Formation Back

Comment collecter des données

Retrouvez tous les documents sur: url.viarezo.fr/back

Front vs Back

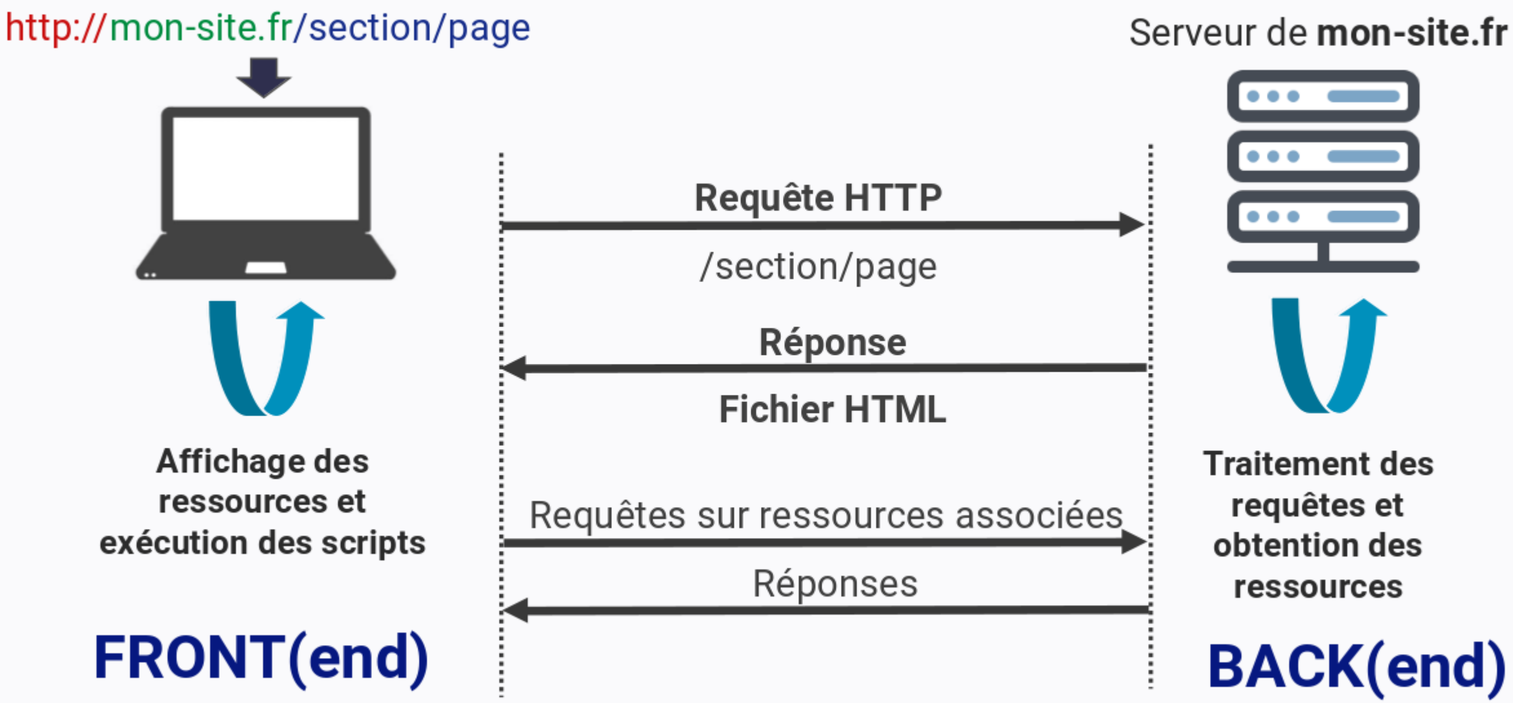
Front

- Affichage graphique
- Interprété par le navigateur
- HTML/CSS/JS

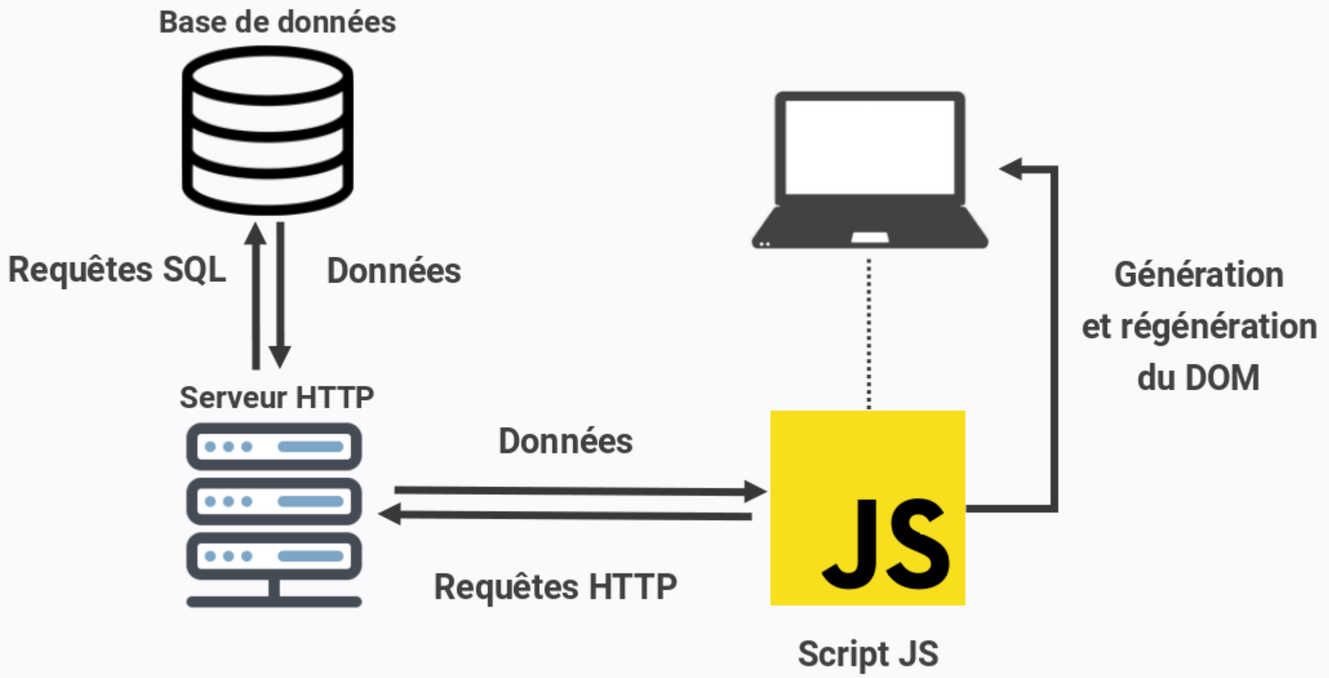
Back

- Base de données, authentification, ...
- Exécuté sur un serveur
- Python, JS, Ruby, ...

Front vs Back



Front vs Back




node


dj


FastAPI


RAILS

FastAPI

framework python permettant de créer un serveur web

FastAPI

API: interface de communication entre deux programmes

Installation de FastAPI

```
pip3 install fastapi[all]
# ou
pip3 install fastapi uvicorn
# ou
py -m install fastapi[all]
uvicorn --version
```

Installation de FastAPI

```python
# main.py
from fastapi import FastAPI

app = FastAPI()
```

Installation de FastAPI

```
uvicorn main:app --reload
```

Les différentes requêtes HTTP

CRUD

Les 4 opérations de base pour les données:

Create

Read

Update

Delete

Read

La requête GET

Requête pour récupérer des données ou des pages web.
C'est la requête exécutée par un navigateur pour afficher une page.

La requête GET

```python
# main.py
import FastAPI from fastapi
app = FastAPI()

@app.get("/")
def response():
  return "Hello world"

@app.get("/users")
def get_users():
  return ["Guillaume","Romain","Hervé"]
```

La requête GET

```python
# main.py
import FastAPI from fastapi
app = FastAPI()

@app.get("/")
def response():
  return "Hello world"

@app.get("/users")
def get_users():
  return ["Guillaume","Romain","Hervé"]
```

La requête GET

Les paramètres

```python
# main.py
import FastAPI from fastapi
app = FastAPI()

# Path parameters
@app.get("/user/{user_id}")
def get_user(user_id):
  return user_id

# Query parameters
@app.get("/users")
def get_users(limit: int):
  return ["Guillaume", "Romain", "Hervé"][:limit]
```

La requête GET

Les paramètres

```python
# main.py
import FastAPI from fastapi
app = FastAPI()

# Path parameters
@app.get("/user/{user_id}")
def get_user(user_id):
  return user_id

# Query parameters
@app.get("/users")
def get_users(limit: int):
  return ["Guillaume", "Romain", "Hervé"][:limit]
```

Le JSON

```
{
  "tasks":[
  {
    "titre": "Sortir le chien",
    "created": "2021-10-06"
  },
  {
    "titre": "Réviser la CIP",
    "created": "2021-10-01"
  }]
}
```

Le JSON

```python
# main.py

@app.get("/tasks")
def get_tasks():
  return [{"titre":"Sortir le chien", "created": "2021-10-06"},
          {"titre":"Rejoindre ViaRézo", "created": "2021-10-01"}]
```

Create

La requête POST

Elle permet de créer une nouvelle donnée.

La requête POST

```python
# main.py

@app.post("/route/")
def fonction():
  # fonction
```

La requête POST

Les paramètres, la suite

```python
# main.py
from pydantic import BaseModel

class User(BaseModel):
  name: str
  age: int

# Body parameter
@app.post("/route/")
def create_user(user: User):
  return user
```

La requête POST

Les paramètres, la suite

```python
# main.py
from pydantic import BaseModel

class User(BaseModel):
  name: str
  age: int

# Body parameter
@app.post("/route/")
def create_user(user: User):
  return user
```

La requête POST

Les paramètres, la suite

```python
# main.py
from pydantic import BaseModel

class User(BaseModel):
  name: str
  age: int

# Body parameter
@app.post("/route/")
def create_user(user: User):
  return user
```

Update et Delete

Les requête PUT et DELETE

Les requête PUT et DELETE

```python
# main.py

@app.put("/route/")
def update():
  # fonction

@app.delete("/route/")
def delete():
  # fonction
```

Les requête PUT et DELETE

```python
# main.py

@app.put("/route/")
def update():
  # fonction

@app.delete("/route/")
def delete():
  # fonction
```

Le TP

Créer un back avec:

- Une route GET renvoyant la liste des tâches
- Une route POST permettant de rajouter une tâche
- Une route permettant de modifier une tâche

Bon courage 😉

On continue ?

Comment intégrer ces données à ma page ?


JS

La fonction fetch

La fonction fetch: GET

```
fetch("http://localhost:8000")
  .then(response => {response.json().then(
    value => console.log(value)
  )})
```

La fonction fetch: POST

```
fetch("http://localhost:8000",{
  method: "POST",
  headers: {
    "Content-Type": "application/json"
  },
  body: JSON.stringify(data)
})
```

Pour aller plus loin, venez nous voir au local 💙