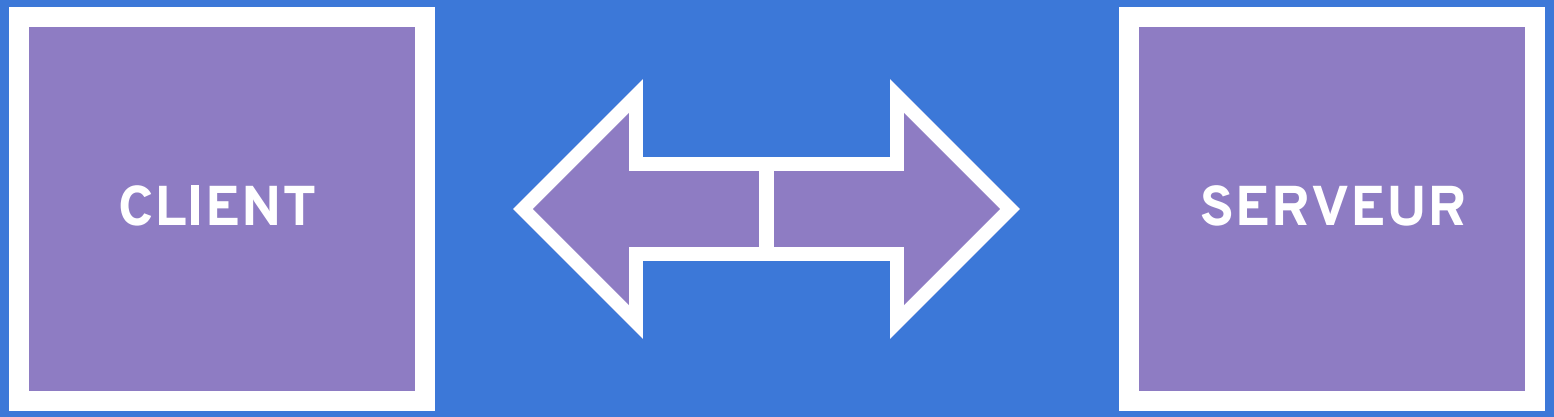
Cloud et Système web

II. COMMUNICATION CLIENT/SERVEUR
DANS LE DOMAINE DU WEB

Stéphane WOUTERS

École WIS 2019/2020

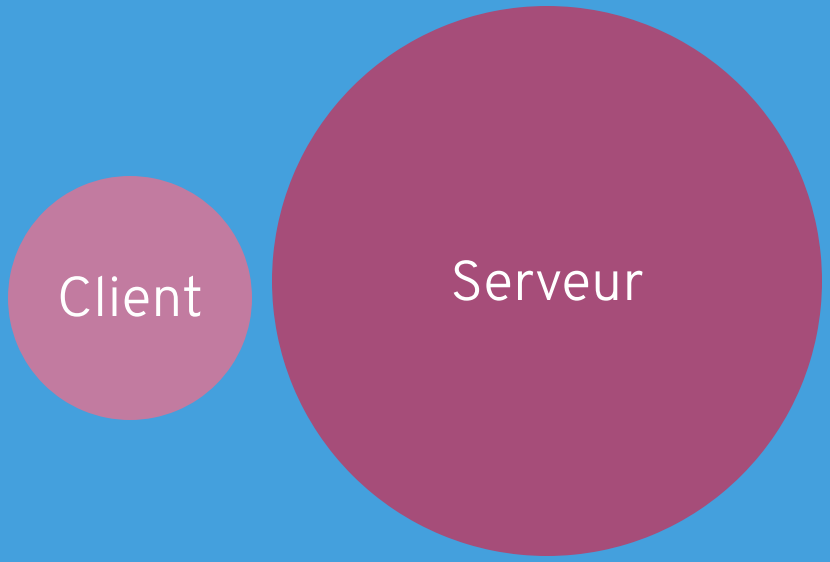
ARCHITECTURE CLIENT SERVEUR



Logiciel, application web/mobile, site internet, objet connecté...

FTP, API, Base de données...

RESPONSABILITÉ CLIENT/SERVEUR



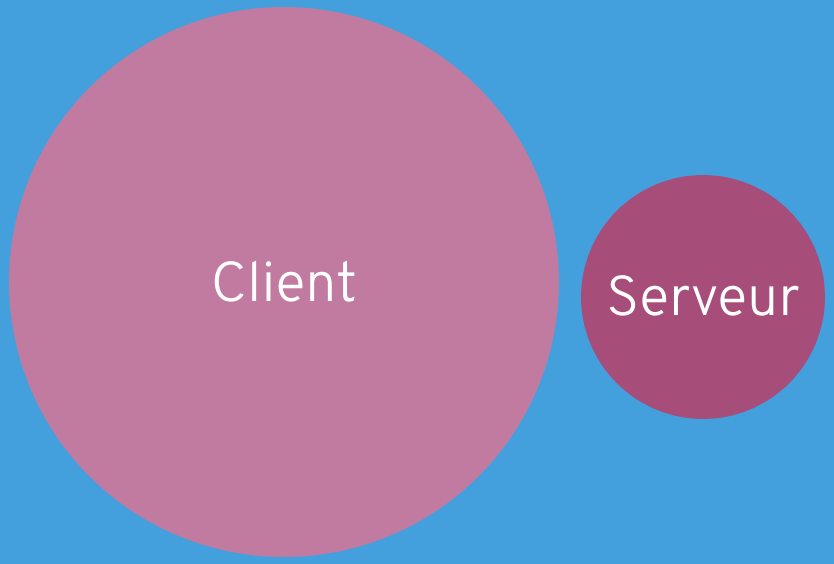
STRATÉGIE CLIENT LÉGER

Le client ne s'occupe que de l'affichage.

Exemples :

Un Site web en PHP sans javaScript.

Un accès à un poste teamviewer



STRATÉGIE CLIENT LOURD

La logique métier est côté client.

Exemples :

Une application web javascript connectée à une API.

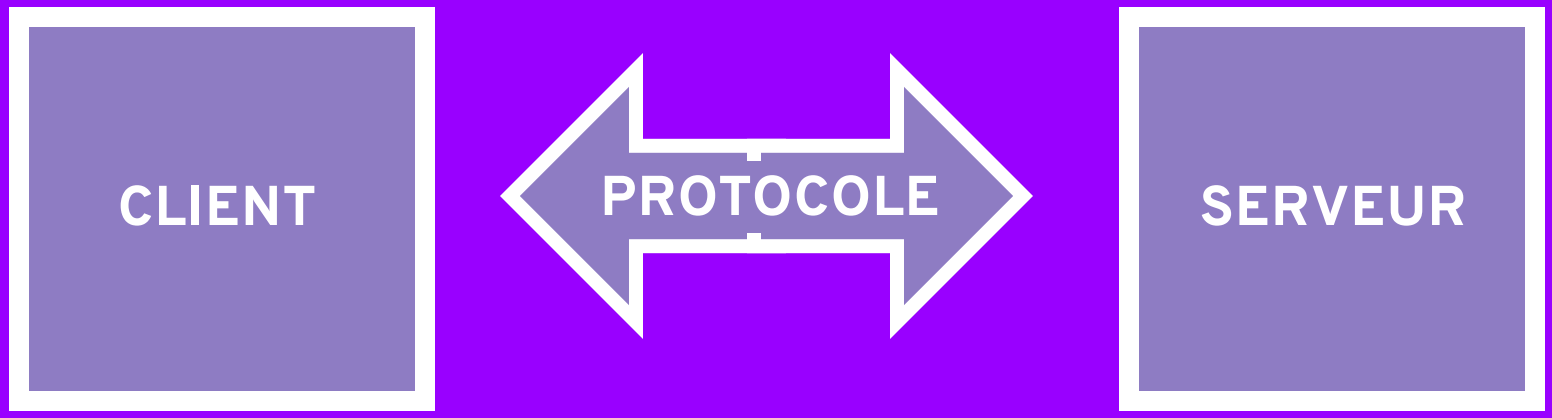
Un logiciel connecté à une base de donnée.

QUIZZ CLIENT/SERVEUR

www.wooclap.com/ENIULO

COMMUNICATION CLIENT / SERVEUR

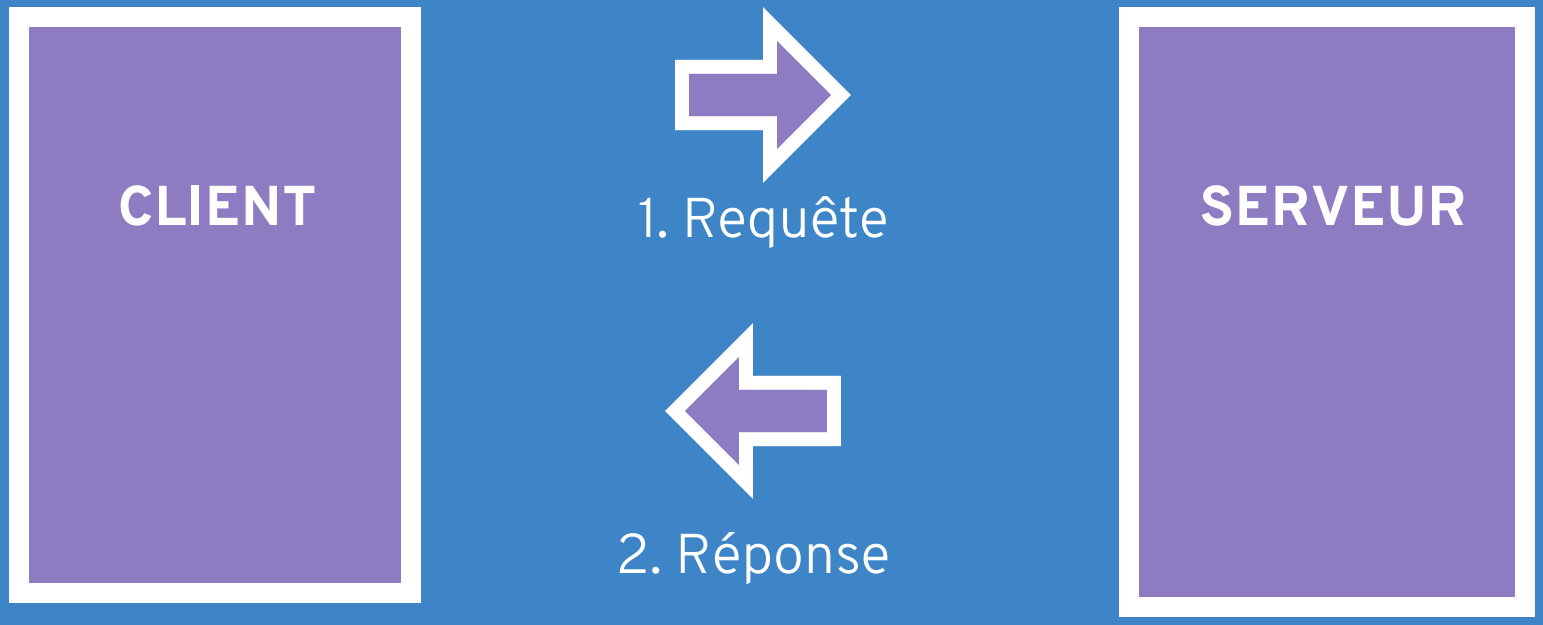
COMMUNICATION CLIENT/SERVEUR



Exemples de protocoles utilisés sur le web :

- **HTTP/HTTPS**
- **WebSockets**
- FTP, SFTP, SSH
- POP/IMAP/SMTP pour les emails

PROTOCOLE HTTP



HTTP fonctionne par transactions : Le client envoie une requête au serveur, puis le serveur envoie sa réponse

✗ Le serveur ne peut jamais envoyer une requête au client

PROTOCOLE HTTP

Exemples d'utilisations :

- Servir une page web (html) et ses assets (css/javascript/images/videos...)
- Servir une API
- Télécharger des fichiers

COMPOSITION D'UNE REQUETE HTTP

- **Une requête est composée de :**
  - Un chemin. Exemple /api/weather/montpellier
  - Une méthode. Exemple : GET, POST, PUT, DELETE, OPTIONS
    - Seuls POST et PUT peuvent avoir un contenu
  - Des paramètres URL (?page=1)
  - Un contenu (formulaire, json, fichier…)
  - Des headers (Authentification, type de donnée souhaité…)
- **Une réponse est composée de :**
  - Des headers
  - Un contenu

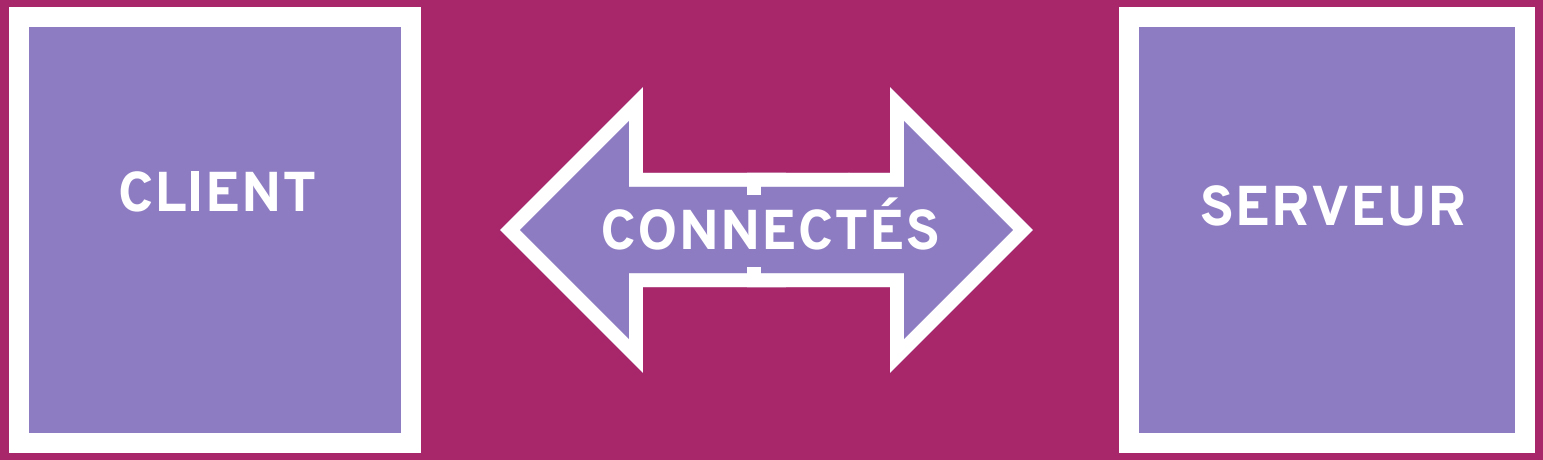
HTTPS

“ *HyperText Transfer Protocol Secure*

Version sécurisée de HTTP

Permet de chiffrer la connexion grâce à un certificat SSL

PROTOCOLE WEBSOCKET



Le client et le serveur échangent via une connexion persistante. Le client et le serveur peuvent s'envoyer un message à tout moment.

PROTOCOLE WEBSOCKET

UTILISÉ POUR LES APPLICATIONS INTERACTIVES

Exemples :

- Chats
- Outils collaboratifs (google doc...)
- Jeux vidéos multijoueurs
- Notifications Push

SUPPORTS DE DONNÉES

Exemples de supports de données utilisés

TEXTE

```
UNE PIZZA n°1 A 8 EUROS
```

JSON

```json
{
  "orders": [{
    "pizza": {
      "id": "1",
      "price": 3
    }
  }]
}
```

XML

```xml
<order>
  <pizza id="1">
    <price>3</price>
  </pizza>
</order>
```

CSV

```
product;id;price
pizza;1;3
```

LES API WEB

API HTTP

UNE API HTTP PERMET D'ACCÈDER À DES SERVICES EN LIGNE.

- **API à usage public** : Utiliser des données externes dans son application (météo, vols disponibles, données géographiques...)
- **API publique à usage sécurisé :** Récupérer sous forme JSON son profil facebook, github...
- **API à usage privé** : Développer une API sur mesure pour faire le lien entre son application et sa base de données

UTILISER UNE API

Pour les API plus complexes, il est utile de structurer la donnée. Les normes les plus connues sont :

- **RESTful** (Actuel)
- **SOAP** (Plus ancien)
- **GraphQL** (Plus récent)

Les api peuvent aussi ne pas suivre de protocole existant

UTILISER UNE API

POUR UTILISER UNE API, SA DOCUMENTATION TECHNIQUE EST INDISPENSABLE

On trouvera dans une documentation :

- Le protocole de sécurité (Clé API, Basic auth, oAuth, JWT...)
- Les routes (endpoints), leurs méthodes (GET, POST...)
- Le format et schéma des données attendues

TESTER SES REQUÊTES HTTP

POURQUOI PAS SIMPLEMENT UN NAVIGATEUR ?

- Fonctionne uniquement pour les requêtes GET
- Difficile d'envoyer des headers personnalisés (authentification...)

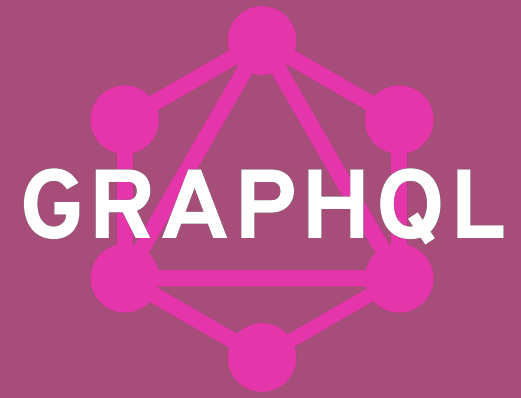
Un outil complet : Postman

Démo

TP1

ÉCRIRE LA COLLECTION POSTMAN D'UNE API

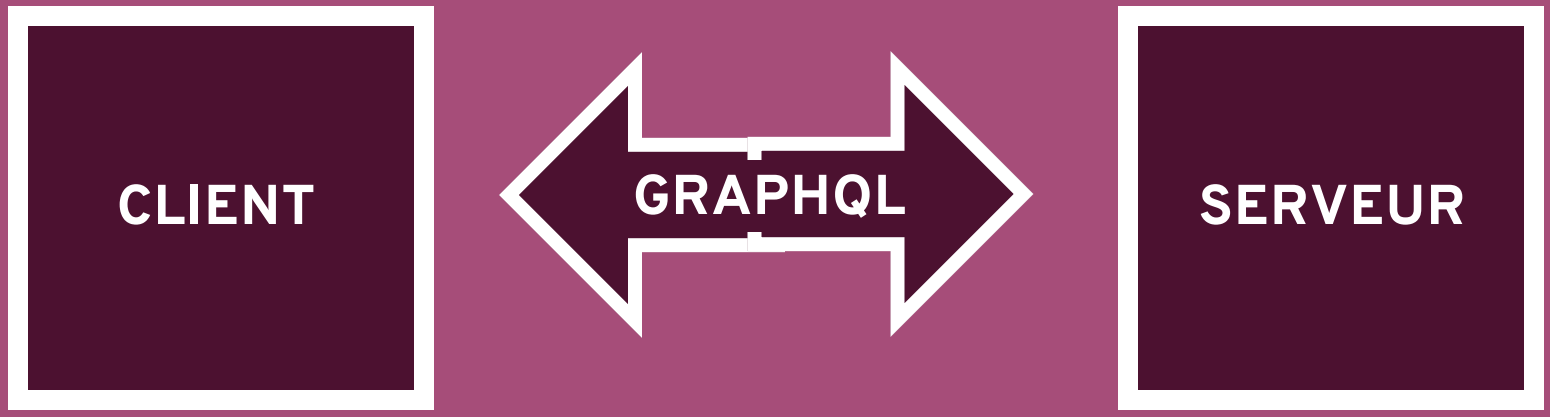
https://bit.ly/2Fnj32k



Langage de requête pour les API

GRAPHQL

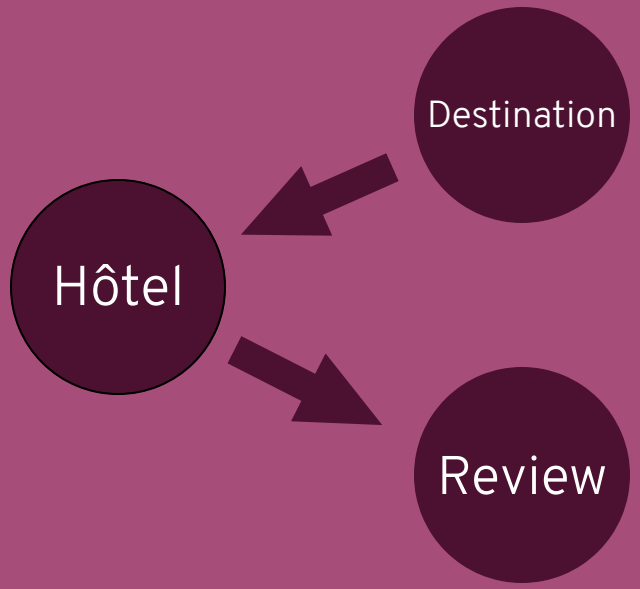
GraphQL est un langage de requête pour les API



GRAPHQL

Intêrets

```
hotel {
    name: string
    description: string
    rooms: number
    photos: Picture
    destinations: Destination[]
}
```



Typé

Structuré

GRAPHQL

Différence avec les API classiques

CLASSIQUE

Une route par action :

- GET **/hotels**
- POST **/hotels**
- GET **/reviews**
- POST **/reviews**
- DELETE **/reviews/45**

GRAPHQL

Un seul endpoint :

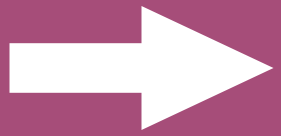
- exemple : POST **api.monsite.com/api**
- Sur lequel on envoie nos requêtes

GRAPHQL

C'est l'utilisateur qui décide des données qu'il recevra

```
{
  user {
    name
  }
}
```

Requête

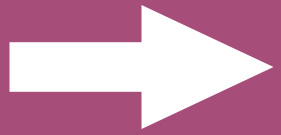


```
{
  "user": {
    "name": "John Doe"
  }
}
```

Réponse

```
{
  user {
    name
    email
    account {
        login
        last_connexion
    }
  }
}
```

Requête

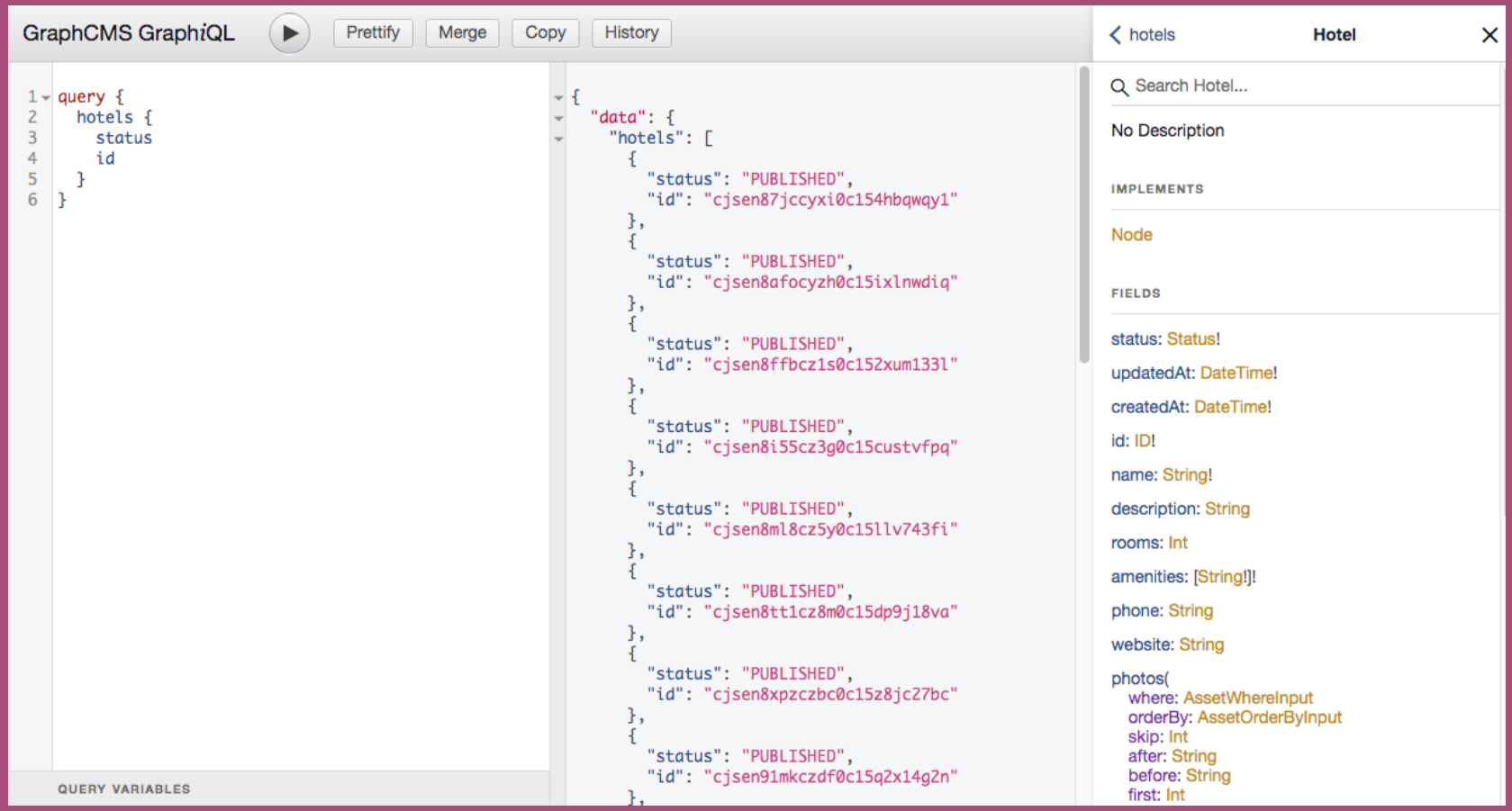


```
{
  "user": {
    "name": "John Doe"
    "email": "john.doe@gmail.com"
    "account" {
        "login": "john"
        "last_connexion": "27/12/2019"
    }
  }
}
```

Réponse

GRAPHQL

Documentation intégrée (= schéma)



Requête Réponse Schéma

TP2

I. Pré requis : Génerer une API en SAAS avec GraphCMS
https://bit.ly/2FjxB32

STRUCTURE D'UNE REQUÊTE GRAPHQL

Type Fields Arguments

TYPES DE REQUÊTES GRAPHQL

QUERY

Permet de récupérer des données

```
query {
  user
}
```

MUTATION

Permet de modifier des données

```
mutation {
  updateReview(data: {
    content: "Hello"
  }) {
    id
  }
}
```

SUITE TP2

Query & Mutations GraphQL (II. et III.)

https://bit.ly/2FjxB32