

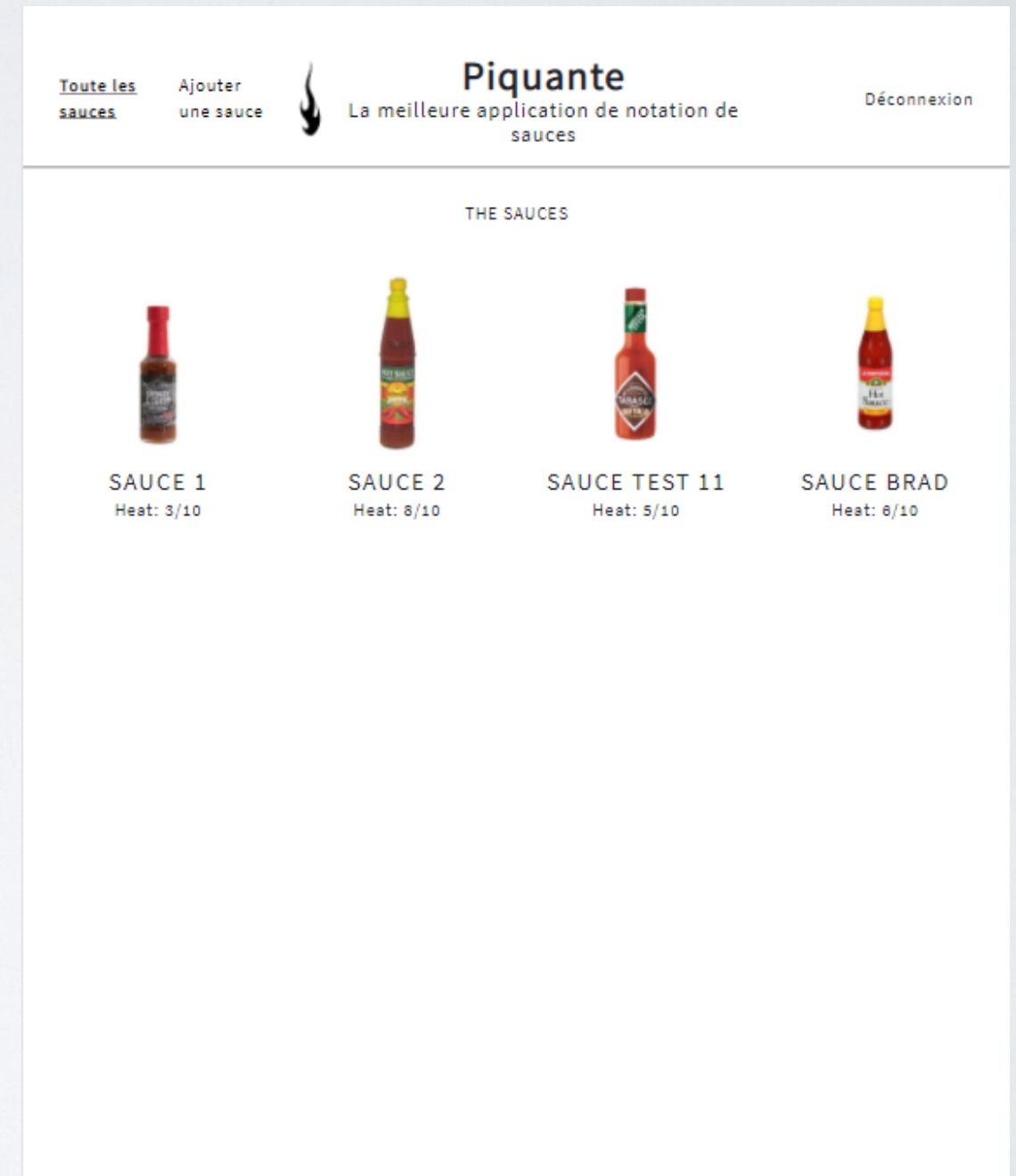
PROJET 6

So Pekocko

OBJECTIFS

- **L'utilisateur doit pouvoir:**

- S'inscrire
- Se Connecter
- Créer des sauces
- Supprimer **ses** sauces
- Modifier **ses** sauces
- Consulter les sauces du site
- Liker ou Disliker les sauces du site



OBJECTIFS

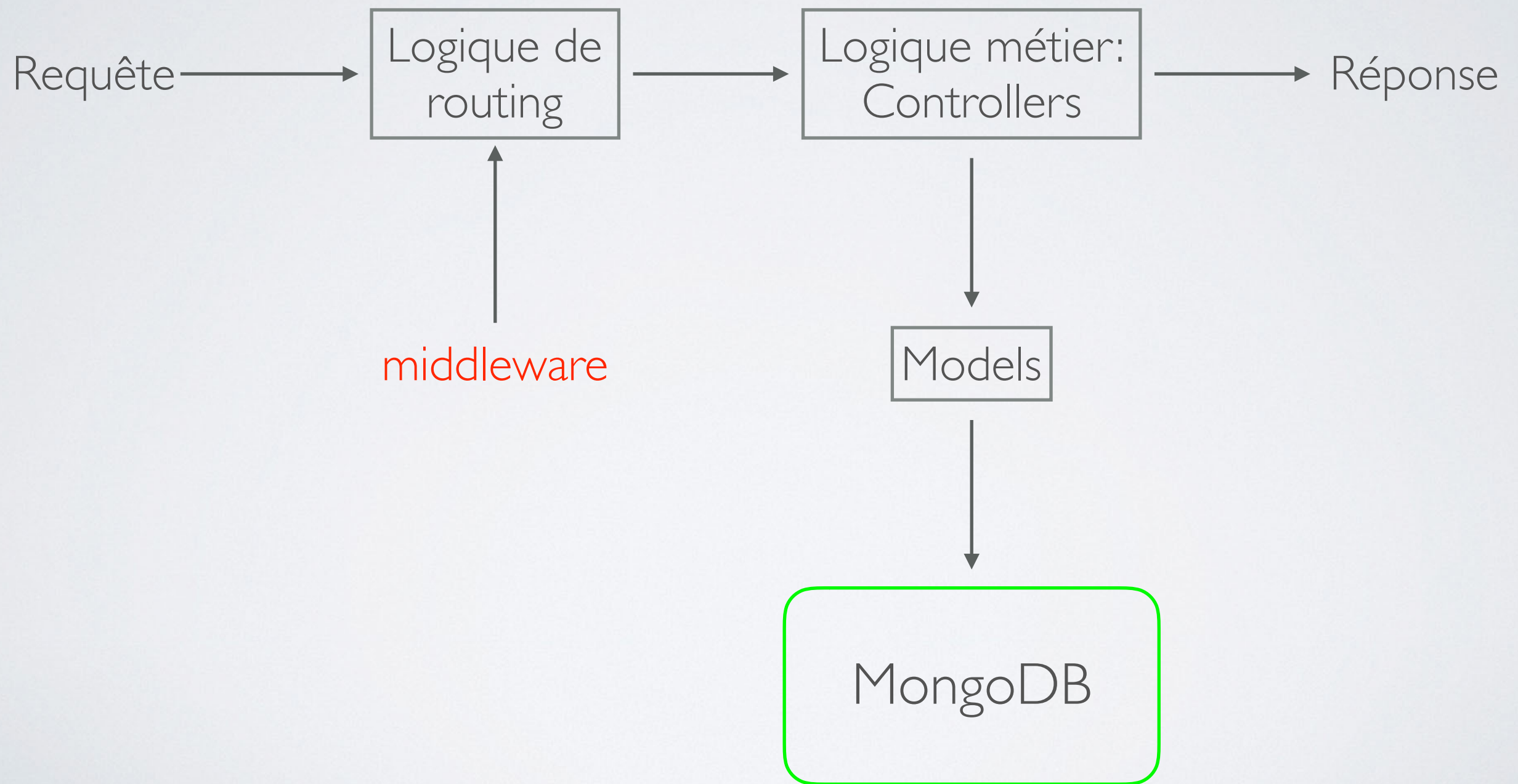
- **Sécurité:**
 - Authentification renforcée
 - Mot de passe sécurisé
 - Unicité des adresses mails
 - Utilisation d'un système de token
 - Masquage des données
 - Anticiper les attaques



TECHNOLOGIES UTILISÉES

- Nodejs (installation de package, serveur)
- Express (framework)
- MongoDB (base de données)

STRUCTURE DU CODE



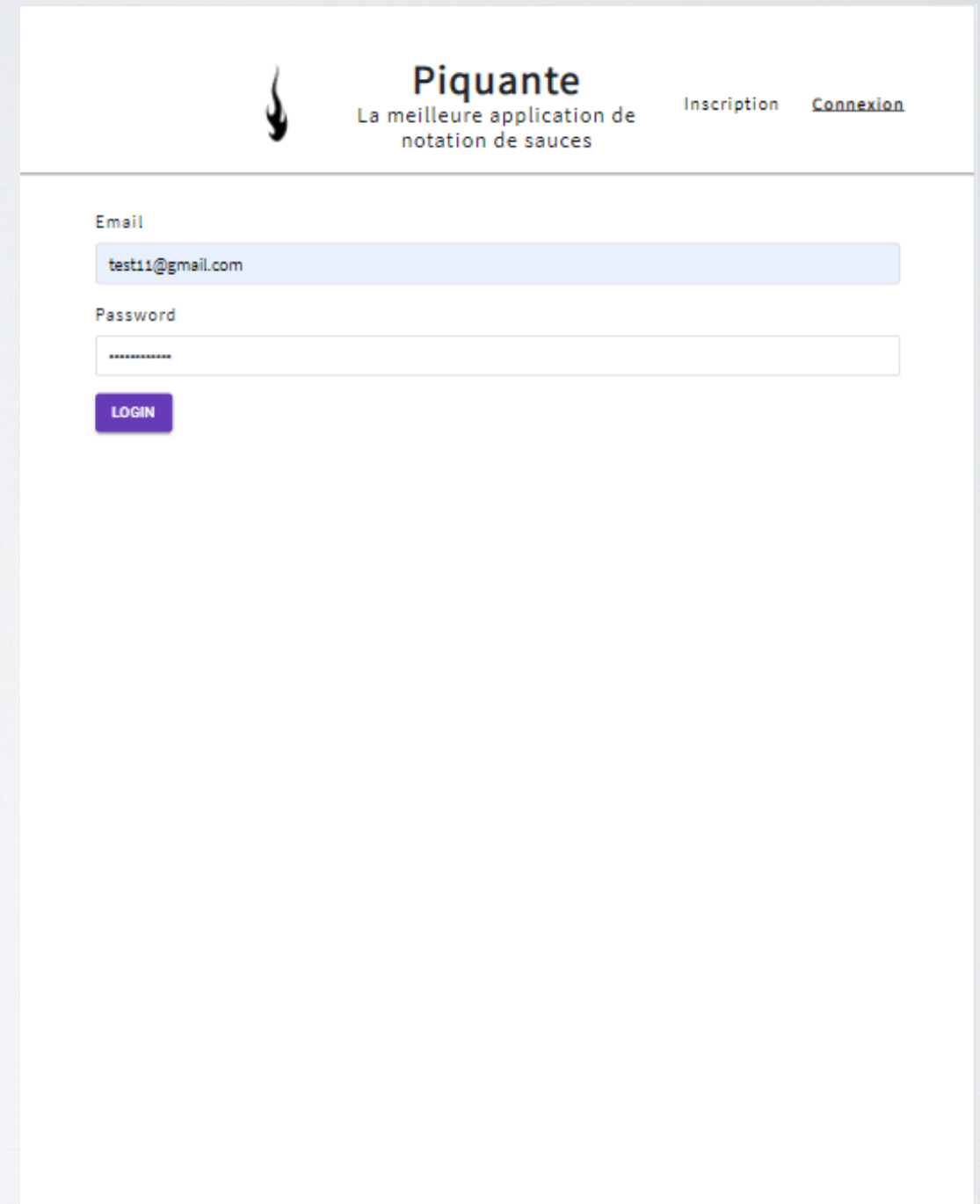
INSCRIPTION / CONNEXION

- **INSCRIPTION:**

- Adresse mail correcte
- Unicité de l'adresse mail (plug-in `mongoose-unique-validator`)
- Mot de passe soumis un regex
- Mot de passe crypté (bcrypt)

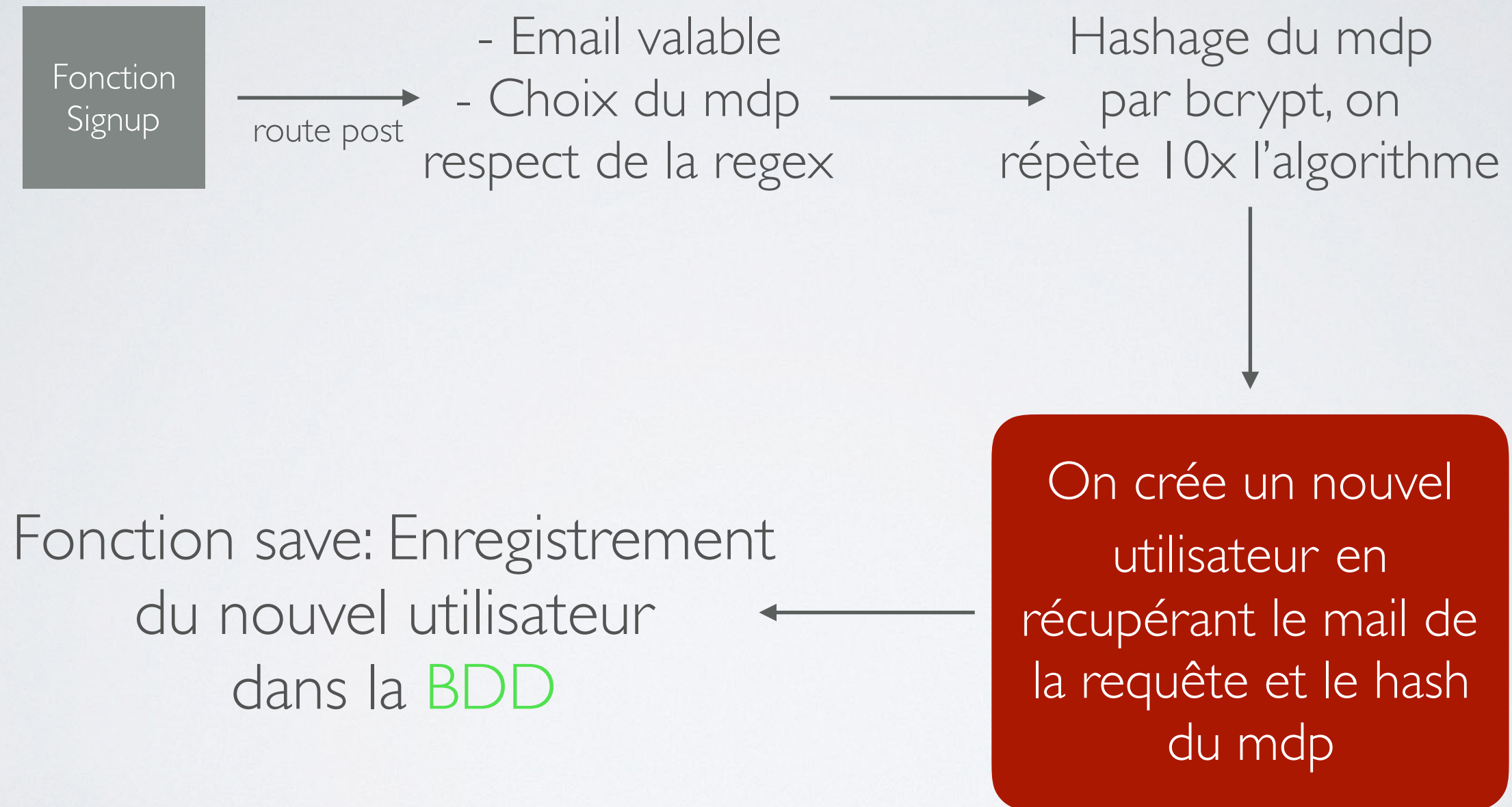
- **CONNEXION:**

- Correspondance entre l'adresse mail et la BDD
- Comparaison entre le mdp avec le hash enregistré

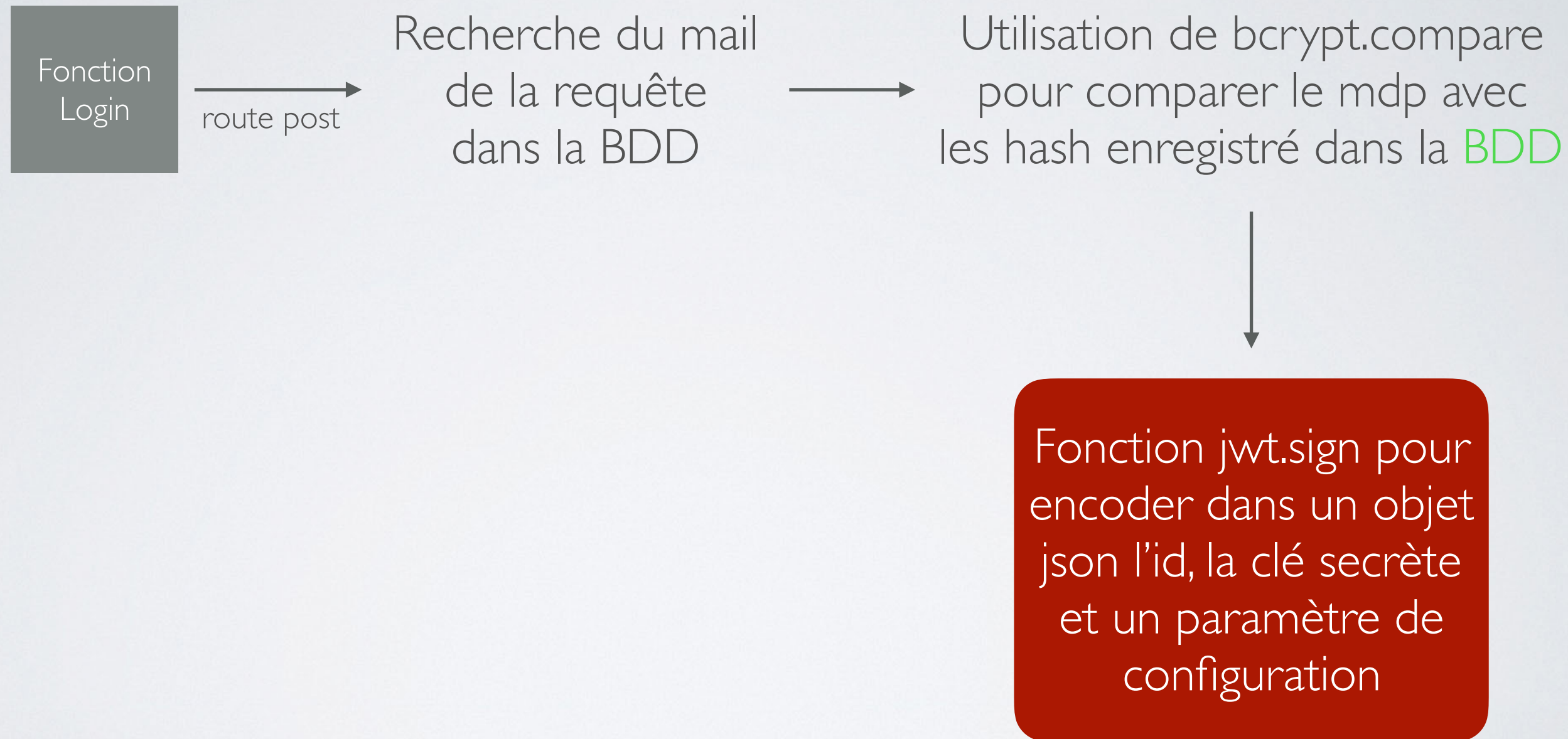


The screenshot shows the login interface for 'Piquante', described as 'La meilleure application de notation de sauces'. The header includes a flame logo, the app name, and links for 'Inscription' and 'Connexion'. The login form contains an 'Email' field with 'test11@gmail.com', a 'Password' field with masked characters, and a purple 'LOGIN' button.

INSCRIPTION

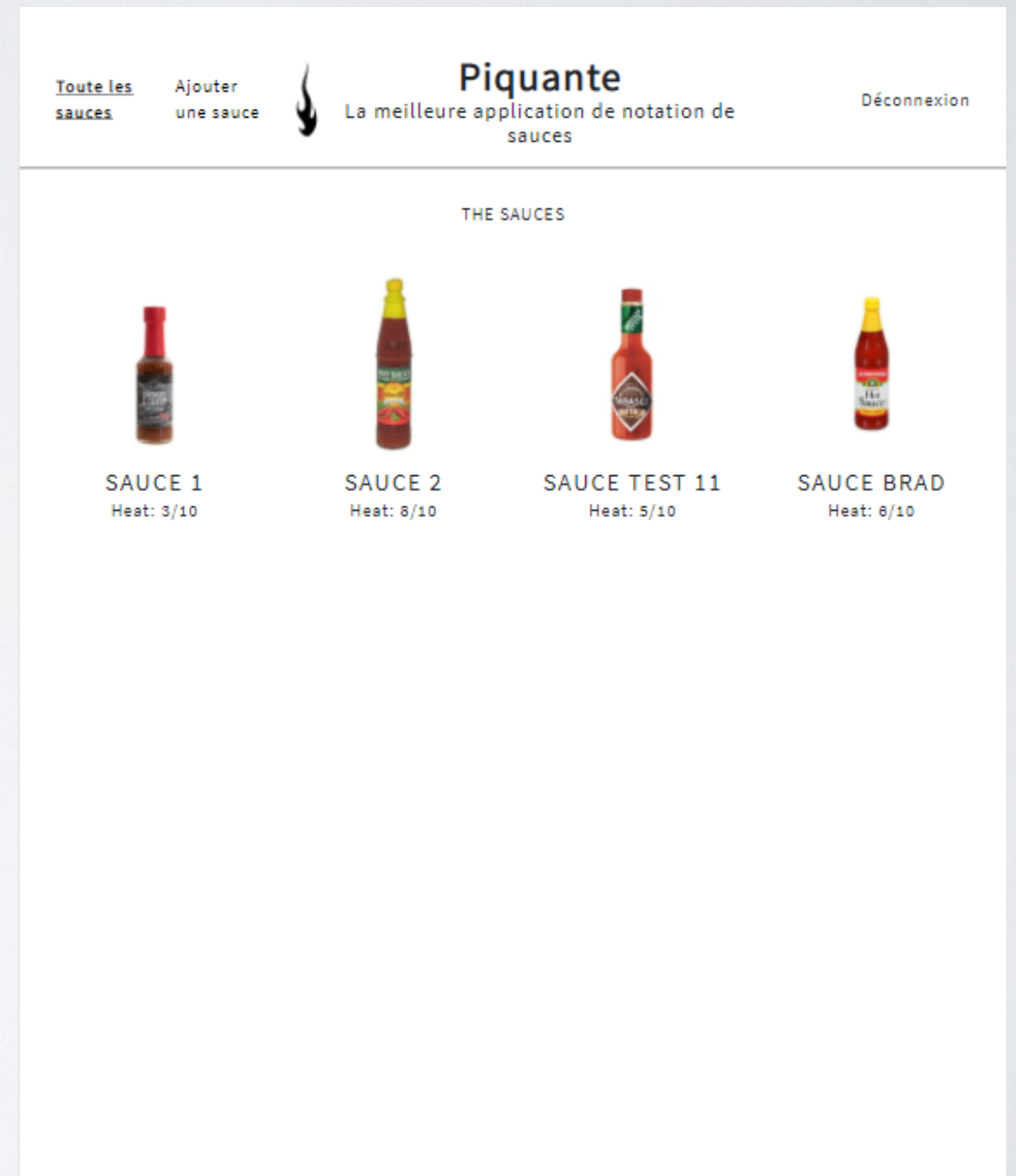


CONNEXION

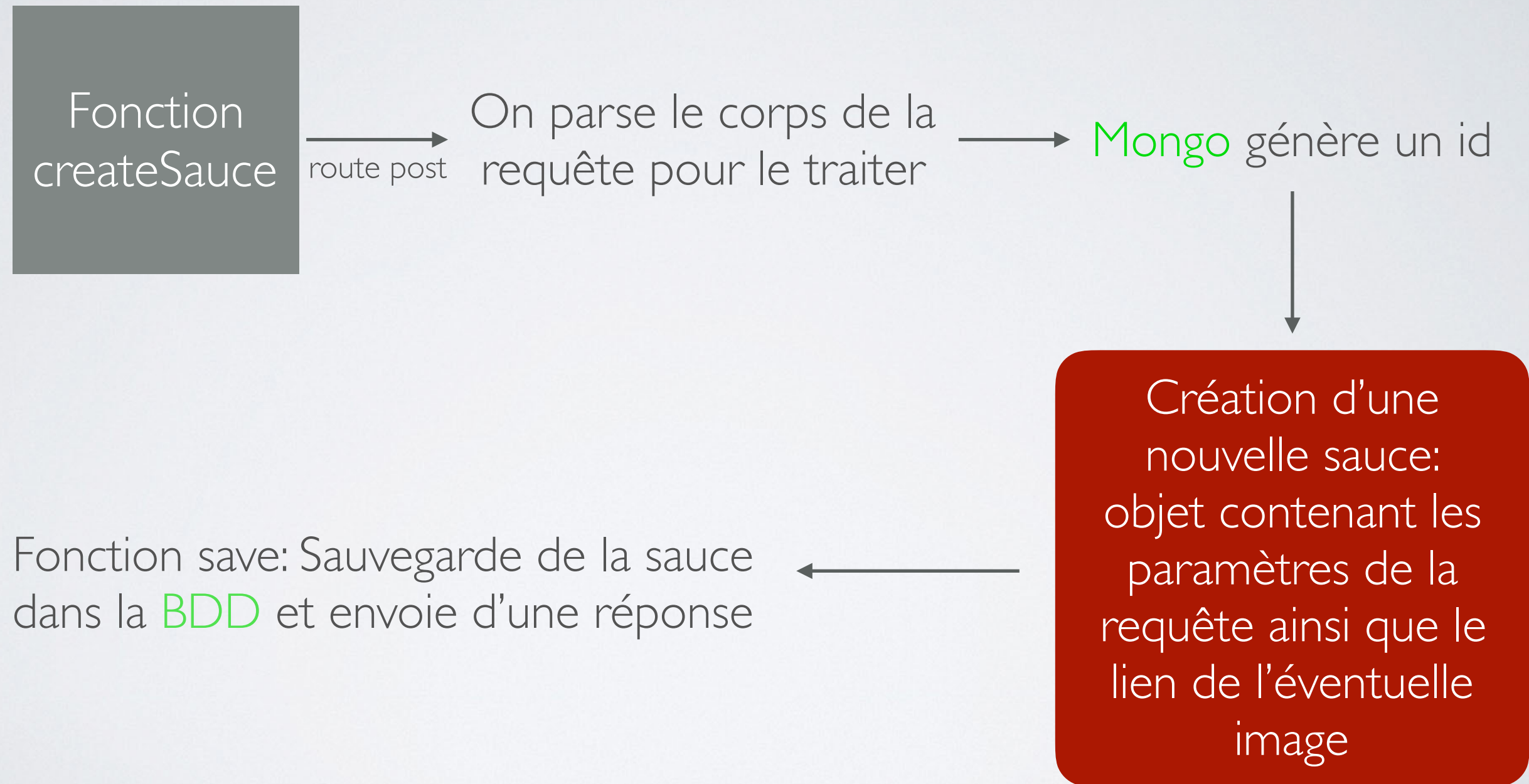


INTERACTIONS

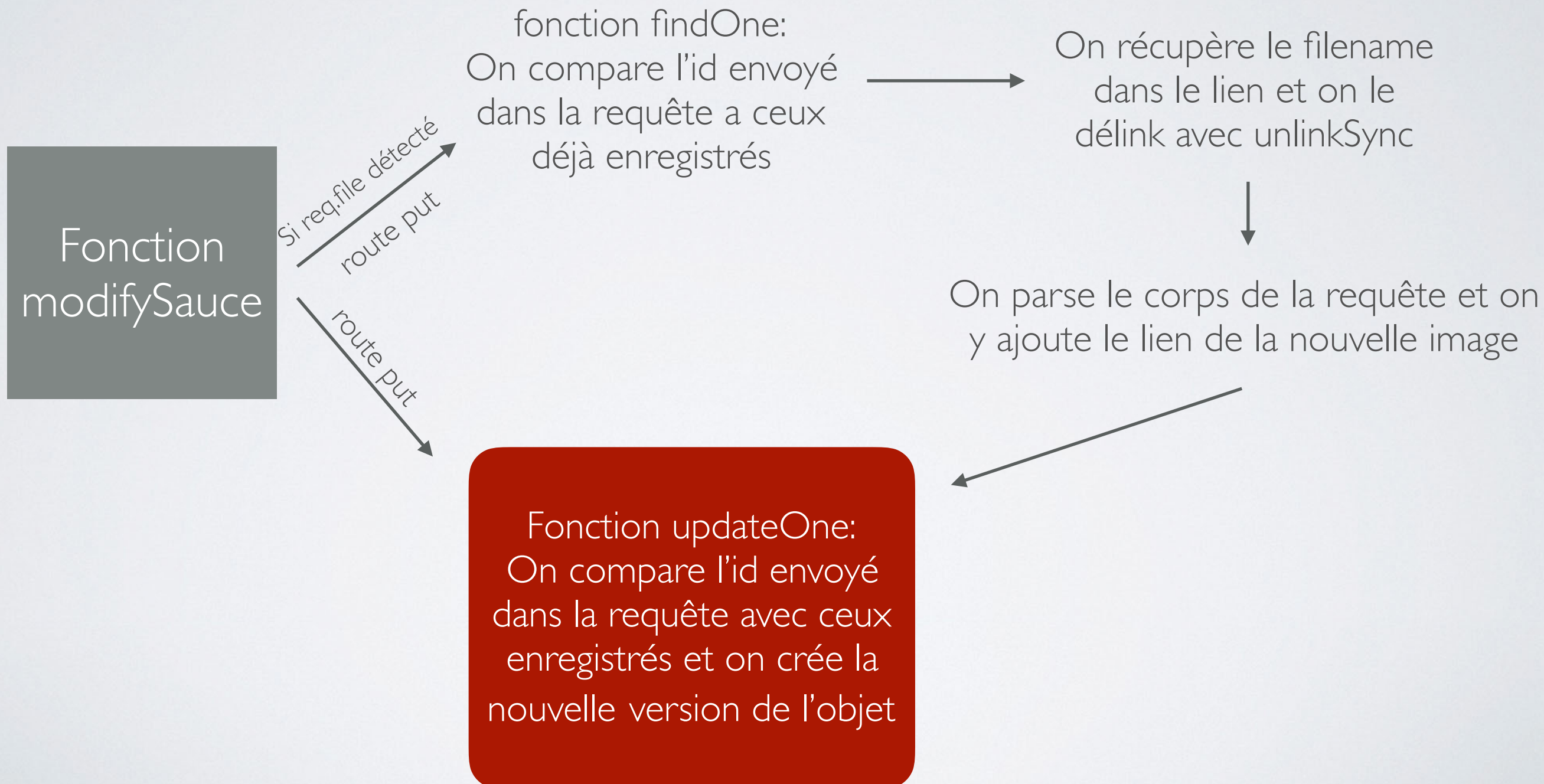
- Sauce:
- **C**reate
- **R**ead
- **U**ppdate
- **D**elete
- Liker
- Disliker



CRÉER



MODIFIER



SUPPRIMER

Fonction
deleteSauce

→ route delete

fonction findOne:
On compare l'id envoyé
dans la requête a ceux
déjà enregistrés

→

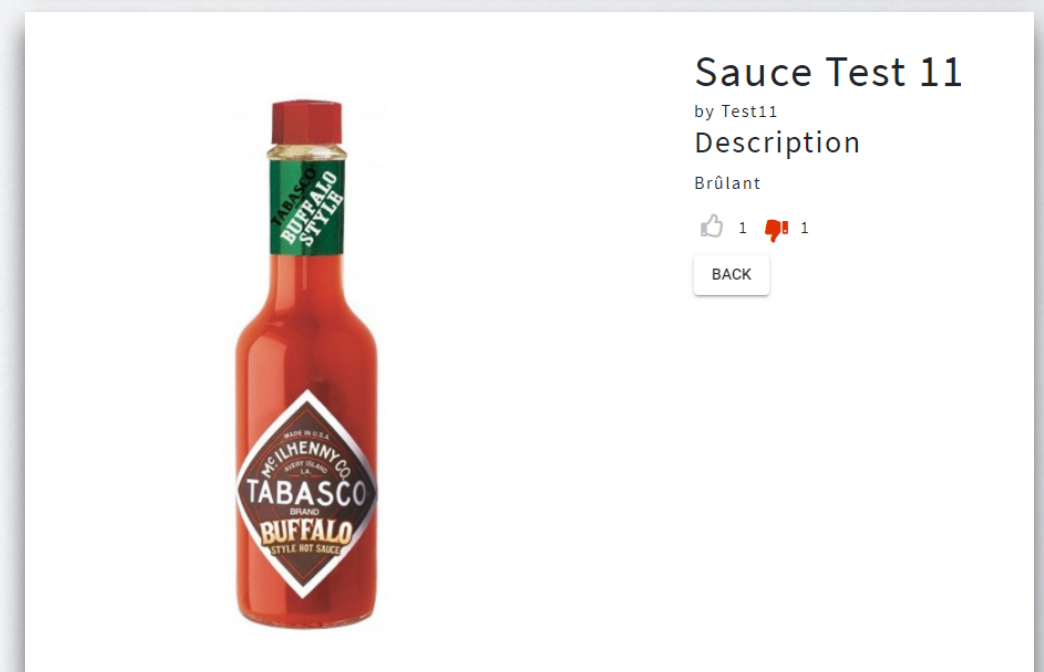
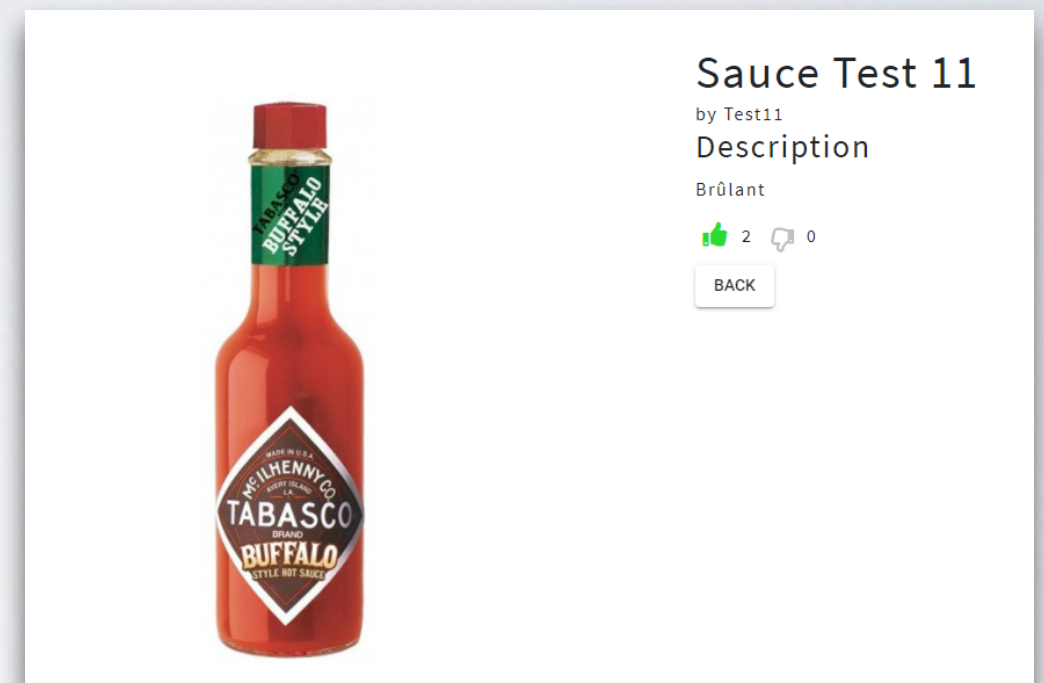
On récupère le filename
dans le lien et on le
délink avec unlinkSync

↓

Fonction deleteOne:
On compare l'id envoyé
dans la requête avec ceux
enregistrés et on supprime
cet objet de notre **BDD**

LIKER / DISLIKER

- L'utilisateur ne doit pouvoir liker une sauce qu'une seule fois
- L'utilisateur ne doit pouvoir disliker une sauce qu'une seule fois
- l'utilisateur doit pouvoir annuler son like ou son dislike



POUR LIKER OU DISLIKER UNE SAUCE

Fonction
likeSauce

→
route post

fonction updateOne:
On compare l'id envoyé
dans la requête a ceux
déjà enregistrés

→

Operator \$inc:
On incrémente une valeur
pour le paramètre
like ou dislike | ou |
selon si l'on like ou dislike

↓

On \$push dans le corps de
la requête le userId dans le
tableau usersLiked ou
usersDisliked

RETIRER UN LIKE OU UN DISLIKE

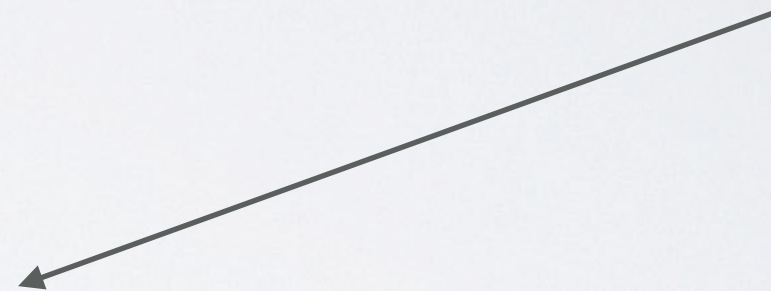
Fonction
likeSauce

→
route post

fonction findOne:
On compare l'id envoyé
dans la requête a ceux
déjà enregistrés

→

On compare le userId de la
requête avec les tableaux
usersLiked ou usersDislike



Si une correspondance est trouvée —> Fonction updateOne:
On incrémente avec \$inc la valeur -1 à like ou dislike et on \$pull du tableau
correspondant le userId de l'utilisateur

SÉCURITÉ

- UTILISATION DE PLUG-IN UNIQUEVALIDATOR ET DU PARAMÈTRE UNIQUE DANS NOTRE SCHÉMA UTILISATEUR
- AJOUT DU MIDDLEWARE AUTH SUR TOUTES LES ROUTES. CE MIDDLEWARE RÉCUPÈRE LE TOKEN ET LE DÉCODE. IL RÉCUPÈRE UN OBJET JSON DANS LEQUEL ON RÉCUPÈRE LE USERID
- UTILISATION D'UNE REGEX POUR FORCER L'UTILISATION DE MDP ROBUSTE: (AU MOINS 8 CARACTÈRES, AU MOINS UNE MAJUSCULE, AU MOINS UNE MINUSCULE, AU MOINS UN NOMBRE)
- MASQUAGE DES DONNÉES GRÂCE À DOTENV
- UTILISATION DE HELMET PRÉVENANT LES ATTAQUES DE TYPE XSS

MULTER

middleware
multer
config.js

Fonction diskstorage: 2 arguments

- la destination (dossier 'images')
- quel nom de fichier utiliser

fonction destination:
dossier 'images'

fonction filename:
génère un nouveau nom de fichier:

- Nom d'origine auquel on remplace les espaces dans le nom du fichier par des '_'
- On génère l'extension du fichier grâce à la correspondance du mime types du fichier avec un dictionnaire qu'on a préalablement créé
- On crée le nouveau nom:
`name` + `Date.now()` + `'.'` + `extension`