Navision Sync

# Призначення

Програма призначена для синхронізації даних між CRM і базою даних іншої системи

# Компоненти

Програмний код розміщено в каталозі Navision\_sync. Для розповсюдження програми достатньо файлової копії зазначеного каталогу.

Основні компоненти (файли):

* Node.exe – програмне середовище для запуску сервісу і виконання програмного коду.
* run.cmd – командний файл, призначений для запуску програми у вигляді консольного вікна під управлінням операційної системи MS Windows
* dist/config.json – файл конфігурації програми.
  + особливості – після внесення змін в конфігураційний файл програма потребує перезапуску.

Управління системним сервісом **navision\_sync\_service**:

* serviceStart.cmd – сомандний файл, призначений для запуску програми у вигляді системного сервісу. Призначений для запуску на операційній системі MS Windows.
  + особливості – файл призначиний як для запуску так і для встановлення сервісу. Якщо при першому запуску в системі буде виявлено що сервіс не встановлено то відбудеться його інсталяція.
* serviceStop.cmd – командний файл, призначений для зупмнки Winddows сервісу.
* serviceRestart.cmd – командний файл, призначений для перезапуску Windows сервісу.
* serviceRemove.cmd - командний файл, призначений для видалення Windows сервісу.

# Налаштування

Для налаштування програми використовується файл dist/config.json. Після внесення змін в конфігураційний файл програма потребує перезапуску.

Основні параметри файлу dist/config.json:

* Параметр **port** –визначає системний порт на якому буде запущено сервіс. По замовчуванню - 8080
* Параметр **windowsSerwice** – визначає основну конфігурацію сервісу Windows

"windowsService":{  
 "name":"navision\_sync\_service",  
 "description": "The navision\_sync web server.",  
 "script": "./dist/index.js",  
 "user":{  
 "domain": **null**,  
 "account": "администратор",  
 "password":"\*\*\*\*\*\*\*"  
 }  
}

* name – назва сервісу. За назвою відбувається вся робота по управлінню сервісом.
* description – опис сервісу
* script – командний файл для запуску сервісу **!!! НЕ ЗМІНЮВАТИ !!!.**
* user – параметри користувача від котрого буде запускатись сервіс
* user.domain – домен (якщо використовується доменний користувач)
* user.account – ім’я користувача
* ucer.password – пароль користувача
* Параметр **logger** - визначає основну конфігурацію системи логування

"logger":{  
 "logFolder": "./logs/",  
 "logLevel": "error",  
 "logType": "file"  
}

* logType – тип логування (в поточній версії доступне лише одне значення - file)
* logFolder – каталог для збереження файлових логів.
* logLevel – рівень логування. Допустимі значення erro, warn, info, debug. Кожен вищий рівень включає попередні. Помилки логуються завжди.
* Параметр **orders** – визначає конфігурацію для роботи з CRM та параметри підключення до бази MS SQL Server

"orders":{  
 "crmApiKey": "HvJ2KOiy4UGccwj3bAgo6V8GQQQQQQQ",  
 "crmServerUrl": "https://mirs.retailcrm.ru/",  
 "sqlServerConnectionConfig": {  
 "user": "WEBGate",  
 "password": "\*\*\*\*\*\*\*",  
 "server": "192.168.50.251",  
 "database": "Views"  
 }  
}

* crmApiKey - ключ авторизації для підключення сервера CRM
* crmServerUrl – URL адреса за котрою розміщено сервер CRM
* sqlServerConnectionConfig – параметри підключення до MS SQL Server
  + user – ім’я користувача для підключення
  + password – пароль користувача для підключення
  + server – ім’я сервера
  + database – назва бази даних

# API

Після запуску програми буде доступний HTTP сервіс за адресою

GET <http://server:port/api/orders/:externdlId>

Де замість :externalId необхідно підставити параметр externalId реального замовлення в CRM.

Приклад:

GET <http://127.0.01:8080/api/orders/PKB-к0000335>

Після виклику сервісу за переданим externalId буде виконано запит до CRM та збережено замовлення в базу MS SQL Server, що вказана в конфігурації.

Результатом роботи сервісу буде повернення результату обробки та збереження замовлення у вигляді JSON із зазначенням статусу по кожному пункту замовлення

{  
 "status":"OK",  
 "itemsInfo":[  
 {  
 "status":"OK",  
 "type":"item",  
 "code":0  
 },{  
 "status":"OK",  
 "type":"item",  
 "code":0  
 },{  
 "status":"OK",  
 "type":"item",  
 "code":0  
 },{  
 "status":"OK",  
 "type":"delivery",  
 "code":0  
 }  
 ]  
}

* status – загальний статус обробки запиту. Може мати значення OK або ERROR. Якщо хоч один з пунктів замовлення буде оброблено з помилкою значення міститиме ERROR.
* message – (опціонально) текст помилки котра виникла в процесі роботи. Якщо помилок не було поле не повертається.
* itemsInfo – масив з результатами обробки кожного пункту замовлення
  + status – статус обробки пункту замовлення. Може мати значення OK або ERROR.
  + type – признак замовлення чи доставки. Може мати значення item (замовлення) або delivery (доставка).
  + code – код обробки що повернула SQL процедура.

# Тестування

Для тестування збереження замовлення в базу даних розроблено сервіс до котрого можна звернутись за адресою:

GET <http://server:port/api/orders/get?test=true>

Приклад:

GET http://127.0.01:8080/api/orders/get?test=true

При обробці такого запиту не буде виконуватись запит до CRM, дані будуть братись з файлу **test/api/orders/order.json**