w.c.s. - Development #38073

tests sur les accès aux formdata (accès directs ou via le code de suivi)

29 novembre 2019 11:03 - Nicolas Roche

Statut:	Fermé	**Début:**	29 novembre 2019
Priorité:	Normal	**Echéance:**	
Assigné à:	Nicolas Roche	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	Non

Description

Le but de ce ticket est de proposer un test qui reprend les différent cas d'accès aux ressources accessibles via un code de suivi :

- accès direct ou par code de suivi
- accès aux demandes ou aux brouillons
- accès par différents utilisateurs

Demandes liées:

Lié à w.c.s. - Development #38077: Accès aux brouillons via le code de suivi	**Fermé**	**29 novembre 2019**

Historique

#1 - 29 novembre 2019 12:03 - Nicolas Roche

- Fichier 0001-tracking_code-add-tests-on-formdata-access-38073.patch ajouté

- Statut changé de Nouveau à Solution proposée

- Patch proposed changé de Non à Oui

L'idée serait d'avoir une matrice récapitulative :

```
    1- Direct access to ressources :

    | sumitter / accesser | anonymous | user1 | user2 | agent1 | agent2 | admin1 |
    +---------------------+-----------+-------+-------+--------+--------+--------+
    | anonymous           |  login    | deny  | deny  | deny   | (*)    | (*)    |
    | agent1 (submiter)   |  login    | deny  | deny  | deny   | (*)    | (*)    |
    | user1               |  login    | allow | deny  | deny   | (*)    | (*)    |

    (*) Agent2 is the receiver.
        Redirected into backoffice for demands.
        Access denied for drafts.

    2- Access using tracking code :

    All access is granted,
    On restoring draft, the logged user become the new draft owner,
    this affect the computed and prefill fields.
```

#2 - 29 novembre 2019 12:43 - Nicolas Roche

- Lié à Development #38077: Accès aux brouillons via le code de suivi ajouté

#3 - 29 novembre 2019 13:59 - Thomas Noël

Je ne comprends pas la matrice.

Par exemple, j'y lis que si sumitter=anonymous et accesser=user1, alors deny ? Mais non, il faut au contraire accepter... non ?

#4 - 29 novembre 2019 16:36 - Nicolas Roche

- Statut changé de Solution proposée à En cours

C'est bien ce que tu lis.
Sauf qu'en vrai ça marche, et donc mon test est mauvais...

#5 - 29 novembre 2019 17:43 - Nicolas Roche

- Statut changé de En cours à Solution proposée

En fait si, c'est bon :
c'est proche de ce que tu décris dans https://dev.entrouvert.org/issues/37808#note-2 :

> si elle revient "plus tard" sur l'URL de sa demande, par hasard ou historique de son navigateur

J'arrive sur la même exception "en vrai" (en utilisant un chrome tout neuf, et non pas une instance privé de FF), quand je pointe tout de go sur https://wcs.dev.publik.love/arbres-espaces-verts-aires-de-jeux/30/ :

```
/wcs/forms/common.py::FormStatusPage
    def check_receiver(self):
 ...
        if not self.filled.formdef.is_user_allowed_read(user, self.filled):
            raise errors.AccessForbiddenError()  # <-- ici
```

Peut-être que ce n'est pas le comportement voulu, mais j'ai l'impression que c'est bien le comportement observé actuellement.

#6 - 29 novembre 2019 17:48 - Thomas Noël

Nicolas Roche a écrit :

> En fait si, c'est bon :

Oui, j'avais lu trop vite croyant que tu respectais le sujet du ticket ;-) En fait la matrice concerne "Direct access to ressources", rien à voir avec le code de suivi...

#7 - 29 novembre 2019 18:19 - Nicolas Roche

- Sujet changé de tests sur les accès via le code de suivi à tests sur les accès aux formdata (accès directs ou via le code de suivi)

- Description mis à jour

Tu as raison c'est plutôt confus, j'essaie d'améliorer.

#8 - 02 janvier 2020 17:19 - Nicolas Roche

- Fichier 0001-tracking_code-add-tests-on-formdata-access-38073.patch ajouté

J'ai ajouté un test sur le pré-remplissage qui à lieu dans les formulaires de workflow.

Cependant, je n'ai pas réussi à tester avec les choix proposés par l'IHM :
ex: prefill={'type': 'user', 'value': '_first_name'})

Le pré-remplissage avec {{form_user_display_name}},
qui change de valeur pour prendre le nouvel utilisateur logué sur les brouillons,
garde l'utilisateur qui est à l'origine de la demande sur les workflows.

```
    2- New user on prefill fields when accessing using tracking code :
     a-  Drafts

    | sumitter / accesser | anonymous | user1 | user2 | agent1 | agent2 | admin1 |
    +---------------------+-----------+-------+-------+--------+--------+--------+
    | anonymous           | anonymous | user1 | user2 | agent1 | agent2 | admin  |
    | agent1 (submiter)   | anonymous | user1 | user2 | agent1 | agent2 | admin  |
    | user1               | anonymous | user1 | user2 | agent1 | agent2 | admin  |

     b-  Demands (evolving into Workflows forms)

    | sumitter / accesser | anonymous | user1 | user2 | agent1 | agent2 | admin1 |
    +---------------------+-----------+-------+-------+--------+--------+--------+
    | anonymous           | anonymous | anon. | anon. | anony. | anony. | anon.  |
    | agent1 (submiter)   | anonymous | anon. | anon. | anony. | anony. | anon.  |
    | user1               | user1     | user1 | user1 | user1  | user1  | user1  |
```

Le problème qui a motivé l'élaboration de ces grilles, semble mis en évidence : (https://dev.entrouvert.org/issues/36086#note-64)
L'accès à un brouillon via code de suivi change l'identité du demandeur (puisque la demande n'est pas encore déposée),
ce qui a des conséquences sur les champs (pas encore) pré-remplis (ou pré-remplis à nouveau).
On constate ce comportement :

- si un utilisateur logué utilise le code de suivi d'un brouillon anonyme.
  (un brouillon anonyme est également obtenu sur une saisie backoffice)
- si un utilisateur revient sur un brouillon de façon anonyme
- si un utilisateur utilise le code de suivi du brouillon d'un autre utilisateur.

#9 - 07 janvier 2020 16:37 - Nicolas Roche

- Fichier 0001-tracking_code-add-tests-on-formdata-access-38073.patch ajouté

J'ai ajouté les champs préremplis avec les données utilisateur (ceux disponibles via l'IHM).

```
prefill={'type': 'user', 'value': '_first_name'}),
```

Et en fait non, avec ce pré-remplissage, on a le même comportement dans les workflows que sur les brouillons :
dans les formulaires de workflow, ces champs préremplis vont aussi changer de valeur quand un autre utilisateur reprendra la demande via son code de suivi.

```
     b-  Demands (evolving into Workflows forms)
```

```
    | sumitter / accesser | anonymous | user1 | user2 | agent1 | agent2 | admin1 |
    +---------------------+-----------+-------+-------+--------+--------+--------+
    | anonymous           | anonymous | user1 | user2 | agent1 | agent2 | admin  |
    | agent1 (submiter)   | anonymous | user1 | user2 | agent1 | agent2 | admin  |
    | user1               | anonymous | user1 | user2 | agent1 | agent2 | admin  |
```

#10 - 08 janvier 2020 11:48 - Nicolas Roche

- Fichier 0001-tracking_code-add-tests-on-formdata-access-38073.patch ajouté

juste le renommage d'une variables (s/expected/access/)

#11 - 13 janvier 2020 10:36 - Frédéric Péters

Il m'a l'air d'y avoir mille choses dans ce test, genre vraiment besoin de six champs dans la démarche pour tester l'accès, vraiment besoin d'un workflow particulier, etc. ? je serais pour drastiquement réduire, et ne pas mélanger d'histoire de préremplissage ici.

#12 - 03 février 2020 16:28 - Nicolas Roche

Il y a bien 2 thématiques : la gestion des accès aux demandes (et brouillons)

- directs (via une URL)
- et indirects (via code de suivi)

La problématique des accès directs a été ajouté pour couvrir les éventuelles régression liées à la résolution de #37808. Je dirais qu'elle reste secondaires ici, car ce qui a motivé ces tests c'est d'abord #38239 (qui vient solutionner #37095).

La problématique des accès via code de suivi est liée à celle de l'appartenance du formulaire (formdata_user) comme le suggère #37095. Dit autrement, ces tests sont proposés afin de pouvoir mesurer les régressions des futures corrections relatives aux champs pré-remplis avec les données de l'utilisateur.

Ci-dessous, voici ce que vérifient les tests qui couvrent cette seconde problématique :

- commentaire : pour voir que {{form_user_display_name}} évolue dans le commentaire ; et que c'est la valeur la plus proche de formdata_user (qui fait référence), sans l'être complètement.

1) Sur la première page, remplie par le premier usager qui accède au formulaire :

- prefill-string-page1 : pré-remplissage avec {{form_user_display_name}}
- prefill-user-page1 : pré-remplissage avec les variables de sessions (solution mise en avant par l'IHM)

On constate que ces 2 pré-remplissages sont cohérents : ils ont été figés lors du pré-remplissage et ne suivent donc pas l'évolution de la valeur formdata_user.

2) Sur la seconde page du formulaire, remplie par un second usager qui accède au brouillon :

- prefill-string-page2 : pré-remplissage avec {{form_user_display_name}}
- prefill-user-page2 : pré-remplissage avec les variables de sessions

On constate que ces 2 champs qui suivent la valeur du champ commentaire respectent la règle : "Taper le code de suivi = devenir/être la personne à l'origine de la demande." ; ce que décrit #37095.

3) Sur le formulaire de workflow, remplie par un second usager qui accède à la demande :

- prefill-string-workflow : pré-remplissage avec {{form_user_display_name}}
- prefill-user-workflow : pré-remplissage avec les variables de sessions

On constate que la règle citée ci-dessus ne s'applique globalement pas, mais suffisamment quand même pour confirmer les doutes évoqués en réunion concernant l'élargissement du périmètre couvert par cette problématique.

Tout ça pour dire que tant qu'on ne sait pas trop vers où on va, je ne saurais pas quoi supprimer ici.

#13 - 03 février 2020 16:48 - Frédéric Péters

Ce test est inmaintenable; dans ton commentaire tu décris les multiples aspects qui sont testés; mon souhait c'est que tous les aspects ne soient pas testés dans un unique test, qui se trouve définir ses propres fixtures, fonctions, dans les fonctions 'autres fonctions, context manager, etc.

À noter que ce qui me heurte vraiment est la taille et la structure de l'affaire, davantage que l'idée de tester plusieurs trucs à la fois (même si je ne suis pas fan et que ça fera un truc beaucoup trop pollué lors d'un futur debug).

À noter (bis) que déjà simplement la docstring est inmaintenable.

À reprendre, je dirais que si on veut des tests englobant tout avec plein de redondance, il faut plutôt multiplier les fixtures paramétriques. (ex: les tests sur l'API dans Chrono, ils sont courts, mais ils sont répétés dans différents fuseaux horaires, à différentes dates).

#14 - 04 février 2020 13:56 - Nicolas Roche

- Fichier 0001-tracking_code-add-tests-on-formdata-access-38073.patch ajouté

> multiplier les fixtures paramétriques

oui, ça permet de voir directement où les tests ne passent pas.

> si on veut des tests englobant tout avec plein de redondance

test_tracking_code.py deviendrait le 4ème test le plus long (56 secondes).

#15 - 04 février 2020 14:47 - Frédéric Péters

Faut laisser à l'ordinateur le soin de faire les combinaisons.

#16 - 04 février 2020 14:49 - Frédéric Péters

i.e.

Avoir is_draft, submitTer (deux T), is_front, assessor (?), owner, être chacuns des fixtures paramétriques.

#17 - 04 février 2020 18:59 - Nicolas Roche

J'ai mis "assessor" pour l'utilisateur qui accède à la demande (par opposition à "submitter")
C'est pas terrible, ni comme mot, ni comme traduction, mais je n'ai pas trouvé mieux.
https://fr.wiktionary.org/wiki/accesseur
http://www.granddictionnaire.com/ficheOqlf.aspx?Id_Fiche=17051753

> nested pytest.mark.parametrize

Je ne l'ai pas fait sur le premier lot car les résultats attendus ne me semblent pas factorisables en fonction des paramètres.

#18 - 04 février 2020 19:00 - Nicolas Roche

- Fichier 0001-tracking_code-add-tests-on-formdata-access-38073.patch ajouté

#19 - 04 février 2020 21:16 - Frédéric Péters

> J'ai mis "assessor" pour l'utilisateur qui accède à la demande (par opposition à "submitter")

Ailleurs dans les tests on met agent.

> Je ne l'ai pas fait sur le premier lot car les résultats attendus ne me semblent pas factorisables en fonction des paramètres.

Ça reste illisible, il faut vraiment trouver autre chose. Si ça simplifie tu peux te dire qu'on se fiche du cas particulier des admins. Tu peux aussi pointer des trucs qui sont incohérents, et ne pas faire un test compliqué pour des trucs incohérents.

Mais déjà, vu comment l'intérieur du test c'est juste quasi directement deux branches d'un "if is_draft", ça semble facilement pouvoir se couper en deux, un test pour quand c'est brouillon, un test pour quand c'est pas brouillon.

~~

Par ailleurs il reste des mélanges de bout de fonctions dans les fonctions, et j'ai l'impression que les tests en eux-mêmes donneraient l'impression d'être bien moins long si ce n'était pas le cas.

#20 - 05 février 2020 16:59 - Nicolas Roche

- Fichier 0001-tracking_code-add-tests-on-formdata-access-38073.patch ajouté

> "assesssor" : l'utilisateur qui accède à la demande

En fait je voulait dire l'utilisateur qui y ré-accède à la demande pour la compléter ou pour la suivre, via le code de suivi... mais ok pour agent.

> ça semble facilement pouvoir se couper en deux,

Oui (je trouvais ça intéressant d'avoir tout au même endroit : ça rejoignait l'idée d'avoir une matrice).

> on se fiche du cas particulier des admins.

Oui, via les test on voit qu'admin possède les mêmes droit que l'agent de traitement (ça me semblait intéressant de l'avoir parce que c'est notre configuration à nous en tant que collaborateur).

> ne pas faire un test compliqué pour des trucs incohérents.

Je pense que c'est quand même pratique pour éviter les régressions.
Mon idée ici serait que ces valeurs finissent par devenir cohérentes.

#21 - 05 février 2020 17:15 - Frédéric Péters

> > "assesssor" : l'utilisateur qui accède à la demande
>
> En fait je voulait dire l'utilisateur qui y ré-accède à la demande pour la compléter ou pour la suivre, via le code de suivi... mais ok pour agent.

Ok alors tout simplement user.

~~

Vraiment il faut trouver un truc pour cette longue liste; j'ai toujours du mal à comprendre ce que tout ça teste et j'espère enfin comprendre une fois arrivé au bout des simplifications.

Si je prends test_direct_access_to_draft

On a un mode login/deny/deny/deny/deny les trois fois, SAUF le dernier où on a login/FRONT/deny/deny/deny.

Je serais ainsi à dégager expected_access des fixtures, à avoir dans le code du test :

```
result = 'login' if user == 'anonymous' else 'deny'
if submitter == 'user1' and user == 'user1':
    result = 'front'  # user is allowed access to its draft
```

Réfléchir à un truc similaire pour test_direct_access_to_demand.

#22 - 05 février 2020 17:45 - Nicolas Roche

> j'ai toujours du mal à comprendre ce que tout ça teste

Ces tests sont dans un premier temps un outils pour mesurer les changement apportés par les patchs qui concerneront l'accès via code de suivi, et dans un second temps ils valideront (ou donneront les limites) de règles sur lesquelles on pourra s'appuyer, comme celle actuellement en vigueur : "Taper le code de suivi = devenir/être la personne à l'origine de la demande."

> je trouvais ça intéressant d'avoir tout au même endroit : ça rejoignait l'idée d'avoir une matrice

J'ai vraiment en tête de proposer des tests dont les valeurs attendues varieraient dans le temps, en fonctions de l'applications des patchs liés au ticket chapeau. Donc, mon idée c'est que ces valeurs (ou la matrice par abus de langage) soient faciles à faire évoluer. Or, si je factorise les valeurs attendues dans le tests, d'une certaine façon je les fige, ce qui est l'inverse de l'effet que je recherche. Une fois que ces valeurs attendues deviendront cohérentes alors, oui, cela prendra tout son sens de les factoriser.

#23 - 05 février 2020 18:04 - Frédéric Péters

Ce patch ne passera pas ainsi.

#24 - 05 février 2020 18:19 - Frédéric Péters

Vraiment je ne sais pas trop comment l'expliquer une nouvelle fois différemment, cette approche ne me convient pas; on ne fait comme ça nulle part ailleurs dans w.c.s.; faire un truc compliqué sans savoir où ça va.

Et peut-être prendre de là alors. Savoir/décider où aller. Faire les tests qui correspondent à ça. Et marquer en xfail là où la réalité ne correspond pas encore.

#25 - 05 février 2020 18:44 - Nicolas Roche

Ok, compris.
C'est juste un outil dont j'ai besoin pour comprendre où je met les pieds et pour pouvoir communiquer sur les patchs proposés (parce que je n'ai pas de vision d'ensemble sur ces mécanismes).
Je continuerai à utiliser ma branche locale pour rebaser les tickets, et donc s'il n'y a que moi qui en voit l'utilité, autant ne pas les pousser dans master.
Après, comme tu dis, on pourra (ou pas) les intégrer quand on saura vers où aller.

#26 - 06 octobre 2020 11:26 - Frédéric Péters

- Statut changé de Solution proposée à Fermé

Je ferme tout ça parce que mélangé sur cinq tickets, se partageant plus de 50 commentaires, et que je ne vois pas où ça mène.

Fichiers

0001-tracking_code-add-tests-on-formdata-access-38073.patch	13,4 ko	29 novembre 2019	Nicolas Roche
0001-tracking_code-add-tests-on-formdata-access-38073.patch	15,3 ko	02 janvier 2020	Nicolas Roche
0001-tracking_code-add-tests-on-formdata-access-38073.patch	17,1 ko	07 janvier 2020	Nicolas Roche
0001-tracking_code-add-tests-on-formdata-access-38073.patch	17 ko	08 janvier 2020	Nicolas Roche
0001-tracking_code-add-tests-on-formdata-access-38073.patch	15,8 ko	04 février 2020	Nicolas Roche
0001-tracking_code-add-tests-on-formdata-access-38073.patch	14,3 ko	04 février 2020	Nicolas Roche
0001-tracking_code-add-tests-on-formdata-access-38073.patch	15,2 ko	05 février 2020	Nicolas Roche