En effet nous avons introduit la notion de graphql et souligne quelque uns de ces avantages, le graphql est une api largement performant permettant de pallier a un certain problématique non résolue par les api rest tel que le over fecthing, de plus la structure sous forme de graph nous permet de faire des requêtes très complexes , simplement. Le graphql a l'aide du schéma et des revolvers permettent de définir et d'implémenter des 'méthodes', ces méthodes peuvent être des méthodes de sélections ou de modifications. Dans le cas des méthodes de sélections, on parlera des requêtes de type **query** et les méthodes de type modification sont des **mutations.** Et bien pour lancer une application graphql nous avons besoin d'un serveur, il peut s'agir d'un serveur **Express** ou d'un serveur **Apollo,** ces deux serveurs sont largement utilisés mais Apollo, néanmoins garde des spécificités comme la rupture des cycles qui sont difficilement réalisable avec express. Leurs interfaces web du point de vue design semblent différentes mais fondamentales, elles sont identiques.

Pour le test nous nous rendons sur l'adresse suivante : **api.dev.eneotransportation.com** l'interface de cet api a été construite avec Apollo, et comme dans l'interface d'express, nous avons toutes les informations qui sont présentes. Donc on peut cliquer sur le bouton DOCS pour voir toutes les méthodes présentes dans notre api et recherche si nécessaire dans la barre de rechercher présent le schéma d'une query définie ainsi dans cette **documentation introspective** ou d'une mutation ou SCHEMA pour voir le schéma définit dans le cadre de cet api.

Donc on peut déjà envisage de faire un query car nous n'avons pas d'autorisation pour faire une mutation. Mais pour pouvoir faire cette query nous avons besoin d'un token d'authentification :

{"Authorization": "Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJsb2dpbklkIjoxNjUsInVzZXJuYW1lIjoiY2hyaXN0b3BoZWZpbmFsIiwicm9sZXMiOiJDTElFTlQiLCJpYXQiOjE2NDE5MTM4OTV9.rCCYGqwvJZ9u2eLk600aLldHt87s7PoEilRU-m9XInw"}

Une fois ce token obtenu, nous nous rendons dans l'espace header de l'interface Apollo, a l'heure actuelle situe à l'extrême bas gauche et puis nous saisissons ce token.

Apres cette étape nous pouvons réaliser une query exemple :

```
query{
 getAllVehicleDetailsByUniqueDeviceId(uniqueDeviceId:"it_861937063269691"){
 vehicleNumber
 vehicleModel
…
 }}
```

Ici nous pouvons ajouter plus d'informations si nécessaire, donc cette requête peut être affinée en vue d'obtenir un résultat plus conséquent.