Description of the Electronic Invoice Service

version 4.0.4

Prepared by the IT Center of the LEPL Revenue Service

2012-2023

History of documentation:

Implemented change:	Date
1. get_waybills_v1 method will return waybills where the authorized user is listed as both the purchaser and the seller of the waybills with a specific company 2. get_waybills_v1 The accuracy of the method's parameters last_update_date_s and last_update_date_e has been changed from day-level precision to hour, minute, and second-level precision.	03.10.2024
Validation has been added for the goods_list field and its "child" fields in the save_waybill method.	15.05.2024
A new method get_waybills_v1 has been added – it returns waybills based on the update date	05.09.2023
A new field has been added to the seller's waybill registry method	13.05.2014
A new field has been added to the buyer's waybill registry method	13.05.2014
A new function has been added, featuring a filter field	16.05.2014
New methods have been added: Sending an electronic waybill to the carrier; Submitting waybill fields by the carrier; Closing the waybill by the carrier	1.07.2014
In accordance with Order N257 of the Minister of Finance, a new category for timber waybills has been added.	10.08.2015
Waybill confirmation has been added.	03.02.2016
In accordance with Order N122 of 15.06.2016 issued by the Minister of Finance, the form of the timber category waybill has been modified: Field 21 – "Marking Numbers" – has been removed, and a new Column VIII – "Marking Number" – has been added to the product registry	01.08.2016
Integration of the pharmaceutical (medication) waybill	15.03.2023

About documentation

The purpose of this document is to:

- Describe the process of electronic communication;
- Outline the technical requirements for electronic communication;
- Establish and agree upon the technical standard and protocol for electronic communication between the Revenue Service and the service user;
- Define and agree upon the terms and conditions of electronic communication

Definitions:

1. **Revenue Service** – The LEPL Revenue Service.
2. **Working Hours** – Unlimited.
3. **Seller** – A legal entity that is a user of the "Taxpayer Portal."
4. **Buyer** – A legal or natural person who receives the electronic waybill.
5. **Activation** – Acknowledgment of the start of transportation.
6. **Completed** – An electronic waybill that has been delivered to the buye

Connection to the Revenue Service's Information System and Method of Electronic Communication

Electronic communication is enabled via web services.

Web address: https://services.rs.ge/WayBillService/WayBillService.asmx

---

When testing the web service, you can use the following test data:

User:	tbilisi	User: satesto2
Password:	123456	Password: 123456
Identification Number:		**Identification Number:**
		: 12345678910
206322102		

process description:

The generation of an electronic goods waybill involves issuing, correcting, and canceling an electronic waybill (across five types), acknowledging the start of transportation, and maintaining a registry of issued electronic waybills.
It also includes retaining the existing product codes and nomenclature associated with the taxpayer, maintaining its registry, and registering transportation means designated for distribution.
Based on the form of goods delivery, there are five types of electronic goods waybills:

1. **Internal transportation**
2. **Delivery with transportation**
3. **Delivery without transportation**

(You can continue with the remaining two types, and I'll translate them as well.)

4. **Distribution** (includes main waybills and their dependent sub-waybills)
5. **Return of goods**

The electronic goods waybill utilizes reference data available in the Revenue Service's information system, including:

1. Buyer's (recipient's) identification/personal number
2. Buyer's (recipient's) name
3. VAT payer verification
4. Product names with barcode and code references, based on the individual taxpayer
5. Unit of measurement
6. Driver's personal number verification
7. Vehicle registration capability

After completing the electronic goods waybill, it is activated (by specifying the transportation start time), after which the electronic waybill is assigned a unique number. Once transportation is completed, it is possible (but not mandatory) to fill in the recipient's details and mark the electronic waybill as completed.

In the case of distribution, within a main goods waybill, it is possible to issue sub-waybills for specific buyers.

When transportation is carried out by a carrier company, the seller has the option to send two types of goods waybills (delivery with transportation and internal transportation) to the carrier. The carrier, upon receiving the waybills, fills in the fields assigned to them and activates the goods waybill by specifying the transportation start date. The carrier also has the ability to make modifications.

(Edited)

The carrier can modify the fields designated for them and, after the transportation is completed, finalize the goods waybill.

In the case of a timber goods waybill, it is necessary to specify the category as "Timber." Completing this type of waybill is mandatory for the following goods: round timber (logs), products of primary processing of round timber (logs) or woody plants, woody plants themselves, and trimmed round timber up to one meter in length. When filling out this specific goods waybill, additional fields must also be completed: the name of the document certifying the origin of the timber, its number, date of issue, and a registry of marking numbers.

In cases of timber goods export, the waybill must be confirmed by the customs checkpoint. After confirmation, no changes can be made to the waybill.

It is also possible to generate VAT invoices from goods waybills.

The seller is responsible for filling in the waybill, after which the buyer can either confirm or reject the entered information. The “Accepted” and “Disagree” functions are used solely for communication between the buyer and the seller.

Service User Administration:

The creation of a new service user is performed through the authorized user page of the Revenue Service’s electronic services. This involves setting a username, password, and defining the appropriate roles (permissions).

Service User Management Methods:

1. User Update:

public bool update_service_user(string user_name, string user_password, string ip, string name, string su, string sp)

Inputs:

- user_name – Username of the electronic declaration system user
- user_password – Password of the electronic declaration system user
- ip – IP address from which the services will be accessed
- name – Name of the object
- su – Service user
- sp – Service user password

Returns:

Logical Variable:

- true if the user data was successfully updated
- false if the update was not successful

2. Retrieving the List of Users

public XmlNode get_service_users(string user_name, string user_password) Input Parameters:

: user_name - Username of the electronic declaration system user

user_password – Password of the electronic declaration system user

inputs:

this kind Xml :

```
<ServiceUsers>
<ServiceUser>
<ID>62</ID>
<USER_NAME>tbilisi</USER_NAME>
<UN_ID>731937</UN_ID>
<IP>12.12.12.12</IP>
<NAME>test_wb</NAME>
</ServiceUser>
</ServiceUsers>
```

Where:

- ID – ID of the service user
- USER_NAME – Username of the service user
- UN_ID – Taxpayer's unique number
- IP – IP address
- NAME – Name of the store

3. Password Verification for a User

public bool chek_service_user(string su, string sp, out int un_id, out int s_user_id)

inputs:
su – service user ; sp – Password of service user

inputs:

un_id - Taxpayer's Unique Number;

s_user_id

- service user's ID-;

Retrieving Service Reference Data

The system utilizes the following types of reference data: excise goods codes, waybill type retrieval, product unit retrieval, and transportation type retrieval. These reference datasets ensure accurate, standardized, and validated input across the electronic service interfaces.

1. Excise Goods Codes

public XmlNode

`get_akciz_codes` (string su,

string sp) Input Parameters::

su – service user; sp

- service Password;

inputs:

this kind Xml :

```
<AKCIZ_CODES>
  <AKCIZ_CODE>
    <ID>224</ID>
    <TITLE>Ethyl Alcohol (Ethanol) (2207)_litre_ (2.6)</TITLE>
    <MEASUREMENT>litre</MEASUREMENT>
<SAKON_KODI>2207</SAKON_KODI> <AKCIS_GANAKV>2.6</AKCIS_GANAKV>
  </AKCIZ_CODE>
</AKCIZ_CODES>
```

where, ID – excize ID;

- TITLE – Name of the excise good (e.g., Ethyl Alcohol)
- MEASUREMENT – Unit of measurement (e.g., liters, kilograms)
- SAKON_KODI – Excise code assigned to the product
- AKCIS_GANAKV – Excise rate applied to the product (can be a fixed amount or percentage)

2. Retrieving Waybill Types

public XmlNode get_waybill_types(string su, string sp)

Inputs:

su – service users ; sp – service users

password;

Returns:

The following types Xml :

```
<WAYBILL_TYPES>
```

```
        <WAYBILL_TYPE>
            <ID>6</ID>
            <NAME>Sub-Waybill</NAME>
        </WAYBILL_TYPE>
    </WAYBILL_TYPES>
```

where: ID waybill type ID; NAME – type name;

3. Retrieving Product Units

public XmlNode get_waybill_units(string su, string sp)

inputs:

su – service user; sp – service user's password

Returns following type Xml :

```
<WAYBILL_UNITS>
  <WAYBILL_UNIT><ID>1</ID>
   <NAME>piece</NAME>
  </WAYBILL_UNIT>
</WAYBILL_UNITS>
```

Where, ID – unit's ID;

NAME - name;

4. Retrieving Transportation Types

public XmlNode get_trans_types(string su, string sp) inputs:

su – service user; sp -

service users password;

Returns following type of Xml :

```
<TRANSPORT_TYPES> <TRANSPORT_TYPE>
    <ID>1</ID>
    <NAME>name</NAME>
  </TRANSPORT_TYPE>
</TRANSPORT_TYPES>
```

where, ID – Transportation type

ID; NAME - name;

5. Retrieving Timber Types

public XmlNode get_wood_types(string su, string sp) unputs:

su -service user; sp – service user's

password

Returns following types of Xml :

```
<WOOD_TYPES>
  < WOOD_TYPES >
    <ID>1</ID>
    <NAME>name</NAME>
    <DESCRIPTION>name<
    /DESCRIPTION>
  </ WOOD_TYPES>
</ WOOD_TYPES >
```

Methods for Electronic Waybill Generation in the Service:

Core Functions of Electronic Waybills:

1. Saving a Waybill

public XmlNode save_waybill(string su, string sp, XmlNode waybill) inputs:

su – service user; sp -

service user's password ;

following type Xml :

```
<WAYBILL>
  <SUB_WAYBILLS></SUB_WAYBILLS>
  <GOODS_LIST>
    <GOODS>
      <ID>0</ID>
      <W_NAME>Medicine #001</W_NAME>
      <UNIT_ID>1</UNIT_ID>
      <UNIT_TXT>Medicine</UNIT_TXT>
        <QUANTITY>5</QUANTITY>
          <PRICE>4</PRICE>
        <STATUS>1</STATUS>
       <AMOUNT>20</AMOUNT>
     <BAR_CODE>psp101</BAR_CODE>
          <A_ID>0</A_ID>
       <VAT_TYPE>1</VAT_TYPE>
      <QUANTITY_EXT>5</QUANTITY_EXT>
      <WOOD_LABEL>asd</ WOOD_LABEL>
      <W_ID>1</ W_ID >
    </GOODS>
    <GOODS>
      <ID>109416</ID>
      <W_NAME>Medicine #002</W_NAME>
      <UNIT_ID>1</UNIT_ID>
    <UNIT_TXT>Medicine</UNIT_TXT>
<QUANTITY>5</QUANTITY> <PRICE>3</PRICE>
 <STATUS>1</STATUS> <AMOUNT>15</AMOUNT>
<BAR_CODE>psp102</BAR_CODE> <A_ID>0</A_ID>
      <VAT_TYPE>0</VAT_TYPE>
      <QUANTITY_EXT>5</QUANTITY_EXT>
      <WOOD_LABEL>asd</ WOOD_LABEL>
      <W_ID>1</ W_ID >
    </GOODS>
  </GOODS_LIST>
```

```
<WOOD_DOCS_LIST>
<WOODDOCUMENT>
<ID>0</ID>
<DOC_N>tree1</DOC_N>
<DOC_DATE>2012-03-09T12:15:21</DOC_DATE>
<DOC_DESC>license</ DOC_DESC >
<STATUS>1</STATUS> </WOODDOCUMENT>
<WOODDOCUMENT>
<ID>0</ID>
<DOC_N>tree2</DOC_N>
<DOC_DATE>2012-03-09T12:15:21</DOC_DATE>
<DOC_DESC>license</ DOC_DESC >
<STATUS>1</STATUS>
</WOODDOCUMENT>
</WOOD_DOCS_LIST>
<ID>0</ID>
<TYPE>2</TYPE>
<BUYER_TIN>12345678910</BUYER_TIN>
<CHEK_BUYER_TIN>0</CHEK_BUYER_TIN>
<BUYER_NAME>Giorgi Giorgadze</BUYER_NAME>
<START_ADDRESS>tbilisi, Saburtalo street</START_ADDRESS>
<END_ADDRESS>tbilisi, Gulua street</END_ADDRESS>
<DRIVER_TIN>1</DRIVER_TIN>
<CHEK_DRIVER_TIN></CHEK_DRIVER_TIN>
<DRIVER_NAME>Irakli</DRIVER_NAME>
<TRANSPORT_COAST>0</TRANSPORT_COAST>
<RECEPTION_INFO></RECEPTION_INFO>
<RECEIVER_INFO></RECEIVER_INFO>
<DELIVERY_DATE></DELIVERY_DATE>
<STATUS>1</STATUS>
<SELER_UN_ID>731937</SELER_UN_ID>
<PAR_ID></PAR_ID>
<FULL_AMOUNT>35</FULL_AMOUNT>
<CAR_NUMBER></CAR_NUMBER>
<WAYBILL_NUMBER>0000014722</WAYBILL_NUMBER>
<S_USER_ID>20350</S_USER_ID>
```

```
<BEGIN_DATE>2012-03-09T12:15:21</BEGIN_DATE>
<TRAN_COST_PAYER>1</TRAN_COST_PAYER>
<TRANS_ID>1</TRANS_ID>
<TRANS_TXT>truck</TRANS_TXT>
<COMMENT></COMMENT>
<CATEGORY><CATEGORY >
<IS_MED></IS_MED>
<WOOD_LABELS><WOOD_DOC_LABELS > Withdrawn 2016.08.01
</WAYBILL>
```

where:

ID – waybill's ID, inputs 0 If a new one is created TYPE

– Type of waybill;

BUYER_TIN – Buyer's personal or identification number

CHEK_BUYER_TIN – If foreigner: 0; if Georgian citizen: 1

BUYER_NAME – Buyer's name

START_ADDRESS – Transport starting address

END_ADDRESS – Transport destination address

DRIVER_TIN – Driver's personal number

CHEK_DRIVER_TIN – If foreigner: 0; if Georgian citizen: 1

DRIVER_NAME – Driver's name

TRANSPORT_COAST – Transportation cost

RECEPTION_INFO – Supplier information

RECEIVER_INFO – Receiver information

DELIVERY_DATE – Delivery date, filled in before completing an already existing waybill

STATUS – Waybill status:

0 – Saved , 1 – Activated , 2 - Completed

SELER_UN_ID – Seller's unique number; returned by chek_service_user

PAR_ID – Parent waybill ID (used for sub-waybills when TYPE = 6)

CAR_NUMBER – Vehicle number

BEGIN_DATE – Transportation start date

TRAN_COST_PAYER – Who pays for the transportation cost:

1 - Buyer

2 - Seller

TRANS_ID – Transport type ID
TRANS_TXT – Transport type (if "Other" is selected, then TRANS_ID = 4)
COMMENT – Comment or note
CATEGORY – Empty or 0: Regular; 1: Wood
IS_MED – Empty or 0: Regular; 1: Medication
WOOD_LABELS – Wood label number (e.g., log label)

ID – Item record ID in the waybill (pass 0 if creating a new item). If the tag/field is not passed or is empty, it defaults to 0
W_NAME – Product name (**required**)
UNIT_ID – Unit ID (**required**)
UNIT_TXT – Unit name (required only when UNIT_ID = 99, i.e., "Other")
QUANTITY – Quantity (**required**)
PRICE – Price (**required**)
STATUS – 1 or -1. If -1 is passed, the respective item will be deleted
AMOUNT – Total amount (**required**)
BAR_CODE – Barcode (in the case of medicine, the "Medicinal Product Registration Number") (**required**)
A_ID – Excise ID. If the item is not excisable, pass 0. If the tag/field is not passed or is empty, it defaults to 0

VAT_TYPE – Taxation type. If the tag/field is not passed or is empty, it defaults to 0
DOC_N – Document number for wood products
DOC_DESC – Document name for wood products
DOC_DATE – Document date for wood products

For issuing an **invoice** using the save_invoice function, pass the following values:

- 0: Regular
- 1: Zero-rated
- 2: Exempt from VAT

QUANTITY_EXT – Auxiliary quantity (optional field)

Returns following type of Xml “

```
<RESULT>
  <STATUS>0</STATUS>
  <ID>1551</ID>
<GOODS_LIST>
  <GOODS>
<ERROR>-1</ERROR>
    <ID>0</ID>
 <W_NAME>Bag</W_NAME>
   <UNIT_ID>99</UNIT_ID>
 <UNIT_TXT>test</UNIT_TXT>
  <QUANTITY>5</QUANTITY>
    <PRICE>4</PRICE>
  <AMOUNT>20</AMOUNT>
   <BAR_CODE>1</BAR_CODE>
        <A_ID/>
    <STATUS>1</STATUS>
      </GOODS>
    </GOODS_LIST>
      </RESULT>
```

where:

STATUS with a value of 0 means the waybill was saved successfully. A negative value indicates an error. Error details can be found using the get_error_codes function.

If STATUS indicates an error, the waybill will not be saved, and the goods list will also not be saved. In this case, the ERROR field will contain the relevant error information, and the list of goods that failed to save will appear in the GOODS_LIST field.

If the waybill is saved successfully, it will return the waybill ID, and the GOODS_LIST will contain all saved goods along with their record IDs.

2. Waybill Retrieva

public XmlNode get_waybill(string su, string sp, int waybill_id)

Inputs:

su – service user;

sp - service user’s password;

waybill_id waybill's ID inputs:

Returns following type XML :

```
<WAYBILL>
   <SUB_WAYBILLS>
     <SUB_WAYBILL>
       <ID>1670</ID>
       <WAYBILL_NUMBER>0000000002/1</WAYBILL_NUMBER>
     </SUB_WAYBILL>
   </SUB_WAYBILLS>
   <GOODS_LIST>
     <GOODS>
       <ID>2774</ID>
       <W_NAME>Sugar</W_NAME>
       <UNIT_ID>2</UNIT_ID>
        <QUANTITY>1000</QUANTITY>
            <PRICE>1</PRICE>
          <AMOUNT>1000</AMOUNT>
         <BAR_CODE>01</BAR_CODE>
       <QUANTITY_EXT>5</QUANTITY_EXT>
        <WOOD_LABEL>asd</ WOOD_LABEL>
              <W_ID>1</ W_ID >
     </GOODS>
   </GOODS_LIST>
   <WOOD_DOCS_LIST>
      <WOODDOCUMENT>
       <ID>0</ID>
       <DOC_N>tree1</DOC_N>
       <DOC_DATE>2012-03-09T12:15:21</DOC_DATE>
       <DOC_DESC>document</ DOC_DESC >
<STATUS>1</STATUS> </WOODDOCUMENT>
   </WOOD_DOCS_LIST>
   <ID>1669</ID>
   <TYPE>4</TYPE>
   <CREATE_DATE>2012-02-06T16:16:41</CREATE_DATE>
   <BUYER_TIN></BUYER_TIN>
<CHEK_BUYER_TIN>0</CHEK_BUYER_TIN>
    <BUYER_NAME></BUYER_NAME>
```

```
<START_ADDRESS>Tbilisi, saburtalo street</START_ADDRESS>
<END_ADDRESS></END_ADDRESS>
<DRIVER_TIN>11111111111</DRIVER_TIN>
<CHEK_DRIVER_TIN>0</CHEK_DRIVER_TIN>
<DRIVER_NAME>Bakhva Khorava</DRIVER_NAME>
<TRANSPORT_COAST>0</TRANSPORT_COAST>
<RECEPTION_INFO></RECEPTION_INFO>
<RECEIVER_INFO></RECEIVER_INFO>
<DELIVERY_DATE></DELIVERY_DATE>
<STATUS>-2</STATUS>
<SELER_UN_ID>731937</SELER_UN_ID>
<ACTIVATE_DATE>2012-02-06T16:16:41</ACTIVATE_DATE>
<PAR_ID></PAR_ID>
<FULL_AMOUNT>1000</FULL_AMOUNT>
<CAR_NUMBER>aaa555</CAR_NUMBER>
<WAYBILL_NUMBER>0000000002</WAYBILL_NUMBER>
<CLOSE_DATE>2012-02-06T16:22:43</CLOSE_DATE>
<S_USER_ID>0</S_USER_ID>
<BEGIN_DATE>2012-02-06T16:16:38</BEGIN_DATE>
<TRAN_COST_PAYER>2</TRAN_COST_PAYER>
<TRANS_ID>1</TRANS_ID>
<TRANS_TXT></TRANS_TXT>
<COMMENT>comment</COMMENT>
<CATEGORY >< CATEGORY >
<IS_MED >< IS_MED >
<WOOD_LABELS><WOOD_DOC_LABELS >
<CUST_STATUS>< CUST_STATUS >
<CUST_NAME>< CUST_NAME >
</WAYBILL>
```

where, ID – waybill unique code;

TYPE – waybill type; CREATE_DATE –

creation date;

BUYER_TIN - Buyer's personal or identification number

CHEK_BUYER_TIN - If foreign – 0, if Georgian citizen – 1

; BUYER_NAME – buyer's name;

START_ADDRESS – Transportation start address
END_ADDRESS – Transportation end address
DRIVER_TIN – Driver's personal number
CHEK_DRIVER_TIN – If foreign: 0, if Georgian citizen: 1
DRIVER_NAME – Driver's name
TRANSPORT_COAST – Transportation cost
RECEPTION_INFO – Supplier information
RECEIVER_INFO – Receiver information
DELIVERY_DATE – Delivery date
STATUS – Waybill status
SELER_UN_ID – Seller's unique number
ACTIVATE_DATE – Activation date
PAR_ID – Parent waybill ID (used for sub-waybills)
FULL_AMOUNT – Total amount of the waybill
CAR_NUMBER – Vehicle plate number
WAYBILL_NUMBER – Waybill number
CLOSE_DATE – Waybill close date
S_USER_ID – Service user ID
BEGIN_DATE – Transportation start date
TRAN_COST_PAYER – Transportation cost paid by: 1 – Buyer; 2 – Seller
TRANS_ID – Transportation type ID
TRANS_TXT – Transportation type (if "Other" is selected)
COMMENT – Comment
CUST_STATUS – 1: Confirmed by customs; 2: Rejected
CUST_NAME – Customs office name
CATEGORY – 1: Wood; 0 or empty: Regular

```
// GOODS
ID – Product ID (pass 0 if a new one is being created)
W_NAME – Product name

UNIT_ID – Unit ID
UNIT_TXT – Unit name
QUANTITY – Quantity
QUANTITY_EXT – Auxiliary quantity (required field)
PRICE – Price
AMOUNT – Total amount
BAR_CODE – Barcode
A_ID – Excise ID
W_ID – Wood type ID
VAT_TYPE – VAT type: 0 – Standard ; 1 Zero-rated; 2 – Exempt
```

// SUB_WAYBILL

ID – sub-waybill ID

WAYBILL_NUMBER – sub-waybill number

3. List of Waybills Issued by the Seller (Seller's Side)

```
public XmlNode get_waybills(string su, string sp, string itypes,
        string     buyer_tin,
        string        statuses,
        string car_number,
        DateTime begin_date_s,
        DateTimebegin_date_e,
        DateTime create_date_s,
        DateTime create_date_e,
        string driver_tin,
        DateTimedelivery_date_s,
        DateTime
        delivery_date_e,    decimal
        full_amount,            string
        waybill_number,
        DateTime      close_date_s,
        DateTime      close_date_e,
```

strings_user_ids, string

comment)

inputs:

su – Service user
sp – Service user password
types – Waybill types (comma-separated)
buyer_tin – Buyer's personal or identification number
statuses – Waybill statuses (comma-separated)
car_number – Vehicle (car) number
begin_date_s – Transportation start date range (start)
begin_date_e – Transportation start date range (end)
create_date_s – Waybill creation date range (start)
create_date_e – Waybill creation date range (end)
driver_tin – Driver's personal or identification number
delivery_date_s – Delivery date range (start)
delivery_date_e – Delivery date range (end)
full_amount – Total amount
waybill_number – Waybill number
close_date_s – Waybill close date range (start)
close_date_e – Waybill close date range (end)
s_user_id – Service user IDs (comma-separated)
comment – Comment

returns following type XML :

```
<WAYBILL_LIST>
  <WAYBILL>
    <ID>1668</ID>
    <TYPE>1</TYPE>
    <CREATE_DATE>2012-02-06</CREATE_DATE>
    <BUYER_NAME>Giorgi Giorgadze</ BUYER_NAME  >
```

```
<START_ADDRESS>Tbilisi, saburtalo street</START_ADDRESS>
<END_ADDRESS>Tbilisi, Nutsubidze street</END_ADDRESS>
<DRIVER_TIN>12345678910</ DRIVER_TIN >
<TRANSPORT_COAST>0</TRANSPORT_COAST>
<RECEPTION_INFO>Information</RECEPTION_INFO>
<RECEIVER_INFO>information</ RECEIVER_INFO >
<DELIVERY_DATE>2012-02-06</ DELIVERY_DATE >
<STATUS>0</STATUS>
<ACTIVATE_DATE>2012-02-06</ACTIVATE_DATE>
<PAR_ID></PAR_ID>
<FULL_AMOUNT>3</FULL_AMOUNT>
 <CAR_NUMBER>BBB123</ CAR_NUMBER >
<WAYBILL_NUMBER>0000000003</WAYBILL_NUMBER>
<CLOSE_DATE>2012-02-06</CLOSE_DATE>
<S_USER_ID>0</S_USER_ID>
<BEGIN_DATE>2012-02-06</BEGIN_DATE>
<WAYBILL_COMMENT>sdfg</ WAYBILL_COMMENT >
<BUYER_ST>0</ BUYER_ST >
  </WAYBILL>
 </WAYBILL_LIST>
```

where: Id – Waybill ID
type – Waybill type
create_date – Creation date
buyer_tin – Buyer's personal or identification number
buyer_name – Buyer's name
start_address – Transportation starting address
end_address – Transportation destination address
driver_tin – Driver's personal number
transport_coast – Transportation cost
reception_info – Supplier information
receiver_info – Receiver information
delivery_date – Delivery date
status – Waybill status
activate_date – Activation date
par_id – Parent waybill ID (in case of sub-waybill)
full_amount – Total amount

car_number – Vehicle number
waybill_number – Waybill number
close_date – Completion date
s_user_id – Service user ID
begin_date – Transportation start date
waybill_comment – Comment
buyer_st – (0) No status; (1) Micro business status; (2) Small business status

5. List of waybills received by the buyer (buyer's side)

```
public XmlNode get_buyer_waybills (string su, string sp, string itypes, string seller_tin,
                                   string statuses, string
                                   car_number,
                                   DateTime? begin_date_s,
                                   DateTime? begin_date_e,
                                   DateTime? create_date_s,
                                   DateTime?
                                   create_date_e,      string
                                   driver_tin,
                                   DateTime?   delivery_date_s,
                                   DateTime?   delivery_date_e,
                                   decimal? full_amount, string
                                   waybill_number,
                                   DateTime? close_date_s,
                                   DateTime?
                                   close_date_e,       string
                                   s_user_ids,         string
                                   comment)
```

inputs:

su – service use;

sp – service password;

itypes – Waybill types, separated by a comma ,

seller_tin – Seller's personal or identification number

statuses – Status values, separated by a comma , Example for a single status: ,1, Example for multiple statuses: ,1,2,

Status values: 0 – Saved 1 – Active 2 – Completed 8 – Sent to carrier -1 – Deleted -2 – Cancelled

car_number – carm number

begin_date_s - Start date interval – beginning

begin_date_e - Start date interval – ending

create_date_s - creation date interval – beginning

create_date_e - creation date interval – ending

driver_tin - Driver's personal or identification number

number_delivery_date_s – delivery date interval start

delivery_date_e – delivery date interval end

full_amount – total amount

waybill_number – waybill number

close_date_s – closing date interval start

close_date_e – closing date interval end

s_user_id – service user ID(s), separated by commas with ','

returns:

following type XML :

```
<WAYBILL_LIST>
```

```
<WAYBILL>
  <ID>1668</ID>
  <TYPE>1</TYPE>
  <CREATE_DATE>2012-02-06</CREATE_DATE>
  <BUYER_NAME>Giorgi giorgadze</BUYER_NAME >
  <SELLER_NAME>Irakli</ SELLER_NAME >
  <SELLER_TIN>12345678910</ SELLER_ TIN >
  <START_ADDRESS>Tbilisi, saburtalo street</START_ADDRESS>
  <END_ADDRESS>Tbilisi , nutsubidze street</END_ADDRESS>
  <DRIVER_TIN>12345678910</ DRIVER_TIN >
  <TRANSPORT_COAST>0</TRANSPORT_COAST>
  <RECEPTION_INFO>Information</RECEPTION_INFO>
  <RECEIVER_INFO>information</ RECEIVER_INFO >
  <DELIVERY_DATE>2012-02-06</ DELIVERY_DATE >
  <STATUS>0</STATUS>
  <ACTIVATE_DATE>2012-02-06</ACTIVATE_DATE>
  <PAR_ID></PAR_ID>
  <FULL_AMOUNT>3</FULL_AMOUNT>
  <CAR_NUMBER>BBB123</ CAR_NUMBER >
  <WAYBILL_NUMBER>0000000003</WAYBILL_NUMBER>
  <CLOSE_DATE>2012-02-06</CLOSE_DATE>
  <S_USER_ID>0</S_USER_ID>
  <BEGIN_DATE>2012-02-06</BEGIN_DATE>
  <WAYBILL_COMMENT>comment</ WAYBILL_COMMENT >
  <SELLER_ST>0</ SELLER_ST >
</WAYBILL>
</WAYBILL_LIST>
```

Where:
Id – Waybill ID
type – Waybill type
create_date – Creation date
buyer_tin – Buyer's personal or identification number
buyer_name – Buyer's name
seller_tin – Seller's identification number
seller_name – Seller's name

start_address – Transportation starting address

end_address – Transportation ending address

driver_tin – Driver's personal identification number

transport_coast – Transportation cost

reception_info – Supplier information

receiver_info – Recipient information

delivery_date – Delivery date

status – Waybill status

activate_date – Activation date

par_id – Parent waybill ID

full_amount – Total amount

car_number – Vehicle number

waybill_number – Waybill number

close_date – Waybill closing date

s_user_id – Service user ID

begin_date – Transportation start date

seller_st – (0) No status, (1) Micro business status, (2) Small business status

6. List of Waybills Issued by Seller (Seller's Side) with Filter Parameters:

```
public XmlNode get_waybills_ex (string su, string sp, string itypes,
                                string buyer_tin,
                                string statuses,
                                string
                                car_number,
                                DateTime? begin_date_s,
                                DateTime? begin_date_e,
                                DateTime? create_date_s,
                                DateTime? create_date_e,
                                string driver_tin,
                                DateTime? delivery_date_s,
                                DateTime? delivery_date_e,
                                decimal? full_amount,
                                string waybill_number,
```

```
DateTime? close_date_s,
DateTime? close_date_e,
string s_user_ids,
string comment,
int is_confirmed)
```

`is` passed:

su – service user

sp – service password

itypes – waybill types, separated by commas (",")

buyer_tin – buyer's personal or identification number

statuses – statuses, separated by commas (",")

car_number – vehicle number

begin_date_s – start date (range start)

begin_date_e – start date (range end)

create_date_s – creation date (range start)

create_date_e – creation date (range end)

driver_tin – driver's personal or identification number

delivery_date_s – delivery date (range start)

delivery_date_e – delivery date (range end)

full_amount – total amount

waybill_number – waybill number

close_date_s – close date (range start)

close_date_e – close date (range end)

s_user_id – service user IDs, separated by commas (",")

comment – comment

is_confirmed – 0: unconfirmed; 1: confirmed; -1: rejected

returns:

following type XML :

```
<WAYBILL_LIST>
  <WAYBILL>
    <ID>1668</ID>

    <TYPE>1</TYPE>
    <CREATE_DATE>2012-02-06</CREATE_DATE>
    <BUYER_NAME>Giorgi Giorgadze</ BUYER_NAME >
    <START_ADDRESS>Tbilisi, saburtalo street</START_ADDRESS>
    <END_ADDRESS>Tbilisi, nutsubidze street</END_ADDRESS>
    <DRIVER_TIN>12345678910</ DRIVER_TIN >
    <TRANSPORT_COAST>0</TRANSPORT_COAST>
    <RECEPTION_INFO>information</RECEPTION_INFO>
    <RECEIVER_INFO>Information</ RECEIVER_INFO >
    <DELIVERY_DATE>2012-02-06</ DELIVERY_DATE >
    <STATUS>0</STATUS>
    <ACTIVATE_DATE>2012-02-06</ACTIVATE_DATE>
    <PAR_ID></PAR_ID>
    <FULL_AMOUNT>3</FULL_AMOUNT>
    <CAR_NUMBER>BBB123</ CAR_NUMBER >
    <WAYBILL_NUMBER>0000000003</WAYBILL_NUMBER>
    <CLOSE_DATE>2012-02-06</CLOSE_DATE>
    <S_USER_ID>0</S_USER_ID>
    <BEGIN_DATE>2012-02-06</BEGIN_DATE>
    <WAYBILL_COMMENT>comment</ WAYBILL_COMMENT >
    <BUYER_STt>0</ BUYER_STt>
    < is_confirmed>-1</ is_confirmed>
  </WAYBILL>
</WAYBILL_LIST>
```

Where: **Id** – Waybill ID
type – Waybill type
create_date – Creation date
buyer_tin – Buyer's personal or identification number
buyer_name – Buyer's name
start_address – Transport starting address
end_address – Transport destination address
driver_tin – Driver's personal number
transport_coast – Transportation cost
reception_info – Supplier information
receiver_info – Receiver information

delivery_date – Delivery date
status – Waybill status
activate_date – Activation date
par_id – Parent ID (for sub-waybills)
full_amount – Total amount
car_number – Vehicle number
waybill_number – Waybill number
close_date – Closing date
s_user_id – Service user ID
begin_date – Transportation start date
waybill_comment – Comment
buyer_st – Buyer status:
(0) No status
(1) Micro business
(2) Small business

7. List of Waybills Received by the Buyer with Parameter-Based Filtering (Buyer's Side)

```
public XmlNode get_buyer_waybills _ex  (string su, string sp, string itypes, string
                                        seller_tin, string statuses,
                                        string car_number,
                                        DateTime? begin_date_s,
                                        DateTime? begin_date_e,
                                        DateTime?
                                        create_date_s,
                                        DateTime?
                                        create_date_e, string driver_tin,
                                        DateTime? delivery_date_s,
                                        DateTime? delivery_date_e,
                                        decimal? full_amount, string
                                        waybill_number,
                                        DateTime? close_date_s,
                                        DateTime?
                                        close_date_e,        string

                                        s_user_ids,
                                        string comment,
                                        int is_confirmed)
```

inputs:

su – service user
sp – service password
itypes – waybill types, separated by ","
seller_tin – seller's personal number or identification code
statuses – statuses, separated by ","
car_number – car number
begin_date_s – start date of the start date range
begin_date_e – end date of the start date range
create_date_s – start date of the creation date range
create_date_e – end date of the creation date range
driver_tin – driver's personal number or identification code
delivery_date_s – start date of the delivery date range
delivery_date_e – end date of the delivery date range
full_amount – total amount
waybill_number – waybill number

close_date_s – start date of the close date range
close_date_e – end date of the close date range
s_user_id – service user IDs, separated by ","
is_confirmed –
0 – unconfirmed;
1 – confirmed;
-1 – rejected

returns:

following type XML:

```
<WAYBILL_LIST>
  <WAYBILL>
    <ID>1668</ID>
    <TYPE>1</TYPE>
    <CREATE_DATE>2012-02-06</CREATE_DATE>
```

```
<BUYER_NAME>Giorgi giorgadze</ BUYER_NAME >
<SELLER_NAME>Irakli asatiani</ SELLER_NAME >
<SELLER_TIN>12345678910</ SELLER_ TIN >
<START_ADDRESS>Tbilisi, saburtalo street</START_ADDRESS>
<END_ADDRESS>Tbilisi, nutsubidze street</END_ADDRESS>
<DRIVER_TIN>11223344551</ DRIVER_TIN >
<TRANSPORT_COAST>0</TRANSPORT_COAST>
<RECEPTION_INFO>information</RECEPTION_INFO>
<RECEIVER_INFO>information</ RECEIVER_INFO >
<DELIVERY_DATE>2012-02-06</ DELIVERY_DATE >
<STATUS>0</STATUS>
<ACTIVATE_DATE>2012-02-06</ACTIVATE_DATE>
<PAR_ID></PAR_ID>
<FULL_AMOUNT>3</FULL_AMOUNT>
<CAR_NUMBER>BBB123</ CAR_NUMBER >
<WAYBILL_NUMBER>0000000003</WAYBILL_NUMBER>
<CLOSE_DATE>2012-02-06</CLOSE_DATE>
<S_USER_ID>0</S_USER_ID>
<BEGIN_DATE>2012-02-06</BEGIN_DATE>
<WAYBILL_COMMENT>comment</ WAYBILL_COMMENT >
<SELLER_ST>0</ SELLER_ST >
<IS_CONFIRMED>-1</ IS_CONFIRMED >
</WAYBILL>
</WAYBILL_LIST>
```

where: **Id** – waybill ID
type – waybill type
create_date – creation date
buyer_tin – buyer's personal or identification number
buyer_name – buyer's name
seller_tin – seller's code
seller_name – seller's name
start_address – transportation starting location
end_address – transportation ending location
driver_tin – driver's personal number
transport_coast – transportation cost
reception_info – supplier information

receiver_info – receiver's information

delivery_date – delivery date

status – waybill status

activate_date – activation date

par_id – parent ID (main waybill ID)

full_amount – total amount

car_number – car number

waybill_number – waybill number

close_date – close (completion) date

s_user_id – service user ID

begin_date – transportation start date

seller_st –

0 – no status

1 – micro business status

2 – small business status

8. Waybill List with Parameter Filters
public XmlNode get_waybills_v1(string su, string sp, string? buyer_tin, DateTime last_update_date_s, DateTime last_update_date_e)

inputs:
su – service user
sp – service password
buyer_tin – buyer's identification number
last_update_date_s – last update date (range start), e.g., 2024-10-07T12:08:47
last_update_date_e – last update date (range end)
for example: 2024-10-07T14:34:49

returns:

Returns a list of waybills within the specified time range using filters.
The response will include waybills that appear on the web both in the "Received" and "Issued" tabs, where the authorized user is listed as either the **buyer_tin** or the **seller_tin**.
In other words, it will return waybills related to both purchases and sales made by the user with a specific company.
Limitations:

The maximum time range is 3 days.

9. Waybill activation

public string send_waybill(string su, string sp, int waybill_id) inputs:

su – service user ;

sp – service users's password;

waybill_id - waybill ID

returns: waybill number

10. Waybill confirmation

public bool confirm_waybill (string su, string sp, int waybill_id) inputs:

su – service user ;

sp – service user's password;

waybill_id - waybill ID

returns:

true if confirmed

False if not confirmed

11. Waybill Activation with Transportation Start Date

public string send_waybill_vd (string su, string sp, DateTime begin_date, int waybill_id)

inputs:

su – service ;
sp - password;

begin_date - transportation begin date;

waybill_id - waybill ID

inputs:

waybill number

12. Waybill Completion

public int close_waybill(string su, string sp, int waybill_id) input:

su – service user ;

sp – service password;

waybill_id - waybill ID

returns:

- 1 – successfully closed
- -1 – not closed
- -101 – the waybill belongs to another user and cannot be closed
- -100 – invalid service user or password

13. Waybill Completion with Delivery Date

public int close_waybill_vd (string su, string sp,DateTime delivery_date ,int waybill_id)

input:

su – service user ;
sp – service password;

delivery_date – delivery date;

waybill_id - waybill ID

returns:

Return Values (Status Codes):

- 1 – waybill successfully closed
- -1 – waybill was not closed
- -101 – the waybill belongs to another user and cannot be closed
- -100 invalid service user or password

14. Waybill deletion

public int del_waybill(string su, string sp, int waybill_id)
input:

su – service user ;

sp – service password;

waybill_id - waybill ID

inputs:

1-closed; -1 not closed; -101 the waybill belongs to another user and cannot be deleted -100 invalid service user or password

15. Waybill Cancellation

public int ref_waybill (string su, string sp, int waybill_id) input:

su – service user ;

sp – service password ;

waybill_id - waybill ID

returns:

- 1 – waybill successfully closed
- -1 – waybill was not closed
- -101 – the waybill belongs to another user and cannot be closed
- -100 invalid service user or password

Waybill Activation When Transportation is Performed by a Carrier Company

1. Saving a Waybill by the Carrier

int save_waybill_transporter(string su, string sp, int waybill_id, string car_number, string driver_tin, int chek_driver_tin, string driver_name,
int trans_id, string trans_txt, string reception_info, string receiver_info)

input string su, string sp service user and password

where:

waybill_id waybill ID

car_number car number

driver_tin Driver's personal number

chek_driver_tin drivers residency

driver_name

trans_id transportation type id

trans_txt transportation type name

reception_info- reception information

receiver_inf-receiver information

2. Sending a Goods Waybill

int send_waybill_transporter(string su, string sp, int waybill_id, DateTime begin_date, out string waybill_number)

input:

string su, string sp service user and password where: waybill_id waybill id

begin_date transportation begin date

waybill_number – waybill number

3. waybill reject by client

Bool reject_waybill(string su, string sp, int waybill_id)

4. waybill completion

int close_waybill_transporter(string su, string sp, int waybill_id, string reception_info, string receiver_info, DateTime delivery_date)

input

string su, string sp service user and password where:

waybill_id waybill id

reception_info- reception information

receiver_inf- receiver information

delivery_date

XmlNode save_waybill(string su, string sp, XmlNode waybill)

Waybill – Xml
Tag TRANSPORTER_TIN Carrier company's identification code

Service Method for Generating an E-Invoice from an E-Waybill – Invoice Issuance

1. Invoice Issuance for Waybills

public int save_invoice(string su, string sp, int waybill_id, int in_inv_id, out int out_inv_id) input:

su - service user ;

sp - service password;

waybill_id - waybill ID;

in_inv_id invoice number or 0 if its new

input:

Return Values (for Invoice Issuance):

- out_inv_id – the ID/number of the generated invoice
- 1 – invoice successfully created
- -1 – invoice was not created
- -101 – the waybill belongs to another user and cannot be used to issue an invoice
- -100 – invalid service user or password

1. Saving the waybill template

public int save_waybill_tamplate(string su, string sp, string v_name, XmlNode waybill)

inputs:

su – service user; sp – service password;

v_name- template name;

waybill – template in XML which inputs save_waybill

returns:

1-saved; -1 not saved ;-100 The service user or the user password is incorrect

2. get waybill tamplates

public int get_waybill_tamplates(string su, string sp, out XmlNode waybill_tamplates)

inputs:

su – service user ; sp – service password;

returns:

waybill_tamplates XML Where the template names and IDs are located

1- Operation completed; -1 No; -100 The service user or the user password is incorrect

3. get waybill tamplate

public int get_waybill_tamplate(string su, string sp, int id, out XmlNode waybill_tamplate)

inputs:

su -service user; sp – service password; id - tamplates ID

returns:

waybill_template is the same as the XML that is passed to save_waybill. 1 - Operation completed; -1 - No; -100 - The service user or the user password is incorrect

4. delete waybill tamplate

public int delete_waybill_tamplate(string su, string sp, int id) inputs:

su – service user ; sp - service password; id - tamplates ID

returns:

1 - Operation completed; -1 - No; -100 - The service user or the user password is incorrect

Method of generating personal certificates

Product code registration (barcode database) (used only for working within the portal)

1. save bar-codes

public int save_bar_code(string su, string sp, string bar_code, string goods_name, int

unit_id, string unit_txt, int a_id)

inputs:

- **su** – Service user
- **sp** – Service user password
- **bar_code** – Product code (barcode)
- **goods_name** – Name of the product
- **unit_id** – Unit ID
- **unit_txt** – Unit (textual representation)
- **a_id** – Excise ID

Returns:

- **1** – Operation completed successfully
- **-1** – Operation failed / No
- **-100** – The service user or the user password is incorrect

2. delete bar-code

public int delete_bar_code(string su, string sp, string bar_code) inputs:

su – Service user ; **sp** – Service user password ;**bar_code** – Barcode

returns:

1 - Operation completed; -1 - No; -100 - The service user or the user password is incorrect

3. get bar-codes

public int get_bar_codes(string su, string sp, string bar_code, out XmlNode bar_codes)

inputs:

su – service user;

sp - service password

bar_code – barcode

Returns:

1 – Operation completed successfully
-1 – Operation failed / No
-100 – The service user or the user password is incorrect
bar_codes – XML structure that contains the following fields:
bar_code – Barcode
goods_name – Product name
unit_id – Unit ID
unit_txt – Unit (textual description)
a_id – Excise ID

Method for registering vehicles intended for distribution

1. save car numbers

public int save_car_numbers(string su, string sp, string car_number) inputs:

su – servise user;

sp – service password;

car_number – car number

Returns:

- **1** – Operation completed successfully
- **-1** – Operation failed / No
- **-100** – Invalid service user or user password

2. delete care number

public int delete_car_numbers(string su, string sp, string car_number) inputs:

su - service user;

sp – service password;

car_number – car number

Returns:

- **1** – Operation completed successfully
- **-1** – Operation failed / No
- **-100** – Invalid service user or user password

3. get car numbers

public int get_car_numbers(string su, string sp, XmlNode car_numbers) inputs:

su – service user ; sp – service password;

returns:

car_numbers xml – where are car numbers

Returns:

- **1** – Operation completed successfully
- **-1** – Operation failed / No
- **-100** – Invalid service user or user password

Additional functions:

1. Retrieving a name using an identification code or personal number public string get_name_from_tin(string su, string sp, string tin)

Parameters passed:

- **su** – Service user
- **sp** – Service user password
- **tin** – Taxpayer Identification Number or Personal ID number

Returns:

- Buyer's name

2. Error codes

public XmlNode get_error_codes (string su, string sp) passed:

su – service user;

sp – service password

returns:

Following type Xml

```
<WAYBILL_TYPES>
      <WAYBILL_TYPE>
            <ID>-1000</ID>
            <TEXT>TEXT</ TEXT >
            <TYPE>1</ TYPE >
      </WAYBILL_TYPE>
</WAYBILL_TYPES>
```

where, ID – error code ID;

TEXT – error name;

TYPE – 1 main waybill's, 2 Goods entries 3 Invoice Issuance Errors