PLAN

Plan de cours sur GraphQL

1. Introduction à GraphQL

1.1. Qu'est-ce que GraphQL ?

- Historique et contexte (développement par Facebook en 2012, open-source en 2015).
- Différences entre GraphQL et REST.

1.2. Avantages de GraphQL

- Requêtes précises.
- Évolution sans versionnage.
- Documentation intégrée.

1.3. Quand utiliser GraphQL ?

- Cas d'usage et scénarios pratiques.
- Limites et inconvénients.

---

2. Les concepts fondamentaux de GraphQL

2.1. Le schéma GraphQL (Schema)

- Rôle du schéma dans une API GraphQL.
- Définition des types (Query, Mutation, et Subscription).
- Exemple pratique : schéma simple.

2.2. Requêtes (Queries)

- Structure d'une requête.
- Champs et sous-champs.
- Alias et renommage de champs.

2.3. Mutations

- Différence entre requêtes et mutations.
- Exemple pratique : création et mise à jour de données.

2.4. Résolveurs (Resolvers)

- Définition et rôle des resolvers.
- Exemple de résolution d'une requête.

---

3. Fonctionnalités avancées

3.1. Fragments

- Réutilisation de blocs de requêtes.
- Exemple avec un fragment.

3.2. Directives

- Utilisation de `@include` et `@skip`.
- Exemple pratique avec conditions.

3.3. Pagination et filtrage

- Approches courantes : offset/limit et Relay (connections).
- Exemple avec `pageInfo` et `edges`.

3.4. Subscriptions (temps réel)

- Définition et cas d'utilisation.
- Exemple : notifications en temps réel.

---

4. Architecture et mise en œuvre

4.1. Outils pour GraphQL

- Serveurs populaires : Apollo Server, GraphQL Yoga, Express-GraphQL.
- Clients : Apollo Client, Relay, ou Fetch simple.

4.2. Exemple pratique avec Apollo Server

- Installation et configuration.
- Création d'un schéma et des resolvers.
- Test avec GraphQL Playground.

4.3. Interaction avec une base de données

- Lier GraphQL à une base de données (PostgreSQL, MongoDB).

- Exemple avec Prisma pour les bases relationnelles.

---

5. Sécurité et optimisation

5.1. Authentification

- Protéger les requêtes avec des tokens (JWT).
- Gestion des autorisations (permissions par rôle).

5.2. Gestion des erreurs

- Types d'erreurs courantes (exemple : "user not found").
- Structure des réponses d'erreurs GraphQL.

5.3. Optimisation des performances

- DataLoader pour réduire les appels redondants.
- Gestion des limites de profondeur des requêtes (query depth).

---

6. Fonctionnalités avancées et écosystème

6.1. Fédérations de services

- Introduction à Apollo Federation.
- Composition d'APIs multiples.

6.2. Extensions et outils

- GraphQL Voyager (visualisation du schéma).
- GraphQL Inspector (vérification de compatibilité).

6.3. Monitoring et débogage

- Utilisation d'Apollo Studio ou d'outils similaires.

---

7. Exercices pratiques

7.1. Créer une API GraphQL simple

- Définir un schéma pour une application de gestion de tâches.
- Implémenter les resolvers pour lire, créer et modifier des tâches.

7.2. Ajouter des fonctionnalités avancées

- Pagination pour les listes de tâches.

- Notifications en temps réel pour les nouvelles tâches.

7.3. Sécuriser l'API

- Ajouter une authentification basée sur JWT.

---

8. Conclusion et ressources supplémentaires

8.1. Récapitulatif des points clés

- Forces et limites de GraphQL.
- Comparaison avec d'autres solutions.

8.2. Ressources pour aller plus loin

- Documentation officielle GraphQL : https://graphql.org
- Tutoriels et cours en ligne (Apollo, Udemy, etc.).
- Projets open-source pour approfondir.

Durée suggérée

- **Cours complet :** 8 à 10 heures (théorie + pratique).
- **Introduction rapide :** 2 heures.