IMA as a service - Service Urgence Habitation

Plateforme de services IMA

Ce document est un support à l'intégration de l'api ima-as-a-service et du widget de confirmation de commande associé, dans le cadre de l'exposition par un partenaire du service Urgence Habitation de la Plateforme de services IMA.

swagger : https://developers.ima.eu/api/ima-as-a-service

Remarque : La partie api management n'est pas abordée et doit être gérée avec le portail développeur (https://developers.ima.eu/).

Table des matières

Préambule 2

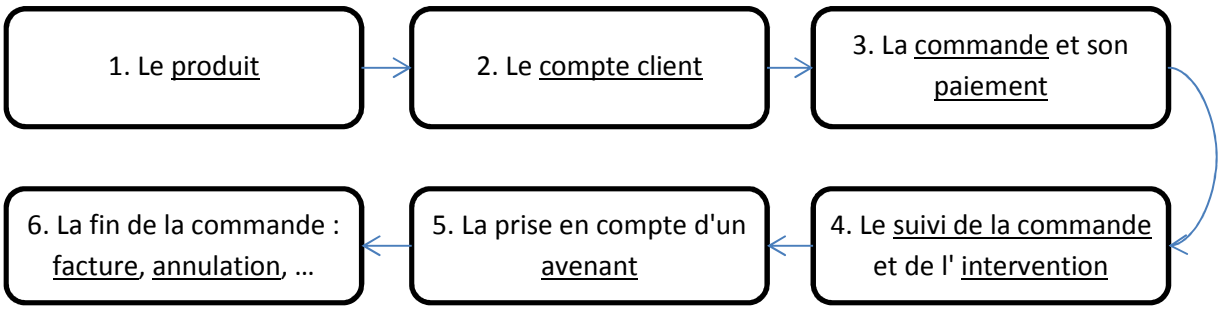
1. Le produit 3
2. Le compte client 4
3. La commande et son paiement 6
4. Le suivi de la commande 9
5. La prise en compte d'un avenant 15
6. La fin de la commande : facture, annulation, … 18
7. Annexe : Widget de confirmation de commande 20

Suivi des versions

Version	Date	Contenu
1.0	octobre 2021	Initialisation du document.
1.1	octobre 2021	Ajout du préambule. Version finalisée des enrichissements de l'api évoqués dans la version v1.0 : - prise en compte de la langue pour récupérer le nom du produit et les CGV ; - transmission de la langue du client ; - informations locataire/propriétaire et ancienneté de l'habitation ; - prise en compte de l'ancienneté de l'habitation lors de l'estimation tarifaire et de la commande ; - callback lorsque la facture est disponible. Renommage des données liées aux CGV pour uniformiser en 'TermsOfSale'. Rq : Ces évolutions seront déployées en production le 01/12/2021.
1.2	19/11/2021	Correction sur la donnée 'Locataire/Propriétaire' d'une habitation : $.ownerOrTenant au lieu de $.tenantOwner.

Préambule

Ce document a pour objectif d'apporter un éclairage sur l'utilisation de l'api ima-as-a-service tout au long d'une commande d'un service Urgence Habitation, en allant de la récupération des informations du produit commandé jusqu'à la fin de la commande. Pour cela, le document suit les grandes étapes proposées dans la figure ci-après.



Pour faciliter la lecture du document et le parallèle avec l'api, les correspondances suivantes mettent en vis-à-vis la terminologie de l'api, en anglais, et celle utilisée dans le document, en français.

Terminologie du document	Terminologie API
produit	product
devis (ou estimation tarifaire)	estimation price
compte client	account
habitation	house
commande	demand
intervention	operation
prestataire	provider
avenant (ou devis complémentaire)	amendment

1. Le produit

Catalogue de produits

L'api retourne l'ensemble des produits de la Plateforme de Services configurés dans votre catalogue. Le nom du produit est retourné dans la langue demandée dans le header de la requête (si la langue demandée n'est pas gérée, une langue par défaut est appliquée). Certains filtres sont disponibles (univers, famille, produits) ainsi que quelques critères de tri (par nom, id, …). En indiquant la géolocalisation du lieu d'intervention, vous avez une information sur la disponibilité géographique de chacun des produits.

Quels produits du service Urgence Habitation sont disponibles à la position GPS [+50.85450,+04.33079] ?

header.lang = 'fr'

GET /products?service=DEPANNAGE%20HABITATION&showavailabilityatlocation=50.85450,4.33079

```
[
  {
    "id": "DEPANNAGE CHAUFFAGE 1",
    "universe": "HABITATION",
    "service": "DEPANNAGE HABITATION",    ← filtre demandé dans la requête
    "family": "DEPANNAGE CHAUFFAGE",
    "name": "Recherche de fuite et remplacement de joints de robinet sur radiateur",  ← selon header.lang
    "configuration": {
      "unit": "INTERVENTION"
    },
    "geographicAvailability": "POTENTIALLY_AVAILABLE"    ← produit "DEPANNAGE CHAUFFAGE 1" disponible
  },
  …
]
```

Fiche produit

Sur la fiche produit, l'api donne des informations à prendre en compte dans le formulaire de commande d'un produit, ainsi que le document pdf des CGV associées au produit dans la langue demandée dans le header (si la langue demandée n'est pas gérée, une langue par défaut est appliquée). Sur les produits du service Urgence Habitation, un élément important est le délai maximum en nombre de jours dans lequel le rendez-vous peut-être demandé. Ce délai s'exprime en nombre de jours, sans distinction entre les jours ouvrés ou non-ouvrés.

Les détails du produit "Recherche de fuite et remplacement de joints de robinet sur radiateur"

header.lang = 'fr'

GET /products/DEPANNAGE%20CHAUFFAGE%201

```
{
  "id": "DEPANNAGE CHAUFFAGE 1",
  "universe": "HABITATION",
  "service": "DEPANNAGE HABITATION",
  "family": "DEPANNAGE CHAUFFAGE",
  "linkTermsOfSale": "https://assets.ctfassets.net/...../CGV_Urg_Habitation_fr.pdf",  ← selon header.lang
  "name": "Recherche de fuite et remplacement de joints de robinet sur radiateur",  ← selon header.lang
  "configuration": {
    "unit": "INTERVENTION",
```

```
    "maximumInterventionTime": 5,     ← ce produit ne peut pas être commandé pour une intervention
                                        au-delà de 5 jours (week-end et jours fériés compris)
    "maximumQuantityAllowed": 1,
    "minimumQuantityAllowed": 1,
    "availability": "24/7"
  }
}
```

Devis (ou estimation tarifaire)

Pour un produit donné, l'api retourne le devis TTC d'une intervention à partir des informations transmises : la date et heure souhaitée, la localisation, et l'ancienneté de l'habitation.
L'ancienneté de l'habitation est obligatoire en Belgique car elle conditionne le taux de TVA, et donc le prix du service, selon que l'habitation ait plus ou moins de 10 ans. Elle n'est pas nécessaire en France.

Un devis pour une intervention de "Recherche de fuite et remplacement de joints de robinet sur radiateur", le 08/10 à 10H30, à la position GPS [+50.85450,+04.33079], pour une habitation de plus de 10 ans

```
POST /products/ DEPANNAGE%20CHAUFFAGE%201/estimation-price
{
  "desiredDateAndTime": "2021-10-08T10:30:00+02:00",
  "location":{
    "position":{
      "latitude":"50.8545089",
      "longitude":"4.3307959"
    }
  },
  "additionalInformations":{
    "propertyAge": "10-999"    ← ancienneté de l'habitation, obligatoire en Belgique, conditionne le taux de TVA
  }
}
```

```
{
  "unitPrice": "375.00",          ← le montant du devis ici est de 375€ par intervention
  "unit": "INTERVENTION",
  "productId": "DEPANNAGE CHAUFFAGE 1",
  "currency": "EUR",
  "includingTaxes": true
}
```

2. Le compte client

Compte client

La création du compte client est une étape obligatoire avant de créer une nouvelle commande : vous devez transmettre à l'api les informations de votre client (à minima celles qui nous sont nécessaires pour opérer le service), et conserver l' identifiant **$.id** qui est retourné.

Remarque : La création du compte client auprès de la Plateforme de services est décorrélée de la création du compte de connexion de votre client sur votre site ou votre application.

Voici les données nécessaires à la Plateforme de Services pour le service Urgence Habitation (les données obligatoires sont soulignées) :

- civilité, nom et prénom (seront transmis au prestataire)
- téléphone (sera transmis au prestataire)
- email **(doit être unique)**
- adresse complète (sera l'adresse de facturation du client)

Création du compte de votre client "Jean Peeters" au sein de la Plateforme de services

```
POST /accounts
{
  "salutation": "Mr.",
  "lastname": "Peeters",
  "firstname": "Jean",
  "phone": "0470123456",          ← vous pouvez également transmettre l'indicatif "+32470123456"
  "email": "unclient@yopmail.com",  ← un email ne peut pas être partagé par 2 comptes client
  "address1": "Rue de Courtrai 45",
  "address2": "",
  "zipCode": "1080",
  "city": "Molenbeek-Saint-Jean",
  "countryCode": "BEL",
  "customerId": "0123456A",
  "technicalId": "6e959164806dbfd",
  "language": "nl"      ← langue du client, sera indiquée au prestataire (utile dans les pays multilingues)
}
```

← address1, zipCode, city et countryCode doivent être renseignés, soit dès la création du compte client, soit plus tard en modifiant le compte client, mais obligatoirement avant le passage de la commande

```
{
  "id": "uINjzKFwkrngWzAhZk",   ← identifiant du compte client, à conserver
  "salutation": "Mr.",
  "lastname": "Peeters",
  "firstname": "Jean",
  "phone": "0470123456",
  "email": "unclient@yopmail.com",
  "address1": "Rue de Courtrai 45",
  "address2": "",
  "zipCode": "1080",
  "city": "Molenbeek-Saint-Jean",
  "countryCode": "BEL",
  "customerId": "0123456A",
  "technicalId": "6e959164806dbfd",
  "language": "nl"      ← une langue par défaut est appliquée si la langue transmise n'est pas gérée
}
```

Habitation

Indépendamment de l'adresse de facturation, l'api propose d'enregistrer les habitations d'un compte client. Cela permet, par exemple, d'enrichir l'expérience utilisateur en proposant dans le tunnel de commande les adresses déjà créées afin de sélectionner le lieu d'intervention de la commande.

Ajout d'une habitation sur le compte client de Jean Peeters

```
POST /accounts/uINjzKFwkrngWzAhZk/houses
{
   "address1": "Rue Mavis 7",
   "address2": "",
   "zipCode": "4460",
   "city": "Grâce-Hollogne",
   "countryCode": "BEL",
   "ownerOrTenant": "TENANT",
   "propertyAge": "0-5"      ← l'ancienneté de l'habitation est transmise en intervalle ou en valeur numérique
}
```

```
{
   "id": "IJZ98466JBJBjhvlh54",
   "address1": "Rue Mavis 7",
   "address2": "",
   "zipCode": "4460",
   "city": "Grâce-Hollogne",
   "countryCode": "BEL",
   "ownerOrTenant": "TENANT",
   "propertyAge": "0-9",    ← attention, l'ancienneté retournée peut être un intervalle différent, cette information étant
                               gérée actuellement uniquement en plus ou moins de 10 ans dans notre système
   "isBillingAddress": false
}
```

Nous verrons dans la suite du document que l'habitation peut être transmise lors du passage de la commande (cf. §3).

IMPORTANT : L'adresse de facturation du client est créée en tant qu'habitation lors de la création du compte client, et on accède à son identifiant en récupérant les habitations du compte client avec withbillingaddress à true. Malgré tout, il est préférable de ne pas manipuler l'adresse de facturation comme une habitation, en particulier si vous liez les habitations aux commandes : si l'intervention a lieu à la même adresse que l'adresse de facturation, il est préconisé de créer une habitation.

3. La commande et son paiement

Nouvelle commande

Les données à transmettre à l'api sont :

- l'identifiant du compte client, le compte client étant préalablement correctement renseigné, en particulier son adresse de facturation (cf. partie Compte client)
- les données nécessaires au calcul du tarif de la commande (cf. partie Devis)
- les autres données nécessaires à la mise en œuvre du service : pour le service Urgence Habitation, un 'contact sur place' doit être transmis au prestataire

La commande de Jean Peeters d' une intervention de "Recherche de fuite et remplacement de joints de robinet sur radiateur", le 08/10 à 10H30, à son adresse de facturation

```
POST /demands
{
   "desiredDateAndTime":"2021-10-08T10:30:00+02:00",      ← date et heure souhaitée
```

```
  "product":{
    "id":"DEPANNAGE CHAUFFAGE 1",          ← produit commandé
    "customerInformations" : [             ← informations complémentaires pour préciser la commande
      {
        "name" : "propertyAge",            ← ancienneté de l'habitation, à transmettre systématiquement en
        "value" : "10-999"                   Belgique même si $.propertyId est utilisée
      }
    ]
  },
  "customer":{
    "accountId":"uINjzKFwkrngWzAhZk"       ← identifiant du compte client
  },
  "contact" : {
    "lastname" : "Mon contact",            ← les informations du contact sur place sont transmises au prestataire ;
    "firstname" : "Sur place",               merci de renseigner par défaut les informations de contact du client
    "phone" : "0476543210"
  },
  "propertyId":"JFzAO1633198670335",       ← identifiant de l'habitation qui fait l'objet de la commande
  "locations":[{
    "address":"Rue de Courtrai 45",
    "zipCode":"1080",                      ← lieu de l'intervention, à transmettre dans tous les cas,
    "city":"Molenbeek-Saint-Jean",           même si $.propertyId est utilisée
    "countryCode":"BEL",
    "position": {
      "latitude":"50.8545089",             ← nous préconisons que vous nous envoyez également les
      "longitude":"4.3307959"                coordonnées GPS du lieu de l'intervention
    }
  }]
}
```

```
{
  "id": "OAMmx1633198757068",  ← identifiant de la commande, à utiliser pour obtenir un paymentToken
  "functionalId": "B21Q06451",     ← référence de la commande ; sera visible par votre client au moment
                                     du paiement, sur le ticket de caisse de la transaction bancaire
  "desiredDateAndTime": "2021-10-08T10:30:00+02:00",
  "locations": [
    {
      "role": "Lieu intervention",
      "address": "Rue de Courtrai 45",
      "zipCode": "1080",
      "city": "Molenbeek-Saint-Jean",
      "countryCode": " BEL",
      "position": {
        "longitude": "4.330796",
        "latitude": "50.854509"
      }
    }
  ],
  "contact": {
    "firstname": "Sur place",
    "lastname": "Mon contact",
    "phone": "0476543210"
  },
  "customer": {
    "accountId": "uINjzKFwkrngWzAhZk"
  },
  "propertyId":"JFzAO1633198670335",
  "status": "DEMAND_INITIATED",    ← statut initial de la commande
```

```
  "product": {
    "id": "DEPANNAGE CHAUFFAGE 1",
    "unitPrice": 135,
    "quantity": 1,        ← la quantité est optionnelle pour passer une commande, elle est égale à 1 par défaut
    "customerInformations" : [
      {
        "name" : "propertyAge",
        "value" : "10-999"
      }
    ]
  },
  "notes": []
}
```

Une modification dans les données de la commande ?

Puisque l'api ne permet pas encore de modifier une commande, nous préconisons que vous nous transmettiez la commande "le plus tard possible", soit juste avant son paiement. A ce moment-là, si dans votre tunnel de commande le client a la possibilité de revenir en arrière pour par exemple modifier l'heure de son rendez-vous, vous devrez nous transmettre ces modifications par une nouvelle commande.
Il ne sera pas nécessaire d'annuler la commande initiale, elle restera non traitée dans notre système.

Widget de confirmation de commande : accès au paiement

Une fois la commande créée, vous disposez de son identifiant, et vous allez pouvoir générer un *paymentToken*.

Générer un paymentToken pour utiliser le widget de confirmation de commande

```
POST /demands/OAMmx1633198757068/payment-page
{
    aucune donnée n'est nécessaire ; malgré tout, merci de transmettre à minima un body vide {}
}
```

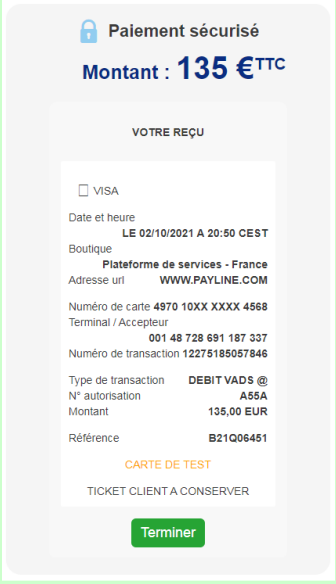
```
{
  "paymentToken": "7c8ad1ff3aa0c4f0bd1e988beac739582001b7ac8e8ade176c62adaa73d59e74",
  "expiresIn": 60
}
```

Pour afficher le widget de confirmation de commande afin que votre client puisse confirmer et payer sa commande, ce paymentToken doit être transmis à l'élément HTML "ImaPayment" *(cf. §7. Annexe : Widget de confirmation de commande)*.

Confirmation et paiement de la commande

```
<div id="ImaPayment"
   data-token="7c8ad1ff3aa0c4f0bd1e988beac739582001b7ac8e8ade176c62adaa73d59e74"
   lang="fr"              ← paramètre facultatif ; la langue du compte client est appliquée par défaut
   returnUrlSuccess       ← pas de returnUrlSuccess, le paiement ok est géré dans cet exemple avec paymentsuccessful
   returnUrlError="https://monurl/monid/payment-error"
   redirectTimeOut="0"
```

>



Rappel : La Plateforme de services n'assure aucune communication vers vos clients, ces communications sont assurées pour vous-mêmes afin de garantir leur homogénéité et leur cohérence. A l'issue du paiement, vous pouvez par exemple envoyer un mail de confirmation de commande. Les CGV peuvent être récupérées dans la fiche produit pour être jointes au mail si nécessaire (cf. § Fiche produit).

4. Le suivi de la commande

Aussitôt après son paiement, la commande d'un service d'Urgence Habitation est mise en œuvre par la Plateforme de services. L'intervention *(équivalent de 'operation' dans l'api)* qui avait été initiée dès la création de la commande et associée à celle-ci, porte ce processus.

1. Recherche automatique des prestataires (en France dans un rayon 30km) : jusqu'à 10 prestataires sont sollicités par sms et dans leur extranet (en France les prestataires ont 20mn pour répondre)
2. Missionnement du 1er prestataire qui accepte : ce prestataire est positionné sur l'intervention
3. Si aucun prestataire ne répond favorablement dans le temps imparti, prise en charge de l'intervention par un agent Plateforme de services **(en heures ouvrées uniquement)**
4. Confirmation de l'heure d'intervention par le prestataire dans son extranet suite à son appel vers le client
5. !! Réalisation de l'intervention !! ☺

A ce moment-là, le prestataire peut être amené à ajouter dans son extranet un avenant (ou devis complémentaire) à la commande initiale. Le client pourra alors accéder à la confirmation et au paiement de cet avenant (cf. § suivant).

6. Déclaration de la fin de l'intervention par le prestataire dans son extranet
7. Dépôt de la facture du client par le prestataire dans son extranet et validation de cette facture par un agent Plateforme de services

Certains de ces événements doivent obligatoirement vous être notifiés, car ils impliquent que vous préveniez votre client. Par exemple, dans le cas où aucun prestataire ne se positionne alors que l'on est en dehors des heures ouvrées, la commande ne peut pas être réalisée et le client doit être prévenu.

D'autres événements sont optionnels, selon l'expérience utilisateur que vous souhaitez mettre en place. Par exemple, vous pouvez choisir d'être alertés lorsque le prestataire a confirmé l'heure de rendez-vous dans son extranet, afin de déclencher une notification de votre application sur le mobile de votre client, mais ce n'est pas primordial.

Nous allons voir dans la suite du document :
- comment accéder aux données de l'intervention
- comment fonctionnent les callbacks qui vous alertent sur les événements

Les données de l'intervention

Directement au niveau de l'intervention, l'api retourne son statut, les informations du prestataire missionné (nom, adresse, téléphone, …), ainsi que les dates et heures demandées par le client d'une part, et celles confirmées par le prestataire d'autre part.

L'intervention peut être lue soit à partir de son identifiant (lorsque celui-ci est retourné par un callback par exemple, cf. plus bas), soit directement à partir de l'identifiant de la commande.

Les données de l'intervention de la commande de Jean Peeters pendant la recherche du prestataire

GET /demands/OAMmx1633198757068/operations

```
[      ← un tableau d'interventions est retourné, puisqu'une commande pourrait être associée à plusieurs
         interventions dans l'absolu ; néanmoins, la commande d'un service Urgence Habitation
         se traduit toujours par 1 et 1 seule intervention
  {
    "id": "UTtON1633198757088",     ← identifiant de l'intervention
    "status": "BEING_ASSIGNED",     ← dans ce statut, nous sommes en attente des réponses des prestataires
    "demandId": "OAMmx1633198757068",
    "dates": {
      "requested": "2021-10-08T08:30:00.000Z",
      "scheduled": null
    },
    "provider": null                ← aucun prestataire n'est missionné pour le moment
  }
]
```

Quant à la facture du client, elle sera accessible parmi les documents joints à l'intervention. Nous détaillerons comment l'api permet de l'uploader dans le § "6. La fin de la commande".

Le principe des callbacks

Lors d'un événement, nous vous transmettons une notification à l'url que vous nous aurez indiquée, avec :

- la nature du callback
- l'identifiant de l'élément concerné
- la nouvelle valeur de la donnée qui déclenche le callback

L'authentification auprès de votre système se fait en oAuth2 ou par API Key.

Exemple : callback vers votre système au moment du missionnement (operation.status devient "ACTIVE")

[POST | PUT] {votreUrlDeCallback} ← *Votre url de callback peut contenir l'identifiant de la commande*
*Ex : https://votreappli/**{demandId}**/notification*

```
{
  "notification":{
    "subject":"STATUS_CHANGED",
    "type":"operation",
    "operation":{
      "id":"UTtON1633198757088",
      "status":"ACTIVE"
    }
  }
}
```

← *ce callback est déclenché par le changement de statut d'une intervention, on transmet donc la ressource 'operation' avec l'identifiant de l'intervention et son nouveau statut*

On peut souligner que bien souvent, suite à un callback, vous serez amenés à utiliser l'api pour avoir une vue complète de l'événement, par exemple pour connaître l'ensemble des données à transmettre à votre client. L'identifiant communiqué dans le callback peut alors être utilisé directement.

Exemple : Lecture de l'intervention suite au callback pour récupérer les données du prestataire missionné

GET /operations/UTtON1633198757088 ← *identifiant de l'intervention fourni dans le callback*

```
{
  "id": "UTtON1633198757088",
  "status": "ACTIVE",
  "demandId": "OAMmx1633198757068",
  "dates": {
    "requested": "2021-10-08T08:30:00.000Z",
    "scheduled": null
  },
  "provider": {
    "name": "SOSTEAM-GROUP",
    "phone": "0470112233",
    "email": "emailpresta@yopmail.com"
  }
}
```

← *identifiant de la commande correspondante*

← *les informations du prestataire missionné*

La liste des callbacks disponibles

Le tableau ci-après liste les callbacks disponibles dans le contexte d'une commande Urgence Habitation. Ils sont de 4 natures.

- STATUS_CHANGED : déclenché sur un changement de statut
- FIELD_CHANGED : déclenché sur la modification d'une donnée
- PAYMENT_AUTHORIZED : déclenché lorsqu'un paiement est possible (*pour le service Urgence Habitation, ce dernier cas survient uniquement dans le cadre des avenants ; cf. §5)*
- DOCUMENTS_AVAILABLE : déclenché lorsque des documents sont disponibles

Nous pouvons activer / désactiver la plupart de ces callbacks en fonction de vos besoins. Cette configuration est à travailler pendant votre intégration de l'api.

Evénement	$.notification.subject	$.notification.type	Donnée principale du callback	Remarque
La commande est payée.	demand	STATUS_CHANGED	demand.status= DEMAND_PREPAID	Si le client ne clique pas sur le bouton Terminer du ticket de caisse Monext, ce callback vous permet d'être prévenu du paiement de la commande (cf. §7). **Attention, le statut DEMAND_PREPAID va disparaître et sera remplacé par des informations de paiement accessibles sur la commande.**
Des prestataires ont été trouvés et sollicités, le système est en attente de leurs réponses.	operation	STATUS_CHANGED	operation.status= BEING_ASSIGNED	

Evénement	$.notification.subject	$.notification.type	Donnée principale du callback	Remarque
Aucun prestataire n'a été trouvé, ou aucune sollicitation n'a été acceptée au bout du temps imparti. callback obligatoire	operation	STATUS_CHANGED	operation.status= SUPPORT	En heures ouvrées, un agent plateforme prend la main, il n'est pas nécessaire de prévenir votre client. En heures non ouvrées, une communication client est nécessaire ; l'agent plateforme se chargera du dossier à l'ouverture du service.
Un prestataire a été missionné sur l'intervention. callback conseillé	demand ---------- operation	STATUS_CHANGED ---------- STATUS_CHANGED	demand.status= ORDER_IN_PROGRESS ---------- operation.status= ACTIVE	2 callbacks : 1 au niveau demand, 1 au niveau operation. Il est possible d'activer les 2, mais également uniquement celui qui sera le plus pratique pour votre intégration. Suite à ce callback, les informations du prestataire missionné doivent être lues sur l'operation.
La date et heure de rdv souhaitée est modifiée.	operation	FIELD_CHANGED	operation.dates. requested= "2021-03-30T09:00:00Z"	Exceptionnellement, cette date pourrait être modifiée par un agent Plateforme de services.
La date et heure de rdv planifiée est saisie ou modifiée.	operation	FIELD_CHANGED	operation.dates. scheduled= "2021-10-08T09:00:00Z"	
Un avenant a été créé par le prestataire et doit être payé par le client. callback obligatoire	amendment	PAYMENT_ AUTHORIZED	amendment. paymentPage. paymenToken= "jefER4645676GBRZBvezztj43eg"	Les informations de l'avenant (montant et statut de l'avenant, tel et mail du client) peuvent être lues par la route GET/amendments/{amendmentId}. **Atttention, pas de callback disponible aujourd'hui dans l'api sur les changements de statut des avenants.**

Evénement	$.notification.subject	$.notification.type	Donnée principale du callback	Remarque
L'intervention est terminée, ce qui termine également la commande.	demand ---------- operation	STATUS_CHANGED ---------- STATUS_CHANGED	demand.status= ORDER_ENDED ---------- operation.status= ENDED	2 callbacks : 1 au niveau demand, 1 au niveau operation. Il est possible d'activer les 2, mais également uniquement celui qui sera le plus pratique pour votre intégration. **Attention, la facture n'a pas forcément été déposée dans le système par le prestataire à ce moment là.**
La facture client est disponible. callback systématique	operation	DOCUMENTS_ AVAILABLE	operation.documents[0].type=INVOICE operation.documents[0].id= a1V1q000000aUI3EAM	La facture peut ensuite être téléchargée par la route GET/operations/{operationId}/documents/{documentId}. Remarque : Callback générique pour tout type de document, mais disponible uniquement pour les factures dans la version actuelle de l'api.
La commande et l'intervention sont annulées. callback conseillé	demand ---------- operation	STATUS_CHANGED ---------- STATUS_CHANGED	demand.status= DEMAND_CANCELLED ---------- operation.status= CANCELLED	2 callbacks : 1 au niveau demand, 1 au niveau operation. Il est possible d'activer les 2, mais également uniquement celui qui sera le plus pratique pour votre intégration. Suite à ce callback, les informations de l'annulation (frais d'annulation et origine client/prestataire de l'annulation) peuvent être lues par la route GET/demands/{demandId}/cancellation.

Configuration des callbacks

En plus de la liste des événements à activer, voici les éléments qui devront nous être communiqués pour la configuration des callbacks. Les urls et credentials seront à transmettre pour chacun des environnements : recette, preprod, prod.

- Endpoint du callback

  - url : 1 unique url (par environnement) pour tous les callbacks ; cette url peut contenir {demandId} dans son path
  - method : POST / PUT
  - header : éléments à ajouter dans le header, en plus des éléments dynamiques de sécurisation

- Sécurisation de ce endpoint

  - En oAuth2
    - url : url (par environnement) appelée avant chaque callback pour récupérer un access token ; cette url est appelée avec le body "grant_type=client_credentials"
    - method : POST / PUT
    - header : éléments à ajouter dans le header ; pour oAuth2, le header contiendra de base "Content-Type: application/x-www-form-urlencoded"
    - client_id / client_secret : nos credentials d'accès à votre endpoint

  - En API Key :
    - key : clé/valeur de l'api key à ajouter dans le header du callback

Reprise sur incident

A l'heure actuelle les callbacks de l'api ne bénéficient pas d'une reprise sur incident automatique en cas d'erreur. Les erreurs sont détectées par une supervision et sont gérées manuellement.

5. La prise en compte d'un avenant

Nouvel avenant

Lorsque le prestataire saisit un avenant dans son extranet, un callback est déclenché pour vous alerter qu'un lien de paiement est à transmettre à votre client. Il contient un *paymentToken* qui permet d'utiliser le widget de confirmation de commande pour cet avenant.

Callback de nouvel avenant
[POST | PUT] {votreUrlDeCallback}

```
{
  "notification":{
    "subject":"PAYMENT_AUTHORIZED",
    "type":"amendment",
    "amendment":{
      "id":"czbdF1633279697020",
      "operationId":"UTtON1633198757088",
      "demandId":"OAMmx1633198757068",
      "paymentPage": {
        "paymentToken":"413490495b49df3f883a13d2d9e918e46ca77321da480bd731b62c49c3351175",
        "expiresIn":60
      }
    }
  }
}
```


La lecture de l'avenant est nécessaire pour récupérer les informations de l'avenant, notamment les informations de contact du client à privilégier pour envoyer le lien de paiement. Celles-ci ont été saisies par le prestataire et peuvent être différentes de vos données client. En effet, au moment de l'ajout de l'avenant, le prestataire et le client peuvent être physiquement ensemble sur le lieu de l'intervention, et le client a pu communiquer un numéro de téléphone autre que le sien si par exemple ce n'est pas lui qui paie directement l'avenant.

Lecture de l'avenant de 115€ ajouté sur la commande de Jean Peeters

```
GET /amendments/czbdF1633279697020        ← identifiant de l'avenant fourni dans le callback
```


```
{
  "id": "czbdF1633279697020",
  "amount": 115,
  "functionalId": "A21Q06451AA",          ← référence de l'avenant ; sera visible par votre client au moment du paiement, sur le ticket de caisse de la transaction bancaire
  "operationId": "UTtON1633198757088",
  "status": "VALIDATED",
  "createdDate": "2021-10-08T09:48:17.000Z",
  "customerContact": {
    "email": "unclient@yopmail.com",      ← les informations de contact client à privilégier si elles sont renseignées
    "phone": "0477332211"
  },
  "demandId": "OAMmx1633198757068"
}
```


Une fois les informations recueillies, vous devez prévenir votre client pour l'inviter à payer le devis complémentaire et lui donner accès à ce paiement. Si possible, la notification se fera à la fois par mail et sur le téléphone mobile (sms, notification dans votre application, …), afin que le client puisse réaliser le paiement "dans la foulée", surtout s'il est en ce moment même avec le prestataire.

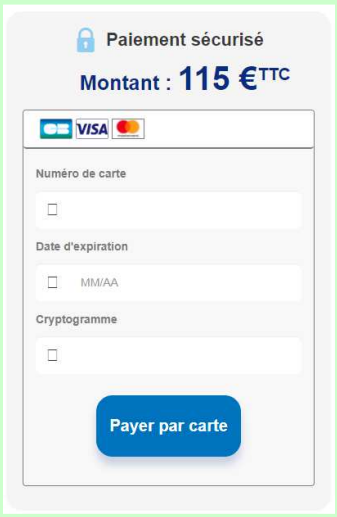
A noter que de notre côté, le prestataire a été notifié par la Plateforme de services que son avenant a bien été pris en compte et qu'une notification a été envoyée au client.

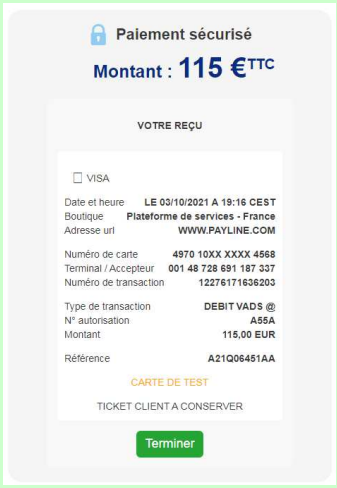
Paiement de l'avenant dans le widget de confirmation de commande

Pour afficher le widget de confirmation de commande afin que votre client puisse confirmer et payer son avenant, vous pouvez utiliser le *paymentToken* fourni par le callback et le passer à l'élément HTML "ImaPayment" *(cf. §7. Annexe : Widget de confirmation de commande)*.

Confirmation et paiement de l'avenant

```
<div id="ImaPayment"
   data-token="413490495b49df3f883a13d2d9e918e46ca77321da480bd731b62c49c3351175"
   lang="fr"
   returnUrlSuccess
   returnUrlError="https://monurl/monid/payment-error"
   redirectTimeOut="0"
>
```





Dans la mesure où le paymentToken a une durée de validité limitée, vous pouvez également générer un nouveau paymentToken au moment même de l'affichage du widget de confirmation de commande.

Générer un paymentToken pour utiliser le widget de confirmation de commande pour un avenant

```
POST /amendments/czbdF1633279697020/payment-page
{
    aucune donnée n'est nécessaire dans notre contexte ; malgré tout, merci de transmettre à minima un body vide {}
}
```

```
{
  "paymentToken": "cb840fc83610fde1f947eda8e277d70a4b13a3980d8a607567914abb2a7fdbcd",
  "expiresIn": 60
}
```

Suite au paiement, le prestataire sera notifié par la Plateforme de services que le devis complémentaire a été réglé par le client. Vous devrez prendre en charge la confirmation de paiement auprès de votre client.

6. La fin de la commande : facture, annulation, …

La fin de l'intervention

La fin de l'intervention est confirmée par le prestataire dans son extranet, ce qui modifie son statut. Vous pouvez en être informés par callback (cf. §4).

Le callback suite à la fin de l'intervention commandée par Jean Peeters

```
[POST | PUT]  {votreUrlDeCallback}
{
   "notification":{
      "subject":"STATUS_CHANGED",
      "type":"operation",
      "operation":{
         "id":"UTtON1633198757088",
         "status":"ENDED"
      }
   }
}
```

NB : Actuellement, la fin de l'intervention déclenche aussi la clôture de la commande. A terme il est possible que ces deux événements ne soient plus simultanés.

La facture du client

La facture du client est déposée par le prestataire dans son extranet (cela peut être fait même après la fin de l'intervention). La facture est ensuite vérifiée par un agent Plateforme de services. Suit à cette vérification, vous serez informés par callback que la facture du client est disponible.

Le téléchargement de la facture est alors possible par l'api grâce à l'identifiant de la facture forni dans les données du callback.

Callback de la disponibilité de la facture de Jean Peeters

```
[POST | PUT]  {votreUrlDeCallback}
{
   "notification":{
      "subject":"DOCUMENTS_AVAILABLE",
      "type":"operation",
      "operation":{
         "id":"UTtON1633198757088",
         "documents": [{
            "type": "INVOICE",
            "id": "a1V1q000000aUI3EAM",          ← identifiant du document de la facture
            "createdDate": "2021-10-09T12:33:55.000Z"
         }]
      }
   }
}
```

Le téléchargement de la facture de Jean Peeters

GET /operations/UTtON1633198757088/documents/a1V1q000000aUI3EAM


```
{
  "type": "INVOICE",
  "createdDate": "2021-10-09T12:33:55.000Z",
  "name": "facture_B21Q06451AA.pdf",
  "mimeType": "application/pdf",
  "size": 143201,
  "data": "{DataEnBase64}"
}
```


Annulation de la commande

Vous pouvez intégrer dans votre application la possibilité pour vos clients de demander l'annulation d'une commande qui a été payée, tant que la commande n'est pas clôturée ou déjà annulée.

Si Jean Peeters avait demandé l'annulation de sa commande …

POST /demands/OAMmx1633198757068/cancellation


```
{
  "origin": "CUSTOMER",
  "status": "IN_PROGRESS",   ← la demande d'annulation nous a été transmise et va être prise en charge
  "type": "CANCELLATION"
}
```


Un callback vous préviendra lorsque la demande d'annulation aura été traitée. Selon les cas, des frais d'annulation peuvent être retenus sur le paiement initial de la commande, les conditions d'annulation étant précisées dans les CGV. Il vous appartiendra de choisir comment vous souhaitez communiquer à votre client la bonne prise en compte de sa demande d'annulation.

Le callback suite à l'annulation effective de la commande

[POST | PUT] {votreUrlDeCallback}


```
{
  "notification":{
    "subject":"STATUS_CHANGED",
    "type":"demand",
    "demand":{
      "id":"czbdF1633279697020",
      "status":"DEMAND_CANCELLED"
    }
  }
}
```


La récupération des frais d'annulation à la charge de Jean Peeters

GET /demands/OAMmx1633198757068/cancellation


```
{
  "origin": "CUSTOMER",
  "status": "DONE",          ← la demande d'annulation a été traitée
  "fee": "60",               ← les frais d'annulation sont de 60 euros
  "type": "CANCELLATION"
}
```

7. Annexe : Widget de confirmation de commande

Description

Le widget se décompose en plusieurs écrans.

Ecrans du widget de confirmation de commande	Exemple	Considérations importantes
1. Récapitulatif de la commande ou **Devis complémentaire** (dans le cas d'un avenant)		Pour les commandes uniquement : - L'acceptation des CGV est obligatoire pour confirmer la commande avec le bouton "Commander" et accéder au paiement de celle-ci. - Le client peut cliquer sur le lien pour télécharger le pdf des CGV. - Puisque la commande se fait dans un contexte d'urgence, le renoncement au délai de rétractation n'est pas demandé.
2. Paiement de la commande dans le widget Monext	Le paiement est géré par le widget Monext (partie encadrée en jaune) qui est intégré dans le widget de confirmation de commande.	
2.a. Monext / Saisie des informations bancaires		- Vous pouvez utiliser les cartes bancaires de test de Monext : o cartes visa classiques https://docs.payline.com/display/DT/Les+cartes+de+test#Lescartesdetest-Cartedetests o cartes 3D Secure https://docs.payline.com/display/DT/Les+cartes+de+test#Lescartesdetest-Carte3DSV2France
2.b. Monext / Authentification 3D Secure (si activé)	<<selon méthode 3D Secure>>	- Si l'authentification 3D Secure est activée, le client est redirigé vers un autre site. Lorsqu'il revient sur sa page de paiement, le paramètre paylinetoken présent dans l'url permet au widget de reprendre le "tunnel de paiement".

Ecrans du widget de confirmation de commande	Exemple	Considérations importantes
2.c. Monext / Ticket de caisse		- La référence "Plateforme de services" de la commande est visible sur le ticket de caisse. - Au clic sur le bouton Terminer, Monext déclenche un callback vers notre backend, qui se charge d'effectuer les traitements post-paiements et qui ensuite redirige systématiquement vers votre page hébergeante du widget avec dans l'url les paramètres paymentsuccessful et ordername. Le widget redirigera alors, selon les paramètres d'appel du widget : o soit vers l'écran de confirmation interne au widget de confirmation de commande, o soit vers votre *returnUrlSuccess*. - **Attention, dans le cas où le client ne clique pas sur le bouton Terminer et par exemple ferme la fenêtre, Monext déclenchera un callback vers notre backend au bout d'1 minute. Nous réaliserons alors les traitements post-paiements, mais sans redirection vers la page hébergeante.** **Cf. considérations techniques ci-après pour gérer au mieux les redirections après le ticket de caisse monext.**
2.d. Monext / Erreur avec 2ème essai		- En cas d'erreur, le client a la possibilité de revenir sur l'écran de saisie des informations bancaires. Dans cet enchaînement, nous sommes toujours dans le widget monext.
2.e. Monext / Erreur avec btn Terminer		- Au bout de 2 erreurs, Monext affiche un écran d'erreur avec un bouton Terminer. On a alors les mêmes mécanismes que pour le bouton Terminer du ticket de caisse : o au clic, Monext appelle notre backend qui redirige vers votre page hébergeante, où le widget redirige soit vers le récapitulatif de commande, soit vers votre *returnUrlError* ; o sans clic, Monext appellera notre backend au bout d'1mn, et il n'y aura pas de redirection vers votre page hébergeante

Ecrans du widget de confirmation de commande	Exemple	Considérations importantes
3. Confirmation		- Le widget propose un écran de confirmation basique avec la référence "Plateforme de services" de la commande. - Pour afficher votre propre page de confirmation, vous pouvez utiliser le paramètre returnUrlSuccess ou reprendre le contrôle dès le retour de notre backend sur la base des paramètres payentsuccessful et ordername (cf. plus bas).
4. Erreur technique		- Cet écran est affiché si le widget de confirmation de commande rencontre une erreur technique lors de son initialisation (ex : commande déjà payée, token non valide, …). - Il est possible de ne pas afficher cet écran ou de l'afficher temporairement en utilisant les paramètres redirectTimeOut et returnUrlError (cf. ci-après).

Utilisation

Pour implémenter le widget, placer un élément HTML portant l'id **ImaPayment** :

```
<div id="ImaPayment" data-token="{{string}}" lang="{{alpha-2}}"
withConfirmationPage="{{boolean}}" returnUrlSuccess="{{string}}"
returnUrlError="{{string}}" redirectTimeOut="{{integer}}">
```

Paramètre	Type	Description	Valeur par défaut	Obligatoire ?
id	string	Identifiant technique de l'élément contenant le widget. **Doit toujours être égal à "ImaPayment"**	-	Oui
data-token	string	Le token contient les informations nécessaires à l'initialisation du widget. **À renseigner dynamiquement : obtenu lors de la génération du lien/token**	-	Oui
lang	alpha-2	Permet au consommateur de forcer la langue du widget	Langue du compte client	Non
withConfirmationPage	boolean	*Voir descriptif plus bas*	false	Non
returnUrlSuccess	string	Indique l'URL de redirection vers une page externe de confirmation de commande en cas de paiement réussi si **withConfirmationPage = false**	-	Non
returnUrlError	string	Indique l'URL de redirection vers une page externe d'erreur en cas d'échec du paiement ou d'erreur Monext, ou en cas d'erreur interne	-	Non

		au widget si **withConfirmationPage = false**		
redirectTimeOut	integer *en ms*	Détermine le temps s'écoulant avant qu'une redirection ait lieu en cas d'erreur interne au widget, si **withConfirmationPage = false**	3000	Non

Descriptif de l'attribut **withConfirmationPage** :

1. Si la valeur est **true** :

   - Affiche la page de confirmation de paiement interne au widget en cas de paiement réussi (après le ticket de caisse Monext)
   - Renvoie vers la page de récapitulatif de commande si le paiement a échoué (après le ticket de caisse Monext)
   - Affiche la page d'erreur interne au widget en cas d'erreur propre à celui-ci, sans aucune redirection

2. Si la valeur est **false** :

   - Redirige vers l'URL saisie dans **returnUrlSuccess** en cas de paiement réussi (après le ticket de caisse Monext)
   - Redirige vers l'URL saisie dans **returnUrlError** en cas d'échec du paiement (après le ticket de caisse Monext)
   - Affiche la page d'erreur interne au widget en cas d'erreur propre à celui-ci, puis applique une redirection vers **returnUrlError** après un délai configurable avec l'attribut **redirectTimeOut**

Si **returnUrlSuccess** et/ou **returnUrlError** ne sont pas renseignés alors que **withConfirmationPage** vaut **false**, c'est le comportement avec **withConfirmationPage = true** qui s'applique pour chaque valeur returnUrl manquante.

Considérations techniques sur les redirections

1. **Redirection après le ticket de caisse Monext ou le message d'erreur Monext**

   Suite à la confirmation par l'utilisateur des informations affichées sur le ticket de caisse Monext (paiement accepté ou refusé), la plateforme de services est appelée pour effectuer les traitements post-paiement, **et redirige systématiquement vers la page qui héberge le widget avec des query strings supplémentaires :**

   - en cas de succès du paiement : paymentsuccessful (valeur true) et ordername (valeur = numéro de commande)

   Par exemple, si le widget est hébergé à l'adresse https://apiconsumer.com/payment : https://apiconsumer.com/payment?paymentsuccessful=true&ordername=P21Q00049

- en cas d'échec du paiement : paymentsuccessful (valeur false) et ordername (valeur = numéro de commande)

Ex :
https://apiconsumer.com/payment?paymentsuccessful=false&ordername=P21Q00049

/!\ La page qui héberge le widget doit alors contenir l'élément HTML ImaPayment avec ses paramètres d'appel : data-token, les url de return, … C'est uniquement suite à cette redirection que le widget exploitera les paramètres liés à la redirection : withConfirmationPage, returnUrlSuccess et returnUrlError.

/!\ Les query strings initiaux de la page hébergeante ne sont actuellement pas conservés lors de la redirection.

Remarque : Il est tout à fait possible de "reprendre la main" dès le retour de notre backend en exploitant le paramètre paymentsuccessful pour poursuivre l'expérience utilisateur telle que vous le souhaitez, notamment en cas de paiement ok. Par exemple, dans le cas d'une application, si vous avez intégré le widget de confirmation de commande dans une page web, vous pouvez alors rediriger le client vers votre application.

2. Redirection après l'authentification 3D Secure

De la même manière, suite à l'authentification 3D Secure, le client est redirigé vers la page hébergeante, cette fois avec des query strings propres à Monext : paylinetoken et paymentEndpoint.

Ex :
https://apiconsumer.com/payment?paylinetoken=1hbUGFOdWNHaStYn9q79163320536 8434&paymentEndpoint=1

Considérations techniques sur les css

A VENIR