Специфікація використання

Cashälot API Bridge

v. 4.4

Зміст

Зміст
Поняття і визначення
Загальна інформація
Апаратні та програмні вимоги

API ПРРО Cashälot з .NET інтерфейсом CashalotApiBridge
Відповідь на виконання методів
Порядок роботи обліково-розрахункової системи
з API бібліотекою ПРРО Cashälot
1. Встановлення параметрів
2. Відкриття зміни
3. Службове внесення грошових коштів в касу
4. Службова видача грошових коштів з каси
5. Фіскалізація чека продажу товарів, послуг
6. Проведення операції повернення товару
7. Визначення фіскального номера чека
8. Визначення локального номера чека по фіскальному номеру
9. Отримання XML-структури чека
10. Відображення фіскалізованого чека
11. Створення X-звіту
12. Закриття зміни та створення Z-звіту
13. Ручний перевод ПРРО в режим офлайн
14. Ручний перевод ПРРО в режим онлайн
15.Синхронізація залишків по товарам
16.Синхронізація товарів та залишків
ЛОМБАРДНІ ОПЕРАЦІЇ
Приклад реалізації функції для виклику API методів ПРРО Cashälot

Поняття і визначення

КЕП – кваліфікований електронний підпис.
Локальний номер чека – номер, призначений чеку системою виписки електронних касових чеків, що використовується продавцем.
Офлайн сесія – сукупність документів, створених в режимі офлайн, між припиненням і відновленням зв'язку ПРРО з ФСКО.
ПРРО – програмний реєстратор розрахункових операцій. Система виписки електронних касових чеків, що використовується продавцем.
Фіскальний номер чека(ФН) – номер, призначений чеку сервером ФСКО.
ФСКО – фіскальний сервер контролюючого органу.

Загальна інформація

Cashälot API Bridge використовується для інтеграції ПРРО з розрахунково-обліковими системами за допомогою підключення бібліотеки CashaLotApiBridge.dll. В даному документі приведено специфікацію методів бібліотеки Cashälot.

Інтеграція надає користувачам можливість, безпосередньо з розрахунково-облікової системи, скористатися наступним функціоналом програмного забезпечення Cashälot:

- Відкриття та закриття касової зміни;
- Створення та реєстрація чеків продажу/повернення;
- Створення та реєстрація чеків з типами оплати передоплата, післяплата, інтернет-продажа;
- Створення видаткового чека на основі чека продажу та його реєстрація;
- Формування та друк Z-звіту за даними фіскального сервера контролюючого органу (скорочено ФСКО)/X-звіту за даними ПРРО;
- Створення та реєстрація службових чеків;
- Службове внесення;
- Службова видача;
- Робота в режимі офлайн;
- Можливість відображення на екрані зареєстрованого чека на пристрої продавця з QR-кодом після успішної реєстрації;
- Друк зареєстрованого чека;
- Формування періодичних звітів.

Апаратні та програмні вимоги

Для оптимальної роботи ПРРО Cashälot рекомендовані наступні системні та програмні вимоги:

Операційна система

Програмний комплекс коректно функціонує на комп'ютерах з сучасними операційними системами Microsoft та компонентами, що входять до її складу. Рекомендовано використовувати останню версією ОС, наприклад, Windows 10 або Windows Server 2019 відповідно до потреб, з встановленими актуальними оновленнями.

Апаратне забезпечення:

- процесор з мінімальною тактовою частотою від 2 ГГц;
- оперативна пам’ять від 2 GB;
- вільне місце на жорсткому диску від 1,5 GB;
- кольоровий графічний дисплей;
- маніпулятор типу миша та клавіатура;
- встановлений український або російський мовний стандарт регіональних налаштувань операційної системи;
- підтримка кирилиці в операційній системі;
- доступ до поштового сервера;
- наявний зв’язок з фіскальним сервером контролюючого органу.

Мережа

Для роботи користувача необхідний доступ до мережі інтернет. Також, необхідно мати доступ до кабінету Cashälot, фіскального сервера контролюючого органу та до ресурсу центру сертифікації ключів. Наприклад, якщо Ви використовуєте ключі від “АЦСК Україна”, необхідно надати доступ до ресурсу: https://uakey.com.ua/ - 212.90.186.150 - 80/443. Якщо ж ви використовуєте КЕП (ЕЦП) інших кваліфікованих надавачів (АЦСК), то необхідно надати доступ до серверів TSP, OCSP, CMP відповідних центрів.

Загальний список ресурсів:

- https://load.cashalot.org.ua/update/ - 94.131.247.22:443 - Сервіс завантаження інсталяцій програми (дистрибутиви, оновлення);
- http://fs.tax.gov.ua:8609/fs/cmd - Сервіс API ДПС;
- https://my.cashalot.org.ua/ - 94.131.247.21:443 - Особистий кабінет Cashälot.

API ПРРО Cashälot з .NET інтерфейсом CashalotApiBridge

Підключення ПРРО до розрахунково-облікової системи відбувається за допомогою розробленої API бібліотеки ПРРО Cashälot **CashalotApiBridge.dll**, шлях до якої необхідно вказати для коректного налаштування. Користувачу необхідно розробити функції, що будуть транслювати виклик методів із бібліотеки Cashälot.

Зверніть увагу! *Виклик бібліотеки* CashalotApiBridge.dll *необхідно здійснювати виключно з кореневого каталогу встановленого екземпляру Cashälot.*

Як приклад у цьому документі наведено **C#** реалізацію у вигляді методу:

CallCashalotMethod (“Метод”, “Параметри”).

Оскільки будь-яка дія в ПРРО (відкриття/закриття зміни, відправка чеків тощо) здійснюється шляхом підписання блоку даних КЕП, то під час роботи необхідно вказати шлях до сертифікатів, вибрати користувача і здійснити авторизацію.

За допомогою наявних методів API бібліотека реалізує виконання наступних операцій ПРРО Cashälot:

- Відкриття зміни;
- Службове внесення грошових коштів в касу;
- Службове вилучення грошових коштів з каси;
- Проведення розрахункових операцій продажу товарів/послуг(Фіскалізація чека);
- Проведення операції повернення товару;
- Визначення номера фіскального чека за локальним номером чека в касі;
- Створення X-Звіту;
- Визначення касового локального номера фіскального чека;
- Закриття зміни та створення Z-Звіту.

Відповідь на виконання методів

Виконання методів повертає об’єкт **CashalotApiRsp**, що містить інформацію про результат роботи:

- **Ret –** ознака успішності виконання функції (bool);
- **ErrorString –** текст помилки (string);
- **Values –** масив параметрів (Dictionary<string, string>).

Приклад:

```
public class CashalotApiRsp
{
    //Ознака успішності виконання функції (true - успішно/false - не успішно)
    public bool Ret { get; set; } = false;
    //Рядок з текстом помилки
    public string ErrorString { get; set; } = String.Empty;
    //Масив параметрів, які можуть повернути функції АПІ
    public Dictionary<string, string> Values { get; set; }
}
```

Values, залежно від використаного методу, може містити такі параметри:

Назва параметра	Значення
ReceiptFiscalNumber	Фіскальний номер чека.
ReceiptLocalNumber	Локальний номер чека.
ShiftID	Ідентифікаційний номер поточної зміни.
OfflineMode	Ознака офлайн режиму.
Type	Тип документа.
Base64Str1251ReceiptXML	XML-структура фіскалізованого чека в кодуванні Base64.
ServiceInput	Загальна сума службових внесень по зміні.
ServiceOutput	Загальна сума службової видачі по зміні.
OrderCount	Кількість чеків продажу.
OrderExpCount	Кількість чеків повернення.
SumCash	Поточний залишок готівки в касі.
TotalSUM	Загальна сума продажу.
TotalSUMCash	Загальна сума продажу "Готівка".
TotalSUMCard	Загальна сума продажу "Картка".
TotalTAXSUM	Загальна сума ПДВ по продажу.
ReturnSUM	Загальна сума повернення.
ReturnSUMCash	Загальна сума повернення "Готівка".
ReturnSUMCard	Загальна сума повернення "Картка".
ReturnTAXSUM	Загальна сума ПДВ по продажу.
TotalTAX_<1>_<2>%_<3>	Деталізація податків по фіскальним чекам.
ReturnTAX_<1>_<2>%_<3>	Деталізація податків по чекам повернення.

Назви об'єктів податків чеків оплати TotalTAX та повернення ReturnTAX побудовані з використанням складових, що визначаються на основі карток проданих товарів:

Складова	Зміст складової	Можливі значення	Приклад
<1>	Літера податку зазначена в картках товарів	А, Б, В, ...	TotalTAX_А_20%_SUM ReturnTAX_Б_5%_TURNOVER TotalTAX_Є_10%_REM1OVEDTAXES
<2>	Відсоток податку	0, 5, 20, ...	
<3>	Сума	SUM	
	Обіг	TURNOVER	
	Знижка	DISCOUNTSUM	
	Виключені податки	REMOVEDTAXES	

Також повертаються службові повідомлення або графічні відображення чеків та звітів, які можна відправити на друк.

Порядок роботи обліково-розрахункової системи з API бібліотекою ПРРО Cashälot

Перед початком роботи каси виконується операція відкриття зміни. При відкритій касовій зміні доступні проведення розрахункових операцій, повернення товарів, службові внесення та видача грошових коштів, а також, створення X-звітів.

Для завершення касової зміни необхідно провести операцію закриття зміни, під час якої відбувається службове вилучення грошових коштів з каси і формується Z-звіт.

Всі перелічені операції додатково дозволяють відправку документів на друк.

Деякі методи ПРРО Cashälot, вказані у розділі загальна інформація, виконуються неявно під час обробки запитів від розрахунково-облікової системи і можуть повертати відповідні службові повідомлення (наприклад, перехід у офлайн-режим тощо).

---

1. Встановлення параметрів

Метод **SetParameter** призначений для встановлення значень параметрів налаштувань API бібліотеки. Рекомендовано встановити всі параметри на початку роботи. Дані параметри можна змінювати в процесі роботи з API. Наприклад, для того, щоб виконувати автоматичний друк тільки Z-звіту в процесі роботи програми.

public string SetParameter(string Name, string Value);

Кожен параметр повинен містити в собі:

Ім'я	Тип	Опис
Name	string	Ім'я параметра
Value	string	Значення параметра

Приклад:

```
instparameter Param = new instparameter();

Param.PathToCashalotDir = "C:\\ProgramData\\Cashalot\\Cashalot";
Param.DeviceIDFnRRO = "4000000000";

Object[] PathToCashalotDir = { "PathToCashalotDir", Param.PathToCashalotDir };
var ret = CallCashaLotMethod("SetParameter", PathToCashalotDir);

Object[] DeviceIDFnRRO = { "DeviceIDFnRRO", Param.DeviceIDFnRRO };
var ret1 = CallCashaLotMethod("SetParameter", DeviceIDFnRRO);
```

SetParameter складається з обов'язкових параметрів (виділені червоним) та не обов'язкових параметрів. Параметри повинні мати послідовність, як в таблиці опису параметрів. Тип всіх параметрів string.

Опис параметрів вказано в таблиці:

Ім'я параметра(Name)	Значення параметра(Value)
PathToCashalotDir	Шлях до каталогу з встановленим ПРРО Cashälot.
DeviceIDFnRRO	Фіскальний номер ПРРО.
NOAUTOUPDATE	Перевірка на наявність нової версії на сервері https://load.cashalot.org.ua/update/ та запит на оновлення програми Cashälot в момент початку роботи з API: "True" – заборонити оновлення; "False" – дозволити оновлення(за замовчуванням).
NOINTERFACEMODE	Ознака використання інтерфейсу РМК для авторизації та службових операцій видачі та внесення готівки в касу. "False" – при роботі з API викликається діалогове вікно. "True" – авторизація відбувається шляхом передачі параметрів авторизації PathToCertificate та PwdToCertificate. Вікна службових

	операцій видачі та внесення готівки не викликаються (проте, результат операції службового внесення/видачі готівки відображається).
PathToCertificate	Шлях до каталогу з файловим ключем користувача. Важливо: в папці повинні міститися ключ і сертифікат тільки одного користувача. Якщо касою користуються декілька касирів, необхідно помістити їх сертифікати і ключі в окремі каталоги.
PwdToCertificate	Пароль від ключа.
USETOKEN	Чи використовується токен для авторизації. "True" – використовується. "False" – не використовується (використовується файловий ключ). Важливо: параметр USETOKEN має бути заданий після параметрів PathToCashalotDir, DeviceIDFnRRO та NOINTERFACEMODE. Для пришвидшення завантаження програми рекомендовано не використовувати токени.
USECLOUDKEY	Чи використовується CloudKey для авторизації. "True" – використовується. "False" – не використовується. Параметр USECLOUDKEY має бути заданий після параметрів PathToCashalotDir, DeviceIDFnRRO та NOINTERFACEMODE.
NOAUTOOPENSHIFT	Перевірка стану зміни (відкрита або закрита). Якщо зміна закрита, при виклику будь-якої COM-функції зміна автоматично відкривається. "False" – автоматично відкривати зміну, якщо вона закрита. "True"– не відкривати автоматично зміну. В цьому випадку робота COM-функції буде перервана і результат виконання буде повернуто у вигляді помилки.
DEFAULTPRINTERNAME	Визначає принтер для друку при натисканні кнопки "Друк" в діалогових вікнах: 1) "" – пустий рядок, виводиться діалог вибору принтера; 2) "DEFAULTWINDOWS" – використовувати принтер Windows за замовчуванням; 3) "ім'я принтера чеків" – в параметр передається ім'я принтера чеків, який буде використовуватись для друку чеків в ПРРО Cashälot.
AUTOPRINTMODE	Параметр визначає автоматичну відправку на друк чека або службового звіту без відображення інформаційного вікна: "True" – відправляти на друк; "False" – не відправляти на друк. Якщо задано принтер за замовчуванням (DEFAULTPRINTERNAME), чек друкується одразу, в іншому випадку відображається вікно вибору принтера для друку.
NOPRINTMODE	Дозвіл на друк: "True" – заборонити друк; "False" – дозволити друк.
SHOWREPORTADDITIONALINFO	Виводити додаткову інформацію в нижній частині Z-звіту. В разі встановленого значення "True", відображаються такі поля: - Залишок готівки в касі; - Сума знижки; - Сума націнки. "True" - відображає додаткову інформацію; "False" – не відображає додаткову інформацію.

USEGOODSSTORAGEMODE	Параметр USEGOODSSTORAGEMODE визначає використання ведення складського обліку по товарам та автоматичну синхронізацію залишків по ним при використанні API Cashälot за умови активованого налаштування в кабінеті користувача в розділі «Склад - Налаштування - Облік товарів у ПРРО». Параметр може приймати наступні значення: «0» - програма працює в стандартному режимі без ведення обліку товарів в API та синхронізації залишків по товарам з кабінетом користувача. «1» - режим передбачає використання автоматичної синхронізації залишків по товарам та інформування користувача окремим повідомленням про перевищення їх ліміту під час спроби реєстрації фіскального чека. Після появи повідомлення користувачу доступна можливість відмовитись від подальшої обробки чека або погодитись та продовжити реєстрацію, але таким чином залишки по товарам будуть мати від'ємне значення кількості. «2» - режим передбачає використання автоматичної синхронізації залишків по товарам та інформування користувача окремим повідомленням про перевищення їх ліміту без можливості продовжити обробку та реєстрацію чека. Візуалізація повідомлення про перевищення ліміту залишків з можливістю продовжити/перервати процедуру фіскалізації чека, використовуючи перший режим параметра USEGOODSSTORAGEMODE, буде виникати також при ознаці використання режиму роботи без інтерфейса (NOINTERFACEMODE=true). *Кроки по практичному застосуванню параметра USEGOODSSTORAGEMODE 1. Імпортувати номенклатуру, яка використовується в API Cashälot, до кабінету користувача. Після імпорту необхідно перенести номенклатуру до еталонної групи. 2. В кабінеті користувача в модулі «Склад» сформувати та провести документ «Надходження» по необхідним товарним позиціям. 3. Налаштувати та встановити використання параметра USEGOODSSTORAGEMODE у необхідному режимі. 4. Виконати синхронізацію даних по товарам (номенклатурі) та залишкам товарів методом SyncGoodsWithBalance. Подальше оновлення інформації по залишкам відбувається в автоматичному режимі при відкриті зміни та реєстрації фіскального чека, а також в ручному режимі, з використанням методів синхронізації товарів та залишків по ним або окремо, виконавши синхронізацію залишків. 5. Сформувати та заповнити фіскальний чек даними по номенклатурі, для якої доступні залишки по товарам. **Зверніть увагу!** Коректна робота обраного параметра відбувається лише за умови, що в API Cashälot використовується одна й та сама номенклатура, що заведена в кабінеті користувача.

2. Відкриття зміни

Для відкриття касової зміни необхідно викликати метод **OpenShift** з параметром, у якому вказано **Фіскальний номер ПРРО.** Cashälot виконує запит щодо стану каси з обраним фіскальним номером, після чого створює XML повідомлення із призначеним локальним номером і типом «Відкриття зміни», засвідчує його кваліфікованим електронним підписом продавця і надсилає на ФСКО.

public string OpenShift(string rroFiscalNumber);

Приклад:

```
Object[] parameters = { "4000000000"/*фіскальний номер ПРРО*/};
var ret = CallCashaLotMethod("OpenShift", parameters);
```

При успішному виконанні метод повертає структуру **CashalotApiRsp** у вигляді JSON, де буде вказаний ID зміни. *Приклад:*

```
{"Ret":true,"ErrorString":"", "Values":{"ShiftID":"e8353c00-f9eb-4b4c-aebc-ad0e55fb9109"}}
```

---

3. Службове внесення грошових коштів в касу

Для проведення операції службового внесення грошових коштів необхідно викликати метод **ServiceInput** з параметрами, де буде вказаний **ФН ПРРО та сума службового внесення.**

public string ServiceInput(string rroFiscalNumber, string sumVal);

Приклад:

```
Object[] parameters = {
                "4000000000",    /* фіскальний номер ПРРО*/
                "150.25"         /* сума службового внесення*/
               };
var ret = CallCashaLotMethod( "ServiceInput", parameters );
```

Cashälot створює службове діалогове вікно для перевірки отриманих параметрів запиту і після підтвердження, відправляє відповідне XML повідомлення на сервер ФСКО.

При виконанні відкривається чек службового внесення та повертається **CashalotApiRsp**.

4. Службова видача грошових коштів з каси

Для проведення операції службової видачі грошових коштів необхідно викликати метод **ServiceOutput** з параметрами, де буде вказаний **ФН ПРРО та сума службової видачі.**

public string ServiceOutput(string rroFiscalNumber, string sumVal)

Приклад:

```
Object[] parameters = {
                "4000000000",    /* фіскальний номер ПРРО*/
                "150.25"         /* сума службової видачі*/
               };
var ret = CallCashaLotMethod( "ServiceOutput", parameters );
```

Cashälot створює службове діалогове вікно для перевірки отриманих параметрів запиту і після підтвердження, відправляє відповідне XML повідомлення на сервер ФСКО.

При виконанні відкривається чек службової видачі та повертається **CashalotApiRsp**.

---

5. Фіскалізація чека продажу товарів, послуг

Для виконання розрахункової операції продажу товарів або послуг необхідно викликати метод фіскалізації чека **FiscalizeCheck**, в якості параметрів якого передаються ФН каси та два JSON-рядки: **strJSONData** для списку товарів/послуг та **payDataJSON** для повної вартості по чеку.

public string FiscalizeCheck(string rroFiscalNumber, string strJSONData, string payDataJSON);

Опис strJSONData:

Структура містить массив **ReceiptLst**, кожен елемент якого містить параметри номенклатури та об'єкти **Comment**, **DocType**, **SaleOrderNumber**.

Опис ReceiptLst:

Поле	Значення
VendorCode*	Артикул.
Name	Найменування товару.
DocNumber**	Номер договору (при використанні ломбардних послуг).
CdUKDZED	Код УКТЗЕД.
CdDKPP	Код ДКПП.
GoodsType	Тип номенклатури. За замовчуванням товар, якщо параметр не задано. 1 – товар; 2 – послуга; 3 – ломбардна послуга **.
Barcode	Штрих-код.
UnitType*	Одиниця вимірювання.
Quantity*	Кількість товару.
Price*	Ціна одиниці товару (в форматі <Гривні, Копійки>).
Amount*	Кінцева сума по позиції чека (з урахуванням всіх націнок та знижок, у форматі <Гривні, Копійки>).
DiscountPrc	Відсоток знижки. Увага! При заповненні DiscountPrc, необхідно також обов'язково заповнювати та передавати DiscountSum
DiscountSum	Сума знижки на позицію.
VATRate	ПДВ у відсотках (від 0 до 99,99).
VATLetter	Літера ставки ПДВ.
IsPriceIncludeVAT	Для випадків, коли сума по позиції не містить суму ПДВ (true/false).
SumVAT	Сума ПДВ за одиницю товару (ігнорується, якщо вказати ознаку, що враховує ПДВ “true”).
IsExcise	Ознака підакцизного товару(true/false).
ExciseLetter	Літера ставки акцизного податку.
ExciseStampBarcode	Штрих-код марки акцизного податку. У випадку коли товарів більше ніж 1, по одній позиції номенклатури, то всі штрих-коди заповнюються через кому. Поле ігнорується, якщо ознака підакцизного товару IsExcise = false.
OtherParametrs	Додаткові параметри у вигляді XML таблиці.

* обов'язкові параметри при передачі даних товару.
** для Ломбардних операцій.

Зауваження! Для передачі кількості товару з дробовою частиною (зокрема, нульовою, наприклад, "2,00") допускається використовувати десятковий розділювач "." (крапка) або "," (кома). Якщо параметр не задано, або задано некоректно, його значення приймається рівним одиниці (1).

Параметр OtherParametrs передається в форматі XML з кодуванням UTF-8. Атрибути XML-рядка вказані в таблиці нижче.

Зазначимо, що параметри XML-рядка OtherParametrs мають пріоритет над параметрами, які передані в **ReceiptLst**.

Атрибути XML-рядка OtherParametrs:

Назва атрибута	Опис атрибута
Nomenclature *	Найменування товару.
UKTZED	Код УКТЗЕД.
VendorCode *	Артикул.
Barcode	Штрих-код.
Dimension *	Одиниця вимірювання.
Discount	Відсоток знижки. Увага! При заповненні Discount, необхідно також обов'язково заповнювати та передавати DiscountAmount
DiscountAmount	Сума знижки.
VAT *	Ставка ПДВ в відсотках.
VATLetter	Літера ставки ПДВ.
PriceIncludesVAT або PriceIncludeVAT	Для випадків, коли сума по позиції не містить суму ПДВ (true/false).
AmountVAT	Сума ПДВ за одиницю товару, у форматі <Гривні, Копійки>.
IsExcisable	Ознака підакцизного товару, True/False.
ExciseLetter	Літера ставки акцизного податку.
ExciseStampBarcode	Штрих-код марки акцизного податку. У випадку коли товарів більше ніж 1, по одній позиції номенклатури, то всі штрих-коди заповнюються через кому. Поле ігнорується, якщо ознака підакцизного товару IsExcise = false.

* обов'язкові параметри в рядку OtherParametrs, якщо не вказані відповідні явні параметри.

Приклад OtherParametrs:

```
<?xml version=\"1.0\" encoding=\"UTF-8\"?> <Table> <Record VAT=\"5\" IsExcisable=\"True\"
Nomenclature=\"Товар1\"      UKTZED=\"0202\"      VendorCode=\"1\"      Barcode=\"1452\"
Dimension=\"шт\"    Discount=\"0.54\"    DiscountAmount=\"0.25\"    PriceIncludeVAT=\"1\"
AmountVAT=\"0\" ExciseStampBarcode=\"11111, 22222, 33333\"/> </Table>
```

Об'єкт **Comment** використовується для додавання нефіскальної інформації:

Comment	Опціональний рядок нефіскальної інформації, коментар. Допускається використання одного коментаря на чек.

Об'єкт **DocType** є обов'язковим при проведенні ломбардних операцій, призначений для визначення типу документа (чека). Ломбардні операції.

DocType	Тип документа: 0 – чек продажу; 4 – ломбардний чек; За замовчуванням DocType = 0, якщо об'єкт не задано.

Об'єкт **SaleOrderNumber** (Номер замовлення/договору) потрібен для заповнення номера договору при проведенні інтернет продажу, передоплати та післяплати. Для відображення номера замовлення в післяплаті обов'язково потрібно заповнювати в **payDataJSON** фіскальний номер чека передоплати ParentReceiptFiscalNumber.

Опис payDataJSON:

Поле	Значення
SumCash	Сума наданої готівки.
SumPayByCard	Сума оплати чека банківською картою.
SumPayByCredit	Сума оплати чека в кредит.
SumPayByCertificate	Сума оплати чека сертифікатом.
SumPayCheck	Сума чека до оплати з урахуванням всіх надбавок, знижок та округлень.
PaymentOrderType	Встановлення порядку сплати: 0 - звичайна; 1 - передоплата; 2 - післяплата; 3 - інтернет продаж.
SumPreparePayed	Сума попередньої оплати.
ParentReceiptFiscalNumber	Фіскальний номер чека, по якому здійснюється доплата/повернення.
ParentRROFiscalNumber	Фіскальний номер РРО, по якому здійснюється доплата/повернення.
CustomerEmail	Опціонально. Електронна адреса покупця, на яку можна відправити фіскалізований чек, передається в поле електронної адреси при натисканні кнопки "Надіслати чек на Email" екрану відображення чека. Рекомендовано використовувати при встановленому параметрі відображення вікон інтерфейсу: SetParameter("NOINTERFACEMODE","False").
OtherParametrs	Додаткові параметри у вигляді XML таблиці.
Необов'язкові параметри інформації по транзакції термінала при безготівкових розрахунках:	
TerminalID	Код термінала.
ApprovalCode	Код підтвердження.
RRN	РРН транзакції.
IssuerName	Платіжна система.
PAN	Номер картки.
TransactionDate	Дата транзакції (в форматі dd.mm.yyyy hh:mm:ss).
SignVerif	Електронний підпис.
AcquireName	Термінал.
InvoiceNumber	Номер чека.
ParentRRN	РРН транзакції чека, по якому здійснено повернення.

Увага! Різниця між сумою оплати чека та SumPayCheck (сумою по всіх видах оплати) вважається значенням заокруглення. Тому розробник має самостійно реалізувати алгоритм округлення значення виданої готівки або суми оплати по чеку.

Заокруглення суми по чеку реалізовано наступним чином:

В **payDataJSON** передається значення готівки, що надана клієнтом на касі, значення оплати картою, сертифікатом та кредитом, а також заокруглена сума, розрахована за певним алгоритмом користувача SumPayCheck (відповідно до Постанови НБУ п.4, 1-4 коп заокруглюється до 0 коп, 5-9 коп заокруглюється до 10 коп). В чеку буде відображено заокруглення, надану готівку та інші форми оплати, заокруглену суму до сплати, решту.

Значення заокруглення, яке буде зареєстроване на сервері ФСКО та відображено в чеку, розраховується по формулі:

Сума заокруглення = SumPayCheck - ***Загальна сума товарів***

Загальна сума товарів – це сума рядків Amount, вказаних в **strJSONData**. Тобто:

Загальна сума товарів = *Сума товару 1(*Amount*) + Сума товару 2(*Amount*) +* ...

Решта по чеку розраховується таким чином:

Сума решти = SumCash + SumPayByCard + SumPayByCredit + SumPayByCertificate – SumPayCheck

Решта розраховується програмою Cashälot автоматично, згідно вказаних користувачем даних.

*Приклад виконання методу **FiscalizeCheck**:*

```
        private void btnFiscalizeReceipt_Click( object sender, EventArgs e )
         {
             //Майже всі поля задаються як строки(для сумісності з різними мовами програмування, щоб уникнути проблем з конвертацією)
             LstCheckInfoRsp lstGoodsInCheck = new LstCheckInfoRsp();
             lstGoodsInCheck.ReceiptLst = new List<CheckInfo>();
             CheckInfo chInfo = new CheckInfo();
             chInfo.VendorCode = "АртикулТовару1";
             chInfo.Name = "НазваТовару1";
             chInfo.CdUKDZED = String.Empty;
             chInfo.Barcode = "123456789";
             chInfo.UnitType = "шт";
             chInfo.Quantity = "1";
             chInfo.Price = "20000.03";
             chInfo.Amount = "19000.03";
             chInfo.DiscountPrc = null;
             chInfo.DiscountSum = "-1";
             chInfo.IsPriceIncludeVAT = true;
             chInfo.VATRate = "20";
             chInfo.SumVAT = "3800.01";
             chInfo.IsExcise = true;
             chInfo.ExciseStampBarcode = "AFFD6723344";
             //chInfo.OtherParametrs = "ХМЛ структура реквізитів, якщо реквізит не заповнено в основній структурі, та заповнено в ХМЛ, значення беруться з ХМЛ"
             //chInfo.OtherParametrs = "<?xmlversion=\"1.0\"encoding=\"UTF-8\"?><Table><Record VAT=\"20\" IsExcisable=\"False\" Nomenclature=\"Товар1\" UKTZED=\"000001\" VendorCode=\"79078\" Barcode=\"1234567890\" Dimension=\"шт\" Discountamount=\"1.83\" PriceIncludeVAT=\"1\" AmountVAT=\"30.2\"/></Table>";

             //додати рядок чека з товаром №1
             lstGoodsInCheck.ReceiptLst.Add( chInfo );

             CheckInfo chInfo = new CheckInfo( chInfo );
             chInfo2.VendorCode = "АртикулТовару2";
             chInfo2.Name = "НазваТовару2";
             chInfo2.Barcode = "9999999999";
             chInfo2.VATRate = null;
             chInfo2.SumVAT = null;
             chInfo2.ExciseStampBarcode = "AFFD55223344,AAFF667788";
             chInfo2.DiscountSum = "2";
             //chInfo2.OtherParametrs = "<?xmlversion=\"1.0\"encoding=\"UTF-8\"?><Table><Record VAT=\"20\" IsExcisable=\"False\" Nomenclature=\"Товар2\" UKTZED=\"000001\" VendorCode=\"79078\" Barcode=\"1234567890\" Dimension=\"шт\" Discountamount=\"1.83\" PriceIncludeVAT=\"1\" AmountVAT=\"30.2\"/></Table>";
             //додати строку чеку з товаром №2
             lstGoodsInCheck.ReceiptLst.Add( chInfo2 );

             PayInfoRsp payInfo = new PayInfoRsp();
             payInfo.SumCash = "20000";    //Сума готівки наданої покупцем( може бути більшою за сумму до сплати, з неї вираховується решта/здача)

             payInfo.SumPayByCard = "19000";//Сума оплати картою

             //-----------------------------------------{
```

```
            // Інформація по транзакції терміналу:
            //Код теміналу
            payInfo.TerminalID = "S1K700J8";
            //Код підтвердження
            payInfo.ApprovalCode = "68014B";
            //РРН транзакції
            payInfo.RRN = "048867173620";
            //Платіжна система
            payInfo.IssuerName = "MASTERCARD";
            //Номер картки
            payInfo.PAN = "XXXXXXXXXXXX6883";
            //Дата транзакції(в форматі dd.MM.yyyy HH:mm:ss)
            payInfo.TransactionDate = DateTime.Now.ToString( "dd.MM.yyyy HH:mm:ss" );
            //Електроний підпис
            //payInfo.SignVerif="12334444";
            //Термінал
            payInfo.AcquireName = "Приват Банк";
            //Номер чека
            payInfo.InvoiceNumber = "224";
            //-----------------------------------------}

            payInfo.SumPayCheck = "38000.10";//Сума оплати загальна з урахуванням заокруглення
(+0,04 грн)
            payInfo.CustomerEmail = edCustomerEmail.Text;
            //Конвертація структур до формату джейсон строки
            string strJSONData = JsonConvert.SerializeObject( lstGoodsInCheck );
            string payDataJSON = JsonConvert.SerializeObject( payInfo );


            //payDataJSON =
"{\"SumCash\":\"240\",\"SumPayByCard\":\"0\",\"SumPayByCredit\":\"0\",\"SumPayByCertificate\":\"0
\",\"SumPayCheck\":\"239,64\"}";
            //Параметри функції
            Object[] parameters =
            {
                edFiscalNumberRRO.Text.Trim(), // фіскальний номер РРО
                strJSONData, // структура інформації зі строками чека
                payDataJSON  // структура інформації про оплату по чеку
            };

            var ret = CallCashaLotMethod( "FiscalizeCheck", parameters );
            if ( ret?.Ret != true )
            {
                MessageBox.Show( String.IsNullOrEmpty( ret?.ErrorString ) ? "Не вдалося
фіскалізувати чек" : ret?.ErrorString );
            }
            else if ( ret?.Values != null )
            {
                //отримання параметра в результаті виконання функції
                var vals = ret.Values.ToArray();
                //перебираемо параметри
                for ( int i = 0; i < vals.Length; i++ )
                {
                    // отримуємо значення по ключу
                    if ( vals[i].Key.ToLower() == "receiptlocalnumber" )
                    {
                        edLocalCheckNum.Text = vals[i].Value;
                    }
                    else
                    if ( vals[i].Key.ToLower() == "receiptfiscalnumber" )
```

```
                {
                    edFiscalCheckNum.Text = vals[i].Value;
                }
            }
            MessageBox.Show( $"Чек було успішно фіскалізовано.\n" +
                             $"Чеку присвоєно локальний номер:{edLocalCheckNum.Text},\n" +
                             $"та фіскальний номер: {edFiscalCheckNum.Text} "
                            );
        }
    }
```

У відповіді на **FiscalizeCheck** повертається **CashalotApiRsp**, в якому заповнений масив параметрів **Values**, що містить:

ReceiptFiscalNumber	Фіскальний номер чека.
ReceiptLocalNumber	Локальний номер чека.
ShiftID	Ідентифікаційний номер поточної зміни.
OfflineMode	Ознака офлайн режиму.
Type	Тип документа.
Base64Str1251ReceiptXML	XML-структура фіскалізованого чека в кодуванні Base64.

Після виконання запиту Cashälot створює діалогове вікно з графічним відображенням результатів реєстрації продажу у вигляді чека.

Тип документа *"Type"* визначається числом, яке формується наступним чином:

1<Клас документа><Тип документа><Додатковий тип документа>

Опис *"Type"* :

Число	Кількість знаків	Значення
<Клас документа>	1	0 - Чек 1 - Z-звіт 2 - X-звіт
<Тип документа>	3	000 - Чек реалізації товарів/послуг 001 - Чек переказу коштів 002 - Чек операції обміну валюти 003 - Чек видачі готівки 100 - Відкриття зміни 101 - Закриття зміни 102 - Початок офлайн сесії 103 - Завершення офлайн сесії
<Додатковий тип документа>	3	000 - Касовий чек (реалізація) 001 - Видатковий чек (повернення) 002 - Чек операції «службове внесення»/«отримання авансу» 003 - Чек операції «отримання підкріплення» 004 - Чек операції «службова видача»/«інкасація» 005 - Чек сторнування попереднього чека

6. Проведення операції повернення товару

Для виконання повернення товару чи товарів з фіскального чека з відомим (визначеним) фіскальним номером використовується метод **FiscalizeReturnCheck**, для якого формується чек повернення аналогічно операції фіскалізації чека продажу товарів, послуг із зазначенням тих товарів, які реєструються для повернення. Тобто, створюються **strJSONData** для списку товарів повернення, **payDataJSON** для повної вартості покупок та додатково зазначається параметр з заповненим **ФН чека, по якому здійснюється повернення**:

Параметр	Значення	Приклад	Примітка
strFiscalNumReturnCheck	Фіскальний номер чека, по якому здійснюється повернення	“275301” “19255.3.7284”	онлайн-чек офлайн-чек

public string FiscalizeReturnCheck(string rroFiscalNumber, string strJSONData, string payDataJSON, string strFiscalNumReturnCheck)

Приклад:

```
// Відправка чека повернення
LstCheckInfoRsp lstGoodsInCheck = new LstCheckInfoRsp();
lstGoodsInCheck.ReceiptLst = new List<CheckInfo>();
CheckInfo chInfo = new CheckInfo();
chInfo.VendorCode = "АртикулТовару1";
chInfo.Name = "НазваТовару1";
chInfo.CdUKDZED = String.Empty;
chInfo.Barcode = "123456789";
chInfo.UnitType = "шт";
chInfo.Quantity = "1";
chInfo.Price = "183";
chInfo.Amount = "181,17";
chInfo.DiscountPrc = String.Empty;
chInfo.DiscountSum = "1.83";
chInfo.IsPriceIncludeVAT = true;
chInfo.VATRate = "20";
chInfo.SumVAT = "30.2";
lstGoodsInCheck.ReceiptLst.Add( chInfo );
CheckInfo chInfo = new CheckInfo();
chInfo.VendorCode = "АртикулТовару2";
chInfo.Name = "НазваТовару2";
chInfo.Barcode = "9999999999";
lstGoodsInCheck.ReceiptLst.Add( chInfo );
PayInfoRsp payInfo = new PayInfoRsp();
payInfo.SumCash = "366";
payInfo.SumPayCheck = "362,34";
string strJSONData = JsonConvert.SerializeObject( lstGoodsInCheck );
string payDataJSON = JsonConvert.SerializeObject( payInfo );
Object[] parametersChRt =
{
    edFiscalNumberRRO.Text.Trim(),
    strJSONData,
    payDataJSON,
    edFiscalCheckNum.Text /*фіскальний номер чека, по якому здійснюється повернення*/
};
var retChRt = CallCashalotMethod( "FiscalizeReturnCheck", parametersChRt );
if ( !retChRt.Ret )
{
    MessageBox.Show( String.IsNullOrEmpty( retChRt.ErrorString ) ? "Не вдалося здійснити
повернення" : retChRt.ErrorString );
}
```

Після виконання методу Cashälot створює діалогове вікно з графічним відображенням результатів реєстрації повернення товару у вигляді чека та повертається **CashalotApiRsp**.

7. Визначення фіскального номера чека

З метою визначення ФН чека, зареєстрованого раніше в ФСКО, за локальним номером, що зареєстрований в касі при продажу, виконується метод **GetReceiptFiscalNumberByLocalNumber** з параметрами, де буде вказаний **ФН ПРРО** та **локальний номер чека.**

public string GetReceiptFiscalNumberByLocalNumber(string rroFiscalNumber,string checkLocalNumber);

Приклад:

```
Object[] parameters = {
                "4000000000",    /* фіскальний номер ПРРО*/
                "1554"        /* локальний номер чека*/
               };
var ret = CallCashaLotMethod( "GetReceiptFiscalNumberByLocalNumber", parameters );
```

Визначений фіскальний номер чека використовується, зокрема, в якості додаткового параметра при фіскалізації чека повернення товару.

---

8. Визначення локального номера чека по фіскальному номеру

В деяких випадках виникає необхідність визначення локального номера чека, зареєстрованого в касі, по відомому фіскальному номеру чека. Для визначення локального номера чека по ФН використовується метод **GetReceiptLocalNumberByFiscalNumber** з параметрами, де заповнені **Фіскальний номер ПРРО** та **ФН чека.**

public string GetReceiptFiscalNumberByLocalNumber(string rroFiscalNumber,string checkLocalNumber);

Приклад:

```
Object[] parameters = {
                "4000000000",    /* фіскальний номер ПРРО*/
                "1234567"        /* фіскальний номер чеку*/
               };
var ret = CallCashaLotMethod( "GetReceiptLocalNumberByFiscalNumber", parameters);
```

---

9. Отримання XML-структури чека

Для отримання XML-структури фіскалізованого чека використовується метод **GetReceiptXML** з параметрами, де заповнені **Фіскальний номер ПРРО** та **ФН чека.**

public string GetReceiptXML(string rroFiscalNumber, string checkFiscalNumber);

Приклад:

```
Object[] parameters = {
                "4000000000",    /* фіскальний номер ПРРО*/
                "1234567"        /* фіскальний номер чеку*/
               };
var ret = CallCashaLotMethod( "GetReceiptXML", parameters );
```

Після виконання методу Cashälot повертається **CashalotApiRsp**, в якому заповнений масив параметрів **Values**, що містить заповнений параметр Base64Str1251ReceiptXML. Дані в ньому закодовані в Base64.

---

10. Відображення фіскалізованого чека

У випадку коли після фіскалізації чека з'являється необхідність відобразити на екрані його повторно, наприклад для друку, використовується метод **ShowReceipt** з заповненими параметрами **Фіскальний номер ПРРО** та **ФН чека.**

public string ShowReceipt(string rroFiscalNumber, string checkFiscalNumber);

Приклад виконання:

```
Object[] parameters = {
                "4000000000",    /* фіскальний номер ПРРО*/
                "1234567"        /* фіскальний номер чеку*/
               };
var ret = CallCashaLotMethod( "ShowReceipt", parameters );
```

---

11. Створення X-звіту

Для створення X-звіту необхідно викликати метод **ShowXReport** з параметром, в якому вказано **Фіскальний номер ПРРО**.

public string ShowXReport(string rroFiscalNumber);

Приклад:

```
Object[] parameters = {"4000000000",    /* фіскальний номер ПРРО*/};
var ret = CallCashaLotMethod( "ShowXReport", parameters );
```

Після виконання методу Cashälot повертається **CashalotApiRsp**, в якому заповнений масив параметрів **Values.** В результаті виконання операції створюється X-звіт та відображається на екрані, після чого його можна роздрукувати.

---

12. Закриття зміни та створення Z-звіту

Для закриття поточної зміни використовується метод **CloseShift**. В параметрах **CloseShift** має бути заповнено тільки **ФН ПРРО**.

Увага! Якщо встановлений параметр NOINTERFACEMODE = True, то службову видачу потрібно передбачити перед виконанням методу **CloseShift.**

public string CloseShift(string rroFiscalNumber);

Приклад виконання:

```
Object[] parameters = {
                "4000000000",    /* фіскальний номер ПРРО*/
               };
var ret = CallCashaLotMethod( "CloseShift", parameters );
```

Перед створенням Z-звіту та закриттям зміни, описаним в цьому розділі, автоматично виконується метод службового вилучення наявних грошових коштів з каси. Також виконується графічне відображення Z-звіту для можливості подальшого друку.

---

13. Ручний перевод ПРРО в режим офлайн

Для ручного ручного переводу ПРРО в режим офлайн використовується метод SetOfflineMode. В якості параметрів використовується **Фіскальний номер ПРРО** та **Ознака автоматичного виходу в онлайн**(true - автоматично виходити з офлайну при наявності інтернету. false - виходити з офлайну лише при виклику методу TryGoToOnlineMode або після закриття зміни).

public string SetOfflineMode(string rroFiscalNumber, bool autoExitToOnline);

Приклад виконання:

```
Object[] parameters = {
                "4000000000",    /* фіскальний номер ПРРО*/
                false,    /* true - автоматично виходити з офлайну при наявності інтернету. false - виходити з офлайну лише при виклику методу TryGoToOnlineMode або після закриття зміни*/
                };
var ret = CallCashaLotMethod( "SetOfflineMode", parameters );
```

Виклик SetOfflineMode не означає автоматичний перехід в офлайн одразу в момент визову, перехід до офлайну буде здійснено при будь-якій фіскальній операції(відкриття/закриття зміни, службовій видачі/внесенню або реєстрації чека) після визову SetOfflineMode.

В якості відповіді повертається стандартна структура параметрів повернення CashalotApiRsp з заповненим параметром Return(Ознака успішності виконання).

---

14. Ручний перевод ПРРО в режим онлайн

Для ручного ручного переводу ПРРО в режим онлайн використовується метод TryGoToOnlineMode.

public string TryGoToOnlineMode(string rroFiscalNumber);

Приклад виконання:

```
Object[] parameters = {
                "4000000000",    /* фіскальний номер ПРРО*/
                };
var ret = CallCashaLotMethod( "TryGoToOnlineMode", parameters );
```

В якості відповіді повертається стандартна структура параметрів повернення CashalotApiRsp з заповненим параметром Return(Ознака успішності виконання).

---

15. Синхронізація залишків по товарам

Метод призначений для ручної синхронізації залишків по товарам з кабінету користувача в локальну базу даних Cashalot. Для можливості виконання методу необхідно встановити параметр ведення обліку товарів в API в режимі 1 або 2 та ознаку ведення обліку товарів в кабінеті користувача в розділі «Склад - Налаштування - Облік товарів у ПРРО».

public string SyncBalanceOnGoods(string rroFiscalNumber);

Приклад виконання:

```
Object[] parameters = {
                "4000000000",    /* фіскальний номер ПРРО*/
                };
var ret = CallCashaLotMethod( "SyncBalanceOnGoods", parameters );
```

В якості відповіді повертається стандартна структура параметрів повернення CashalotApiRsp з заповненим параметром Return(Ознака успішності виконання).

---

16. Синхронізація залишків та товарів

Метод призначений для ручної синхронізації товарів та залишків по ним з кабінету користувача в локальну базу даних Cashalot. Для можливості виконання методу необхідно встановити параметр ведення обліку товарів в API в режимі 1 або 2 та ознаку ведення обліку товарів в кабінеті користувача в розділі «Склад - Налаштування - Облік товарів у ПРРО».

public string SyncGoodsWithBalance(string rroFiscalNumber);

Приклад виконання:

```
Object[] parameters = {
                "4000000000",    /* фіскальний номер ПРРО*/
               };
var ret = CallCashaLotMethod( "SyncGoodsWithBalance", parameters );
```

В якості відповіді повертається стандартна структура параметрів повернення CashalotApiRsp з заповненим параметром Return(Ознака успішності виконання).

ЛОМБАРДНІ ОПЕРАЦІЇ

Для фіскалізації ломбардних послуг введено системні номенклатури, які необхідно використовувати при виклику методу фіскалізації чека **FiskalizeCheck**. Повернення товару при ломбардних операціях не підтримується.

Також необхідно вказати тип документа DocType = 4 в JSON-структурі параметрів чека. У разі потреби можна вказати номер договору DocNumber, за яким надається ломбардна послуга.

Напрям руху необхідно враховувати при обчисленні кінцевого підсумку в чеку: від позицій типу 2 віднімаються позиції типу 3.

Системні номенклатури для ломбардних операцій:

Артикул	Назва	Тип послуги	Напрям руху*
"ЛО-00001"	"Повернення страхування"	"3"	-
"ЛО-00002"	"Надання кредиту"	"3"	-
"ЛО-00003"	"Добір кредиту"	"3"	-
"ЛО-00004"	"Надання додаткового кредиту"	"3"	-
"ЛО-00005"	"Зберігання"	"2"	+
"ЛО-00006"	"Страховий платіж"	"2"	+
"ЛО-00007"	"Часткове повернення кредиту (Частковий викуп)"	"2"	+
"ЛО-00008"	"Повне повернення кредиту (Повний викуп)"	"2"	+
"ЛО-00009"	"Повернення кредиту"	"2"	+
"ЛО-00010"	"Користування кредитом"	"2"	+
"ЛО-00011"	"Сплата відсотків по кредиту"	"2"	+
"ЛО-00012"	"Сплата пені"	"2"	+

* Напрям руху "-" видача грошей з каси; "+" повернення грошей в касу.

Приклад реалізації функції для виклику API методів ПРРО Cashälot

```
//Структура інформації рядка чека (містить всі дані для створення товару в CashaLot)
 public class CheckInfo
 {
     public CheckInfo()
     { }
     public CheckInfo( CheckInfo ckInfo )

     {
         this.VendorCode = ckInfo.VendorCode;
         this.Name = ckInfo.Name;
         this.CdUKDZED = ckInfo.CdUKDZED;
         this.CdDKPP = ckInfo.CdDKPP;
         this.GoodsType = ckInfo.GoodsType;
         this.Barcode = ckInfo.Barcode;
         this.UnitType = ckInfo.UnitType;
         this.Quantity = ckInfo.Quantity;
         this.Price = ckInfo.Price;
         this.Amount = ckInfo.Amount;
         this.DiscountPrc = ckInfo.DiscountPrc;
         this.DiscountSum = ckInfo.DiscountSum;
         this.VATRate = ckInfo.VATRate;
         this.IsPriceIncludeVAT = ckInfo.IsPriceIncludeVAT;
         this.SumVAT = ckInfo.SumVAT;
         this.IsExcise = ckInfo.IsExcise;
         this.OtherParametrs = ckInfo.OtherParametrs;
         this.VATLetter = ckInfo.VATLetter;
```

```
            this.ExciseLetter = ckInfo.ExciseLetter;
            this.ExciseStampBarcode = ckInfo.ExciseStampBarcode;
        }
        //Артикул товару
        public String VendorCode { get; set; } = String.Empty;
        public String Article { get => VendorCode; set { VendorCode = value; } }

        //Назва товару
        public String Name { get; set; } = String.Empty;
        //Код УКДЗЕД
        public String CdUKDZED { get; set; } = String.Empty;
        //Код ДКПП(для послуг)
        public String CdDKPP { get; set; } = String.Empty;
        //Тип (1-товар/2-послуга(+)|3-послуга ломбардна(-)). За замовчуваннял товар, якщо нічого не задано.
        public String GoodsType { get; set; } = String.Empty;
        //Штрихкод
        public String Barcode { get; set; } = String.Empty;
        //Код одиниці виміру
        public String UnitType { get; set; } = String.Empty;
        //Кількість товару
        public String Quantity { get; set; } = String.Empty;
        //Ціна товару
        public String Price { get; set; } = String.Empty;
        //Сума
        public String Amount { get; set; } = String.Empty;
        //Процент знижки
        public String DiscountPrc { get; set; } = String.Empty;
        //Сума знижки
        public String DiscountSum { get; set; } = String.Empty;
        //Процент податку ПДВ
        public String VATRate { get; set; } = String.Empty;
        //Ознака - чи включає сума ПДВ
        public bool? IsPriceIncludeVAT { get; set; } = true;
        //Сума ПДВ
        public String SumVAT { get; set; } = String.Empty;
        //Ознака - підакцизного товару
        public bool? IsExcise { get; set; } = false;
        //Інші параметри у вигляді строки XML
        //Приклад: <?xml version=\"1.0\" encoding=\"UTF-8\"?><Table><Record VAT =\"20\" IsExcisable=\"False\" Nomenclature=\"STEP Пазл 104 Винни Пух\" UKTZED=\"\" VendorCode=\"82106\" Barcode=\"\" Dimension=\"шт\" Discount=\"1\" DiscountAmount=\"0.56\" PriceIncludeVAT=\"1\" AmountVAT=\"9.24\"/></Table>
        public String OtherParametrs { get; set; }
        //Літера ставки податку ПДВ
        public String VATLetter { get; set; } = String.Empty;
        //Літера ставки податку Акцизу
        public String ExciseLetter { get; set; } = String.Empty;
        //Штрихкод акцизної марки
        public String ExciseStampBarcode { get; set; } = String.Empty;
    };

 //Структура з масивом рядків чека
    public class LstCheckInfoRsp
    {
        public List<CheckInfo> ReceiptLst { get; set; }
        //Коментар
        public String Comment { get; set; }
        //Тип чека(0 - продаж, 4 - ломбард)
        public long DocType { get; set; } = 0;
```

```
  };



//шлях до каталогу ПРРО Cashalot
public string edCashaLotPath = "C:\\ProgramData\\Cashalot\\Cashalot";

//Фіскальний номер РРО
public string edFiscalNumberRRO = "4000000000";

//завантажена бібліотека
private Assembly asmCashaLotDll;

//тип класу Апі
private Type asmCashaLotClass;

//екземпляр класу Апі
private object instCashaLotApi;

//клас відповіді апі для десеріалізації
public class CashaLotApiRsp
{
     //Ознака успішності виконання функції (true - успішно/false - не успішно)
     public bool Ret { get; set; } = false;
     //Рядок з текстом помилки
     public string ErrorString { get; set; } = String.Empty;
     //Масив параметрів, які можуть повернути функції АПІ
     public Dictionary<string, string> Values { get; set; }
}

/// <summary>
/// Функція для виклику методу з АПІ Cashalot
/// </summary>
/// <param name="methodName"> Назва методу </param>
/// <param name="parameters"> Список параметрів методу</param>
/// <returns></returns>
public CashaLotApiRsp CallCashaLotMethod(string methodName, Object[] parameters )
{
     CashaLotApiRsp ret = new CashaLotApiRsp();

     if ( edCashaLotPath.Trim().Length == 0 )
         MessageBox.Show( "Не вказано шлях до каталогу Cashalot" );

     if ( edFiscalNumberRRO.Trim().Length == 0 )
         MessageBox.Show( "Не вказано фіскальний номер РРО" );

     string strPath = $"{edCashaLotPath}\\CashaLotApiBridge.dll";
     if ( !File.Exists( strPath ) )
     {
         MessageBox.Show( "За вказаним каталогом не знайдено CashaLotApiBridge.dll" );
         return ret;
     }

     if ( asmCashaLotDll == null )
          asmCashaLotDll = System.Reflection.Assembly.LoadFrom( strPath
);//Завантаження бібліотеки

     if ( asmCashaLotDll != null )
     {
         if( asmCashaLotClass == null)
```

```
            asmCashaLotClass = asmCashaLotDll.GetType( "CashaLotApiBridge.BridgeApi"
);

          if ( asmCashaLotClass != null )
          {
              if( instCashaLotApi == null)
              instCashaLotApi = Activator.CreateInstance( asmCashaLotClass );//Створення
об'єкту класу

              System.Reflection.MethodInfo mi = asmCashaLotClass.GetMethod(methodName
);//отримання інформації по вказанному методу
              if ( mi != null )
              {
                  var retVal = mi.Invoke( instCashaLotApi, parameters ); // виклик методу
з передачею параметрів
                  if( retVal !=null )
                  {
                      try
                      {
                          //десеріалізація отриманої відповіді
                          ret = JsonConvert.DeserializeObject<CashaLotApiRsp>( retVal as
String );
                      }
                      catch
                      {
                          MessageBox.Show( "Не вдалося десеріалізувати відповідь від
Cashalot" );
                      }
                  }
              }
          }
      }
      return ret;
  }
```