Authentic 2 - Development #53902

avoir une classe de base pour les modèles de gestion des moyens d'authentification

10 mai 2021 16:29 - Serghei Mihai

Statut:	Fermé	**Début:**	10 mai 2021
Priorité:	Normal	**Echéance:**	
Assigné à:	Valentin Deniaud	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	Non

Description

Comme suggéré dans #39406#note-36, on doit avoir un modèle de base rassemblant les éléments communs de tous les blocs d'authentification:

- uuid
- nom
- slug
- ordre d'apparition sur la page de connexion
- condition d'affichage sur la page de connexion
- etc.

Il doit proposer les interfaces pour ajouter/édition/suppression (sur le modèle de BaseResource dans passerelle).

Et le modèles qui hériterons ajouterons les paramètres qui leur sont nécessaires.

Demandes liées:

Lié à Authentic 2 - Development #39406: Fournir dans le backoffice (/manage/)...	**Fermé**	**30 janvier 2020**	**05 octobre 2020**
Lié à Authentic 2 - Development #51362: permettre de définir la connexion à u...	**Fermé**	**23 février 2021**	
Lié à Authentic 2 - Development #65472: Crash migration sur mise à jour	**Fermé**	**19 mai 2022**	
Lié à Authentic 2 - Development #65504: auth_oidc : migration 0011 qui casse ...	**Fermé**	**20 mai 2022**	
Précède Authentic 2 - Development #65358: /manage/authenticators/, journalise...	**Fermé**	**11 mai 2021**	**11 mai 2021**
Précède Authentic 2 - Development #65360: /manage/authenticators/, ajouter l'...	**Fermé**	**11 mai 2021**	**11 mai 2021**

Révisions associées

Révision 8532ac64 - 19 mai 2022 16:25 - Valentin Deniaud

authenticators: add new app (#53902)

Révision 46c99d78 - 19 mai 2022 16:25 - Valentin Deniaud

authenticators: migrate login password authenticator (#53902)

Révision 2c6b3d2e - 19 mai 2022 16:25 - Valentin Deniaud

auth_oidc: migrate authenticator to database (#53902)

Révision 8eec403c - 19 mai 2022 17:09 - Valentin Deniaud

build: distribute src/authentic2/apps/authenticators/templates/ (#53902)

Historique

#1 - 10 mai 2021 16:29 - Serghei Mihai

- Lié à Development #39406: Fournir dans le backoffice (/manage/) des écrans de configuration de la gestion et de la fourniture des identités ajouté

#2 - 10 mai 2021 16:29 - Serghei Mihai

- Lié à Development #51362: permettre de définir la connexion à un idp saml externe via des modèles ajouté

#3 - 12 mai 2021 10:28 - Serghei Mihai

- Fichier 4.png ajouté

- Fichier 3.png ajouté

- Fichier 2.png ajouté

- Fichier 1.png ajouté

- Tracker changé de Support à Development

Premier jet poussé dans une wip.

Captures d'écran ilustrant l'interface telle que je l'imagine.

#4 - 12 mai 2021 10:52 - Serghei Mihai

- Fichier 6.png ajouté

- Fichier 5.png ajouté

#5 - 28 juin 2021 10:40 - Frédéric Péters

> Configure display order

Communément on fait du glisser/déposer des lignes, pas dans une popup.

~~

Communément aussi on marque les lignes désactivées en grisant, plutôt qu'avec un truc "bouton switch" affiché à droite.

#6 - 28 juin 2021 14:42 - Serghei Mihai

Frédéric Péters a écrit :

> Communément on fait du glisser/déposer des lignes, pas dans une popup.

J'ai gardé le principe proposé initialement dans #39406 (capture https://dev.entrouvert.org/attachments/43705)

> Communément aussi on marque les lignes désactivées en grisant, plutôt qu'avec un truc "bouton switch" affiché à droite.

Le bouton "switch" est aussi un bouton d'action pour des/activer le moyen d'authentification. Pour marquer un authenticator désactivé je me suis inspiré de Chrono ou on marque avec opacity: 0.7 les événements pour lesquelle les résas ne sont plus possibles.

Mais il y a sûrement d'autres manières, plus efficaces, de marquer un authenticator désactivé. Je suis preneur d'idées.

#7 - 28 juin 2021 15:21 - Frédéric Péters

> J'ai gardé le principe proposé initialement dans #39406 (capture https://dev.entrouvert.org/attachments/43705)

Oui, parce que déplacer de gros blocs est parfois inconfortable, ici ce n'est plus le cas.

> de marquer un authenticator désactivé

"Communément aussi on marque les lignes désactivées en grisant"; en effet dans chrono on touche à l'opacité, c'est une autre possibilité.

Mon propos concernait surtout l'icône inintelligible. Et je dégagerais aussi l'icône « oeil » que j'imagine faire la même chose qu'un clic ailleurs dans la ligne. (et si ça ne fait pas ça, je ne sais pas ce que ça fait). Ah à regarder les captures l'icône « oeil » serait "modifier la condition d'affichage". Je trouve donc cela confus, intégrerait l'édition de ce paramètre dans la même fenêtre que les autres.

#8 - 30 juin 2021 10:15 - Serghei Mihai

Frédéric Péters a écrit :

> > J'ai gardé le principe proposé initialement dans #39406 (capture https://dev.entrouvert.org/attachments/43705)
>
> Oui, parce que déplacer de gros blocs est parfois inconfortable, ici ce n'est plus le cas.

Effectivement.

de marquer un authenticator désactivé

> "Communément aussi on marque les lignes désactivées en grisant"; en effet dans chrono on touche à l'opacité, c'est une autre possibilité.
>
> Mon propos concernait surtout l'icône inintelligible. Et je dégagerais aussi l'icône « oeil » que j'imagine faire la même chose qu'un clic ailleurs dans la ligne. (et si ça ne fait pas ça, je ne sais pas ce que ça fait). Ah à regarder les captures l'icône « oeil » serait "modifier la condition d'affichage". Je trouve donc cela confus, intégrerait l'édition de ce paramètre dans la même fenêtre que les autres.

Je pensais à la base mettre des liens d'action explicites "activer", "desactiver", "éditer la condition d'affichage" mais après je me suis tourné vers des icônes (toujours inspiré de Chrono avec le picto pour charger un ICS avec des exceptions). Je vais revenir sur des liens explicites.

#9 - 22 septembre 2021 17:38 - Valentin Deniaud

- Assigné à mis à Serghei Mihai

J'ai testé et je trouve ça très bien, c'est un code plus court et beaucoup moins complexe que le journal (pour prendre l'exemple de la dernière grosse évolution), il n'y a vraiment rien qui bloque pour que ce soit mergé dans pas longtemps.

Le seul gros truc qui manque à mon sens (mais je débarque et je n'ai pas d'idée précise de ce qui s'est déjà dit/de la dernière version du plan), c'est que je m'attendais à arriver sur la branche et à faire tourner les migrations et pouf à voir les moyens d'authentification que j'avais configuré créés et bien rangés dans le manager. C'est du travail mais ça me paraît assez essentiel et pas inatteignable.

Pour que ce soit atteignable, je propose que ce ticket ajoute juste l'authentification par mot de passe, FC c'est vraiment pas la priorité. Donc :

- Premier commit qui ajoute les modèles/vues de base
- Deuxième commit qui ajoute l'authentification par mot de passe. En plus de ce qu'il contient déjà dans la version sur la branche :
  - Avoir une migration qui lit les settings puis crée le modèle qui va bien
  - Remplace complètement le code qui va regarder les settings A2_* pour le remplacer par des accès direct au modèle
    - (ie abandonner ce qui est fait dans « auth: read password auth settings from manager model » ou autour de la méthode get_authenticator_settings)

(il y a #15846 où je fais ça sur un cas très simple et qui me fait dire que c'est possible)

Et là dessus enchaîner tranquille avec des tickets pour FC et SAML sur le même modèle, et OIDC un cas particulier puisque déjà en db.

Sur l'interface actuelle, remarques :

- Plutôt qu'un enchaînement de popup à l'ajout, en avoir une seule avec un select (cf « Nouvelle plateforme de paiement » dans lingo)
- J'aurais bien aimé ne pas avoir de champ slug, je me retrouve à taper dedans le label slugifié et il faut que je lise le code pour savoir que je peux le laisser blanc et que ça fait ça. L'éditer depuis les options suffit largement (cf l'ajout d'un évènement dans chrono)
- Je n'ai pas trouvé de bouton pour supprimer, mais je comprends que ça a peu de sens pour mot de passe et FC, ça pourra être ajouté quand il y en aura besoin sans problème.

#10 - 23 septembre 2021 09:15 - Serghei Mihai

- Statut changé de Nouveau à En cours

Valentin Deniaud a écrit :

> Le seul gros truc qui manque à mon sens (mais je débarque et je n'ai pas d'idée précise de ce qui s'est déjà dit/de la dernière version du plan), c'est que je m'attendais à arriver sur la branche et à faire tourner les migrations et pouf à voir les moyens d'authentification que j'avais configuré créés et bien rangés dans le manager. C'est du travail mais ça me paraît assez essentiel et pas inatteignable.

Il manque ça en effet. J'ai prévu de créer une migration pour créer les modèles pour les authentifications existantes. Mais je voulais déjà avoir un retour sur l'interface.
Je rajoute ça donc.

Comme FC est présent presque partout, autant le rajouter asap et programmer la gestion de la chose dans Hobo.

> Sur l'interface actuelle, remarques :
>
> - Plutôt qu'un enchaînement de popup à l'ajout, en avoir une seule avec un select (cf « Nouvelle plateforme de paiement » dans lingo)

D'acc.

> - J'aurais bien aimé ne pas avoir de champ slug, je me retrouve à taper dedans le label slugifié et il faut que je lise le code pour savoir que je peux le laisser blanc et que ça fait ça. L'éditer depuis les options suffit largement (cf l'ajout d'un évènement dans chrono)

Je vais le flinguer.

- Je n'ai pas trouvé de bouton pour supprimer, mais je comprends que ça a peu de sens pour mot de passe et FC, ça pourra être ajouté quand il y en aura besoin sans problème.

On rajoutera la suppression dans un second temps.

#11 - 23 septembre 2021 10:06 - Benjamin Dauvergne

BaseAuthenticatorManager ne doit pas être abstract, ça n'apporte rien, on peut utiliser l'héritage Django ici, ça évite de jouer avec *subclasses* et d'avoir à gérer l'unicité globale du slug et du nom.

Ce modèle devrait avoir les propriétés suivantes :

- attribut ou, FK vers OrganizationalUnit , nullable, la plupart des backends autre que password pouvant être liés à une OU (et dans le futur lié Password à une ou pourra avoir une utilité).
- attribut manager_ct comme FK vers le content_type de la classe de base, ça doit permettre étant donné cet id de trouver facilement le Manager réel, à initialiser dans *init*: self.backend_ct = ContentType.objects.get_for_model(self) et une méthode @
  def real_manager():
  real_manager_ct = ContentType.objects.get_for_id(self.manager_ct, for_concrete_model=False)
  real_manager_model_class = real_manager_ct.model_class()
  if isinstance(self, real_manager_model_class):
  return self
  concrete_manager_ct = ContentType.objects.get_for_id(self.manager_ct)
  concrete_manager_model_class = concrete_manager_ct.model_class()
  if isinstance(self, concrete_manager_model_class):
  new_instance = copy.copy(self)
  new_instance.__class__ = concrete_manager_model_class
  return new_instance
  return concrete_manager_model_class.objects.get(pk=self.pk)

- attribut settings = JSONField(..., null=True); on a pas vraiment besoin d'un modèle par backend, pour les plus simples on doit pouvoir s'en sortir avec un modèle class Meta: proxy = True qui n'implémente que des éléments statiques comme la classe du formulaire d'édition, et c'est utile durant la transition où on gère juste l'ordre
- attribut autenticator_id = TextField(...); on y met authenticator.id pendant la transition, mais ça reste utile parce que ça rend la table directement lisible, avec un nouveau get_authenticator_class(...) :

```
def get_authenticator_class(self):
    if getattr(self, 'authenticator_class', None):
        return self.authenticator_class
    return utils.get_authenticator_class_by_id(self.authenticator_id)
```

- authentic2/manager/authenticators_views.py : y ajouter un log des actions pour le journal,
- authentic2/manager/urls.py : il n'y a pas de contrôle d'accès par défaut sur les vue du manager (on demande juste un utilisateur connecté), soit tu peux le faire dans les vues comme c'est fait ailleurs mais c'est un peu lourd, mais nécessaire quand ça doit être fin; soit rajouter un required(is_superuser) (warning pseudocode) pour l'instant dans un authenticator_url_patterns, ou alors envisager un manager_urls.py dans la nouvelle application authenticators (ce serait bien de migrer vers ce modèle de loger les pages managers des applications dans leur code, de ne plus tout concentrer dans authentic2/manager/) et d'y déplacer les vues aussi.

Valentin Deniaud a écrit :

> Le seul gros truc qui manque à mon sens (mais je débarque et je n'ai pas d'idée précise de ce qui s'est déjà dit/de la dernière version du plan), c'est que je m'attendais à arriver sur la branche et à faire tourner les migrations et pouf à voir les moyens d'authentification que j'avais configuré créés et bien rangés dans le manager. C'est du travail mais ça me paraît assez essentiel et pas inatteignable.

Je pense qu'on pourrait avoir un premier commit où on a juste l'infra de la nouvelle application authenticators et uniquement AUTH_FRONTENDS qui soit migré dans les modèles, ensuite on peut avoir un commit par backend qui bouge tout dans un nouveau modèle, ou dans l'attribut settings si ça parait suffisant

> Pour que ce soit atteignable, je propose que ce ticket ajoute juste l'authentification par mot de passe, FC c'est vraiment pas la priorité. Donc :
>
> - Premier commit qui ajoute les modèles/vues de base
> - Deuxième commit qui ajoute l'authentification par mot de passe. En plus de ce qu'il contient déjà dans la version sur la branche :
> - Avoir une migration qui lit les settings puis crée le modèle qui va bien
> - Remplace complètement le code qui va regarder les settings A2_* pour le remplacer par des accès direct au modèle

On doit vraiment migrer AUTH_FRONTENDS avant de migrer le reste; on peut bien sûr migrer les backends Password et FC tout de suite aussi. OIDC on aura intérêt à migrer OIDCProvider pour le faire hériter de AuthenticationBaseManager, ce sera un ticket un peu plus compliqué.

- J'aurais bien aimé ne pas avoir de champ slug, je me retrouve à taper dedans le label slugifié et il faut que je lise le code pour savoir que je peux le laisser blanc et que ça fait ça. L'éditer depuis les options suffit largement (cf l'ajout d'un évènement dans chrono)

Oui, il y a authentic2.manager.forms.SlugMixin qui peut être utilisé pour ça (il y a juste #57138 à corriger avant, mais visiblement c'est en cours)

#12 - 20 janvier 2022 11:22 - Valentin Deniaud

- Assigné à changé de Serghei Mihai à Valentin Deniaud

#14 - 07 avril 2022 18:54 - Valentin Deniaud

Petites questions et remarques pour m'éviter de partir dans la mauvaise direction. J'ai parcouru les bouts de discussion disséminés à droite à gauche mais ne pas hésiter à répéter ce qui a déjà été dit si besoin.

Benjamin Dauvergne a écrit :

> BaseAuthenticatorManager ne doit pas être abstract, ça n'apporte rien, on peut utiliser l'héritage Django ici, ça évite de jouer avec *subclasses* et d'avoir à gérer l'unicité globale du slug et du nom.

Soit, mais il y a des endroits où il y aura toujours besoin d'aller voir les subclasses, genre pour avoir la liste des moyens d'authentification qu'on peut ajouter.

> - attribut manager_ct comme FK vers le content_type de la classe de base, ça doit permettre étant donné cet id de trouver facilement le Manager réel

Très concrètement, mettons qu'on aille voir sur la page d'édition d'un moyen d'authentification, /authenticators/1/edit/, ta méthode permet de trouver la bonne classe depuis l'info pk=1, c'est ça ? Donc on admet d'avoir une requête à faire en plus à plein d'endroits où on doit manipuler un objet spécifique ?
On se différencie ici de passerelle qui va réussir à ne pas faire cette requête en plus en incluant un pointeur vers la bonne classe dans l'URL.
Et la méthode basée sur le ContentType a l'air assez atypique, sur internet les gens ont l'air de préférer une bidouille de ce style https://jeffelmore.org/2010/11/11/automatic-downcasting-of-inherited-models-in-django/.

> - attribut settings = JSONField(..., null=True); on a pas vraiment besoin d'un modèle par backend, pour les plus simples on doit pouvoir s'en sortir avec un modèle class Meta: proxy = True qui n'implémente que des éléments statiques comme la classe du formulaire d'édition, et c'est utile durant la transition où on gère juste l'ordre

Pour moi ça c'est non, tout l'intérêt de ce qu'on fait ici c'est pour nous simplifier la vie à configurer les raccordements, avoir un formulaire avec plein de validation et de textes d'aide, il faut donc totalement s'interdire un JSONField (ce qui fait qu'il n'y aura jamais besoin de modèles proxy, ça nuance peut-être l'intérêt de ne pas utiliser abstract=True).

> - attribut autenticator_id = TextField(...); on y met authenticator.id pendant la transition, mais ça reste utile parce que ça rend la table directement lisible, avec un nouveau get_authenticator_class(...) :
> [...]

Là j'ai du mal à comprendre quand est-ce qu'on utiliserait get_authenticator_class() par rapport à real_manager().

> Je pense qu'on pourrait avoir un premier commit où on a juste l'infra de la nouvelle application authenticators et uniquement AUTH_FRONTENDS qui soit migré dans les modèles

Tu peux expliciter ce tu entends par « migrer AUTH_FRONTENDS » ? Est-ce qu'on veut se débarrasser des classes genre LoginPasswordAuthenticator, et donc bouger ses méthodes login, profile etc dans notre nouveau modèle ?

#15 - 14 avril 2022 17:46 - Valentin Deniaud

Démo vite fait : https://perso.entrouvert.org/~vdeniaud/authenticators.mp4

Le code sur la branche est bien avancé, j'ai décidé comme un grand pour les questions que je posais plus haut, plus besoin d'y répondre.

À noter :

- J'ai choisi de migrer deux moyens d'authentification, le login par mot de passe et les clients OIDC
  - C'est le minimum pour pouvoir tester toutes les fonctionnalités de la classe de base
- Le mot de passe est spécial, il est créé par une migration et on ne peut pas en ajouter d'autre
  - Je n'ai pas migré tous les settings qui tournent autour, ça aussi on le fera au fil de l'eau et on ne veut peut-être pas tout dans l'interface (ou alors ça peut être l'occaz de simplifier, je pense aux quatre paramètres différents pour contrôler le nombre d'essais de mot de passe autorisé)
- La gestion de l'ordre ne sera satisfaisante que quand tout sera migré (on pourra alors faire un joli truc par glisser/déposer comme imaginait Serghei)

#16 - 19 avril 2022 18:14 - Valentin Deniaud

- Fichier 0004-auth_oidc-migrate-authenticator-to-database-53902.patch ajouté

- Fichier 0002-authenticators-add-new-app-53902.patch ajouté

- Fichier 0001-tests-use-serialized_rollback-with-transactional-db-.patch ajouté

- Fichier 0003-authenticators-migrate-login-password-authenticator-.patch ajouté

- Statut changé de En cours à Solution proposée

- Patch proposed changé de Non à Oui

Et voilà les patches.

À noter 0001 grosse galère: le LoginPasswordAuthenticator introduit dans 0003 est créé par une migration, or la fixture transactional_db wipe la db et on perd l'objet. Pour y remédier il y a une option serialized_rollback=True à mettre pour restaurer la base après le wipe. Mais Django n'arrive pas à déserialiser, c'est peut-être une bizarrerie authentic peut-être https://code.djangoproject.com/ticket/31051, j'ai abandonné et ça a donné ce patch.

#17 - 17 mai 2022 10:31 - Serghei Mihai

Dans 0002, pourquoi ne pas utiliser l'uuid au lieu de get_identifier ? et aussi dans {% url 'a2-manager-authenticator-detail' pk=authenticator.pk %} ?
Sinon pour le reste c'est ok pour moi.

#18 - 17 mai 2022 16:35 - Valentin Deniaud

Serghei Mihai a écrit :

> Dans 0002, pourquoi ne pas utiliser l'uuid au lieu de get_identifier ?

Parce que les uuid c'est pas beau, je préfère un truc lisible dans l'html.

> et aussi dans {% url 'a2-manager-authenticator-detail' pk=authenticator.pk %} ?

Pour avoir des urls jolies, et c'est comme ça qu'on fait partout.

J'imagine que les uuid ne seront utiles que pour l'import/export, on pourrait d'ailleurs ne les ajouter qu'à cette occasion.

#19 - 17 mai 2022 16:41 - Valentin Deniaud

- Précède Development #65358: /manage/authenticators/, journaliser les modifications ajouté

#20 - 17 mai 2022 16:42 - Serghei Mihai

- Statut changé de Solution proposée à Solution validée

Valentin Deniaud a écrit :

> Serghei Mihai a écrit :
>
> > Dans 0002, pourquoi ne pas utiliser l'uuid au lieu de get_identifier ?
>
> Parce que les uuid c'est pas beau, je préfère un truc lisible dans l'html.

Ok.

> J'imagine que les uuid ne seront utiles que pour l'import/export, on pourrait d'ailleurs ne les ajouter qu'à cette occasion.

C'est bien de les avoir dès maintenant.
En tout cas go pour moi.

#21 - 17 mai 2022 16:46 - Valentin Deniaud

- Précède Development #65360: /manage/authenticators/, ajouter l'import/export ajouté

#22 - 17 mai 2022 17:21 - Valentin Deniaud

- Fichier 0003-auth_oidc-migrate-authenticator-to-database-53902.patch ajouté

- Fichier 0002-authenticators-migrate-login-password-authenticator-.patch ajouté

- Fichier 0001-authenticators-add-new-app-53902.patch ajouté

- Statut changé de Solution validée à Solution proposée

Merci pour le ack mais malheureusement pour moi Benj m'avait fait des remarques sur jabber et je ne les avais pas encore appliquées.

Revoici les patches avec les (petites) modifications suivantes :

- Unicité globale du slug au lieu d'une unicité (slug, ou) qui n'apportait rien.
- Gestion spécifique de LoginPasswordAuthenticator pour toujours s'assurer qu'il y en est un, ça signifie taper un get_or_create explicite, et ça permet de dégager l'adaptation des tests de 0001.

#23 - 19 mai 2022 09:19 - Serghei Mihai

- Statut changé de Solution proposée à Solution validée

#24 - 19 mai 2022 09:46 - Benjamin Dauvergne

- Statut changé de Solution validée à Solution proposée

Il faudrait des tests des migrations, elles sont un peu critiques.

#25 - 19 mai 2022 13:46 - Valentin Deniaud

- Fichier 0003-auth_oidc-migrate-authenticator-to-database-53902.patch ajouté

- Fichier 0002-authenticators-migrate-login-password-authenticator-.patch ajouté

- Fichier 0001-authenticators-add-new-app-53902.patch ajouté

Revoilà avec des tests sur les migrations, j'ai dû inventer un AlterModelBase pour le modèle OIDCProvider sinon le MigrationExecutor était perdu.

#26 - 19 mai 2022 16:05 - Benjamin Dauvergne

- Statut changé de Solution proposée à Solution validée

À pousser vite que ça tourne un peu.

#27 - 19 mai 2022 16:25 - Valentin Deniaud

- Statut changé de Solution validée à Résolu (à déployer)

```
commit 037ec9f742b14b30377ec83e8a395c98a492ad15
Author: Valentin Deniaud <vdeniaud@entrouvert.com>
Date:   Wed Apr 13 16:28:13 2022 +0200

    auth_oidc: migrate authenticator to database (#53902)

commit 5c83646c681f31c7f986bdb8e8bd67432c5aef47
Author: Valentin Deniaud <vdeniaud@entrouvert.com>
Date:   Wed Apr 13 13:58:40 2022 +0200

    authenticators: migrate login password authenticator (#53902)

commit 435e8a877d8c7f0dbf39473acedad41c99360dc3
Author: Valentin Deniaud <vdeniaud@entrouvert.com>
Date:   Wed Apr 13 13:58:05 2022 +0200

    authenticators: add new app (#53902)
```

#28 - 19 mai 2022 18:53 - Benjamin Dauvergne

- Lié à Development #65472: Crash migration sur mise à jour ajouté

#29 - 19 mai 2022 21:14 - Transition automatique

- Statut changé de Résolu (à déployer) à Solution déployée

#30 - 20 mai 2022 12:15 - Guillaume Baffoin

- Lié à Development #65504: auth_oidc : migration 0011 qui casse la clé étrangère des mapping de claim vers le fournisseur (?) ajouté

#31 - 24 juillet 2022 04:42 - Transition automatique

Automatic expiration

Fichiers

4.png	30,5 ko	12 mai 2021	Serghei Mihai
3.png	67,2 ko	12 mai 2021	Serghei Mihai
2.png	38,4 ko	12 mai 2021	Serghei Mihai
1.png	24,9 ko	12 mai 2021	Serghei Mihai
5.png	29,6 ko	12 mai 2021	Serghei Mihai
6.png	53,4 ko	12 mai 2021	Serghei Mihai
0004-auth_oidc-migrate-authenticator-to-database-53902.patch	25,2 ko	19 avril 2022	Valentin Deniaud
0002-authenticators-add-new-app-53902.patch	35,5 ko	19 avril 2022	Valentin Deniaud
0001-tests-use-serialized_rollback-with-transactional-db-.patch	4,44 ko	19 avril 2022	Valentin Deniaud
0003-authenticators-migrate-login-password-authenticator-.patch	34,1 ko	19 avril 2022	Valentin Deniaud
0003-auth_oidc-migrate-authenticator-to-database-53902.patch	25,3 ko	17 mai 2022	Valentin Deniaud
0002-authenticators-migrate-login-password-authenticator-.patch	35,6 ko	17 mai 2022	Valentin Deniaud
0001-authenticators-add-new-app-53902.patch	33,5 ko	17 mai 2022	Valentin Deniaud
0003-auth_oidc-migrate-authenticator-to-database-53902.patch	28,2 ko	19 mai 2022	Valentin Deniaud
0002-authenticators-migrate-login-password-authenticator-.patch	37 ko	19 mai 2022	Valentin Deniaud
0001-authenticators-add-new-app-53902.patch	34,1 ko	19 mai 2022	Valentin Deniaud