

DOCUMENT D'INTÉGRATION

avec Use Cases

Sommaire

1	But de cette documentation	4
2	Avantages de l'interface XML PIS/POS V2.0	4
3	Comment fonctionne le processus de commande via Galexis?.....	5
3.1	Création & contrôle du panier dans le système PIS/POS	6
3.2	Enregistrer & contrôler le panier	6
3.3	Charger et traiter le panier.....	7
3.4	Supprimer le panier	7
3.5	Libérer le panier - ctSendX	7
3.6	Clôturer le panier & transmettre la commande.....	7
3.7	Interroger la disponibilité article	8
3.8	Consulter les conditions.....	8
3.9	Enregistrer un rachat	8
3.10	Traiter automatiquement l'entrée marchandise (Chargement du bulletin de livraison & box)	9
3.11	Consulter les articles en attente (Backlog).....	9
3.12	Effectuer l'inventaire	9
3.13	Envoyer des colis à d'autres clients Galexis	10
4	Aperçu des Use Cases	10
4.1	Enregistrer le panier	14
4.1.1	Exemple de demande XML à storeShoppingCartRequest	14
4.1.2	Exemple de réponse XML à storeShoppingCartResponse	15
4.2	Charger & traiter le panier	16
4.2.1	Exemple de demande XML à loadShoppingCartRequest	16
4.2.2	Exemple de réponse XML à loadShoppingCartResponse (succès)	17

4.3	Supprimer le panier	18
4.3.1	Exemple de demande XML à deleteShoppingCartRequest	18
4.3.2	Exemple de réponse XML à deleteShoppingCartResponse (succès)	18
4.4	Libérer le panier (ctSendX)	19
4.4.1	Exemple de demande XML à releaseShoppingCartRequest	19
4.4.2	Exemple de réponse XML à releaseShoppingCartResponse (succès)	20
4.5	Clôturer le panier (transmettre la commande)	20
4.5.1	Exemple de demande XML à commitShoppingCartRequest	20
4.5.2	Exemple de réponse XML à commitShoppingCartResponse (succès)	21
4.6	Interroger la disponibilité article	21
4.6.1	Exemple de demande XML à productAvailabilityRequest	22
4.6.2	Exemple de réponse XML à productAvailabilityResponse	22
4.7	Consulter les conditions (KuKo)	23
4.7.1	Exemple de demande XML à customerSpecificConditionsRequest	23
4.7.2	Exemple de réponse XML à customerSpecificConditionsResponse	24
4.8	Enregistrer un rachat	25
4.8.1	Exemple de demande XML à storeReturnCartRequest (pour les rachats)	25
4.8.2	Exemple de réponse XML à storeReturnCartResponse (pour les rachats)	25
4.8.3	Exemple de demande XML à commitReturnCartRequest (pour les rachats)	26
4.8.4	Exemple de réponse XML à commitReturnCartResponse (pour les rachats)	26
4.9	Traiter automatiquement l'entrée marchandise	27
4.9.1	Exemple de demande XML à deliveryNoteDownloadRequest	28
4.9.2	Exemple de réponse XML à deliveryNoteDownloadResponse	29
4.9.3	Exemple pratique - client avec intégration SOAP	30
4.10	Consulter les articles en attente (Backlog)	38
4.10.1	Exemple de demande XML à openBacklogRequest	38

4.10.2 Exemple de réponse XML à openBacklogResponse	38
4.11 Effectuer l'inventaire	39
4.11.1 Commencer l'inventaire	40
4.11.2 Transmettre l'état intermédiaire de l'inventaire	41
4.11.3 Clôturer l'inventaire	42
4.11.4 Consulter le statut de l'inventaire.....	42
4.11.5 Scanner de codes-barres hors ligne	43
4.12 Envoi de colis.....	45
4.12.1 Exemple de demande XML à getAddressLabelsRequest	45
4.12.2 Exemple de réponse XML à getAddressLabelsResponse	46
5 Tests & mise en production	47

1 But de cette documentation

Cette documentation sert de base technique pour les éditeurs de logiciels souhaitant l'intégrer à l'interface XML POIS/PIS de leurs applications (p. ex. systèmes PIS/POS, ERP ou EWM). Pour chaque fonction, deux messages sont définis. Un pour la requête (request) et un pour la réponse (response). Les messages sont échangés au format XML, dont la structure est définie par un schéma (XSD). Les messages sont échangés via l'une des options suivantes :

Requête POST via un protocole HTTPS (SSL) avec le type de contenu « text/xml ». Le contenu de la requête est le message XML lui-même, sans enveloppe supplémentaire (Enveloppe - Element).

Requête SOAP via protocole HTTPS (SSL). Les fonctions sont alors décrites au moyen de définitions WSDL.

L'objectif est de permettre une communication entièrement automatisée avec le système ERP de Galexis, afin de rendre les processus de commande, de disponibilité et de livraison plus efficaces pour les pharmacies, drogueries et autres clients.

Groupe cible en détail : les éditeurs logiciels et les prestataires de services informatiques du domaine de la santé - en particulier les équipes de développement de systèmes PIS/POS, de systèmes ERP & EWM ainsi que les responsables de projet techniques chez les fournisseurs de logiciels pour pharmacies et drogueries.

Lien vers les WSDL et les schémas : <https://xml.e-galexis.com/V2/>

2 Avantages de l'interface XML PIS/POS V2.0

- Requêtes en temps réel pour les prix, la disponibilité et les livraisons en attente
- Traitement intégré du processus de commande, depuis la requête en passant par le panier, jusqu'au bulletin de livraison / bac, qui inclut toutes les données de lot et de péremption ainsi que les quantités et les prix des produits livrés.
- Intégration directe des conditions spécifiques au client.
- Communication flexible via HTTPS (text/xml ou SOAP)

3 Comment fonctionne le processus de commande via Galexis?

Les méthodes intégrées de bout en bout permettent à l'utilisateur PIS/POS d'assurer un traitement efficace, transparent et sécurisé depuis le contrôle des articles à la création et la transmission du panier de commande puis la gestion des stocks.

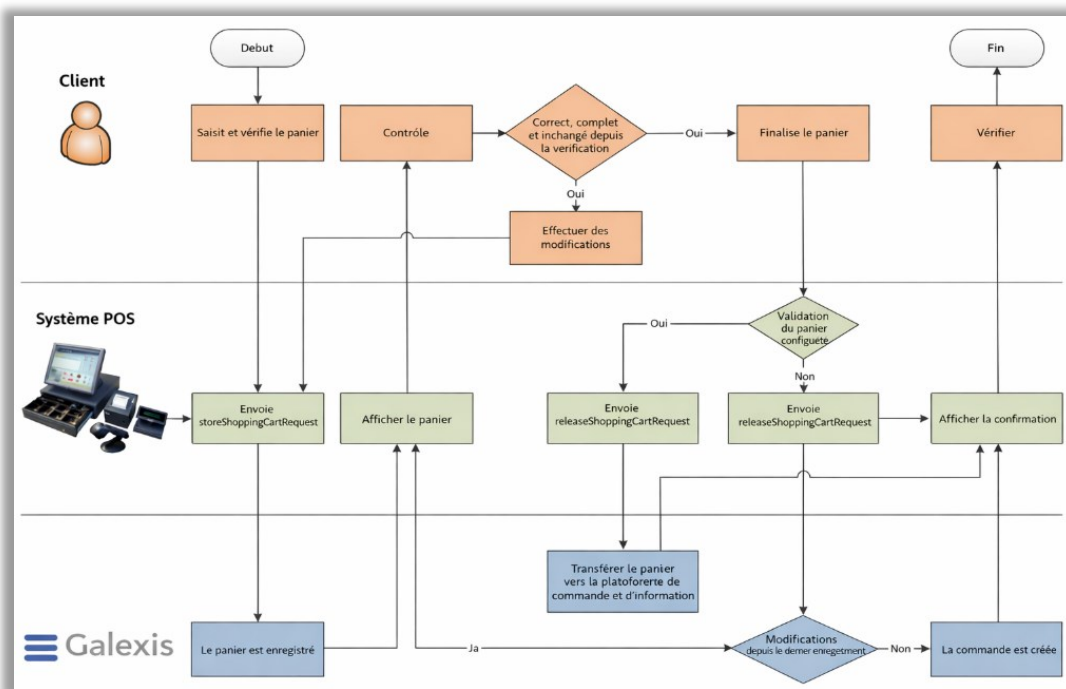


Figure 1: Workflow flux du traitement des commandes

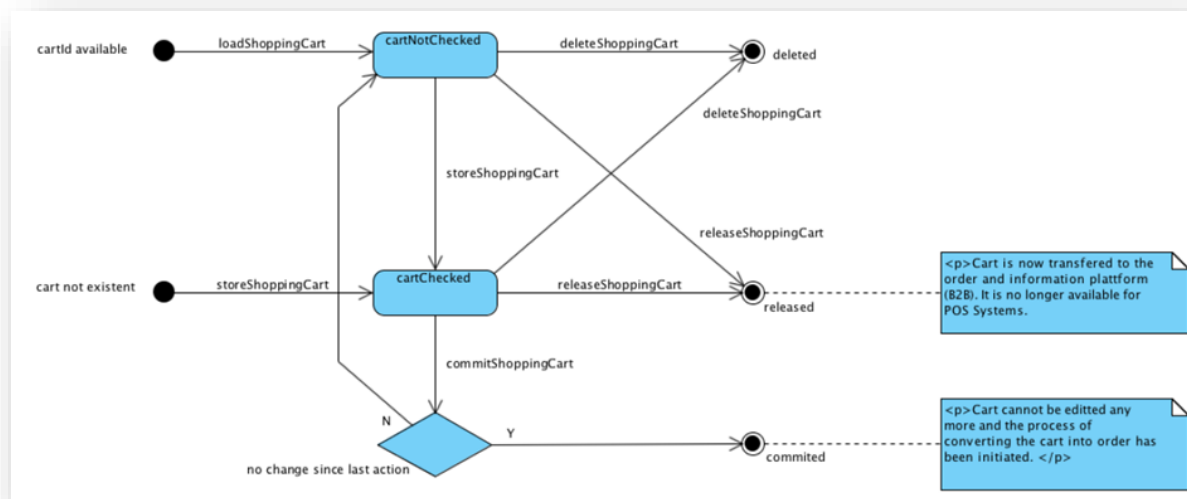


Figure 2: Etats du panier

3.1 Création & contrôle du panier dans le système PIS/POS

L'utilisateur du système PIS/POS crée d'abord son panier dans le système et vérifie qu'il soit complet et correct. Cette fonctionnalité est mise à disposition par le système PIS/POS lui-même.

3.2 Enregistrer & contrôler le panier

Le système PIS/POS envoie le panier à Galexis avec la spécification `storeShoppingCartRequest`. Galexis contrôle ensuite toutes les positions en termes de disponibilité, de droit d'approvisionnement et de calcul correct des prix, puis renvoie le résultat avec la spécification `storeShoppingCartResponse`. L'utilisateur PIS/POS peut alors être certain que le panier contient des données valides et à jour.

3.3 Charger et traiter le panier

Si le panier n'a pas encore été définitivement validé, l'utilisateur du PIS/POS peut le recharger à tout moment avec la spécification `loadShoppingCartRequest` afin d'effectuer des ajustements ou vérifier l'état actuel. La réponse de Galexis est envoyée via `loadShoppingCartResponse` et affiche le contenu actuel du panier.

3.4 Supprimer le panier

Il existe également la possibilité de supprimer un panier enregistré (mais non transmis) avec la spécification `DeleteShoppingCart`, si celui-ci n'est plus souhaité. Cette méthode permet à l'utilisateur PIS/POS de retirer le panier du système, assurant ainsi davantage de clarté et d'ordre dans le processus de commande.

3.5 Libérer le panier - ctSendX

Pour la poursuite du traitement dans le portail central clients (e-Galexis), l'utilisateur du PIS/POS peut libérer le panier à l'aide de `releaseShoppingCartRequest`. Galexis confirme cette libération via `releaseShoppingCartResponse`, rendant ainsi le panier modifiable dans le portail.

Cette fonction a été spécialement conçue pour les cabinets médicaux, dans lesquels plusieurs assistantes médicales (MPA) saisissent des paniers via GaliPhone. Le médecin peut ensuite les consolider dans e-Galexis et ainsi créer une commande définitive.

3.6 Clôturer le panier & transmettre la commande

Si le panier est correct, complet et libéré, l'utilisateur du PIS/POS déclenche la commande à l'aide de `commitShoppingCartRequest`. Galexis confirme la réussite de l'opération via `commitShoppingCartResponse`. Si la commande n'aboutit pas ou si certaines conditions ont changé, l'utilisateur du PIS/POS reçoit un retour correspondant afin de pouvoir apporter les corrections nécessaires.

3.7 Interroger la disponibilité article

De plus, l'utilisateur souhaite pouvoir vérifier à tout moment la disponibilité des produits, les prix et les détails. Pour cela, il envoie une `productAvailabilityRequest` qui reçoit en retour une `productAvailabilityResponse` de Galexis fournissant des informations fiables sur la disponibilité actuelle et la tarification.

3.8 Consulter les conditions

Afin de pouvoir consulter les conditions individuelles Galexis (KuKo's) et les rabais à tout moment, l'utilisateur PIS/POS utilise la méthode `customerSpecificConditionsRequest`. La réponse `customerSpecificConditionsResponse` lui donne un aperçu de sa propre tarification. Les conditions clients Galexis ne doivent pas être saisies manuellement : elles sont fournies automatiquement par l'interface.

3.9 Enregistrer un rachat

Les rachats correspondent à des ventes effectuées à Galexis pour des marchandises se trouvant en dehors du processus de retours (dans un délai de 10 jours¹). Les produits peuvent être proposés à Galexis pour un rachat en raison d'un rappel de lot, d'une reprise initiée par un fournisseur ou d'un assainissement de stock chez le client. La saisie des données de rachat fonctionne de manière analogue à la fonctionnalité du panier de commande : les rachats sont d'abord enregistrés via `storeReturnCart`, puis finalisés à l'aide de `commitReturnCartRequest`. Les rachats erronés peuvent être supprimés au moyen de `deleteReturnCartRequest`. La réponse `storeReturnCartResponse` indique si les produits peuvent être rachetés et à quelles conditions.

¹ La réglementation Swissmedic s'applique : <https://www.swissmedic.ch/swissmedic/fr/home/news/mitteilungen/neue-techn-interpretation-ti-i-smi-ti-28.html>

3.10 Traiter automatiquement l'entrée marchandise (Chargement du bulletin de livraison & box)

Les livraisons sont traitées automatiquement par l'utilisateur PIS/POS via `deliveryNoteDownloadRequest`. Cette méthode permet d'effectuer une requête basée sur le numéro du bulletin de livraison, le numéro de boîte ou la date. La réponse retournée par Galexis via `deliveryNoteDownloadResponse` fournit des informations détaillées telles que le numéro de lot, la quantité, la date d'expiration, les indications de prix, coûts logistiques, totaux etc.

3.11 Consulter les articles en attente (Backlog)

Les articles en attente correspondent aux lignes de commande pour lesquelles Galexis attend encore une livraison de marchandise afin de pouvoir les expédier au client. Ces articles en attente peuvent concerner des articles de stock classiques (NOTA) ou des articles que Galexis ne tient pas en stock, mais qu'il peut obtenir auprès du fournisseur pour le client (BESO). Pour consulter les articles en attente, l'utilisateur PIS/POS utilise la méthode `openBacklogRequest`. Galexis renvoie alors des informations détaillées sur les articles en attente via `openBacklogResponse`.

3.12 Effectuer l'inventaire

Pour effectuer un inventaire, l'utilisateur PIS/POS démarre un inventaire avec `getInventoryIdRequest` et reçoit un inventaire-ID via `getInventoryIdResponse`. Il transmet les résultats intermédiaires avec `inventoryTransmissionRequest`, confirmés par `inventoryTransmissionResponse`. La clôture de l'inventaire est signalée à l'aide de `closeInventoryRequest` et confirmée par `closeInventoryResponse`. L'état de l'inventaire peut être consulté avec `inventoryStatusRequest` et est renvoyé avec `inventoryStatusResponse`. Pour une saisie efficace, le système prend également en charge un scanner de codes-barres hors ligne, basé sur les données fournies par `getProductBaseMasterDataDiff`. Concrètement, les données hors ligne sont d'abord chargées dans le scanner permettant ainsi d'effectuer l'inventaire sans connexion. Une fois l'inventaire terminé, Galexis génère la liste des produits et quantités comptés, complétée par les prix d'achat et la met à disposition du client au format CSV et/ou PDF.

3.13 Envoyer des colis à d'autres clients Galexis

Pour terminer, l'utilisateur PIS/POS a la possibilité de générer une étiquette de colis. Celle-ci permet d'envoyer des paquets via Galexis à d'autres clients Galexis.

4 Aperçu des Use Cases

Use Case	Description	Méthodes
Créer du panier (côté PIS/POS)	L'utilisateur PIS/POS saisit son panier dans le système et en vérifie l'exactitude et la conformité. Cette fonctionnalité relève du système PIS/POS.	<code>storeShoppingCartRequest</code> <code>storeShoppingCartResponse</code>
Enregistrer le panier	Le système PIS/POS envoie le panier à Galexis via <code>storeShoppingCartRequest</code> . Galexis vérifie la disponibilité, les droits d'acquisition et les prix, puis renvoie le résultat via <code>storeShoppingCartResponse</code> . De cette manière, l'utilisateur PIS/POS reçoit des données valides et à jour concernant son panier.	<code>storeShoppingCartRequest</code> <code>storeShoppingCartResponse</code>
Charger et traiter le panier	Si le panier n'a pas encore été finalisé, l'utilisateur PIS/POS peut le changer à tout moment au moyen de <code>loadShoppingCartRequest</code> afin d'effectuer des ajustements ou de vérifier l'état actuel. La réponse est transmise via <code>loadShoppingCartResponse</code> .	<code>loadShoppingCartRequest</code> <code>loadShoppingCartResponse</code>
Supprimer le panier	L'utilisateur PIS/POS peut supprimer un panier enregistré (mais non transmis) au moyen de <code>deleteShoppingCartRequest</code> s'il n'en a plus besoin. Galexis confirme via <code>deleteShoppingCartResponse</code> .	<code>deleteShoppingCartRequest</code> <code>deleteShoppingCartResponse</code>
Libérer le panier (ctSendX)	Pour un traitement ultérieur dans le portail client e-Galexis, le panier peut être libéré au moyen de <code>releaseShoppingCartRequest</code> . La confirmation est transmise via <code>releaseShoppingCartResponse</code> , ce qui rend le panier modifiable dans le portail.	<code>releaseShoppingCartRequest</code> <code>releaseShoppingCartResponse</code>

Clôturer le panier (transmettre la commande)	Lorsque le panier est complet, l'utilisateur PIS/POS déclenche la commande au moyen de <code>commitShoppingCartRequest</code> . Galexis confirme la finalisation réussie via <code>commitShoppingCartResponse</code> . En cas d'erreur ou de conditions modifiées, un message est directement transmis dans la réponse, ainsi que le contenu du panier, afin que celui-ci puisse être à nouveau enregistré et finalisé.	<code>commitShoppingCartRequest</code> <code>commitShoppingCartResponse</code>
Interroger la disponibilité article	Pour obtenir rapidement des informations sur les produits, l'utilisateur PIS/POS envoie une <code>productAvailabilityRequest</code> . Galexis fournit, via <code>productAvailabilityResponse</code> , des informations à jour concernant la disponibilité, les prix et les détails du produit.	<code>productAvailabilityRequest</code> <code>productAvailabilityResponse</code>
Consulter les conditions	L'utilisateur PIS/POS récupère ses conditions et rabais individuels Galexis au moyen de <code>customerSpecificConditionsRequest</code> . La réponse <code>customerSpecificConditionsResponse</code> lui présente sa tarification personnalisée.	<code>customerSpecificConditionsRequest</code> <code>customerSpecificConditionsResponse</code>
Enregistrer un rachat	Pour les retours (rachats), l'utilisateur PIS/POS utilise la fonctionnalité <code>storeReturnCart</code> . Elle fonctionne de manière analogue à la fonctionnalité du panier. Les retours sont d'abord enregistrés via <code>storeReturnCart</code> , puis finalisés avec <code>commitReturnCartRequest</code> . Les retours erronés peuvent être supprimés au moyen de <code>deleteReturnCartRequest</code> .	Ouvrir un rachat <code>storeReturnCartRequest</code> <code>storeReturnCartResponse</code> Fermer un rachat <code>commitReturnCartRequest</code> <code>commitReturnCartResponse</code> Supprimer un rachat <code>deleteReturnCartRequest</code> <code>deleteReturnCartResponse</code>

Traiter automatiquement l'entrée marchandise	L'utilisateur PIS/POS peut consulter les livraisons sur la base du numéro du bulletin de livraison, de la date ou du numéro de bac au moyen de <code>deliveryNoteDownloadRequest</code>. La réponse <code>deliveryNoteDownloadResponse</code> contient des détails tels que le lot, la quantité, la date de péremption, les prix, les coûts logistiques, les rabais ainsi que les informations globales.	<code>deliveryNoteDownloadRequest</code> <code>deliveryNoteDownloadResponse</code>
Consulter les articles en attente (Backlog)	Pour surveiller l'état des arriérés de commande ouverts (nota,beso), l'utilisateur PIS/POS utilise <code>openBacklogRequest</code>. Galexis répond avec <code>openBacklogResponse</code>, qui fournit des informations détaillées sur les reliquats de commande en cours.	<code>openBacklogRequest</code> <code>openBacklogResponse</code>

Effectuer l'inventaire	L'utilisateur PIS/POS commence l'inventaire avec <code>getInventoryIdRequest</code> et reçoit l'inventaire ID via <code>getInventoryIdResponse</code>. Il transmet les résultats intermédiaires au moyen de <code>inventoryTransmissionRequest</code> (confirmation : <code>inventoryTransmissionResponse</code>). Il signale la clôture de l'inventaire avec <code>closeInventoryRequest</code> (réponse : <code>closeInventoryResponse</code>). Il interroge l'état actuel au moyen de <code>inventoryStatusRequest</code> (réponse : <code>inventoryStatusResponse</code>). Pour une saisie efficace, le système prend en charge un scanner de codes-barres offline avec les données issues de <code>getProductBaseMasterDataDiff</code>. Les données offline sont d'abord chargées dans le scanner, puis l'inventaire s'effectue offline.	Commencer l'inventaire <code>getInventoryIdRequest</code> <code>getInventoryIdResponse</code> Envoyer les résultats intermédiaires <code>inventoryTransmissionRequest</code> <code>inventoryTransmissionResponse</code> Fermer l'inventaire <code>closeInventoryRequest</code> <code>closeInventoryResponse</code> Interroger le statut de l'inventaire <code>inventoryStatusRequest</code> <code>inventoryStatusResponse</code> Données du scanner de codes-barres <code>getProductBaseMasterDataDiff</code>
Envoyer des colis (Parcel)	L'utilisateur PIS/POS peut consulter les informations nécessaires pour créer une étiquette de colis. Grâce à celle-ci, les paquets peuvent être envoyés via Galexis à d'autres clients Galexis.	<code>getAddressLabelsRequest</code> <code>getAddressLabelsResponse</code>

4.1 Enregistrer le panier

Le collaborateur du POS saisit un panier contenant des produits dans son système qui est ensuite transmis à Galexis. Galexis vérifie les produits du panier en termes de disponibilité, de droits d'acquisition et de tarification, puis fournit un retour.

4.1.1 Exemple de demande XML à storeShoppingCartRequest

XSD :

<https://xml.e-galexis.com/V2/schemas/shoppingCart/storeShoppingCartRequest.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/shoppingCart/>

 Le collaborateur POS transmet le panier de commande contenant tous les produits à Galexis. Chaque produit est identifié par une pharmacode unique accompagné de la quantité souhaitée, du type de position et de la zone de livraison. La date de livraison désirée est définie dans l'en-tête afin de planifier l'expédition. De cette manière, Galexis reçoit toutes les données nécessaires pour vérifier le panier.

L'élément principal `<storeShoppingCartRequest>` contient, en plus des attributs techniques pour schemaLocation et namespace, les attributs suivants :

- version : version du message. La version 2.0 doit être utilisée.
- language : langue dans laquelle la description du produit doit être renvoyée. Si aucune langue n'est transmise, la langue définie dans le dossier client est utilisée.
- communicationSoftwareId : identification du logiciel qui a transmis la requête.

Le sous élément `client` contient les données d'authentification (ID Webclient & mot de passe).

L'élément `cartIdentifier` contient l'identification du panier. Si un nouveau panier est créé, cet élément ne doit pas être transmis. Si un panier existant est modifié, l'identification du panier doit être fournie.

L'élément `<cartHeader>` contient les métadonnées du panier, par exemple la date de livraison souhaitée.

- `<cartLines>` : La liste des différentes positions dans le panier de commande.

- `<cartLine>` : une ligne individuelle dans le panier de commande.
- `cartLineNumber` numéro de position
- `orderQuantity` quantité commandée
- `cartLineType` Type de position, par exemple marchandise, étiquette ou les deux. Pour les étiquettes, la quantité définie par `orderQuantity` correspond au nombre d'étiquettes à livrer. Pour marchandise et étiquette, la marchandise est livrée dans la quantité souhaitée, accompagnée d'une étiquette.
- `deliveryZone` Zone de livraison. Valeurs autorisées : vide, CUSTOMER, ROBOT, SPECIAL & WEBSHOP.
- `<product>` : Le produit peut être identifié via l'un des trois types suivants : `<pharmaCode>`, `<EAN>` ou `<supplierProductNumber>`.

4.1.2 Exemple de réponse XML à storeShoppingCartResponse

XSD:

<https://xml.e-galexis.com/V2/schemas/shoppingCart/storeShoppingCartResponse.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/shoppingCart/>

 Galexis confirme la réception et valide chaque position. Pour chaque ligne de produit, la disponibilité, la quantité réellement livrable et le prix actuel sont communiqués. Une somme totale hors TVA ainsi que les frais de livraison estimés sont également fournies. Ainsi, le collaborateur POS obtient une vue d'ensemble claire sur l'état actuel du panier.

- `<storeShoppingCartResponse>` : élément principal de la réponse, contenant namespace, la version et la langue
- `<clientResponse>` : confirmation des données de client transmises dans la requête
- `<shoppingCart>` : informations relatives au panier traité
 - `<cartIdentifier>` : ID unique du panier, nécessaire pour les actions ultérieures, par exemple pour finaliser le panier. S'y ajoute un numéro lisible par un utilisateur, utilisé en interne par Galexis.
 - `<cartHeaderResponse>`
 - `deliveryDate` : date de livraison confirmée

- **<cartLinesResponse>** : La liste des réponses concernant chaque position du panier.
 - **<cartLineResponse>** : est affiché pour chaque position :
 - **lineAccepted** : si la position est acceptée (true/false).
 - **deliveryZone** : zone de livraison concernée
 - **productResponse** : détails sur le produit, tels que la description, le prix de base, la quantité livrée et la quantité facturée.
 - **<availability>** : disponibilité du produit (par exemple "yes" = disponible).
 - **<cartTotals>** : résumé des coûts dans le panier :
 - **totalAmountExclVATExclExpLogServiceCosts** : Montant total sans TVA et sans frais logistiques supplémentaires.
 - **totalExpectedLogServiceCosts** : coûts logistiques supplémentaires prévus.

4.2 Charger & traiter le panier

L'utilisateur du POS souhaite charger un panier précédemment enregistré depuis Galexis afin d'en consulter le contenu, de le vérifier ou de le modifier si nécessaire. Le système POS envoie une requête à Galexis pour récupérer la version actuelle du panier

4.2.1 Exemple de demande XML à loadShoppingCartRequest

XSD:

<https://xml.e-galexis.com/V2/schemas/shoppingCart/loadShoppingCartRequest.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/shoppingCart/>

 L'utilisateur du POS envoie la requête accompagnée des données d'authentification afin de charger le panier actuellement enregistré. Il est possible d'indiquer en option quel panier doit être chargé (généralement via un ID). Dans ce cadre, le POS demande le contenu complet du panier, y compris l'ensemble des positions.

L'élément principal `<loadShoppingCartRequest>` contient les informations d'authentification de l'utilisateur du POS. Galexis identifie le client et charge le panier actuel depuis le système. L'élément `<cartIdentifier>` contient l'ID du panier qui doit être chargé.

4.2.2 Exemple de réponse XML à `loadShoppingCartResponse` (succès)

XSD:

<https://xml.e-galexis.com/V2/schemas/shoppingCart/loadShoppingCartResponse.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/shoppingCart/>

 Galexis renvoie une réponse contenant les détails complets du panier, y compris les positions avec les informations produit, les prix et le statut de disponibilité.

L'élément principal `<loadShoppingCartResponse>` confirme le client ainsi que le panier chargé. A l'intérieur de `<cartLinesResponse>`, toutes les lignes du panier sont répertoriées avec les détails concernant le produit, le prix, la quantité et la disponibilité. L'élément `<cart-Totals>` fournit un récapitulatif des coûts hors TVA ainsi que les frais logistiques prévisionnels. Cela permet à l'utilisateur du POS de comprendre précisément l'état actuel de son panier et d'y apporter des modifications si nécessaire.

4.3 Supprimer le panier

L'utilisateur du POS souhaite supprimer un panier enregistré lorsqu'il n'est plus nécessaire. Cela permet de garder le système clair et d'éviter des commandes involontaires. Pour ce faire, le système POS envoie à Galexis une requête de suppression, qui retire du système le panier correspondant à l'ID indiquée.

4.3.1 Exemple de demande XML à deleteShoppingCartRequest

XSD:

<https://xml.e-galexis.com/V2/schemas/shoppingCart/deleteShoppingCartRequest.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/shoppingCart/>



L'utilisateur du POS identifie le panier à supprimer à l'aide de son identifiant unique et s'authentifie auprès de Galexis.

L'élément principal `<deleteShoppingCartRequest>` contient les données d'authentification de l'utilisateur du POS ainsi que l'élément `<cartIdentifier>` avec l'ID unique du panier à supprimer. Galexis utilise ces informations pour identifier correctement le panier concerné et le supprimer.

4.3.2 Exemple de réponse XML à deleteShoppingCartResponse (succès)

XSD:

<https://xml.e-galexis.com/V2/schemas/shoppingCart/deleteShoppingCartResponse.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/shoppingCart/>



Galexis confirme la suppression réussie du panier.

L'élément principal `<deleteShoppingCartResponse>` confirme les détails du client et du panier. La présence de l'élément `<successful/>` indique que le panier a été supprimé avec

succès. L'utilisateur du POS reçoit ainsi une confirmation claire de la suppression effective du document.

4.4 Libérer le panier (ctSendX)

L'utilisateur du POS souhaite libérer un panier enregistré afin de le rendre disponible pour une modification ultérieure dans le portail client central (e-Galexis). Cette action permet au panier d'y être visible et éditable, par exemple pour d'autres personnes autorisées ou dans le cadre de processus de validation.

4.4.1 Exemple de demande XML à releaseShoppingCartRequest

XSD:

<https://xml.e-galexis.com/V2/schemas/shoppingCart/releaseShoppingCartRequest.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/shoppingCart/>

 L'utilisateur du POS identifie le panier à libérer à l'aide de son identifiant et s'authentifie auprès de Galexis.

L'élément central `<releaseShoppingCartRequest>` contient les données d'authentification ainsi que l'identifiant unique du panier qui doit être libéré pour le portail client. Galexis utilise ces informations pour enregistrer la libération dans le système.

4.4.2 Exemple de réponse XML à releaseShoppingCartResponse (succès)

XSD:

<https://xml.e-galexis.com/V2/schemas/shoppingCart/releaseShoppingCartResponse.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/shoppingCart/>

 Galexis confirme la libération réussie du panier.

L'élément de réponse `<releaseShoppingCartResponse>` confirme la libération du panier en incluant les données du client ainsi que les identifiants du panier. L'élément `<successful>` accompagné d'un message approprié indique que le panier est désormais modifiable dans le portail client.

4.5 Clôturer le panier (transmettre la commande)

Après vérification et libération, l'utilisateur du POS souhaite finaliser le panier et ainsi transmettre la commande auprès de Galexis. A cette occasion, l'intégralité du panier est transmise à Galexis afin de placer la commande de manière ferme et définitive.

4.5.1 Exemple de demande XML à commitShoppingCartRequest

XSD:

<https://xml.e-galexis.com/V2/schemas/shoppingCart/commitShoppingCartRequest.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/shoppingCart/>

 L'utilisateur du POS envoie une requête de commande avec l'identifiant unique du panier et les données d'authentification.

L'élément principal `<commitShoppingCartRequest>` contient, en plus de namespace et de la version, les informations d'authentification ainsi que l'ID du panier qui doit être commandé. Les attributs `productDescriptionDesired` et `compressionDesired` déterminent si les

descriptions de produits sont souhaitées dans la réponse et si les données doivent être transmises de manière compressée.

Le client POS peut transmettre les informations `deliverZone` ainsi que `cartIdentifier` si il souhaite que ceux-ci soient renvoyés lors de l'entrée marchandise avec `deliveryNoteDownloadResponse`.

4.5.2 Exemple de réponse XML à `commitShoppingCartResponse` (succès)

XSD:

<https://xml.e-galexis.com/V2/schemas/shoppingCart/commitShoppingCartResponse.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/shoppingCart/>

 L'élément de réponse `<commitShoppingCartResponse>` contient l'élément `clientResponse` avec les données du client confirmées, ainsi que l'élément `cartIdentifier` avec l'ID du panier. La réponse contient soit l'élément `successful` soit l'élément `unsuccessful`, qui informe de la réussite ou non de la finalisation du panier.

L'élément de réponse `<commitShoppingCartResponse>` contient l'élément `clientResponse` avec les données du client confirmées, ainsi que l'élément `cartIdentifier` avec l'ID du panier. Si la réponse contient l'élément `<successful>`, le panier est clôturé avec succès, respectivement la commande est créée.

Si l'élément de réponse contient `unsuccessful`, le panier n'a pas pu être finalisé, par exemple en raison de conditions modifiées. Dans ce cas, le panier est renvoyé avec le sous élément `shoppingCart`. Il doit alors être à nouveau enregistré et transmis.

4.6 Interroger la disponibilité article

L'utilisateur du POS aimerait interroger rapidement et de façon fiable la disponibilité, le prix et d'autres détails produits auprès de Galexis. L'interrogation s'effectue sur l'identification unique du produit ainsi que sur la quantité à vérifier.

4.6.1 Exemple de demande XML à productAvailabilityRequest

XSD:

<https://xml.e-galexis.com/V2/schemas/productAvailability/productAvailabilityRequest.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/productAvailability/>

 L'utilisateur du POS envoie une demande avec l'identification du produit et la quantité souhaitée.

L'élément principal `<productAvailabilityRequest>` contient, en plus des attributs techniques relatifs à schemaLocation et namespace, les attributs suivants :

- version : version du message. La version 2.0 doit être utilisée.
- language : langue dans laquelle la description du produit doit être renvoyée. Si aucune langue n'est transmise, la langue définie dans le dossier client est utilisée.
- communicationSoftwareId : identification du logiciel ayant transmis la requête.

Le sous élément `client` contient les données d'authentification (ID client & mot de passe). Les données d'authentification ainsi que la liste qui doivent être interrogées sont également transmises. Chaque produit est identifié de manière unique par son `pharmacode` et la quantité souhaitée `requestedQuantity` est indiquée afin de vérifier la disponibilité.

4.6.2 Exemple de réponse XML à productAvailabilityResponse

XSD:

<https://xml.e-galexis.com/V2/schemas/productAvailability/productAvailabilityResponse.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/productAvailability/>

 Galaxis fournit les informations de disponibilité, les prix et d'autres attributs pour chaque produit interrogé.

La réponse contient l'élément `productAvailabilityResponseLine` pour chaque ligne interrogée. Celle-ci comprend :

`productAvailabilityLine` : les informations demandées dans la requête

productResponse : les informations produit, y compris les prix

availability : l'indication de la disponibilité du produit dans la quantité souhaitée

genericsResponse : la liste des produits du même groupe générique

productConditionLevels : les informations relatives aux conditions commerciales

4.7 Consulter les conditions (KuKo)

L'utilisateur du POS souhaite pouvoir consulter à tout moment ses conditions de rabais individuels et rabais pour chaque produit auprès de Galexis afin de garantir une tarification transparente et compréhensible. Comme le nombre de lignes pour une seule requête est trop important, la consultation paginée. La première requête est effectuée sans **requestKey** (un identifiant de position). Un **requestKey** est envoyé dans la réponse et doit être fourni lors de la requête suivante. Ce processus se répète jusqu'à ce que la réponse indique que la fin des données est atteinte.

! Pour pouvoir interroger les données KuKo, Galexis AG doit intégrer le client dans le processus de préparation des données.

4.7.1 Exemple de demande XML à **customerSpecificConditionsRequest**

XSD:

<https://xml.e-galexis.com/V2/schemas/customerSpecificConditions/customerSpecificConditionsRequest.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/customerSpecificConditions/>



L'utilisateur du POS envoie une requête pour consulter ses conditions spécifiques.

L'élément principal **<customerSpecificConditionsRequest>** contient, en plus des attributs techniques relatifs à **schemaLocation** et **namespace**, les attributs suivants :

- **version** : version du message. La version 2.0 doit être utilisée.
- **language** : langue dans laquelle la description du produit doit être renvoyée. Si aucune langue n'est transmise, la langue définie dans le dossier client est utilisée.
- **communicationSoftwareId** : identification du logiciel ayant transmis la requête.

Sous l'élément principal, on trouve les sous éléments suivants :

- `client` contient les données d'authentification (Webclient-Id & mot de passe).
- `browseRequest` : contient la `direction` (avant ou arrière) ainsi que le `requestKey`. Lors de la première requête, ce champ doit être transmis vide.
- `customerSpecificConditionsBrowseDefinition` : contient la `pageSize` qui permet de définir le nombre de conditions renvoyées par page.

4.7.2 Exemple de réponse XML à `customerSpecificConditionsResponse`

XSD:

<https://xml.e-galexis.com/V2/schemas/customerSpecificConditions/customerSpecificConditionsResponse.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/customerSpecificConditions/>



Galexis fournit un aperçu des conditions et rabais individuels du client.

La réponse `<customerSpecificConditionsResponse>` confirme les données du client et répertoire, pour chaque produit bénéficiant d'une condition individuelle, le pharmacode, le rabais en pourcentage ainsi que la période de validité. Ainsi, l'utilisateur du POS dispose à tout moment d'une transparence complète sur ses avantages tarifaires actuels.

4.8 Enregistrer un rachat

4.8.1 Exemple de demande XML à storeReturnCartRequest (pour les rachats)

XSD:

<https://xml.e-galexis.com/V2/schemas/returnCart/storeReturnCartRequest.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/returnCart/>

 Envoie les articles retournés, avec les quantités et le motif de retour à Galexis.

L'utilisateur du POS transmet les produits à reprendre, avec leur pharmacode, la quantité retournée et le motif du retour. Cela permet à Galexis d'enregistrer les retours de manière systématique.

4.8.2 Exemple de réponse XML à storeReturnCartResponse (pour les rachats)

XSD :

<https://xml.e-galexis.com/V2/schemas/returnCart/storeReturnCartResponse.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/returnCart/>

 Fournit le statut d'acceptation ou de refus des retours, y compris les conditions.

Galexis confirme pour chaque produit si la ligne est acceptée `<lineAccepted>` ou refusée `<lineNotAccepted>` pour un rachat et fournit un message de retour. L'élément `<repurchase-CartLineResponse>` contient également les informations relatives aux conditions auxquelles le produit est repris. Ainsi, le processus de rachat est transparent et facilement compréhensible pour l'utilisateur du POS.

4.8.3 Exemple de demande XML à commitReturnCartRequest (pour les rachats)

XSD:

<https://xml.e-galexis.com/V2/schemas/returnCart/commitReturnCartRequest.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/returnCart/>

 Transmet la confirmation des conditions acceptées.

L'utilisateur du POS accepte les conditions communiquées dans le `storeReturnCartResponse` et souhaite enregistrer/clôturer définitivement les positions.

4.8.4 Exemple de réponse XML à commitReturnCartResponse (pour les rachats)

XSD:

<https://xml.e-galexis.com/V2/schemas/returnCart/commitReturnCartResponse.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/returnCart/>

 Confirme l'enregistrement définitif des retours.

L'élément de réponse `<commitReturnCartResponse>` contient l'élément `clientResponse` avec les données clients confirmées et l'élément `cartIdentifier` avec l'ID du panier. Si la réponse contient l'élément `<successful>`, cela signifie que les positions enregistrées ont été traitées et clôturées avec succès.

4.9 Traiter automatiquement l'entrée marchandise

L'utilisateur du POS souhaite automatiser le traitement des livraisons entrantes en récupérant les données de livraison auprès de Galexis. La requête est effectuée sur la base du numéro du bulletin de livraison, de la date ou du numéro de bac. L'objectif est d'obtenir systématiquement toutes les informations importantes telles que le lot, la quantité, la date d'expiration, les prix, les frais logistiques et les rabais afin de les enregistrer correctement dans le système de gestion de stock.



Le sous-élément diffère selon les informations entrées : numéro de bulletin de livraison, numéro de bac ou date.

Ci-dessous, trois variantes montrent comment la réception marchandise peut être gérée techniquement. Comme variante standard, la réception des marchandises par scannage de bac est recommandée. Vous pouvez décider vous-même laquelle des trois variantes fait finalement le plus de sens pour vous. En cas d'incertitude, votre conseiller Galexis se tient à votre disposition.

Entrée marchandise par scannage de bulletin de livraison : une entrée marchandise est effectuée dans le système POS par l'utilisateur pour chaque bulletin de livraison fourni. L'utilisateur POS doit alors trier les bacs par commande transmise. Cela nécessite d'effectuer plusieurs entrées marchandises par livraison.

Entrée marchandise par scannage de bac : une seule entrée marchandise est effectuée par livraison dans le système POS par l'utilisateur. En scannant un numéro de bac, l'utilisateur peut importer toutes les données de la livraison actuelle dans le système POS. Cela permet d'introduire la marchandise livrée en une seule fois peu importe le nombre de commandes que la livraison contient.

Les données de la livraison sont transmises par Galexis via le numéro de tour (numéro de livraison) qui correspond à la livraison actuelle de toutes les commandes, NOTA, BESO, etc pour le client final. Les numéros de tous les bacs livrés à un moment précis étant enregistrés chez Galexis afin de permettre un meilleur suivi de livraison, ces données peuvent également être reprise par l'utilisateur POS afin d'effectuer une entrée marchandise groupée.

Entrée marchandise par date : une seule entrée marchandise peut être effectuée par date. Cela signifie que même si le client POS reçoit plusieurs livraisons par jour, il peut les introduire en une seule fois s'il attend que toutes aient été livrées.

4.9.1 Exemple de demande XML à deliveryNoteDownloadRequest

XSD:

<https://xml.e-galexis.com/V2/schemas/deliveryNote/deliveryNoteDownloadRequest.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/deliveryNote/>

 Avec cette requête, l'utilisateur du POS demande les informations concernant un bulletin de livraison ou une livraison.

L'utilisateur du POS indique soit le numéro précis du bon de livraison, soit une date de livraison ou un numéro de tour.

L'élément principal `<deliveryNoteDownloadRequest>` contient, en plus des attributs techniques relatifs à schemaLocation et namespace, les attributs suivants :

- version : version du message. La version 2.0 doit être utilisée.
- language : langue dans laquelle la description du produit doit être renvoyée. Si aucune langue n'est transmise, la langue définie dans le dossier client est utilisée.
- communicationSoftwareId : identification du logiciel ayant transmis la requête (Web-client-Id & mot de passe).

Sous l'élément principal, on trouve les sous éléments suivants :

- `requestedDeliveryNotes` : contient l'information à transmettre en fonction des données utilisées pour effectuer l'entrée marchandise.
 - `deliveryBox barcode` : contient le numéro du bac scanné afin d'effectuer une entrée marchandise unique pour la livraison actuelle.
 -  Le numéro de code barre du bac contient 7 chiffres. Si le bac est scanné, 8 chiffres doivent apparaître dans le système POS car le code barre contient un 8^{ème} chiffre qui est une clé de contrôle (la clé de contrôle est renvoyée comme réponse dans le champ `deliveryBoxNumber`). Il est également possible d'envoyer 8 chiffres en incluant la clé de contrôle.
 - `deliveryNote number` : contient le numéro de bulletin de livraison scanné afin d'effectuer une entrée marchandise par bulletin de livraison.
 - `deliveryDay date` : contient la date de la livraison afin d'effectuer une seule entrée marchandise pour la date renseignée.

Sur l'environnement de test, aucune donnée de bulletin de livraison n'est disponible. Pour cette raison, l'attribut `useStaticTestResponse` peut être défini sur « true » dans l'élément principal `deliveryNoteDownloadRequest`. Lorsque cet attribut est à « true », il est possible d'interroger les bons de livraison de test portant les numéros : 100001, 100002, 200001, 200002, 300001, 400001, 500001.

4.9.2 Exemple de réponse XML à `deliveryNoteDownloadResponse`

XSD :

<https://xml.e-galexis.com/V2/schemas/deliveryNote/deliveryNoteDownloadResponse.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/deliveryNote/>

 Galexis renvoie des informations détaillées sur la livraison, y compris les données importantes des positions.

Galexis fournit l'ensemble des détails pertinents relatifs à la livraison tels que `<deliveryNoteDownloadResponse>` élément principal de la réponse, contenant `schemaLocation`, version et langue. Les éléments de réponses dépendent de la requête initialement transmise.

Pour un scannage de bulletin de livraison `deliveryNote number` :

- `<clientResponse>` confirmation des données client transmises dans la requête
- `<deliveryNoteData>` contient la date de livraison et la référence de la livraison
- `<order>` informations relatives à la commande
 - o `<ordreLine>` données relatives à la ligne de la commande comme la description & le pharmacode de l'article, les informations de prix ainsi que la quantité.
 - o `<deliveryBoxInformation>` fournit les informations de numéro de bac `physicalBoxNumber` et quantité livrée `lineDeliveryQuantity`
 - `<expiryAndBatchInformation>` fournit les informations de date d'expiration (pour les médicaments des listes A à E, éventuellement pour d'autres articles) et numéro de lot
- `<orderLine>` contient `Zone deliveryZone` si celle-ci a été transmise `commitShoppingCartReques` Valeurs autorisées : vide, CUSTOMER, ROBOT, SPECIAL & WEBSHOP.

Lors du scannage d'un numéro de bac ou de recherche par date, les informations complémentaires suivantes sont également fournies :

- `<deliveryNoteData>` contient les informations de date de livraison `deliveryDate`, numéro de tour `tourId`
- Si la valeur `referenceNmber` est transmise dans `shoppingCart`, le numéro de commande du POS est retourné.

! Les 8 chiffres du code barre (7 chiffres visibles sur le bac et numéro de clé de contrôle) sont systématiquement fournis dans la réponse, peu importe le nombre de chiffres fournis dans la requête.

Ainsi, l'utilisateur du POS peut intégrer automatiquement la livraison dans son système et gérer l'entrée de marchandise efficacement.

4.9.3 Exemple pratique – client avec intégration SOAP

L'exemple suivant illustre l'ensemble du processus de commande d'un client réel (client 224946) via l'interface POS v2 SOAP. L'intégration est réalisée au moyen du logiciel « XY » et couvre toutes les étapes, de la transmission du panier jusqu'à la consultation du bon de livraison.

Le client transmet son panier. Dans la requête, une adresse de livraison différente est fournie; celle-ci s'applique uniquement aux documents de livraison et ne modifie pas l'adresse de livraison enregistrée dans les données de base. La communication s'effectue en français et utilise l'ID du logiciel du provider PIS/POS. La requête est documentée dans `storeShoppingCartRequest.xml`, la réponse dans `storeShoppingCartResponse.xml` confirme l'enregistrement réussi du panier avec l'ID 623837.

storeShoppingCartRequest.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:galexis="http://www.galexis.com/XMLSchema">
  <SOAP-ENV:Body>
    <galexis:storeShoppingCartRequest version="2" language="fr"
      communicationSoftwareId="XY">
      <galexis:client number="007457" password="*****"/>
      <galexis:customerSetupOverride>
        <galexis:deliveryAddress line1="PHARMACIE ***"
          line4="CHEMIN DE ***" line5PostalCode="1206" line5City="GENEVE"/>
      </galexis:customerSetupOverride>

      <galexis:cartHeader referenceNumber="116064"/>
      <galexis:cartLines>
        <galexis:cartLine cartLineNumber="1" orderQuantity="1">
          <galexis:product><galexis:pharmaCode id="2474044"/></galexis:product>
        </galexis:cartLine>
        <galexis:cartLine cartLineNumber="2" orderQuantity="1">
          <galexis:product><galexis:pharmaCode id="7771215"/></galexis:product>
        </galexis:cartLine>
        <!-- weitere cartLine-Einträge gekürzt -->
        <galexis:cartLine cartLineNumber="7" orderQuantity="1">
          <galexis:product><galexis:pharmaCode id="7798958"/></galexis:product>
        </galexis:cartLine>
      </galexis:cartLines>
    </galexis:storeShoppingCartRequest>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

storeShoppingCartResponse.xml

```

<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <storeShoppingCartResponse xmlns="http://xml.e-galexis.com/V2/schemas/" language="fr"
version="2.0">
      <clientResponse customerNumber="224946" number="007457"/>
      <shoppingCart>
        <overriddenCustomerSettings>
          <deliveryAddress line1="PHARMACIE ****" line4="CHEMIN DE ****" line5City="GENEVE"
line5PostalCode="1206"/>
        </overriddenCustomerSettings>

        <cartIdentifier humanReadableCartNumber="623837" id="y2N+eQjPH9CqmWHS1fdkRQ==" />
        <cartHeaderResponse deliveryDate="2025-10-14" referenceNumber="116064"/>

        <cartLinesResponse>
          <cartLineResponse lineAccepted="true" deliveryDate="2025-10-14" deliveryZone="CUSTOMER"
originalLineNumberList="1">
            <productResponse deliveryQuantity="1" invoiceQuantity="1" lineAmount="00.01"
wholesalerProductCode="2474044" description="BURGERSTEIN Coenzyme Q10 caps 30 mg 180">
              <EAN id="7640121570377"/>
            </productResponse>
            <availability status="yes"/>
          </cartLineResponse>

          <!-- weitere cartLineResponse-Einträge gekürzt -->

          <cartLineResponse backlogLine="true" lineAccepted="true" deliveryDate="2025-10-15"
deliveryZone="CUSTOMER" originalLineNumberList="7">
            <productResponse deliveryQuantity="1" invoiceQuantity="1" lineAmount="00.01"
wholesalerProductCode="7798958" description="PADMED CIRCOSAN N caps 200 pce">
              <EAN id="7680675410026"/>
            </productResponse>
            <availability status="yes"/>
          </cartLineResponse>
        </cartLinesResponse>

        <cartTotals totalAmountExclVATExclExpLogServiceCosts="00.01"
totalExpectedLogServiceCosts="00.01"/>
      </shoppingCart>
    </storeShoppingCartResponse>
  </soapenv:Body>
</soapenv:Envelope>

```

A l'étape suivante, le panier est finalisé via le fichier `commitShoppingCartRequest.xml`, ce qui entraîne la création de la commande dans le système Galexis.

commitShoppingCartRequest.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:galexis="http://www.galexis.com/XMLSchema">
  <SOAP-ENV:Body>
    <galexis:commitShoppingCartRequest phoneCallDesired="false" version="2" language="fr"
      communicationSoftwareId="XY" productDescriptionDesired="false"
      compressionDesired="false">
      <galexis:client number="007457" password="*****"></galexis:client>
      <galexis:cartIdentifier id="y2N+eQjPH9CqmwHs1fdkRQ=="></galexis:cartIdentifier>
    </galexis:commitShoppingCartRequest>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

La réponse (`commitShoppingCartResponse.xml`) contient la confirmation „G: Ordre ELE 2200853330 sauvegardé(e).“, ce qui confirme que la commande a été sauvegardée avec succès.

commitShoppingCartResponse.xml

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <commitShoppingCartResponse xmlns="http://xml.e-galexis.com/V2/schemas/"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" language="de" version="2.0"
      xsi:schemaLocation="http://xml.e-galexis.com/V2/schemas/shoppingCart/commitShoppingCartResponse.xsd">
      <clientResponse customerNumber="224946" number="007457" />
      <cartIdentifier humanReadableCartNumber="623837" id="y2N+eQjPH9CqmwHs1fdkRQ==" />
      <successful message="G: Ordre ELE 2200853330 sauvegardé(e)." />
    </commitShoppingCartResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

La confirmation de commande correspondante affiche l'ensemble des articles commandés, les quantités, les prix ainsi que les dates de livraison (14 & 15 octobre 2025).

Confirmation de commande



Confirmation de la commande

Informations sur la commande Votre référence de commande: 116064 Date de commande: 14.10.2025 Date de livraison prévue: 14.10.2025 Notre numéro de commande: 2200853330 Date du document: 14.10.2025	Donneur d'ordre Numéro de client: 224946 Pharmacie () 1206 Genève Suisse	Destinataire de la marchandise Numéro de client: 224946 Pharmacie () 1206 Genève Suisse	Informations clients PHARMACIE () 1206 GENEVE Suisse
--	--	---	---

Articles livrables

N° pos	Article	Pharmacode	GTIN	Quantité	Prix de référence par pce	Stup Frigo	Cond. %	Valeur march. hors TVA	Commiss. Cond. / Net	Valeur de la ligne hors TVA	Statut / livrable au
10	BURGERSTEIN Coenzyme Q10 caps 30 mg 180 pce	2474044	7640121570377	1	20		11	65	0 14	79	
20	COPAXONE PEN sol inj 40 mg/ml stylo pré 12 x 1 ml	7771215	7680674920021	1	20	F	0	20	0 12	62	Livrable au 15.10.2025
30	ECOMUCYL Sandoz cpr eff 600 mg bte 10 pce	4935934	7680506550266	1	24		0	24	0 0	24	
40	ITULAZAX lyophilisat oral 12 SQ-Bet 90 pce	7776141	7680672750026	2	75		0	50	0 12	92	
50	KEPPRA cpr pel 250 mg 30 pce	2204557	7680552970049	3	07		0	21	0 11	82	
60	MALARONE Junior cpr pel 62.5/25 mg 12 pce	6639708	7680541500011	6	56		0	36	0 19	65	Livrable au 15.10.2025
70	PADMED CIRCOSAN N caps 200 pce	7798958	7680675410026	1	90		0	90	0 12	12	Livrable au 15.10.2025
Total articles livrables										16	

a,b,c,d = Produits stupéfiants, substances contrôles F = Frigo (reprise impossible) Toutes les valeurs sont en CHF

Pour toute question concernant votre commande, veuillez contacter notre service clients.

Cordialement

Après le traitement de la commande, le client peut récupérer le bulletin de livraison électronique correspondant en utilisant `deliveryNoteDownloadRequest.xml` en utilisant le numéro d'envoi 1001087188.

deliveryNoteDownloadRequest.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:galexis="http://www.galexis.com/XMLSchema">
  <SOAP-ENV:Body>
    <galexis:deliveryNoteDownloadRequest useStaticTestResponse="false" version="2" language="fr"
      communicationSoftwareId="ProPharma Systems AG" productDescriptionDesired="false"
      compressionDesired="false">
      <galexis:client number="007457" password="*****"></galexis:client>
      <galexis:requestedDeliveryNotes>
        <galexis:deliveryNote number="1001087188"></galexis:deliveryNote>
      </galexis:requestedDeliveryNotes>
    </galexis:deliveryNoteDownloadRequest>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

La réponse (*deliveryNoteDownloadResponse.xml*) contient les articles livrés, numéros de lot, dates d'expiration ainsi que les numéros de bacs.

deliveryNoteDownloadResponse.xml

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <deliveryNoteDownloadResponse xmlns="http://xml.e-galexix.com/V2/schemas/" language="fr"
version="2">
      <clientResponse customerNumber="224946" number="007457"/>

      <deliveryNoteData deliveryDate="2025-10-14" number="1001087188" tourId="LX67.2">

        <!-- Beispiel: vollständiger Order-Block -->
        <order number="2200845747" orderDate="2025-10-11" referenceNumber="116027">
          <orderLine pharmaCode="7843697"
            description="NUTROF Total Vit Spuren Omega 3 caps Vit D3 90 pce"
            invoicedQuantity="1" lineValue="00.01">
            <deliveryBoxInformation physicalBoxNumber="41086818" lineDeliveryQuantity="1">
              <expiryAndBatchInformation batchId="EXP20270531" expiryDate="2027-05-31"
partialQuantity="1"/>
            </deliveryBoxInformation>
          </orderLine>
          <!-- ggf. weitere orderLine-Zeilen dieses Orders ... -->
          <orderTotal totalBaseLinePriceNet="00.01" totalLineValue="00.01"/>
        </order>

        <!-- Hier kommen weitere <order> ... </order> Blöcke (z. B. Order 2, Order 3) -->

        <deliveryNoteTotal totallines="11" totalLogisticServiceCost="00.01"
          totalPieces="32" totalPrice="00.01"
          totalPriceNet="00.01" totalVolumeL="13.03" totalWeightKg="3.68"/>
      </deliveryNoteData>
    </deliveryNoteDownloadResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Le bulletin de livraison correspondant est disponible.

Bulletin de livraison☰ Galexis

Livraison sortante
LX67.2
14:30
14.10.2025

Numéro d'envoi 1001087188


Donneur d'ordre
Numéro de client: 224946
Pharmacie
Pharmacie
Chemin
CH-1206 GENÈVE

Destinataire de la marchandise
Numéro de client: 224946
Pharmacie
Pharmacie
Chemin d
CH-1206 GENÈVE

Informations clients

PHARMACIE

CH-1206 GENEVE

Votre référence de commande: 116027

Date/heure de commande: 11.10.2025 / 11.42 Heures

Notre numéro de commande: 2200845747

Peut être commandé chez Mepha/Teva: Rivaroxaban-Mepha® / vascular

N° pos	Article	Pharmacode	GTIN	Quantité	B	V°	Frigo Stup Cyto	FEP par unité	Retours Quantité / Motif
30	NUTROF Total Vit Spuren Omega 3 caps Vit	7843697	3662042009068	1		1		58	/

Les retours sont possibles dans un délai de 10 jours calendaires. Les CGV sous www.galexis.com s'appliquent. Coûts de livraison seront facturés séparément.

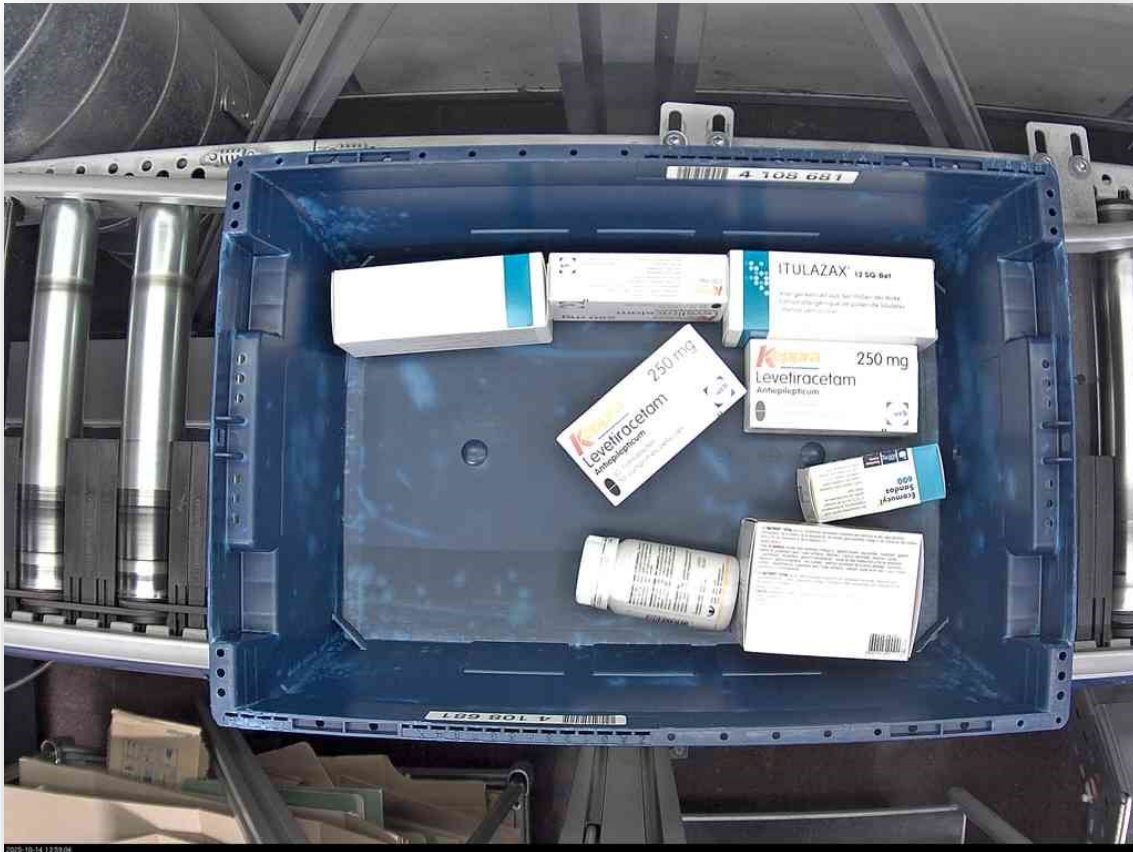
Motifs du retour: 01 Article non commandé et non facturé · 02 Saisie erronée · 03 Reçu endommagé · 04 Reçu trop tard · 05 Date de pérem. · 08 Quantité incorrecte · 10 Article avec date de pérem. · 12 Commandé par erreur

*Je confirme que les articles mentionnés sont légalement commercialisables et non ouverts, qu'ils ont été livrés par Galexis SA et qu'ils ont été stockés et manipulés correctement dès la livraison. En particulier, je confirme qu'ils n'ont pas quitté mon secteur de responsabilité entre temps.

Date, signature: _____

Galexis AG · Industriestrasse 2 · CH-4704 Niederbipp
Service client Tél. 0588517333

De plus, la photo ci-dessous montre le bac physiquement livré avec le numéro 41086818.



4.10 Consulter les articles en attente (Backlog)

Les commandes ouvertes sont des lignes commandées pour lesquelles Galexis attend encore une réception de marchandise afin de pouvoir les livrer au client. Ces commandes peuvent concerner des articles du stock standard (NOTA) ou des articles que Galexis ne tient pas en stock, mais peut se procurer auprès du fournisseur pour le client (BESO).

Pour les consulter, l'utilisateur PIS/POS utilise la méthode `openBacklogRequest`. Galexis renvoie des informations détaillées sur les commandes en attente via `openBacklogResponse`.

4.10.1 Exemple de demande XML à `openBacklogRequest`

XSD :

<https://xml.e-galexis.com/V2/schemas/openBacklog/openBacklogRequest.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/openBacklog/>



L'utilisateur du POS envoie une requête afin d'obtenir toutes les commandes ouvertes.

La requête contient l'identifiant du webclient et le mot de passe pour l'authentification, afin que Galexis puisse identifier correctement les commandes ouvertes du client concerné.

4.10.2 Exemple de réponse XML à `openBacklogResponse`

XSD :

<https://xml.e-galexis.com/V2/schemas/openBacklog/openBacklogResponse.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/openBacklog/>



Galexis livre une liste des commandes ouvertes, incluant les détails relatifs à chaque ligne de commande encore non livrée.

Die La réponse contient le pharmacode, la quantité encore en attente, la date d'entrée & le numéro de la commande pour chaque position ouverte. Ainsi, l'utilisateur du POS dispose à tout moment d'une vue complète de toutes les commandes encore non livrées.

4.11 Effectuer l'inventaire

L'utilisateur du POS aimerait lancer un inventaire, transmettre les résultats intermédiaires, clôturer l'inventaire et interroger son état actuel. Le système prend également en charge l'utilisation d'un lecteur de codes-barres hors-ligne, lequel est alimenté avec les données produit nécessaires. Les chapitres ci-dessous présentent les différents Use Cases de l'inventaire, accompagnés d'exemples de code.

Aperçu :

1. Commencer l'inventaire

- Exemple de requête XML : `getInventoryIdRequest`
- Exemple de réponse XML : `getInventoryIdResponse`

2. Transmettre le statut intermédiaire de l'inventaire

- Exemple de requête XML : `inventoryTransmissionRequest`
- Exemple de réponse XML : `inventoryTransmissionResponse`

3. Clôturer l'inventaire

- Exemple de requête XML : `closeInventoryRequest`
- Exemple de réponse XML : `closeInventoryResponse`

4. Consulter le statut de l'inventaire

- Exemple de requête XML : `inventoryStatusRequest`
- Exemple de réponse XML : `inventoryStatusResponse`

5. Scanner de codes-barres hors ligne

- Exemple de requête XML : `getProductBaseMasterDataDiffRequest`
- Exemple de réponse XML : `getProductBaseMasterDataDiffResponse`

4.11.1 Commencer l'inventaire

4.11.1.1 Exemple de demande XML : getInventoryIdRequest

XSD:

<https://xml.e-galexis.com/V2/schemas/inventory/getInventoryIdRequest.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/inventory/>

 Avec cette requête, l'utilisateur du POS demande un nouvel inventaire ID qui sera utilisé pour l'inventaire en cours.

L'utilisateur du POS ou le système POS, lance un nouvel inventaire en envoyant une requête à Galexis. Dans ce cadre, il transmet les données d'authentification du client. L'objectif est d'obtenir un nouvel ID d'inventaire servant de référence unique pour le processus d'inventaire en cours.

4.11.1.2 Exemple de réponse XML : getInventoryIdResponse

XSD:

<https://xml.e-galexis.com/V2/schemas/inventory/getInventoryIdResponse.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/inventory/>

 Galexis livre un inventaire ID unique que l'utilisateur du POS utilise pour la suite des échanges liés à cet inventaire.

Réponse :

Galexis confirme la création de l'inventaire et renvoie un ID d'inventaire unique. Cet ID est utilisé pour toutes les opérations ultérieures d'inventaire afin d'associer correctement les données à l'inventaire concerné.

4.11.2 Transmettre l'état intermédiaire de l'inventaire

4.11.2.1 Exemple de demande XML : inventoryTransmissionRequest

XSD :

<https://xml.e-galexis.com/V2/schemas/inventory/inventoryTransmissionRequest.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/inventory/>

 L'utilisateur du POS transmet des résultats intermédiaires des stocks comptés pour chaque produit (identifié par le pharmacode).

Au cours de l'inventaire, l'utilisateur POS (ou le système) envoie à Galexis des états intermédiaires. La requête contient l'identifiant d'inventaire, une liste des produits scannés incluant les quantités relevées ainsi que d'autres informations pertinentes. Cela permet une documentation continue et à jour de l'état actuel du stock.

4.11.2.2 Exemple de réponse XML à : inventoryTransmissionResponse

XSD:

<https://xml.e-galexis.com/V2/schemas/inventory/inventoryTransmissionResponse.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/inventory/>

 Galexis confirme la réception réussie des résultats intermédiaires.

Galexis accuse réception des données transmises et renvoie un message de statut actuel indiquant si les données ont été traitées correctement ou s'il existe des erreurs.

4.11.3 Clôturer l'inventaire

4.11.3.1 Exemple de demande XML : closeInventoryRequest

XSD:

<https://xml.e-galexis.com/V2/schemas/inventory/closeInventoryRequest.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/inventory/>

 L'utilisateur POS indique par cette requête que l'inventaire est terminé.

Une fois la saisie finalisée, l'utilisateur du POS envoie une requête à Galexis pour clôturer l'inventaire. La requête contient l'inventaire ID afin de permettre d'identifier et de finaliser clairement l'inventaire concerné.

4.11.4 Consulter le statut de l'inventaire

4.11.4.1 Exemple de demande XML : inventoryStatusRequest

XSD:

<https://xml.e-galexis.com/V2/schemas/inventory/inventoryStatusRequest.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/inventory/>

 Avec cette requête, l'utilisateur du POS interroge le statut actuel de l'inventaire.

L'utilisateur du POS ou le système POS interroge le statut actuel de l'inventaire en cours en transmettant l'inventaire ID. Cela permet de suivre l'avancement du processus ou de vérifier l'état de clôture.

4.11.4.2 Exemple de réponse XML : inventoryStatusResponse

XSD :

<https://xml.e-galexis.com/V2/schemas/inventory/inventoryStatusResponse.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/inventory/>

 Galexis renvoie le statut actuel de l'inventaire (par exemple "inProgress", "completed").

Galexis livre un message de statut contenant des informations sur l'avancée de l'inventaire, par exemple s'il est en cours, terminé ou en erreur.

4.11.5 Scanner de codes-barres hors ligne

L'utilisateur du POS peut utiliser un lecteur de codes-barres hors ligne afin de saisir rapidement les codes-barres des produits lors de l'inventaire sans dépendre du réseau. Cela améliore l'efficacité et permet de réaliser l'inventaire même dans les zones sans connexion réseau.

Fonctionnement

- **Préparation :** Le système POS télécharge les données de base actuelles des produits via la méthode `getProductBaseMasterDataDiff` (Master-Daten-Differenz) qui les transfère ensuite dans le lecteur de codes-barres.
- **Saisie hors ligne :** Le scanner utilise ces données localement pour reconnaître les codes-barres et afficher les informations des produits sans nécessiter de connexion directe avec Galexis.
- **Synchronisation :** Une fois la saisie d'inventaire effectuée, les données collectées sont retransmises du scanner vers le système POS puis envoyées à Galexis via les méthodes d'inventaire standard (`inventoryTransmissionRequest`).

4.11.5.1 Exemple de demande XML : getProductBaseMasterDataDiffRequest

XSD :

<https://xml.e-galexis.com/V2/schemas/getProductBaseMasterDataDiff/getProductBaseMasterDataDiffRequest.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/getProductBaseMasterDataDiff/>

 L'utilisateur du POS demande les modifications récentes des données de base des produits depuis la dernière synchronisation afin de mettre à jour le lecteur de codes-barres hors ligne.

L'utilisateur du POS (le système POS) initie une requête auprès de Galexis pour récupérer les changements de données de base des produits depuis la dernière synchronisation. Le système transmet le numéro client, les données d'authentification ainsi que l'horodatage de la dernière mise à jour. Cette requête garantit que seules les données modifiées sont transférées, ce qui permet d'économiser de la bande passante et du temps.

4.11.4.2 Exemple de réponse XML : getProductBaseMasterDataDiffResponse

XSD:

<https://xml.e-galexis.com/V2/schemas/getProductBaseMasterDataDiff/getProductBaseMasterDataDiffResponse.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/getProductBaseMasterDataDiff/>

 Galexis fournit les modifications de données de base des produits dont le lecteur de codes-barres hors ligne a besoin pour mettre à jour sa base de données locales.

Galexis renvoie en réponse une liste de toutes les modifications produites (nouveaux produits, actualisés ou supprimés). Ces données comprennent toutes les informations

essentielles, telles que le pharmacode, la description et les prix. Le système POS utilise ces informations pour maintenir à jour le lecteur de codes-barres hors ligne de sorte que celui-ci puisse fonctionner correctement même sans connexion réseau.

4.12 Envoi de colis

L'utilisateur POS souhaite envoyer des colis à d'autres clients Galexis via Galexis. Pour cela, il transmet les destinataires souhaités. Pour chaque destinataire, une référence peut également être fournie. La réponse contient ensuite la valeur du code-barre permettant au collaborateur POS de générer l'étiquette du colis.

Exemple d'une étiquette parcel :

<https://xml.e-galexis.com/V2/schemas/parcelService/doc/ParcelEtiquette.pdf>

4.12.1 Exemple de demande XML à getAddressLabelsRequest

XSD:

<https://xml.e-galexis.com/V2/schemas/parcelService/getAddressLabelsRequest.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/parcelService/>

 Le collaborateur du POS envoie à Galexis une requête contenant les destinataires souhaités ainsi qu'une référence optionnelle pour chaque destinataire.

L'élément principal `<getAddressLabelsRequest>` contient, en plus des attributs techniques relatifs à schemaLocation et namespace, les attributs suivants :

- `version` : version du message. La version 2.0 doit être utilisée.
- `language` : langue dans laquelle, par exemple, la description du produit doit être renvoyée. Si aucune langue n'est transmise, la langue définie dans le dossier client est utilisée.
- `communicationSoftwareId` : identification du logiciel ayant transmis la requête.

Dans le sous-élément `addressLabels`, il existe un élément `addressLabel` par destinataire.

Chaque `addressLabel` contient les éléments suivants :

- `recipient` : destinataire souhaité, identifié par le numéro client Galexis (`customerNumber`) ou par le GLN (`customerGLN`).
- `sendersReference` : référence pour chaque destinataire

4.12.2 Exemple de réponse XML à `getAddressLabelsResponse`

XSD:

<https://xml.e-galexis.com/V2/schemas/parcelService/getAddressLabelsResponse.xsd>

Documentation HTML avec des exemples de code et des documents complémentaires :

<https://xml.e-galexis.com/V2/schemas/parcelService/>

 Galexis confirme le traitement réussi de la requête. Pour chaque destinataire valide, une valeur de code-barre est renvoyée, permettant au collaborateur du POS de générer l'adresse pour le service de colis. Cela garantit un envoi fluide et sans erreur.

Un élément adresse `addressLabelResponse` est renvoyé pour chaque destinataire. Il contient les éléments suivants :

- `addressLabel` : informations demandées concernant le destinataire et la référence
- `senderAddress` : adresse de l'expéditeur
- `recipient` : numéro de client Galexis du destinataire
- `recipientAddress` : adresse du destinataire
- `parcelServiceBarcode` : valeur utilisée comme code-barre imprimé sur l'étiquette du colis

5 Tests & mise en production

Exercices de test :

- URL : <https://test.e-galexis.com/testPOS>
- Accès via un login de test (après prise de contact)

Mise en production :

- Après des tests concluants, l'accès au système live est accordé
- Contact: xm1@galexis.com
- [Simuler et tester l'approvisionnement avec l'interface Swagger U](#)