complexes

Comment optimiser une requête SQL?

Réponse:

- Utiliser des index appropriés
- Éviter SELECT * (sélectionner uniquement les colonnes nécessaires)

- Utiliser des clauses WHERE efficaces
- Optimiser les JOINs (ordre et type)
- Éviter les sous-requêtes si possible
- Analyser le plan d'exécution
- Dénormaliser stratégiquement si nécessaire
- Utiliser des vues matérialisées pour les requêtes fréquentes
- Partitionner les grandes tables

Expliquez les concepts de normalisation de base de données

Réponse: La normalisation est le processus d'organisation des données pour:

- Réduire la redondance
- Améliorer l'intégrité des données
- Faciliter la maintenance

Formes normales principales:

- 1NF: Atomicité des valeurs (pas de valeurs multiples dans une colonne)
- 2NF: 1NF + chaque attribut non-clé dépend totalement de la clé primaire
- 3NF: 2NF + pas de dépendances transitives
- BCNF: 3NF + chaque déterminant est une clé candidate

Comment gérer les transactions dans une base de données?

Réponse: Une transaction est un ensemble d'opérations qui doivent être exécutées ensemble.

Dans SQL:

- BEGIN TRANSACTION;
- Opérations
- COMMIT; (ou ROLLBACK en cas d'erreur)

Dans Spring:

- @Transactional pour décorer des méthodes

Propriétés ACID:

- Atomicité: tout ou rien
- Cohérence: maintient l'intégrité
- Isolation: transactions indépendantes
- Durabilité: résistance aux pannes

MÉTHODOLOGIES

Quelle est la différence entre les méthodes Agile et Waterfall?

Réponse:
Waterfall (cascade):

- Séquentiel et linéaire
- Phases distinctes (exigences, conception, développement, test, livraison)
- Documentation complète à l'avance
- Peu de flexibilité pour les changements

Agile:

- Itératif et incrémental
- Livraisons fréquentes de petits incréments
- Collaboration étroite avec le client
- Adaptabilité aux changements
- Amélioration continue

Comment fonctionne Scrum et quel est le rôle du développeur?

Réponse: Scrum est un framework Agile avec:

- Sprints (itérations de 1-4 semaines)
- Rôles: Product Owner, Scrum Master, Équipe de développement
- Artefacts: Product Backlog, Sprint Backlog, Incrément
- Cérémonies: Planning, Daily Scrum, Review, Rétrospective

Rôle du développeur:

- Auto-organisation pour atteindre les objectifs du sprint
- Participation aux cérémonies Scrum
- Développement selon les standards de qualité
- Estimation des tâches
- Amélioration continue des pratiques de développement

Comment gérez-vous les tests unitaires et l'intégration continue?

Réponse:
Tests unitaires:

- Frameworks comme JUnit, TestNG, Jest, Jasmine
- Principes: isolement, répétabilité, automatisation
- TDD: écrire les tests avant le code
- Mocks/stubs pour les dépendances

Intégration continue:

- Outils: Jenkins, GitLab CI, GitHub Actions, Travis CI
- Pratiques: commits fréquents, builds automatiques
- Tests automatisés à chaque commit
- Feedback rapide sur la qualité du code
- Déploiement continu (CD) comme extension naturelle

NODE.JS

Qu'est-ce que Node.js et pourquoi l'utiliser?

Réponse: Node.js est un environnement d'exécution JavaScript côté serveur, basé sur le moteur V8 de Chrome. Ses avantages incluent:

- Modèle non-bloquant et asynchrone
- Performances élevées pour les opérations I/O
- Utilisation du même langage (JavaScript) côté client et serveur
- Grande bibliothèque de packages (npm)
- Idéal pour les applications temps réel et API

Expliquez le modèle événementiel de Node.js

Réponse: Node.js utilise un modèle basé sur des événements avec une boucle d'événements (event loop) unique. Au lieu de créer de nouveaux threads pour chaque requête, Node.js utilise un modèle non-bloquant où les opérations I/O s'exécutent de manière asynchrone. Le programme continue à s'exécuter pendant que les opérations I/O se déroulent en arrière-plan, et des callbacks sont exécutés une fois ces opérations terminées.

Quelle est la différence entre les méthodes asynchrones et synchrones dans Node.js?

Réponse:

- Méthodes synchrones: bloquent l'exécution jusqu'à ce que l'opération soit terminée
- Méthodes asynchrones: ne bloquent pas l'exécution, utilisent des callbacks ou des promesses pour traiter le résultat une fois l'opération terminée

Les méthodes asynchrones permettent à Node.js de gérer de nombreuses connexions simultanément.

Comment gérer les erreurs dans Node.js?

Réponse: Dans Node.js, les erreurs peuvent être gérées de plusieurs façons:

- Callbacks avec le modèle (err, data)
- Try/catch (pour code synchrone)
- Promises avec .catch()

- Async/await avec try/catch
- Événements error
- Middleware pour les erreurs (dans Express)
- Process.on('uncaughtException')

Expliquez le concept de middleware dans Express.js

Réponse: Les middlewares sont des fonctions qui ont accès à l'objet requête (req), l'objet réponse (res) et la fonction suivante (next) dans le cycle requête-réponse. Ils peuvent:

- Exécuter du code
- Modifier les objets req et res
- Terminer le cycle requête-réponse
- Appeler le middleware suivant

Les middlewares sont utilisés pour l'authentification, la journalisation, la gestion des CORS, le parsing de corps, etc.

LARAVEL

Qu'est-ce que Laravel et quels sont ses avantages?

Réponse: Laravel est un framework PHP open-source pour le développement web. Ses avantages incluent:

- Syntaxe élégante et expressive
- Architecture MVC
- ORM puissant (Eloquent)
- Système de migration de base de données
- Système de template Blade
- Artisan CLI pour automatiser les tâches
- Écosystème riche (packages, outils)
- Documentation complète

Expliquez le cycle de vie d'une requête dans Laravel

Réponse: Le cycle de vie d'une requête Laravel:

1. La requête entre par le point d'entrée (public/index.php)
2. La requête est envoyée au routeur
3. Les middlewares globaux sont exécutés
4. Les middlewares de route sont exécutés
5. La requête est transmise au contrôleur approprié

6. Le contrôleur interagit avec les modèles
7. La vue est rendue
8. La réponse est renvoyée au client

Qu'est-ce qu'Eloquent ORM et comment l'utiliser?

Réponse: Eloquent est l'ORM (Object-Relational Mapping) de Laravel qui permet de travailler avec la base de données de manière orientée objet. Chaque table de la base de données a un modèle correspondant.

Utilisation:

- Création de modèles qui étendent la classe Model
- Définition des relations entre modèles
- Utilisation des méthodes pour les opérations CRUD
- Construction de requêtes élaborées avec query builder

Comment fonctionne le système de migration dans Laravel?

Réponse: Les migrations sont comme un contrôle de version pour la base de données, permettant de créer et modifier le schéma:

- Création de migrations via Artisan CLI
- Définition du schéma dans les méthodes up() et down()
- Exécution des migrations avec la commande migrate
- Annulation avec la commande rollback
- Suivi de l'état des migrations dans la table migrations

Expliquez les différents types de relations dans Eloquent

Réponse: Eloquent supporte plusieurs types de relations:

- One-to-One: hasOne et belongsTo
- One-to-Many: hasMany et belongsTo
- Many-to-Many: belongsToMany
- Has-Many-Through: hasManyThrough
- Polymorphic: morphTo, morphMany, morphToMany
- Many-to-Many Polymorphic: morphToMany et morphedByMany

Comment fonctionne l'authentification dans Laravel?

Réponse: Laravel fournit un système d'authentification complet:

- Système prêt à l'emploi avec artisan make
- Gestion des utilisateurs, inscription et connexion

- Protection des routes avec middleware auth
- Multiple authentification (guards)
- Authentification API avec tokens ou JWT
- Intégration des fournisseurs OAuth

REACT

Qu'est-ce que React et quels sont ses avantages?

Réponse: React est une bibliothèque JavaScript pour construire des interfaces utilisateur. Ses avantages incluent:

- DOM virtuel pour des performances optimales
- Architecture basée sur les composants
- Flux de données unidirectionnel
- JSX pour une syntaxe intuitive
- Grande communauté et écosystème
- Compatible avec le rendu côté serveur
- Facilité d'intégration avec d'autres bibliothèques

Expliquez le concept de composants dans React

Réponse: Les composants sont les éléments de base de React. Ils encapsulent la logique et l'UI. Il existe deux types:

- Composants fonctionnels: fonctions JavaScript qui acceptent des props et retournent des éléments React
- Composants de classe: classes ES6 qui étendent React.Component

Les composants peuvent être composés, réutilisés et testés individuellement.

Qu'est-ce que le state et les props dans React?

Réponse:

- Props (propriétés): données passées d'un composant parent à un composant enfant. Elles sont en lecture seule.
- State: données gérées à l'intérieur d'un composant. Il peut être modifié avec setState() dans les composants de classe ou avec les hooks (useState) dans les composants fonctionnels.

Expliquez le cycle de vie d'un composant React

Réponse: Pour les composants de classe, le cycle de vie comprend:

- Montage: constructor, getDerivedStateFromProps, render, componentDidMount

- Mise à jour: getDerivedStateFromProps, shouldComponentUpdate, render, getSnapshotBeforeUpdate, componentDidUpdate
- Démontage: componentWillUnmount

Pour les composants fonctionnels, les hooks remplacent le cycle de vie (useEffect).

Qu'est-ce que les hooks dans React et comment les utiliser?

Réponse: Les hooks sont des fonctions qui permettent d'utiliser l'état et d'autres fonctionnalités de React dans les composants fonctionnels. Les hooks principaux sont:

- useState: gérer l'état local
- useEffect: exécuter des effets de bord (equivalent à componentDidMount, componentDidUpdate, componentWillUnmount)
- useContext: accéder au contexte
- useReducer: gérer l'état complexe
- useRef: créer une référence mutable
- useMemo et useCallback: optimiser les performances

Comment gérer la navigation dans React?

Réponse: React Router est la bibliothèque la plus utilisée pour la navigation:

- BrowserRouter: wrapper principal
- Route: définit un chemin et le composant à afficher
- Link: crée des liens de navigation
- NavLink: comme Link mais avec des styles actifs
- Switch: rend la première Route qui correspond
- Redirect: redirection
- useHistory, useLocation, useParams: hooks pour accéder aux fonctionnalités du routeur

Expliquez Redux et son fonctionnement

Réponse: Redux est une bibliothèque de gestion d'état pour JavaScript. Son fonctionnement:

- Store: objet JavaScript qui contient l'état de l'application
- Actions: objets qui décrivent les changements d'état
- Reducers: fonctions pures qui spécifient comment l'état change en réponse aux actions
- Dispatch: méthode pour envoyer des actions au store

Le flux de données est unidirectionnel: vue → action → reducer → store → vue.

JAVA

Quelles sont les principales caractéristiques de Java?

Réponse: Java est un langage de programmation orienté objet avec plusieurs caractéristiques clés:

- Portable (Write Once, Run Anywhere)
- Orienté objet
- Robuste et sécurisé
- Architecture neutre
- Multithreading intégré
- Haute performance
- Distribué
- Typage statique
- Garbage collection automatique

Expliquez les principes de la programmation orientée objet en Java

Réponse: Java implémente les principes fondamentaux de la POO:

- Encapsulation: masquer les détails d'implémentation via le contrôle d'accès
- Héritage: mécanisme permettant à une classe d'hériter des propriétés d'une autre
- Polymorphisme: capacité d'un objet à prendre plusieurs formes
- Abstraction: simplification d'objets complexes en se concentrant sur les comportements essentiels

Quelle est la différence entre une classe abstraite et une interface en Java?

Réponse:
Classe abstraite:

- Peut avoir des méthodes abstraites et non-abstraites
- Peut avoir des variables d'instance
- Une classe ne peut hériter que d'une seule classe abstraite
- Peut avoir des constructeurs
- Peut avoir des modificateurs d'accès pour les méthodes

Interface:

- Avant Java 8, ne pouvait avoir que des méthodes abstraites
- Depuis Java 8, peut avoir des méthodes par défaut et statiques
- Toutes les variables sont public, static et final
- Une classe peut implémenter plusieurs interfaces
- Pas de constructeurs
- Toutes les méthodes sont implicitement public

Comment fonctionne la gestion des exceptions en Java?

Réponse: Java utilise un mécanisme de gestion des exceptions basé sur:

- Blocs try/catch/finally
- throw pour lever une exception
- throws pour déclarer qu'une méthode peut lever une exception
- Hiérarchie des exceptions avec Exception comme classe mère
- Exceptions vérifiées (checked) qui doivent être gérées
- Exceptions non vérifiées (unchecked) qui n'ont pas besoin d'être déclarées

Expliquez le concept de collections en Java

Réponse: Le framework Collections fournit des interfaces et des classes pour stocker et manipuler des groupes d'objets:

- List: collection ordonnée qui peut contenir des doublons (ArrayList, LinkedList)
- Set: collection qui ne peut pas contenir de doublons (HashSet, TreeSet)
- Queue: collection pour stocker des éléments à traiter (LinkedList, PriorityQueue)
- Map: collection de paires clé-valeur (HashMap, TreeMap)

Ces structures de données offrent différentes caractéristiques en termes de performance, utilisation mémoire et fonctionnalités.

Qu'est-ce que la programmation concurrentielle en Java?

Réponse: Java fournit des outils pour la programmation concurrentielle:

- Threads: légères unités d'exécution
- synchronized: mécanisme pour contrôler l'accès aux sections critiques
- wait/notify: pour la communication entre threads
- java.util.concurrent: framework pour la concurrence
  - Executor et ThreadPool
  - Locks (ReentrantLock, ReadWriteLock)
  - Collections concurrentes (ConcurrentHashMap)
  - Atomic variables
  - CompletableFuture pour la programmation asynchrone

Comment fonctionne le garbage collector en Java?

Réponse: Le garbage collector libère automatiquement la mémoire des objets qui ne sont plus référencés:

- Identifie les objets qui ne sont plus accessibles

- Récupère la mémoire occupée par ces objets
- Plusieurs stratégies: Serial GC, Parallel GC, CMS, G1, ZGC
- Fonctionne dans un thread séparé
- Peut être influencé par des paramètres JVM
- Finalization comme mécanisme de nettoyage avant la collecte

MYSQL

Qu'est-ce que MySQL et quels sont ses principaux avantages?

Réponse: MySQL est un système de gestion de base de données relationnelle open-source. Ses avantages incluent sa facilité d'utilisation, sa stabilité, sa rapidité, sa compatibilité multiplateforme, et son faible coût (gratuit en version Community).

Différence entre MyISAM et InnoDB?

Réponse:

- MyISAM : plus ancien, pas de support de transactions, verrouillage au niveau table, plus rapide pour les lectures
- InnoDB : support ACID, transactions, clés étrangères, verrouillage au niveau ligne, meilleur pour écriture intensive

Qu'est-ce qu'une clé primaire et une clé étrangère?

Réponse:

- Clé primaire : identifie de façon unique chaque enregistrement dans une table
- Clé étrangère : établit une relation entre deux tables en référençant la clé primaire d'une autre table

Expliquez les types de jointures dans MySQL

Réponse:

- INNER JOIN : retourne les enregistrements ayant des correspondances dans les deux tables
- LEFT JOIN : retourne tous les enregistrements de la table de gauche et les correspondances de la table de droite
- RIGHT JOIN : retourne tous les enregistrements de la table de droite et les correspondances de la table de gauche
- FULL JOIN : retourne tous les enregistrements des deux tables (via UNION)

Comment optimiser les performances d'une base MySQL?

Réponse:

- Indexer correctement les colonnes fréquemment recherchées

- Éviter SELECT * et sélectionner uniquement les colonnes nécessaires
- Utiliser EXPLAIN pour analyser les requêtes
- Optimiser la configuration du serveur (buffer, cache)
- Partitionner les grandes tables
- Normaliser correctement les données

Qu'est-ce qu'un trigger et quand l'utiliser?

Réponse: Un trigger est un code SQL exécuté automatiquement en réponse à certains événements sur une table (INSERT, UPDATE, DELETE). Utilisez-les pour maintenir l'intégrité des données, l'audit, ou la validation automatique.

Comment sécuriser une base de données MySQL?

Réponse:

- Utiliser des mots de passe forts
- Limiter les privilèges utilisateurs (principe du moindre privilège)
- Désactiver les comptes non utilisés
- Mettre à jour régulièrement MySQL
- Implémenter des pare-feu
- Utiliser SSL pour le chiffrement des connexions
- Valider toutes les entrées pour prévenir les injections SQL

Différence entre DELETE et TRUNCATE?

Réponse:

- DELETE : supprime des lignes spécifiques, peut utiliser WHERE, journalise chaque suppression, peut être annulé (rollback)
- TRUNCATE : supprime toutes les données, plus rapide, réinitialise AUTO_INCREMENT, ne peut pas être annulé facilement

Comment gérer la haute disponibilité avec MySQL?

Réponse:

- Réplication maître-esclave
- Réplication multi-maître
- Clustering avec MySQL Cluster ou Galera
- Utilisation de solutions comme ProxySQL ou MySQL Router

Expliquez les transactions dans MySQL

Réponse: Les transactions permettent de regrouper plusieurs opérations comme une seule unité atomique. Elles sont importantes pour maintenir l'intégrité des données selon les principes ACID (Atomicité, Cohérence, Isolation, Durabilité).

Comment implémenter des procédures stockées et quels sont leurs avantages?

Réponse: Les procédures stockées sont des ensembles d'instructions SQL stockées dans la base de données. Leurs avantages sont: sécurité accrue, performance améliorée, réutilisation du code, et réduction du trafic réseau.

CONSEILS POUR L'ENTRETIEN

Préparation

- Recherchez l'entreprise et le poste avant l'entretien
- Révisez les concepts fondamentaux des technologies demandées
- Préparez des exemples concrets de projets réalisés
- Préparez des questions pertinentes à poser

Pendant l'entretien

- Soyez honnête sur votre niveau de compétence
- Montrez votre capacité à apprendre rapidement
- Expliquez votre raisonnement lors des questions techniques
- Démontrez votre passion pour le développement
- Écoutez attentivement et demandez des clarifications si nécessaire

Après l'entretien

- Envoyez un email de remerciement
- Faites un suivi si vous n'avez pas de réponse après une semaine
- Analysez ce qui s'est bien passé et ce qui pourrait être amélioré
- Continuez à vous former sur les technologies discutées

Ce guide a été conçu pour vous aider à préparer un entretien technique dans le domaine du développement web et mobile. Les questions et réponses sont adaptées à un niveau junior ou intermédiaire.