Telecom Valley

Soirée du <Test Logiciel>

# Tests d'API à partir de l'analyse des logs

Julien Botella

Jeudi 9 décembre 2021

Smartesting

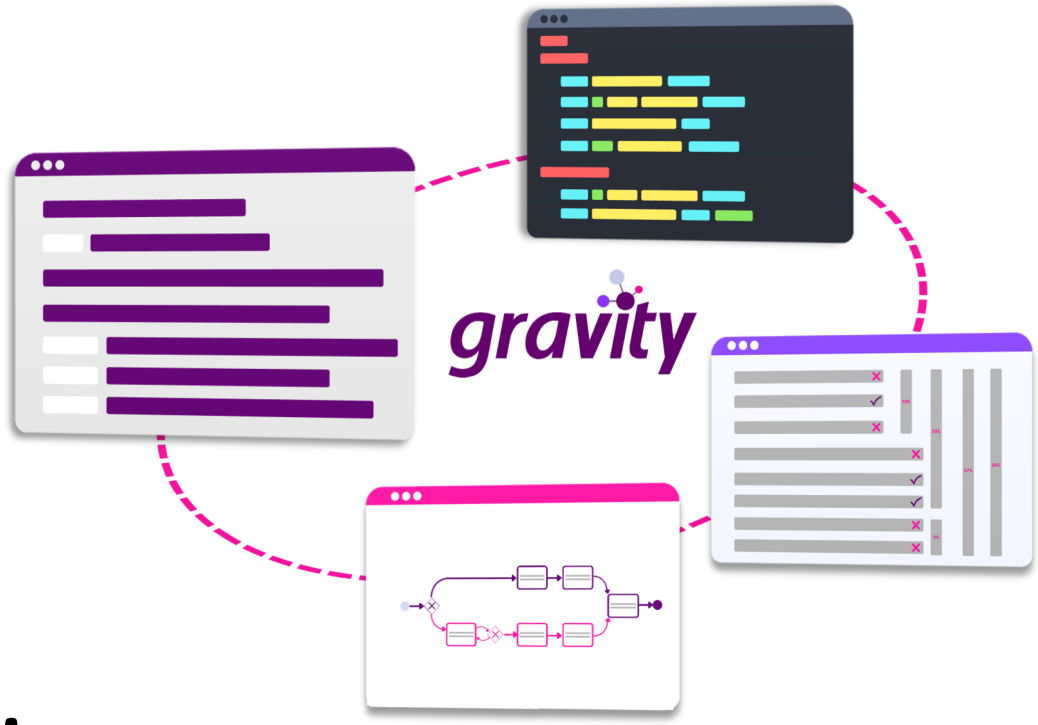
Telecom Valley | Soirée du <Test Logiciel>

# Agenda

- Introduction à Gravity
- Ateliers
  - Import des logs et analyse des traces d'usage
  - Analyse et complétion de la couverture de l'usage par les tests
  - Production de tests automatisés pour Postman
- Conclusion, Questions

**Julien Botella**
RESPONSABLE DU PRODUIT GRAVITY
Smartesting®

Objectifs : découvrir → expérimenter → discuter




# Introduction à Gravity

Des logs aux tests, motivations et démarche

# Des logs aux tests - Motivations

Les **pratiques de test** évoluent :

**Agilité + DevOps**

- releases plus fréquentes
- besoins en tests de régression automatisés

**Monitoring applicatif**

- logs (traces d'exécution) disponibles

→ **Créer** et **maintenir** des **tests de régression automatisés** grâce aux **logs** ?

**Bénéfices attendus** :

1. **Pertinence** des tests automatisés de régression (couverture de l'**usage**)
2. **Réduction de l'effort** de création et de maintenance
3. **Aide à l'analyse d'anomalies**

# Création d'un support outillé


**Outil expérimental développé par Smartesting**

1. Analyse de l'usage
2. Analyse et complétion de la couverture de l'usage par les tests
3. Aide à la production des tests automatisés couvrant l'usage

**Application web**

# Ecosystème

Logs

Tests

kibana

splunk>

Application Insights

DATADOG

dynatrace...

.txt, .log,
.json, .xml, …

gravity

Cucumber
Supported by SMARTBEAR

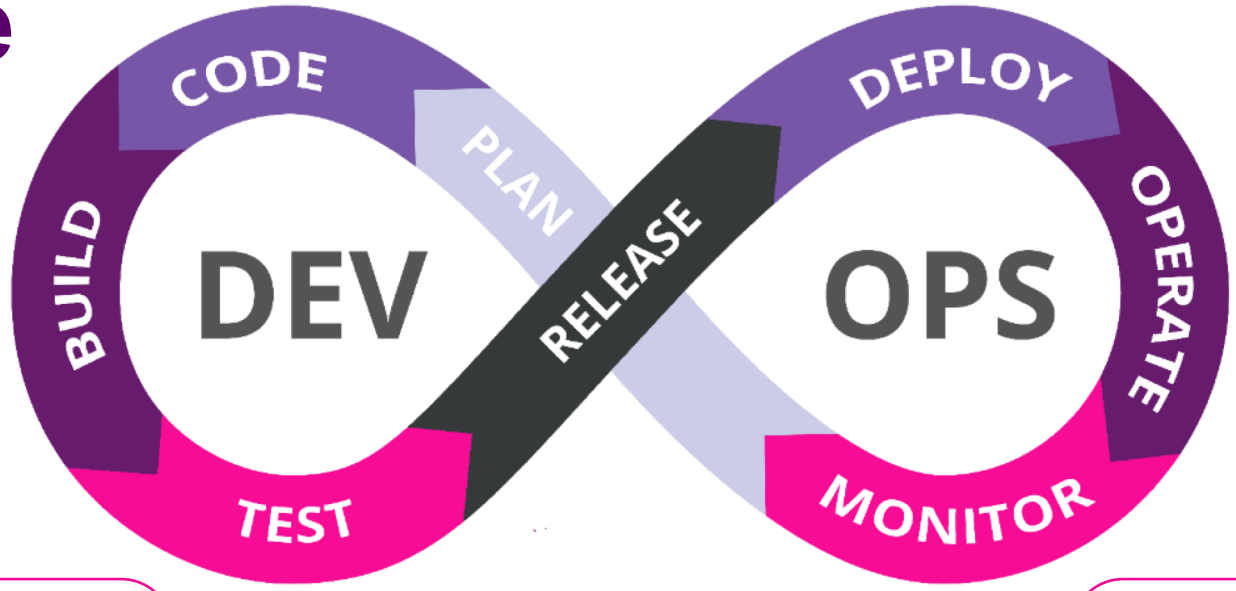
ROBOT
FRAME
WORK/

...

POSTMAN

Swagger
Supported by SMARTBEAR

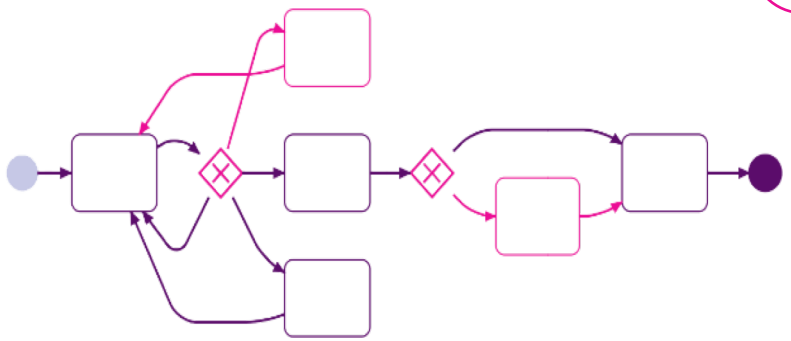
# Démarche




Test
(traces d'exécution)

(1) Analyser l'usage
(2) compléter la couverture de tests

Production
(traces d'usage)

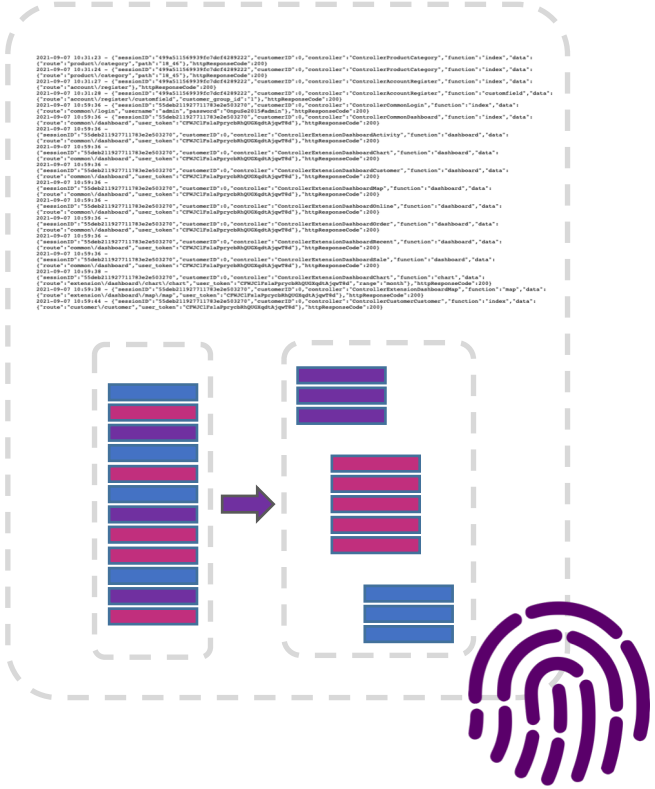


(3) Produire et maintenir les tests automatisés de régression

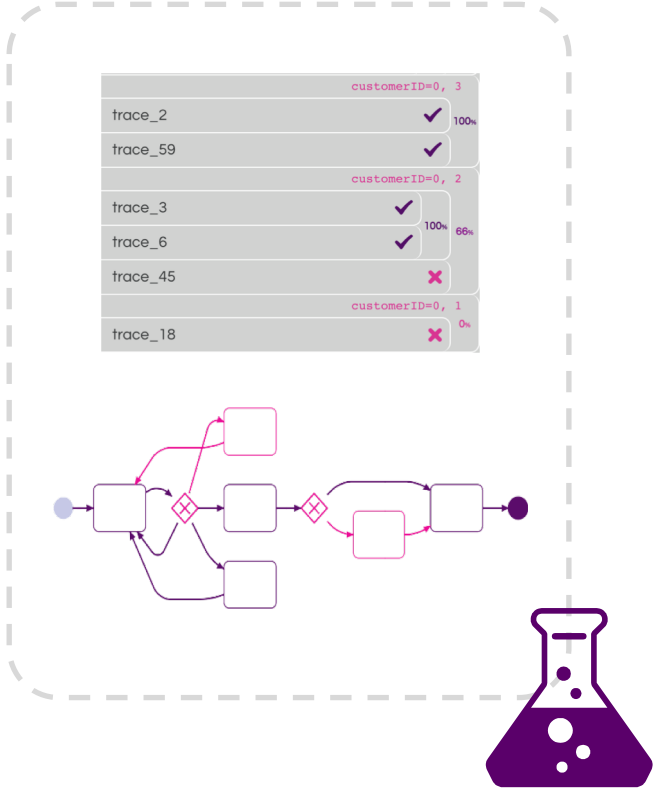
# Glossaire

- **Événement** : étape dans un fichier de log ou un outil de log
- **Donnée** : horodatage, identifiant de session, action, URL, paramètre, statut, ...
- **Donnée clé** : donnée à prendre en compte pour identifier le comportement d'où provient l'événement (statut, …)
- **Trace** : séquence d'événements
- **Trace d'usage** : séquence d'événements enregistrée à partir d'un parcours client dans l'application
- **Trace de test** : séquence d'événements enregistrée à partir d'une exécution de tests sur l'application
- **Scénario** : trace sélectionnée pour produire un test automatisé

# Démarche



Import des logs
et analyse des traces
d'usage




Analyse et complétion de la
couverture de l'usage par les
tests

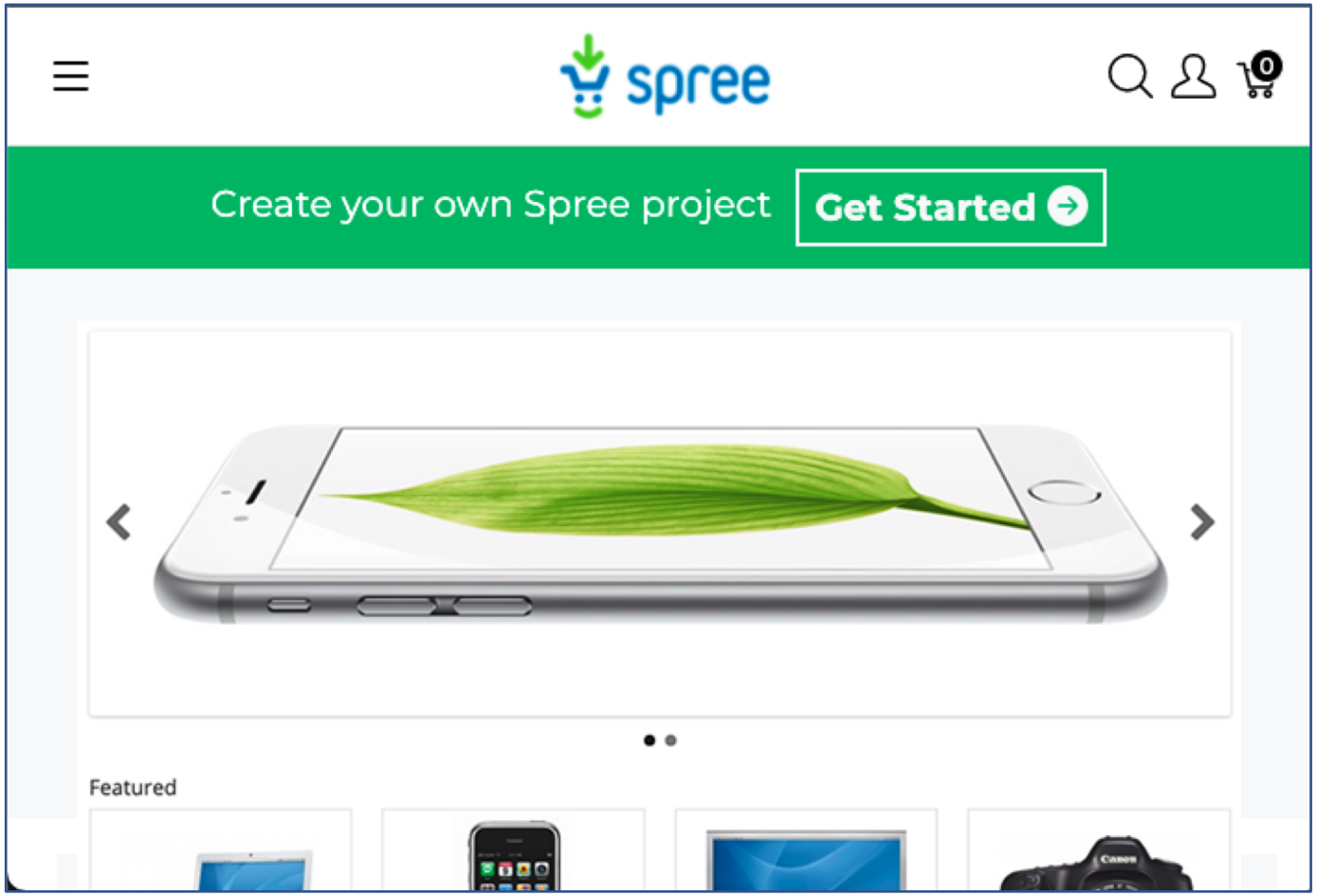

Production des tests
automatisés et export
pour l'exécution

# Cas d'étude : spree

Instance de Spree
(Open Source marchand)

→ API Rest
→ Doc Swagger de l'API

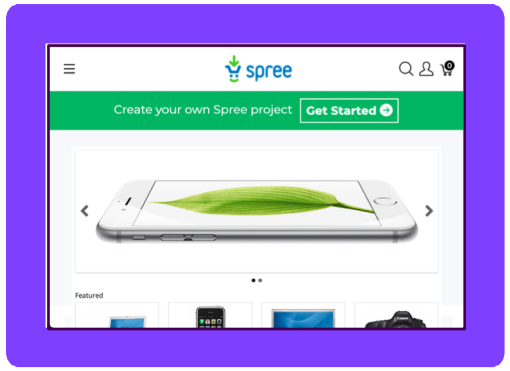
Swagger
Supported by SMARTBEAR



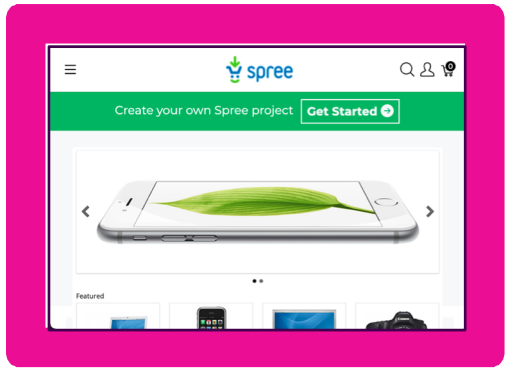
# Cas d'étude : spree

Utilisateurs

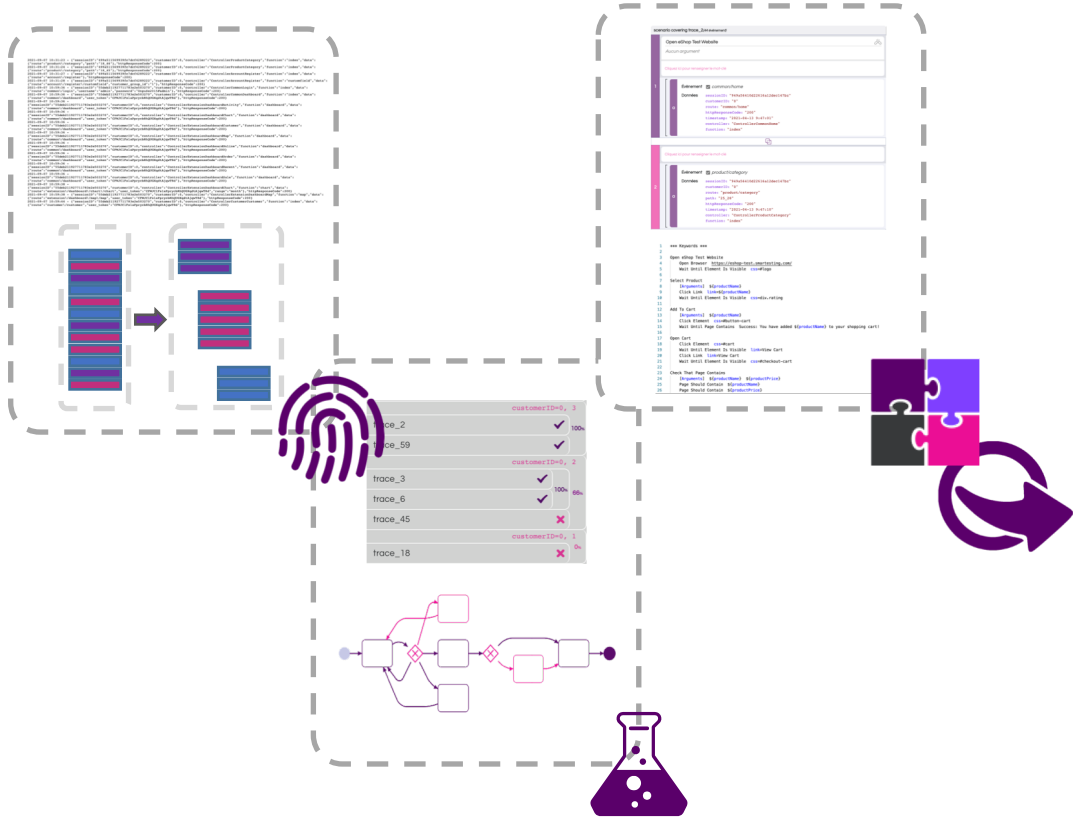
Production




Environnement de tests




Tests automatisés avec Postman




# Ateliers

Import, analyse Des logs aux tests, motivations et démarche

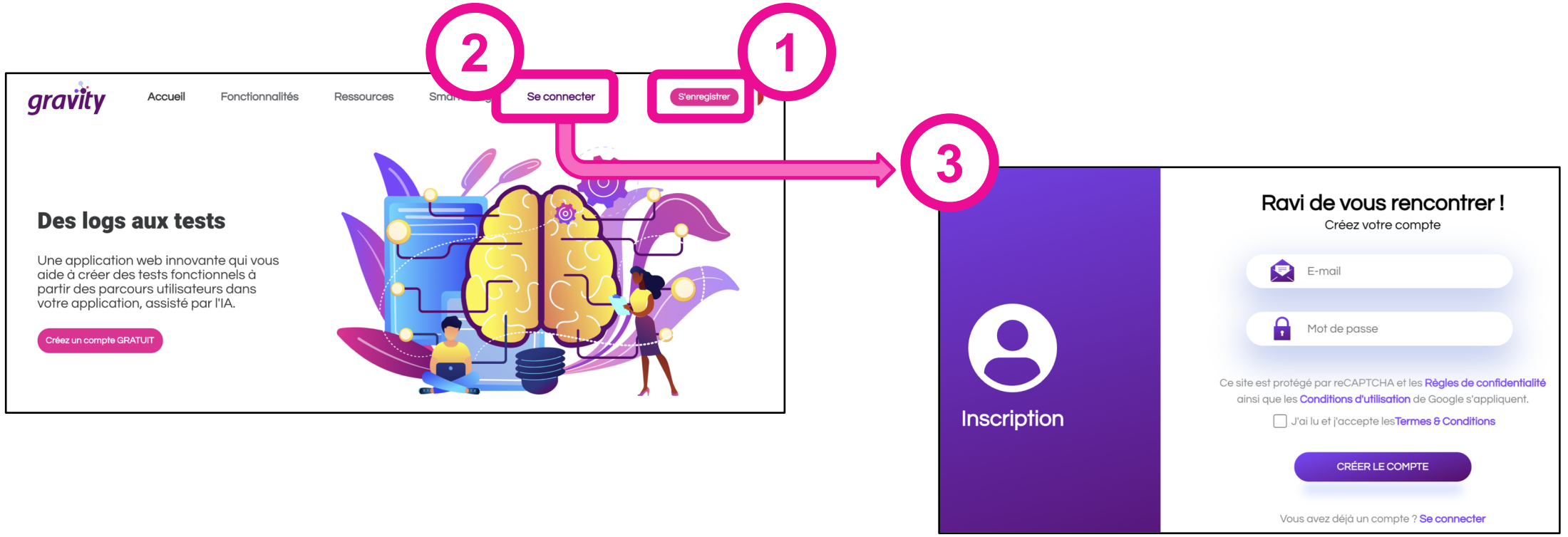
# Matériel

- **Rendez-vous sur https://gravity.smartesting.com/resources puis téléchargez le tutoriel spree (1)**

# Import des logs et analyse des traces d'usage

# Création de compte

- **Rendez-vous sur https://gravity.smartesting.com puis créez un compte si vous n'en possédez pas déjà un (1) ou connectez vous (2, 3)**



# Import de traces d'usage

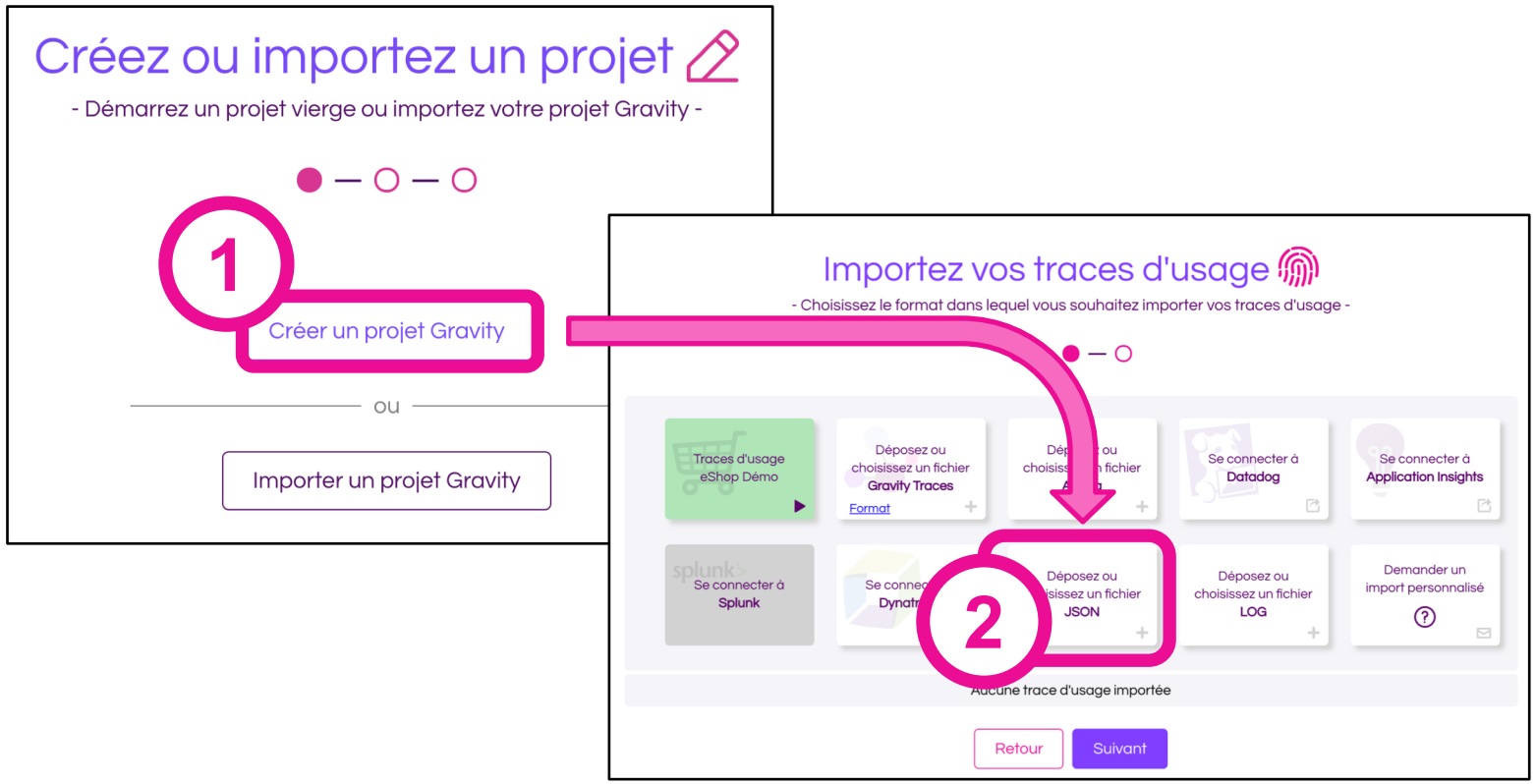
- *Des importeurs ou connecteurs existent pour différents formats ou outils de monitoring*
- *Des imports ou connecteurs personnalisés peuvent être réalisés*

**(1) Créez un projet Gravity**

**(2) Importez des traces d'usage provenant de spree**

**Fichier tutorielUsageTraces.json**

**(préparé spécifiquement pour ce tutoriel)**

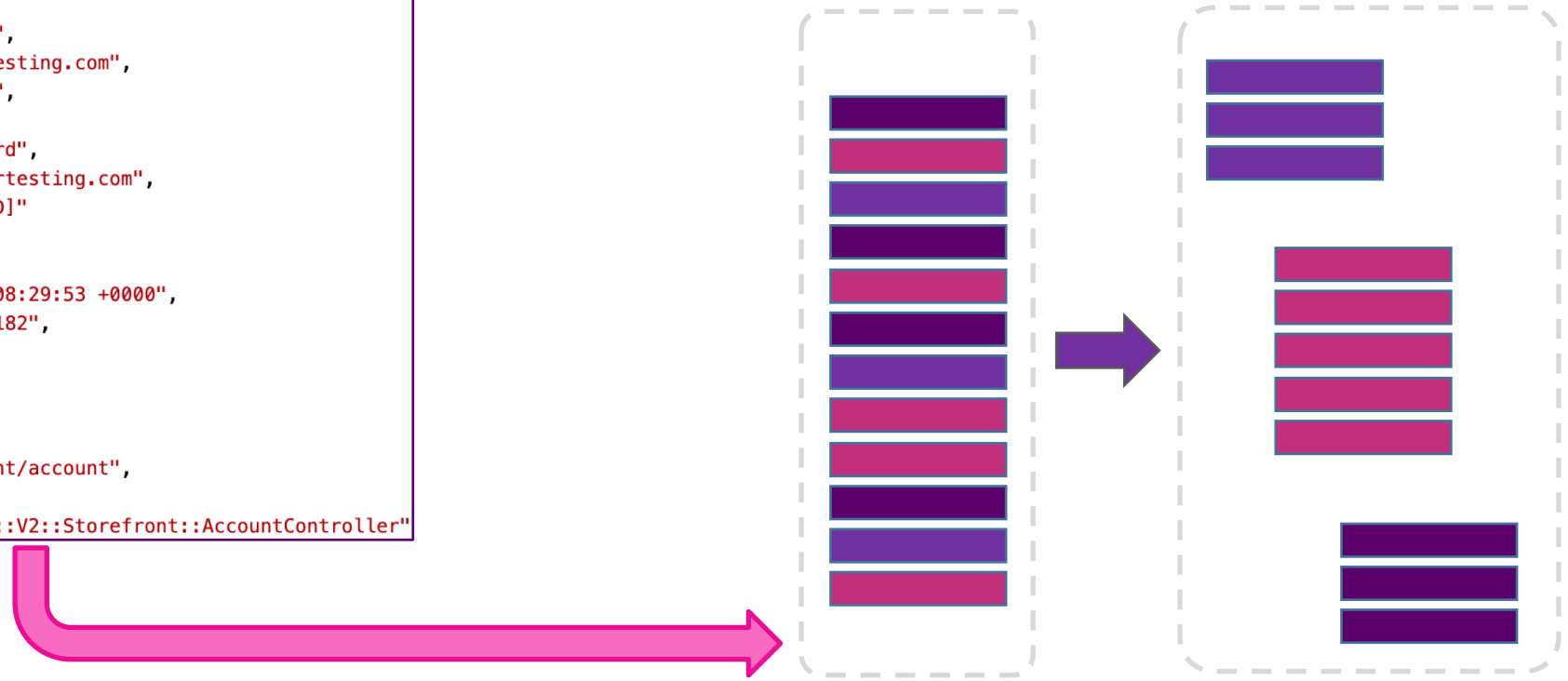


# Préparation des données


```
[
  {
    "method": "POST",
    "path": "/spree_oauth/token",
    "format": "*/*",
    "controller": "Doorkeeper::TokensController",
    "action": "create",
    "status": 200,
    "duration": 13.93,
    "view": 0.2,
    "db": 6.67,
    "ddsource": "ruby",
    "params": {
      "grant_type": "password",
      "username": "test@smartesting.com",
      "password": "[FILTERED]",
      "token": {
        "grant_type": "password",
        "username": "test@smartesting.com",
        "password": "[FILTERED]"
      }
    },
    "timestamp": "2021-11-15 08:29:53 +0000",
    "remote_ip": "158.255.99.182",
    "session_id": 1
  },
  {
    "method": "GET",
    "path": "/api/v2/storefront/account",
    "format": "json",
    "controller": "Spree::Api::V2::Storefront::AccountController"
```


- Identifier les traces d'usage
- Nommer les événements au sein des traces (verbes, actions ?)

→ rendre compréhensibles les séquences



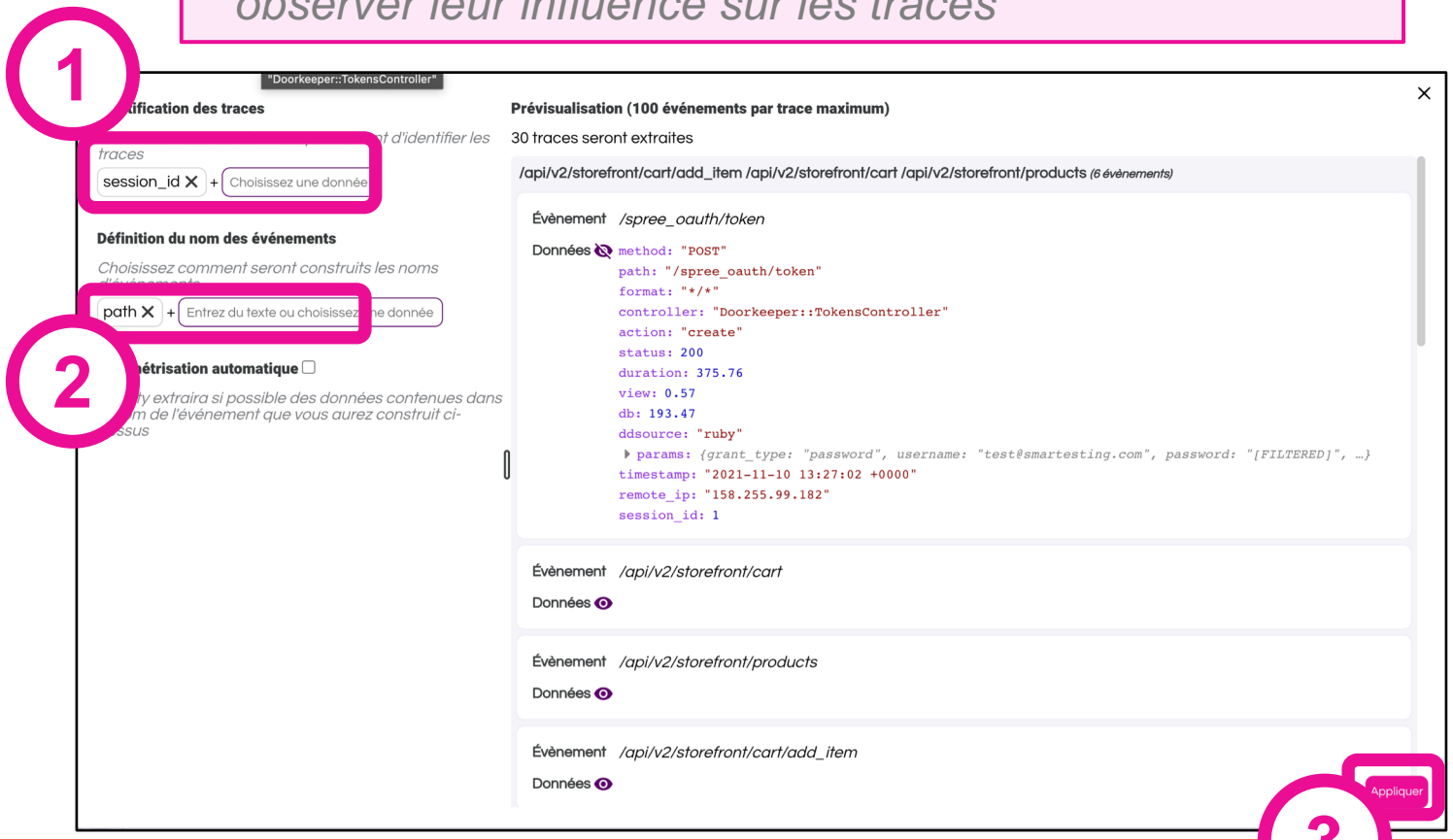
# Identification de sessions et des événements

**(1) Choisissez "session_id" pour identifier les sessions. Vous remarquerez que l'on passe d'une unique trace, à plusieurs.**

**(2) Choisissez "path" pour nommer les événements**

**(3) Appliquez l'import**

- *La prévisualisation permet de voir les événements importés, et leur partitionnement en traces*
- *Jouez avec les paramètres sur la gauche pour observer leur influence sur les traces*



# Import de traces d'usage

**(1) Cliquez sur "Suivant"**

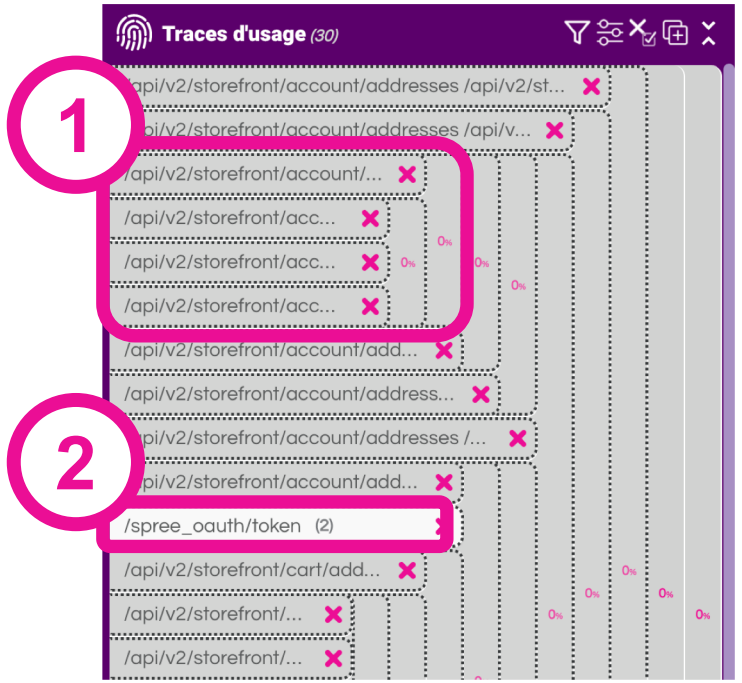
**(2) Nous importerons des traces de tests plus tard, cliquez sur "Terminer"**

# Analyse des traces d'usage : regroupements

- *Des techniques d'apprentissage automatique, plus particulièrement de "Clustering", sont utilisées pour faire les regroupements*
- *Chaque trace peut être renommée sur un clic droit*
- *CTRL + Clic permet d'ajouter ou retirer des éléments de la sélection*

**(1) Une fois les traces chargées, elles sont regroupées par similarité. Plus des traces sont proches fonctionnellement, plus elles sont proches dans le "dendrogramme"**

**(2) Si deux traces exercent le même parcours dans l'application, elles sont regroupées et leur occurrence est affichée**

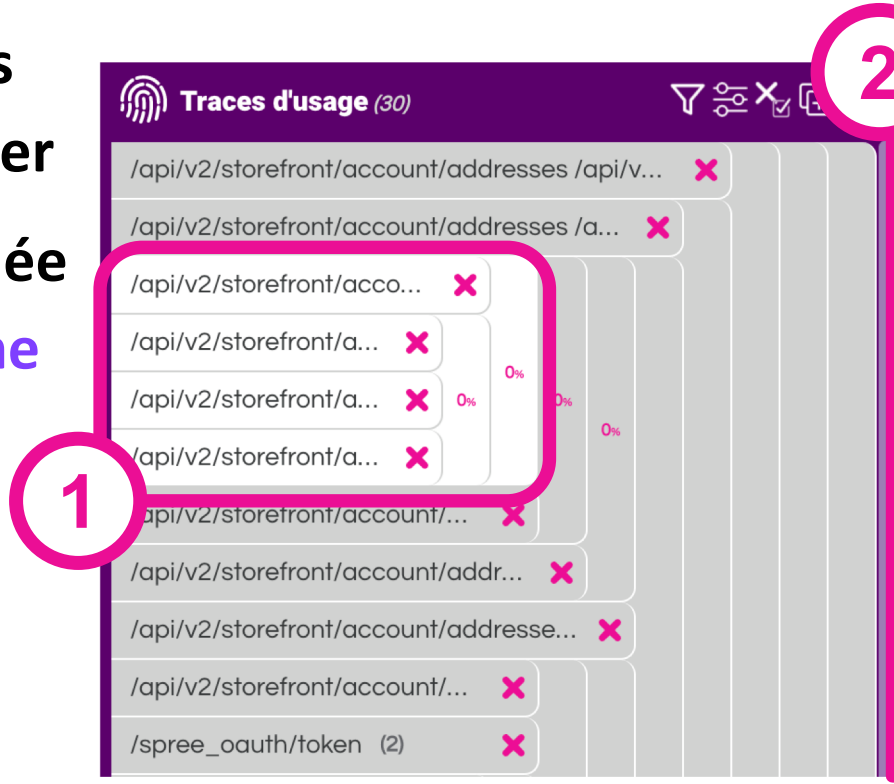


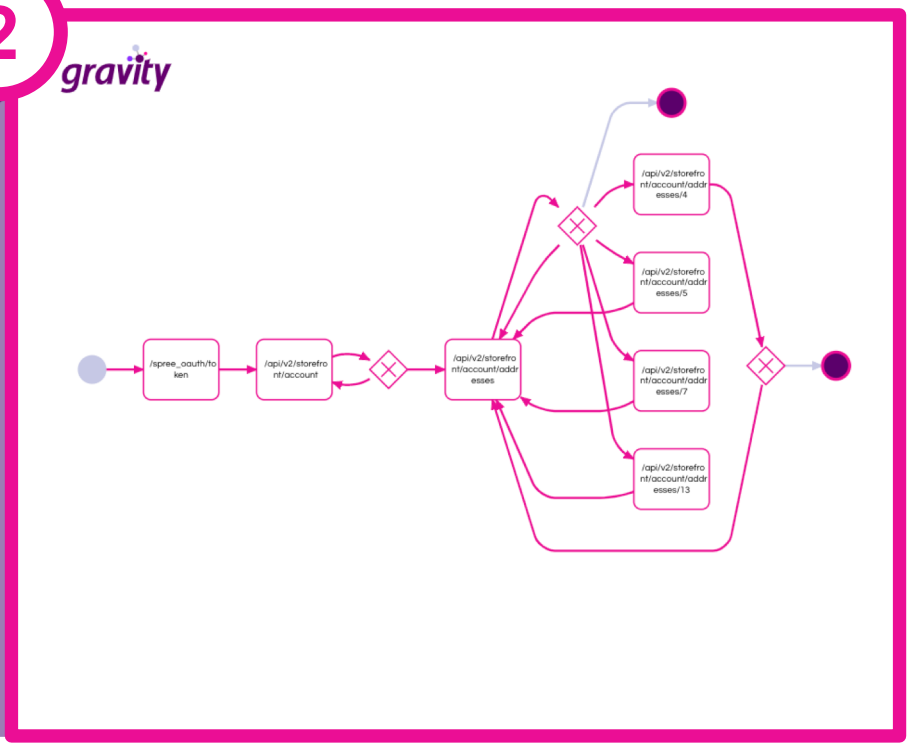
# Analyse des traces d'usage : diagramme

- *Essayer de visualiser des diagrammes issus de différents groupes*
- *Si vous survolez une trace ou groupe de traces sélectionnée*

**(1) Un clic sur tout groupe ou sous-groupe de traces permet de les sélectionner**

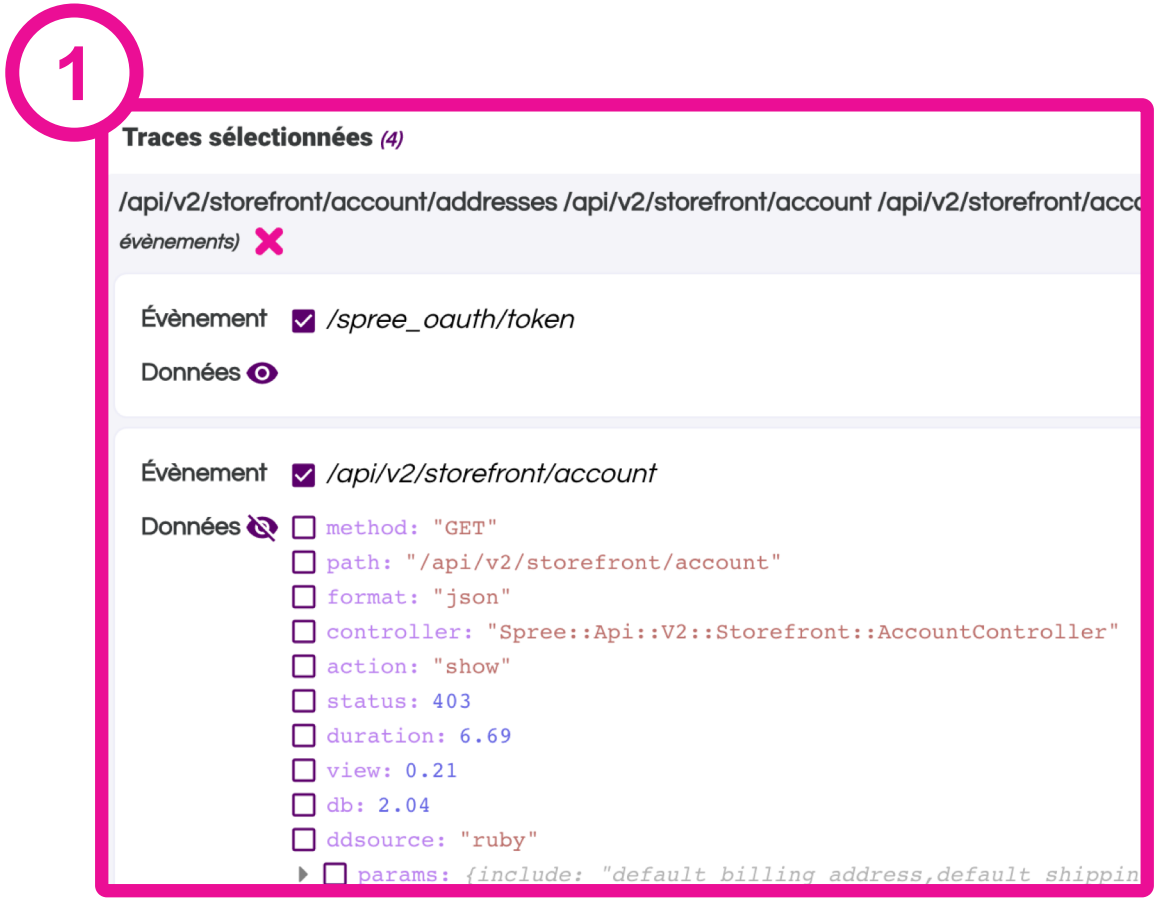
**(2) La sélection est affichée sous forme de diagramme**



# Analyse des parcours utilisateur : contenu

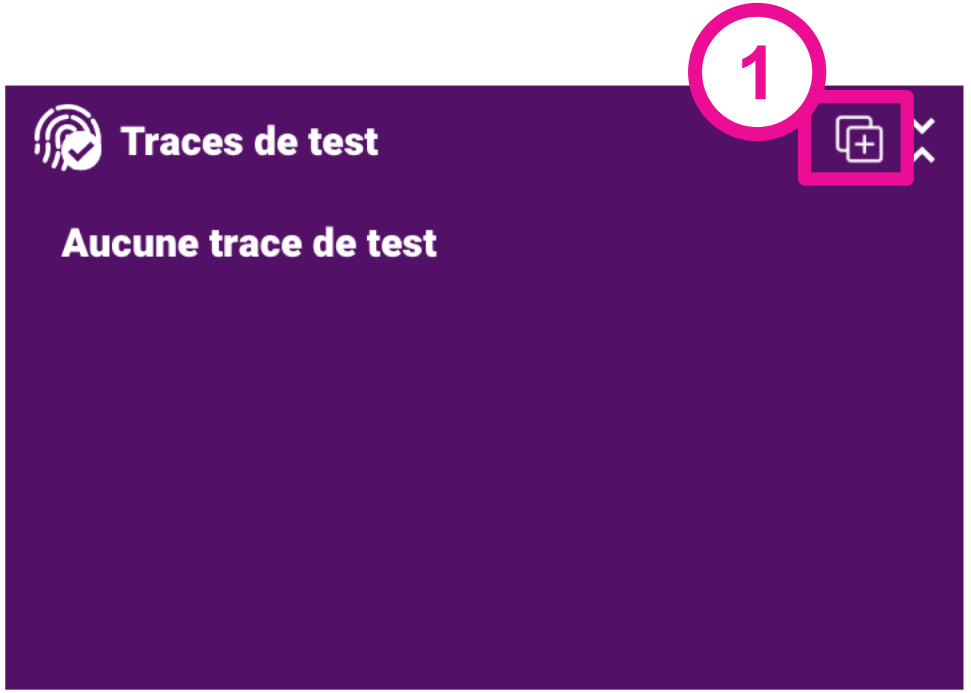
**(1) Les détails de la sélection sont affichés (traces, événements, données)**



# Complétion de la couverture de l'usage par les tests

# Import de traces de test

(1) Accédez à l'import de traces de tests




Puis procédez de même que pour importer les traces d'usage spree

Fichier

tutorielTestTraces.json

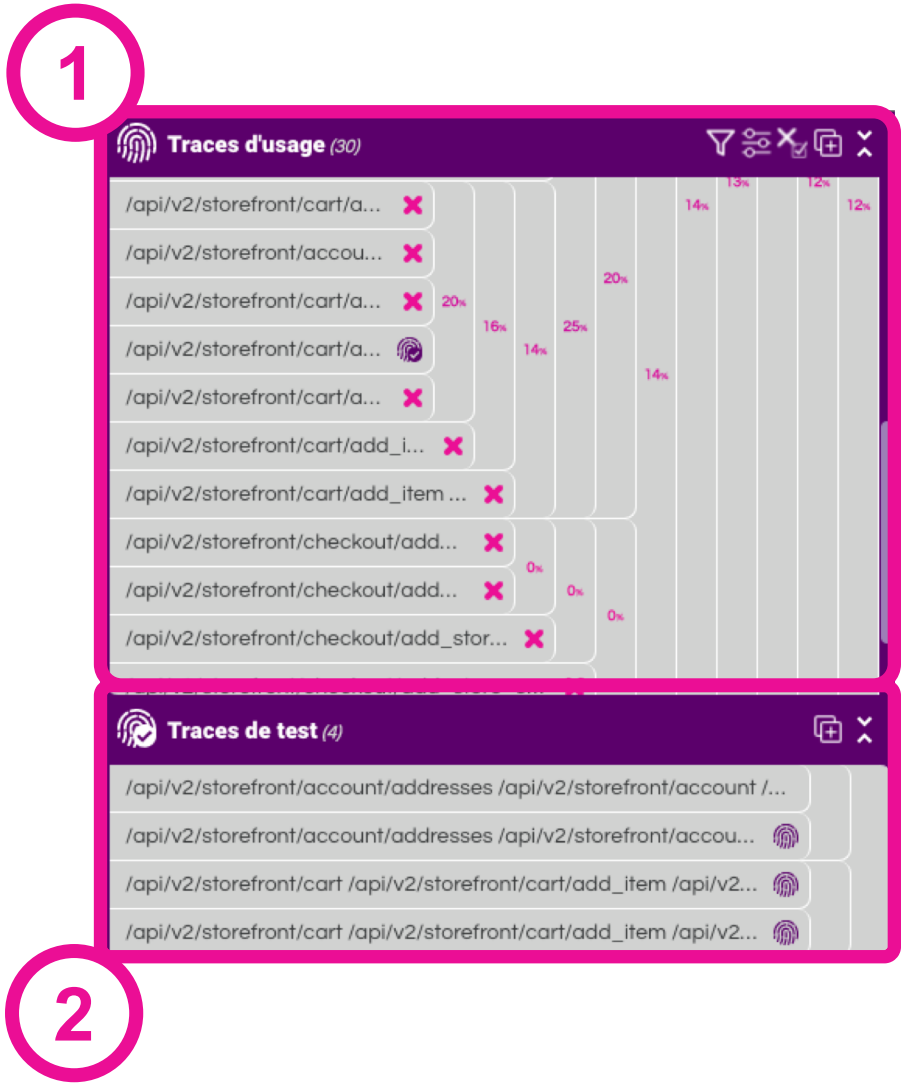
# Analyse de la couverture : regroupements

(1) Pour les traces d'usage,

 signifie qu'au moins une trace de test ou un scénario créé (ensuite) la couvre.

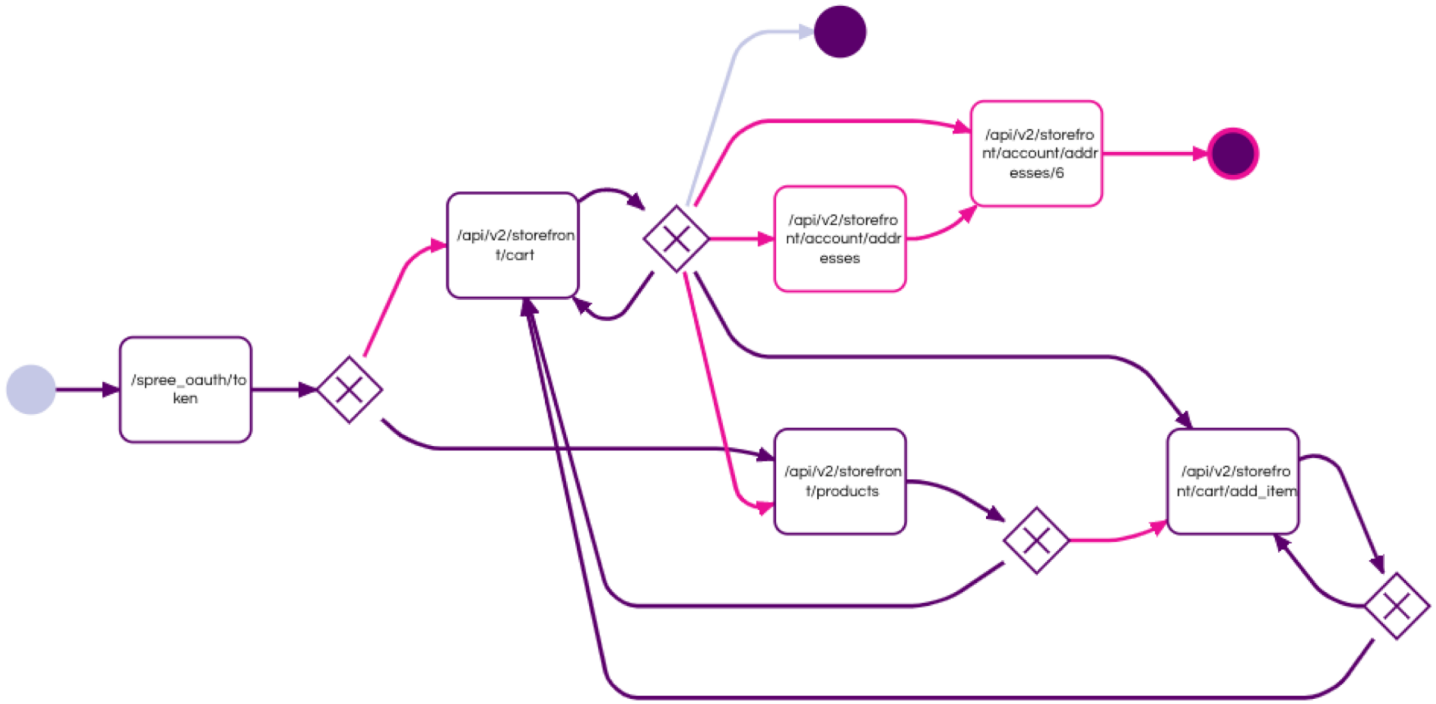
 Signifie que l'usage n'est pas couvert.

(2) Pour les traces de test ou scénarios,  signifie qu'au moins un usage est couvert par l'élément



# Analyse de la couverture : diagramme

**Sur le diagramme les chemins en rose proviennent de traces d'usage, non couvertes, ceux en violet proviennent de traces couvertes**



# Complétion de la couverture

**(1) Sélectionnez les traces suivantes**

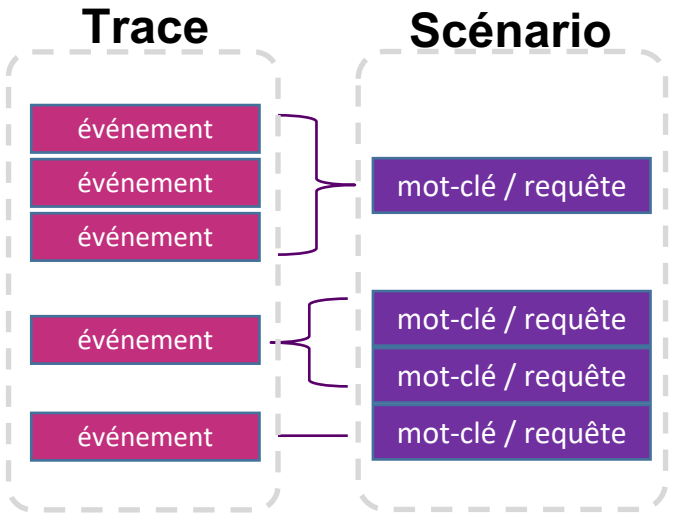
**(2) Des traces sélectionnées peuvent être choisies pour donner lieu à des scénarios en cliquant sur**

**Couvrez 2 traces non couvertes et obtenez 2 scénarios (3)**

- *La couverture sur le dendrogramme et le diagramme est mise à jour lors de la création de scénarios*

# Production de tests automatisés pour Postman

# Correspondance événements / mots-clés ou requêtes



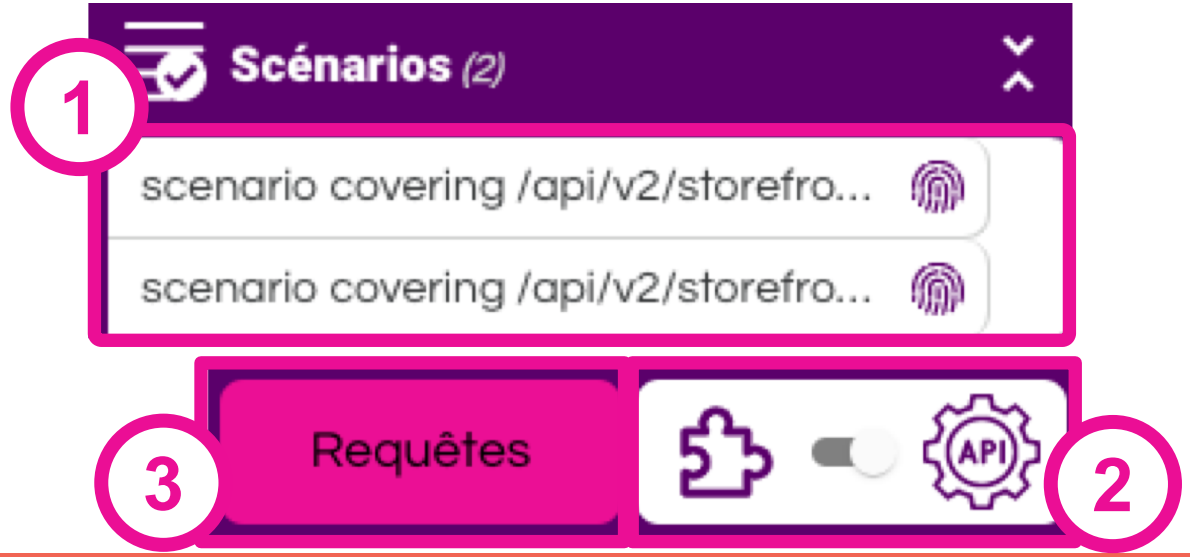

- *Nous voulons passer d'une séquence d'événements, à une séquence de mots-clés ou requêtes*
- *Le but est si possible de réutiliser un maximum de mots-clés existants pour automatiser*
- *Les mots-clés manquants peuvent être créés, et complèteront la bibliothèque d'automatisation*
- *Les paramètres des mots-clés ou requêtes peuvent prendre des valeurs provenant des événements*

**Allez sur l'atelier**

**(1) Sélectionnez les 2 scénarios créés**

**(2) Choisir "API"**

**(3) Ouvrez le panneau de gestion des requêtes**

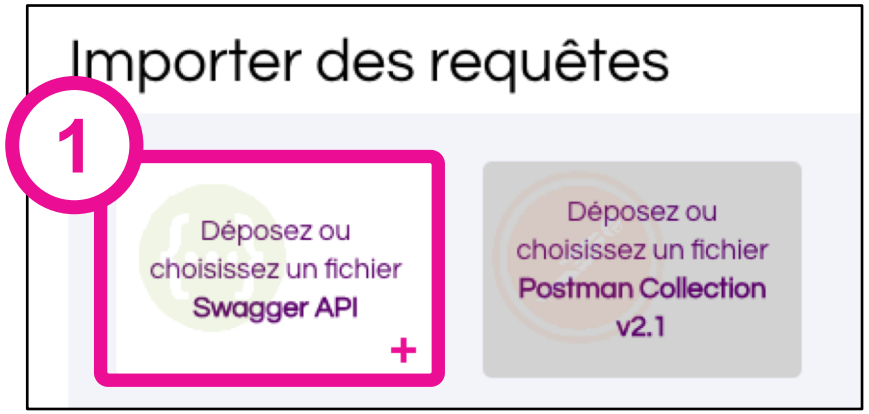


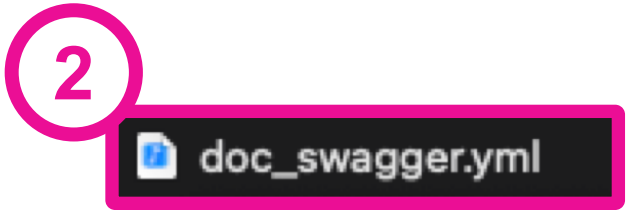
# Correspondance événements / requêtes

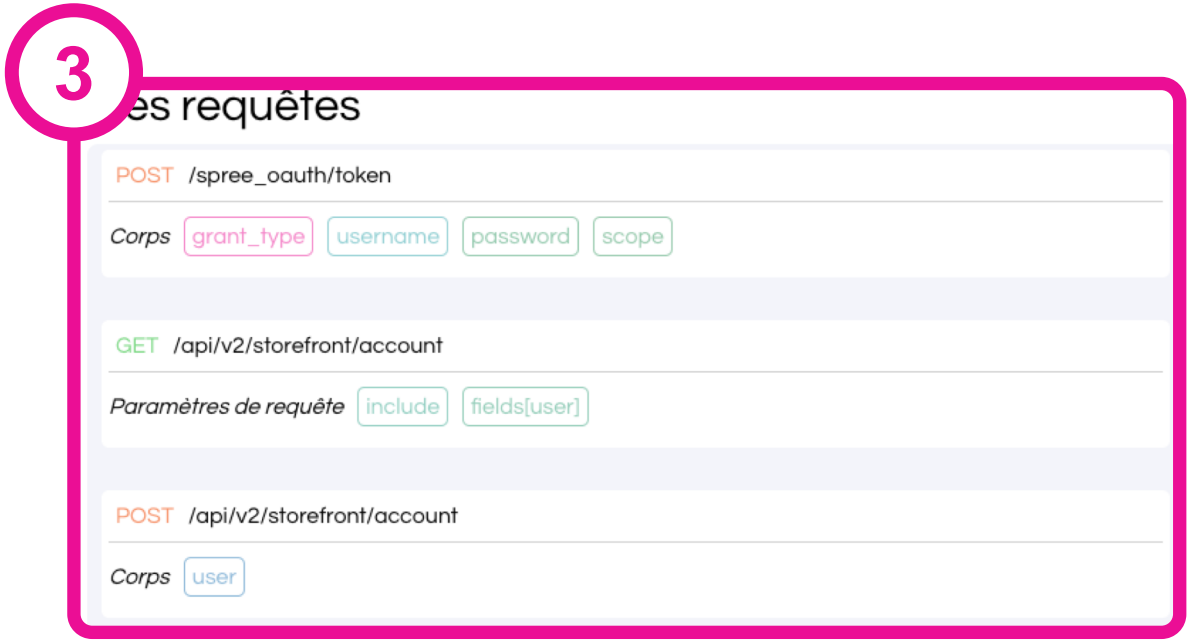
**(1) Choisissez de charger des requêtes provenant d'une documentation Swagger**

**(2) Importez le fichier "doc_swagger.yml"**

**(3) Les requêtes importés pourront être utilisés pour construire les scénarios**

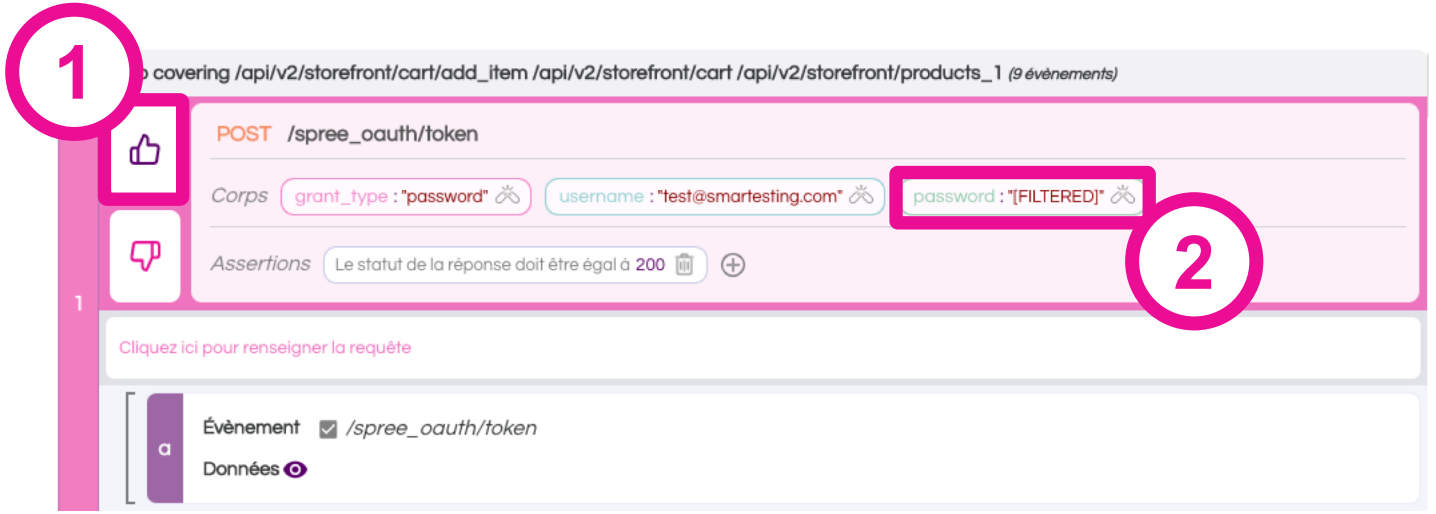


# Correspondance événements / requêtes

**Gravity va proposer des requêtes semblant correspondre aux événements**




**(1) Acceptez la proposition, l'événement est maintenant attaché à la requête proposée**

**(2) Il est possible de modifier les valeurs de paramètres. Choisissez test123 pour le mot de passe**

# Export des scripts de tests

**Allez sur l'atelier**

**(1) Exportez le projet pour Postman**


*Vous avez obtenu vos premiers (probablement) tests automatisés à partir de logs*

# Conclusion

"Des logs aux tests" applicable dan votre contexte ?

# En résumé, nous avons pu

- Importer des logs depuis des environnements de production et de test
- Analyser l'usage à partir de logs
- Analyser et compléter la couverture des tests
- Produire de tests automatisés avec Postman

# Pour réaliser une expérimentation

Pré-requis :

- Avoir accès à des logs de production (et de test)

  splunk, dynatrace, app insight, datadog, kibana

- Avoir des logs riches d'un point de vue fonctionnel

  actions utilisateur, API, paramètres

- Avoir mis en place une approche par mot-clés ou du test d'API pour l'automatisation

  cerberus, java/cucumber, postman

  (possibilité d'export personnalisé)

# Pour réaliser une expérimentation

Approche en 3 étapes :


Devenez Bêta testeur gravity , contactez-nous :
https://gravity.smartesting.com/devenez-beta-testeur
gravity@smartesting.com

Telecom Valley | Soirée du <Test Logiciel>

# Merci !

Avez vous des questions ?

Smartesting