г. Москва от «01» ноября 2024 г.

Программа для ЭВМ MPsklad разработана собственными силами компании ООО «Солвинтек». Свидетельство о государственной регистрации программы для ЭВМ № **2024688942** от «02» декабря 2024 года.

Программа для ЭВМ MPsklad является прикладным программным обеспечением, состоящим из компонентов:

- AuthService // сервис аутентификации и управления пользователями
- HttpUserService // сервис получения информации о текущем пользователе в контексте HTTP-запроса.
- ImpersonationCookieService // сервис по работе с куками
- MandrillEmailSenderService // сервиса для отправки электронных писем с использованием API Mandrill
- OrderService // сервиса, который предоставляет методы получения страниц заказов из различных источников (например, Yandex Market, MoySklad и Wb)
- AdminService // сервис представляет собой асинхронный компонент, который извлекает данные о пользователях и их учетных записях из базы данных, включая информацию о последних заказах в MoySklad
- ApiAuthLimitingService // обеспечивает управление ограничениями на количество одновременных запросов к API для различных пользователей, применяя семафоры для контроля параллелизма
- AppTariffService // сервис, который управляет тарифами приложения, обеспечивая доступ к информации о тарифах пользователей и проверяя соответствие их действий лимитам и возможностям, установленным для каждого тарифа
- AutoStartService // Основная задача этого сервиса — автоматическая настройка и запуск фоновых задач при старте приложения
- FeedService // реализация функционала для генерации и обработки "фидов" (feeds) для платформы Ozon.
- LabelBarcodeService // Сервис отвечает за получение штрих-кодов для заказов из различных торговых платформ (Ozon, Wildberries, Яндекс.Маркет)
- LabelFileService // Сервис отвечает за управление файлами печати для различных торговых платформ (Ozon, Wildberries, Яндекс.Маркет) в контексте системы управления заказами
- LabelService //Сервис отвечает за управление процессом печати этикеток для заказов
- MaintenanceScopedService // Сервис MaintenanceScopedService предназначен для выполнения операций по обслуживанию базы данных, в частности, для "вакуумирования" таблиц. Вакуумирование — это процесс, который помогает оптимизировать производительность базы данных, освобождая место и улучшая скорость выполнения запросов
- MaintenanceSingletonService // Сервис предназначен для выполнения операций по обновлению учетных записей (в системе "MoySklad")
- MoySkladAppService // Сервис для интеграции с API "MoySklad" – предназначен для управления доступом и созданием приложений. Основные функции – Валидация доступа, Создание пользователей приложений, Проверка совместимости типов приложений, Управление конфликтами типов приложений). Сервис обеспечивает надежную и безопасную интеграцию с "MoySklad", минимизируя риски ошибок и конфликтов
- RazorService // предназначен для работы с шаблонами Razor. Он позволяет загружать, компилировать и рендерить шаблоны
- StorageUserSyncService // представляет собой абстрактный сервис для синхронизации данных пользователей с хранилищем

- UrlService // предоставляет методы для создания URL на основе заданных параметров, таких как имя действия и контроллера
- UserLockService // сервис предоставляющий набор методов, предназначенных для блокировки ресурсов (балансов и продуктов) в различных системах (Ozon, Wb, Yandex Market) с использованием подхода, основанного на семафорах и блокировка.
- UserService // Сервис управляет удалением учетных записей пользователей для различных платформ (Ozon, Wildberries, Яндекс.Маркет) в контексте базы данных

Import:

- OzonImportService // часть системы, которая взаимодействует с API Ozon и MoySklad для синхронизации продуктов
- WbImportService часть системы, которая взаимодействует с API Wb и MoySklad для синхронизации продуктов
- YandexMarketImportService // сервис, отвечающий за импорт данных из Yandex Market в систему MoySklad
- ManualUserSyncService // сервис, предназначенный для синхронизации данных пользователя с различными хранилищами.
- MoySkladBackgroundSyncService // это сервис, реализующий фоновую синхронизацию данных с системой MoySklad. Для управления синхронизацией пользователей и их продуктов с MoySklad в фоновом режиме.
- MoySkladBackgroundSyncUserService // отвечает за синхронизацию данных пользователей с системой MoySklad
- PriceSyncService // сервис, который отвечает за синхронизацию цен на товары между системой MoySklad и платформой Wildberries. Он включает в себя логику для отправки цен, проверки успешности отправки и обработки ошибок
- RealizationReportSyncService // отвечает за синхронизацию отчетов о продажах для аккаунтов Ozon и Wildberries. Он использует контекст базы данных и клиент фоновых задач Hangfire для обработки асинхронных операций.
- StoreRemovalService // отвечает за удаление магазинов из системы, поддерживая операции для Ozon, Wildberries и Яндекс.Маркета. Он использует контекст базы данных, клиент фоновых задач и логирование.
- StoreStocksService // отвечает за синхронизацию остатков товаров в интернет-магазинах Ozon, Wildberries и Яндекс.Маркет. Он использует асинхронные задачи для обработки синхронизации и управляет очередями фоновых задач с помощью библиотеки Hangfire
- SyncAllService // предназначен для синхронизации продуктов и остатков товаров между различными системами
- SyncJobLimiterService // предназначен для управления синхронизацией задач, обеспечивая эксклюзивный доступ к выполнению определённых задач, чтобы избежать одновременного выполнения дублирующихся операций
- UserSyncService // сервиса, который управляет синхронизацией данных между различными учетными записями пользователя и внешними системами, такими как Ozon, Wildberries (WB) и Яндекс.Маркет
- WbCommissionsSyncService // обеспечивает синхронизацию заказов и платежей между системой MoySklad и другой системой WB

Product:

- BalanceRecordService // служит для работы с записями баланса продуктов в системе MoySklad. Основная функциональность сервиса включает загрузку, вычисление, добавление и обновление записей баланса для продуктов.
- ProductService // Этот сервис отвечает за синхронизацию товаров между системой управления складом (MoySklad) и внешними торговыми платформами, такими как Ozon, Wildberries и Yandex Market. Он включает несколько ключевых функций – Синхронизация

с платформами, учет остатков на складах, экспорт данных, определение состояния товара, обработка ошибок и логирование.

MoySklad:

- MoySkladApiService // Этот сервис взаимодействует с API MoySklad для выполнения различных операций с сущностями, такими как заказы, вебхуки, контрагенты, организации, контракты, товары, и так далее. Он предоставляет методы для получения метаданных, создания, удаления, обновления и управления сущностями, такими как заказы, платежи, товары и атрибуты сущностей.
- MoySkladSyncService // Этот сервис реализует обработку статусов заказов для нескольких торговых платформ (Ozon, Wildberries, Yandex Market), взаимодействуя с API MoySklad для синхронизации и обработки данных. Функции – Обработка статусов заказов, обработка возвратов и отмен, работа с фоновыми задачами, синхронизация данных с API платформ, логирование и обработка ошибок
- MoySkladUserSyncService // Сервис служит для синхронизации данных с платформой MoySklad для конкретного пользователя
- MoySkladVendorApiService // Выполняет работу с API сервиса MoySklad, ограничивает количество запросов, управляет авторизацией через JWT, а также предоставляет методы для получения и установки контекста, статуса приложения. Использует настраиваемые параметры авторизации, такие как секретные ключи и название приложения, что позволяет безопасно взаимодействовать с API
- MoySkladWebhookService // предназначен для управления вебхуками, связанными с заказами пользователей в системе "MoySklad" (Проверка вебхуков для всех пользователей, Проверка вебхуков для текущего пользователя, Проверка и валидация вебхуков для конкретного пользователя, Синхронизация заказов)

Ozon:

- OzonApiCommonService // сервис используется для базовой настройки и аутентификации при работе с API Ozon
- OzonApiService // сервис предназначен для интеграции с Ozon и автоматизации операций с товарами, заказами, складами и отчетами на платформе Ozon
- OzonApiStocksService // отвечает за управление остатками товаров на складах Ozon
- OzonBackgroundSyncService // отвечает за фоновую синхронизацию данных пользователей с Ozon.
- OzonBackgroundSyncUserService // реализует логику фоновой синхронизации для конкретного пользователя с Ozon
- OzonStocksSenderService // предназначен для управления задачами отправки остатков на Ozon
- OzonSyncService // Этот код представляет реализацию службы синхронизации с Ozon, включающую логику синхронизации товаров, остатков и заказов между базой данных и API Ozon
- OzonUserSyncService // отвечает за синхронизацию данных пользователей между системой хранения и платформой Ozon

WildBerries:

- WbApiService // Сервис взаимодействует с API Wildberries для выполнения различных операций, связанных с товарами, заказами и складами
- WbBackgroundSyncService // выполняет синхронизацию данных между системой и аккаунтами Wildberries
- WbBackgroundSyncUserService // реализует логику фоновой синхронизации для конкретного пользователя с Wildberries
- WbDiscountsPricesApiService // взаимодействует с API Wildberries для управления ценами и скидками товаров. Он выполняет операции, такие как загрузка цен, проверка состояния

задач и получение информации о товарах, для синхронизации данных между системой и Wildberries

- WbOrderService // использует API Wildberries для изменения статусов заказов и ведет логирование ошибок и событий
- WbStatNewApiService // сервис реализует работу с API статистики Wildberries, управляя лимитами запросов и авторизацией

WbSyncService // Это сервис синхронизации данных для работы с Wildberries (WB). Он взаимодействует с API WB и базой данных, чтобы синхронизировать товары, заказы и остатки товаров между системой MS (MoySklad) и WB

WbUserSyncService // реализует синхронизацию данных для пользователей Wildberries

YandexMarket

- YandexMarketApiService // Этот сервис предназначен для взаимодействия с API Яндекс.Маркет для получения информации о товарах, заказах, складах и ценах на Яндекс.Маркет, а также для обновления данных
- YandexMarketBackgroundSyncService // предназначен для обработки фоновых задач синхронизации данных с Yandex Market
- YandexMarketBackgroundSyncUserService // выполняет синхронизацию данных пользователей с Yandex Market
- YandexMarketOAuthApiService // предназначен для работы с OAuth-авторизацией через API Yandex Market. Он предоставляет функционал для обмена кода на OAuth-токен и для обновления токена
- YandexMarketOrderService // предназначен для управления заказами на платформе Yandex Market. Он реализует два основных метода — подтверждение заказа и его отмену
- YandexMarketSyncService // Сервис, описанный в коде, отвечает за синхронизацию товаров и их остатков между системой учета (возможно, MS Sklad) и YandexMarket.
- YandexMarketTokenRefreshService // сервис отвечает за обновление токенов авторизации для аккаунтов YandexMarket, использующих старый формат API-ключа
- YandexMarketUserSyncService // сервис отвечает за синхронизацию пользователей с системой YandexMarket, а также за управление товарами (акциями) в процессе синхронизации

Frontend(React + Mobx)
Backend (ASP.NET)
Database (PostgreSQL и MongoDB)
Logging(NLog)
MoySkladSyncLogger
OzonSyncLogger
SyncLogger
WbSyncLogger
YandexMarketSyncLogger

Разработка Программа для ЭВМ MPsklad завершена в 2024 г.

Правообладатель
Генеральный директор
ООО «Солвинтек» _____ А.А. Кожевников /

ОБЩЕСТВО С ОГРАНИЧЕННОЙ ОТВЕТСВЕННОСТЬЮ «СОЛВИНТЕК»

ПРОГРАММА ДЛЯ ЭВМ

«MPsklad»

Фрагменты исходного текста программы

2024 г. - *год подготовки документа к регистрации*

```
using System;
using System.Threading;
using System.Threading.Tasks;
using Hangfire;
using Microsoft.Extensions.DependencyInjection;
using Microsoft.Extensions.Hosting;
using mpsklad_common;
using mpsklad_core.Entity.Context;
using mpsklad_core.Services.MoySklad;
using mpsklad_core.Services.Ozon;
using mpsklad_core.Services.Synchronization;
using mpsklad_core.Services.WildBerries;
using mpsklad_core.Services.YandexMarket;

namespace mpsklad_core.Services
{
  public class AutoStartService : IHostedService
  {
    private readonly IServiceProvider _serviceProvider;

    public AutoStartService(IServiceProvider serviceProvider)
    {
      _serviceProvider = serviceProvider;
    }

    public async Task StartAsync(CancellationToken cancellationToken)
    {
      // Create a new scope to retrieve scoped services
      using var scope = _serviceProvider.CreateScope();

      var serviceProvider = scope.ServiceProvider;

      // The DB must be migrated before other jobs
      var db = serviceProvider.GetRequiredService<IMpSkladContext>();

      var initialTimeout = db.GetCommandTimeout();
      db.SetCommandTimeout(TimeSpan.FromMinutes(15));

      await db.MigrateAsync(cancellationToken);

      db.SetCommandTimeout(initialTimeout);

      // TODO: Add a maintenance method to recreate the jobs
      // TODO: Also add a method to restart the hangfire server with new workercount, see docs
whether it is possible
      var env = serviceProvider.GetRequiredService<IHostEnvironment>();
      var appSettings = serviceProvider.GetRequiredService<IAppSettings>();
      var jobManager = serviceProvider.GetRequiredService<IRecurringJobManager>();
```

```
    string stocksCron;
    string productsCron;
    string pricesCron;
    string realizationsCron;
    string wbCommissionsCron;
    string feedsCron;
    string storeStocksCron;
    string verifyMsWebhooksCron;
    string trimSyncLogsCron;
    string vacuumTablesCron;
    string yandexMarketTokenRefreshCron;

    if (!env.IsProduction())
    {
      // Manual launch only
      stocksCron = Cron.Never();
      productsCron = Cron.Never();
      pricesCron = Cron.Never();
      realizationsCron = Cron.Never();
      wbCommissionsCron = Cron.Never();
      feedsCron = Cron.Never();
      storeStocksCron = Cron.Never();
      verifyMsWebhooksCron = Cron.Never();
      trimSyncLogsCron = Cron.Never();
      vacuumTablesCron = Cron.Never();
      yandexMarketTokenRefreshCron = Cron.Never();
    }
    else
    {
#pragma warning disable 618 // Why is it obsolete though
      stocksCron = Cron.MinuteInterval(appSettings.PullStocksIntervalMinutes);
      storeStocksCron = Cron.HourInterval(appSettings.PushStoreStocksIntervalHours);
      feedsCron = Cron.Daily(MoscowHoursToUtc(0));
      productsCron = Cron.Daily(MoscowHoursToUtc(9), 15);
      pricesCron = Cron.Daily(MoscowHoursToUtc(10));
      verifyMsWebhooksCron = Cron.Daily(MoscowHoursToUtc(10), 15);
      // Trim logs often to prevent accumulation of logs in case of frequent failures
      trimSyncLogsCron = Cron.Hourly(47);
      vacuumTablesCron = Cron.Weekly(DayOfWeek.Monday, MoscowHoursToUtc(4), 33);
      // Something breaks in WB on Mondays
      wbCommissionsCron = Cron.Weekly(DayOfWeek.Tuesday, MoscowHoursToUtc(11), 5);
      // Ozon guarantees the reports to be fetchable no later than 5th of each month,
      // settings to 6th just to make sure it's alright.
      // WB also makes it inaccessable on mondays from 3:00 to 16:00 - this is checked inside
      realizationsCron = Cron.Monthly(6 /* 03:00 Moscow time */);
      yandexMarketTokenRefreshCron = Cron.Weekly(DayOfWeek.Monday,
MoscowHoursToUtc(1));
#pragma warning restore 618
```

```
    int MoscowHoursToUtc(int moscowHours)
    {
      if (moscowHours < 0 || moscowHours >= 24)
      {
        throw new ArgumentOutOfRangeException(nameof(moscowHours));
      }

      var utcHours = moscowHours - 3;
      return utcHours >= 0 ? utcHours : utcHours + 24;
    }
  }

  jobManager.AddOrUpdate<ISyncAllService>(
    "SyncAllStocksJob", _ => _.SyncAllStocks(CancellationToken.None), stocksCron);

  jobManager.AddOrUpdate<IPriceSyncService>("SyncAllPricesJob", _ => _.SyncAllPrices(),
pricesCron);

  jobManager.AddOrUpdate<ISyncAllService>(
    "SyncAllProductsJob", _ => _.SyncAllProducts(CancellationToken.None), productsCron);

  jobManager.AddOrUpdate<IRealizationReportSyncService>(
    "SyncAllRealizationReportsJob", s => s.SyncAllRealizationReports(), realizationsCron);

  jobManager.AddOrUpdate<IMoySkladBackgroundSyncService>(
    "MoySklad_PullAllUserStocksJob", _ => _.PullAllUserStocks(CancellationToken.None),
    Cron.Never /* Manual launch only */);

  jobManager.AddOrUpdate<IOzonBackgroundSyncService>(
    "Ozon_PullAllUserProductsJob", _ => _.PullAllUserProducts(CancellationToken.None),
    Cron.Never /* Manual launch only */);

  jobManager.AddOrUpdate<IWbBackgroundSyncService>(
    "Wb_PullAllUserProductsJob", _ => _.PullAllUserProducts(CancellationToken.None),
    Cron.Never /* Manual launch only */);

  jobManager.AddOrUpdate<IYandexMarketBackgroundSyncService>(
    "YandexMarket_PullAllUserProductsJob", _ =>
_.PullAllUserProducts(CancellationToken.None),
    Cron.Never /* Manual launch only */);

  jobManager.AddOrUpdate<IWbCommissionsSyncService>(
    "WbCommissionsJob", _ => _.SyncAllCommissions(), wbCommissionsCron);

  jobManager.AddOrUpdate<IFeedService>("SyncAllFeedsJob", _ => _.PrintAllOzonFeeds(),
feedsCron);

  jobManager.AddOrUpdate<IStoreStocksService>(
    "PushAllStoreStocksJob", _ => _.PushAllStoreStocks(), storeStocksCron);
```

```
    jobManager.AddOrUpdate<IMoySkladWebhookService>(
      "VerifyAllUserOrderWebhooksJob", _ => _.VerifyAllUserOrderWebhooks(),
verifyMsWebhooksCron);

    jobManager.AddOrUpdate<MaintenanceScopedService>(
      "Maintenance_TrimSyncLogs", _ => _.TrimSyncLogs(CancellationToken.None),
trimSyncLogsCron);

    jobManager.AddOrUpdate<MaintenanceScopedService>(
      "Maintenance_VacuumTables", _ => _.VacuumTables(CancellationToken.None),
vacuumTablesCron);

    jobManager.AddOrUpdate<IYandexMarketTokenRefreshService>(
      "YandexMarketTokenRefreshJob", _ =>
_.RefreshAllYandexMarketTokens(CancellationToken.None),
      yandexMarketTokenRefreshCron);
  }

  public Task StopAsync(CancellationToken cancellationToken) => Task.CompletedTask;
 }
}
```