

РАЗРАБОТКА ПРОГРАММНОГО КОМПЛЕКСА АВТОМАТИЗАЦИИ СКЛАДСКИХ ЗАПАСОВ

DEVELOPMENT OF A SOFTWARE PACKAGE FOR AUTOMATION OF WAREHOUSE STOCKS

**B. Pruss
A. Bashmakov
O. Pleskacheva**

Summary. The article describes the developed software package for optimization and planning of purchases. The developed package has a user-friendly interface and a sufficient number of functions allowing the user to automate and optimize the process of replenishment of warehouse stocks taking into account the dynamics of sales. The developed software package solves the problem of optimization of warehouse stocks and their timely replenishment to the required level. The results of testing the software package show that it correctly and accurately optimizes warehouse stocks and performs the necessary management decisions using the 1C:Enterprise program. At the same time, the received reports and recommendations are understandable and allow the user to work effectively.

Keywords: information system, automation of calculations, optimization of warehouse stocks.

Прусс Борис Наумович

Кандидат технических наук, доцент,
ФГБОУ ВО «Брянский государственный
инженерно-технологический университет»
prussbor@gmail.com

Башмаков Алексей Геннадьевич

Старший преподаватель,
ФГБОУ ВО «Брянский государственный
инженерно-технологический университет»
alexbs007@rambler.ru

Плескачева Ольга Юрьевна

Кандидат педагогических наук, доцент,
ФГБОУ ВО «Брянский государственный
инженерно-технологический университет»
pleskacheva@inbox.ru

Аннотация. В статье описан разработанный программный комплекс для оптимизации и планирования закупок. Разработанный комплекс имеет удобный интерфейс и достаточное количество функций, позволяющих пользователю оптимизировать процесс пополнения складских запасов с учетом динамики продаж. С помощью разработанного программного комплекса решается задача оптимизации складских запасов и своевременного их пополнения до требуемого уровня. Результаты апробации работы программного комплекса показывают, что он корректно и точно оптимизирует складские запасы и выполняет необходимые управленческие решения с использованием программы 1С:Предприятие. При этом полученные отчеты и рекомендации понятны и позволяют пользователю эффективно работать.

Ключевые слова: информационная система, автоматизация расчетов, оптимизация складских запасов.

Современный мир немаловажен без торговли, и как следствие, товаров. Торговля этими товарами способствует доставке этих товаров конечному потребителю. Но перед тем, как товар дойдет до конечного адресата, его нужно где-то хранить, а физическое место на складах не безгранично.

Оптимизация планирования закупок — процесс, критически важный в современном мире. Помимо оптимизации физического места на складах, он позволяет различным контрагентам избежать финансовых потерь, ведь когда товары «мертвым» грузом лежат на складах это приносит лишь убытки.

Исходя из перечисленного выше можно сформулировать актуальность данной работы — так как планирование закупок довольно важно, создание соответствующей программы, реализующей данный процесс

только «облегчит» жизнь пользователям данной программы.

Цель работы — разработка программного комплекса для планирования и оптимизации закупок. Данную цель можно разделить на соответствующие задачи: проектирование архитектуры программного комплекса (как общей, так и отдельно для каждого модуля), и последующая его реализация.

В данной работе спроектирован и разработан программный комплекс для оптимизации плана закупок. Так как в современном мире данные об оборотах товаров, могут храниться в различных видах (как в разных программах, так и в разных форматах), то программа будет спроектирована с учетом этого факта и будет предоставлять единый унифицированный интерфейс, с использованием распространенных технологий и протоколов,

что позволит пользователю быстрее ориентироваться в заполняемой информации [1].

Из введения можно понять, что разрабатываемая программа не будет являться частью какой-либо другой большей информационной системы, следовательно, архитектура программы будет не монолитной. Программа будет разделена на две части — клиент и сервер.

Основное приложение будет спроектировано как HTTP — сервер, предоставляющий публичный интерфейс вида `http://<адрес_приложения>/predict-optimize`, принимающий POST — запросы. Это позволит соблюсти требования, обозначенные во введении, а именно возможность взаимодействия с различными программами, независимо от их формата хранения данных.

По данному публичному интерфейсу (`/predict-optimize`) будет располагаться обработчик, выполняющий прогнозирование данных и их дальнейшую оптимизацию.

Прогнозирование будет осуществляться на основе симплекс — метода, со следующими параметрами:

Целевая функция: $value + B \rightarrow \max$, где $value$ — это оборот товара за заданный промежуток времени, а B — прогноз закупок.

Также данному уравнению задаются динамические ограничения:

1. Если количество товара на складе больше, чем оборот за заданный период, то ограничение принимает вид $value + B \leq stock$.
2. Если же оборот товара больше, чем текущий остаток на складе, то ограничение принимает вид $value + B \geq stock$,

Где $value$ — оборот товара, B — прогноз закупок данного товара, а $stock$ — количество оставшегося товара на складе.

После прогнозирования закупок, будет выполнена их оптимизация (также с помощью симплекс — метода).

Целевая функция: $B_1 + B_2 + \dots + B_N \rightarrow \max$, где B_i — прогноз закупок i — товара, $i \leq N$.

$$\text{Ограничение 1: } 0.7 * \frac{T1}{t2} \leq \frac{B_1}{t1},$$

$$\text{Ограничение 2: } 1.3 * \frac{T1}{t2} \geq \frac{B_1}{t1}, \text{ где } T1 \text{ — оборот за за-}$$

данный промежуток времени, $t1$ — промежуток времени оборота, $t2$ — промежуток времени прогнозирования, B_1 — прогноз закупок товара на промежуток времени $t2$.

Клиентская часть приложения, формально, является бесконечным множеством программ, каждая из которых будет представлена расширением/дополнительным модулем к уже существующим программам бухгалтерского и административного учета (например, 1С, Битрикс, Контур, БухСофт и т.п.). Так как серверная часть приложения предоставляет публичный интерфейс взаимодействия, разработка клиентских приложений сводится к написанию тривиального кода получения данных соответствующей платформы и отправки их на сервер в определенном формате.

В данной работе представлено клиентское приложение для платформы 1С: Предприятие 8.3 и конфигурации 1С: Управление Торговлей 11.

Так как серверная программа может быть запущена на любой платформе и любой ОС, следовательно, она должна быть кроссплатформенной. Для обеспечения данного свойства был выбран язык Python, помимо встроенной кроссплатформенности, он облегчает разработку благодаря абстракциям высокого уровня, встроенным в язык [2].

Также в Python есть популярная библиотека для развертывания веб — серверов — Flask. Она также предоставляет различные абстракции для создания эффективных веб — приложений серверного типа. В качестве среды разработки будет использована PyCharm Community Edition 2022 от компании JetBrains.

Для создания клиентского приложения будет использована платформа 1С: Предприятие 8.3.24.

Как уже было обозначено ранее, архитектура информационной системы будет клиент — серверная. Ниже представлена схема взаимодействия между элементами ИС.

Как видно из схемы серверное приложение сможет «общаться» сразу с несколькими различными клиентскими приложениями.

Если формализовывать архитектуру информационной системы, то можно определить серверное приложение как backend, а клиентские приложения как frontend.

На клиентское приложение (frontend) помимо функции взаимодействия с пользователем (графического интерфейса), также наложены обязательства по сбору данных и приведению их в соответствующий формат.

Серверное приложение (backend) будет выполнять вычислительные функции, а именно «предсказывание» закупок товаров на указанный период, а также их последующая оптимизация по заданной целевой функции и ограничениям.

Рассматривая архитектуру серверного приложения следует отметить, что оно будет предоставлять публич-

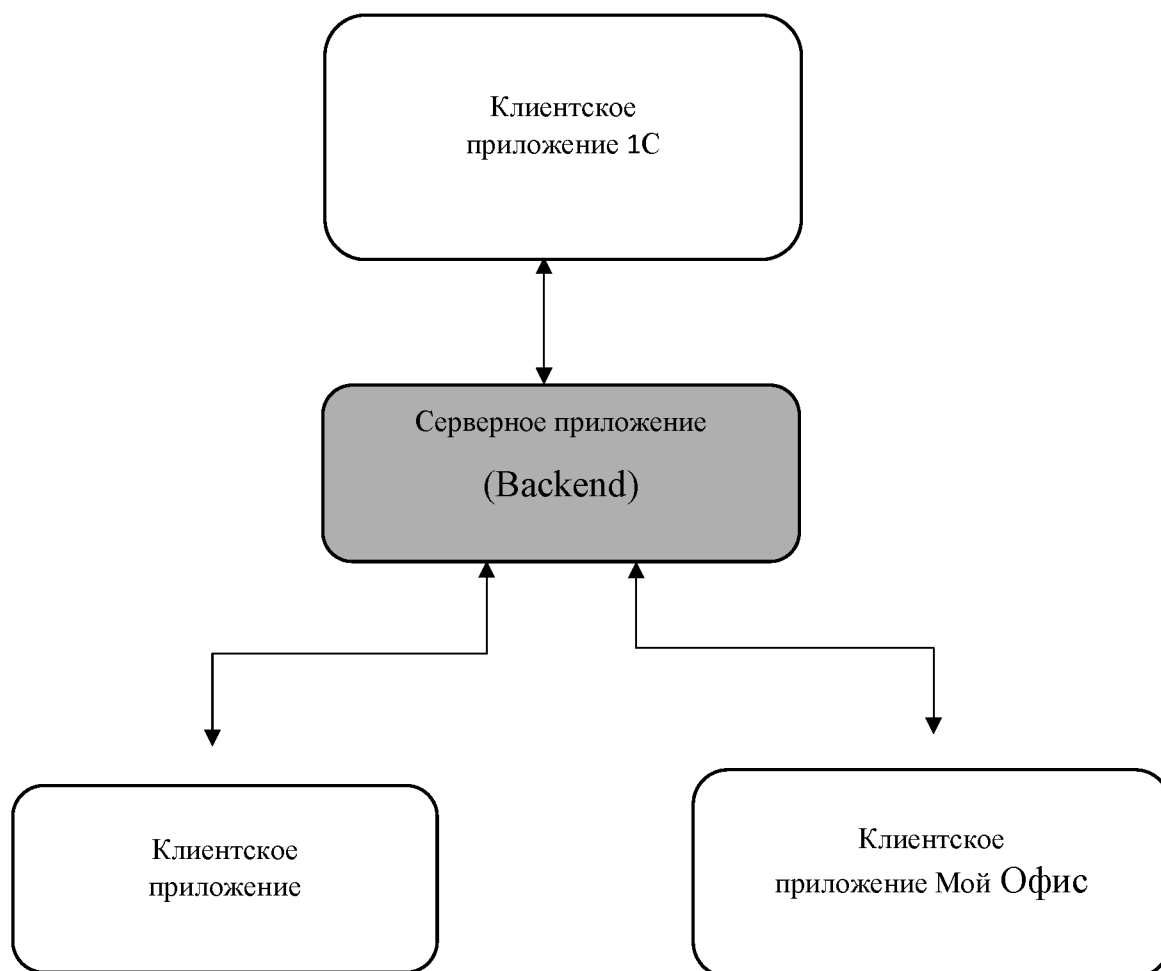


Рис. 1. Схема взаимодействия между элементами ИС

ный интерфейс для взаимодействия с вычислительной частью программного комплекса. Для хранения и взаимодействия с расчётными данными внутри серверного приложения, следует спроектировать соответствующие структуры данных.

Так как данные об оборотах и данные о закупках (до оптимизации и после) имеют схожую структуру, хорошим решением будет их обобщить и создать базовые абстрактные классы, на основе которых уже будут спроектированы основные классы.

Перечислим разработанные структуры данных:

1. Структура данных `AbstractItem` олицетворяет собой абстрактное представление одной единицы товара
2. Структура данных `AbstractData` будет абстрагировать контейнер для хранения всех единиц `AbstractItem`, классы — наследники должны будут реализовывать следующие методы: Теперь можно создать основные структуры данных.
3. Структура данных `SellItem` представляет из себя абстракцию одной единицы входящих на сервер данных (данные оборота товаров). Является наследником класса `AbstractItem` и реализует его абстрактные методы.

4. Структура данных `SellData` представляет собой «ленивый» контейнер для хранения элементов `SellItem`. Фактически `SellData` хранит в себе «сырые» (raw) данные, но перед тем, как передать их далее по потоку управления программы, конструируется объект `SellItem`. Является наследником класса `AbstractData` и реализует его абстрактный метод. Содержит в себе следующие поля и методы.
5. Структура данных `PredictItem` представляет собой один элемент выходных (оптимизированных и/или спрогнозированных) данных закупок. Также является наследником абстрактного класса `AbstractItem`.
6. Структура данных `PredictData` — «ленивый» контейнер для хранения объектов `PredictItem`. Аналогично контейнеру `SellData` хранит «сырой» поток данных (необработанный) и только при соответствующем запросе создает объект класса `PredictItem`. Является наследником класса `AbstractData` и реализует его абстрактные методы.

Ниже на рисунке представлена полная диаграмма иерархии классов.

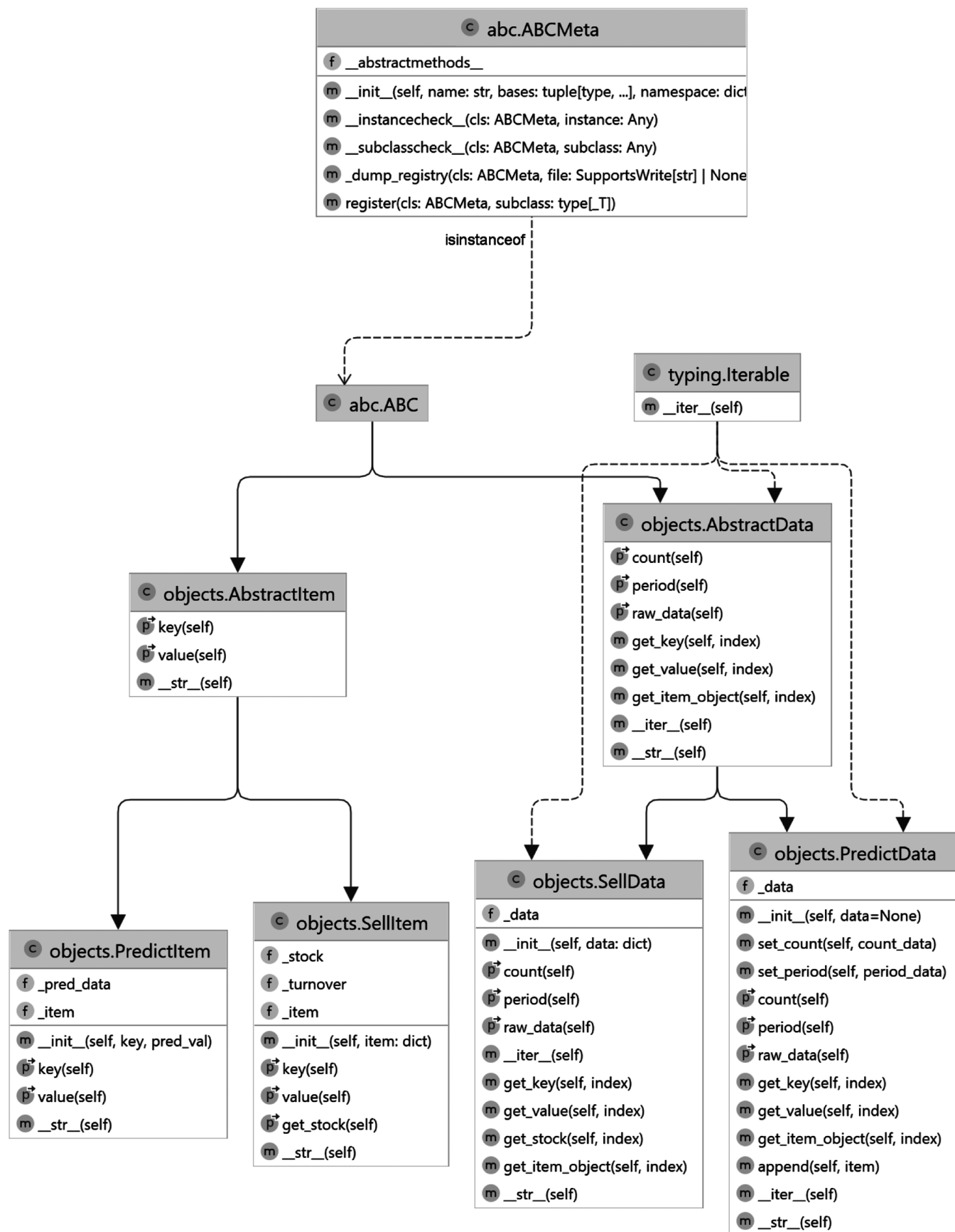


Рис. 2. Иерархия классов структур данных

Рассматривая архитектуру клиентского приложения, следует отметить, что оно представляется неопределенным множеством различных программ. Но если «спуститься» на более практический уровень обсуждения, то здесь будет продемонстрирована архитектура клиентского приложения для платформы 1С: Предприятие 8.3.

Разработка клиентского приложения, фактически, является разработкой пользовательского интерфейса и небольшого количества «внутренних» структур данных и алгоритмов (сбор данных и отправка на серверное приложение). Так как язык разработки 1С: Предприятие 8 не имеет поддержки объектно — ориентированных механизмов, проектирование клиентского приложения сводится к созданию управляемой формы приложения и размещению на ней соответствующих полей.

Ниже на рисунках представлен прототип формы клиентского приложения для платформы 1С: Предприятие 8.3.

Рассматривая процесс реализации серверного приложения, необходимо отметить, что структуры данных были спроектированы ранее и описаны выше, осталось разработать алгоритмы и серверное приложение будет готово к использованию. Но перед тем, как разрабатывать алгоритмы, нужно определиться со структурой передаваемых данных.

Данные между клиентом и сервером будут передаваться в качестве POST — данных HTTP протокола в формате JSON. Этот выбор обусловлен требованиями, представленными ранее, а именно использование распределённых форматов и технологий. Протокол HTTP сейчас используется повсеместно и имеет широкую поддержку в различных библиотеках, аналогичная ситуация с форматом JSON.

Данные от клиента на сервер будут передаваться в качестве массива объектов JSON, первый элемент будет содержать объект count — общее количество пере-

Оптимизация закупок

Дополнительные настройки

Даты
 Дата начала: 26.03.2017
 Дата конца: 30.06.2017
 Период (дни): 96

Настройки сервера
 Адрес сервера: 127.0.0.1
 Протокол: ☒ HTTP Порт: 5000
 Адрес запроса: /predict-optimize

Инфо
 Стандартные значения заданы для конфигурации 1С "Управление торговлей", редакция 11.

Настройки

Источники данных

Источник данных склада: ТоварыНаСкладах.Остатки
 Реквизит номенклатуры склада: Номенклатура
 Реквизит остатка: ВНаличииОстаток

Источник данных оборота: ВыручкаИСебестоимостьПродаж.Обороты
 Реквизит номенклатуры оборота: АналитикаУчетаНоменклатуры.Номенклатура
 Реквизит оборота: КоличествоОборот

Список источников данных:

- БонусныеБаллы
- БонусныеБаллы.Остатки
- ВыручкаИСебестоимостьПродаж
- ВыручкаИСебестоимостьПродаж.Обороты
- ГрафикОтгрузкиТоваров
- ГрафикОтгрузкиТоваров.Остатки
- ГрафикПоступленияТоваров
- ГрафикПоступленияТоваров.Остатки
- ДвижениеТоваров

Бонусная программа лояльности	Бонусные программы лояльности
Клиент	Партнер
Начислено	Число
К списанию	Число


Рис. 3. Прототип формы клиентского приложения 1С

Источники данных

Источник данных склада: <input type="text" value="ТоварыНаСкладах.Остатки"/>	Источник данных оборота: <input type="text" value="ВыручкаИСебестоимостьПродаж.Обороты"/>
<input type="button" value="Установить склад"/>	<input type="button" value="Установить источник оборота"/>
Реквизит номенклатуры склада: <input type="text" value="Номенклатура"/>	Реквизит номенклатуры оборота: <input type="text" value="АналитикаУчетаНоменклатуры.Номенклатура"/>
<input type="button" value="Установить номенклатуру склада"/>	<input type="button" value="Установить номенклатуру оборота"/>
Реквизит остатка: <input type="text" value="ВНаличииОстаток"/>	Реквизит оборота: <input type="text" value="КоличествоОборот"/>
<input type="button" value="Установить остаток"/>	<input type="button" value="Установить оборот"/>

Рис. 4. Настройка источников данных

Дополнительные настройки

Даты	Настройки сервера	Инфо
Дата начала: <input type="text" value="07.12.2016"/>	Адрес сервера: <input type="text" value="127.0.0.1"/>	 Стандартные значения заданы для конфигурации 1С "Управление торговлей", редакция 11.
Дата конца: <input type="text" value="03.07.2017"/>	Протокол: <input checked="" type="radio"/> HTTP Порт: <input type="text" value="5000"/>	
Период (дни): <input type="text" value="208"/>	Адрес запроса: <input type="text" value="/predict-optimize"/>	


Настройки

Оптимизация

Рис. 5. Настройка параметров оптимизации


☆ Оптимизация закупок
Еще ▾

Дополнительные настройки

Даты	Настройки сервера	Инфо
Дата начала: <input type="text" value="07.12.2016"/>	Адрес сервера: <input type="text" value="127.0.0.1"/>	 Стандартные значения заданы для конфигурации 1С "Управление торговлей", редакция 11.
Дата конца: <input type="text" value="03.07.2017"/>	Протокол: <input checked="" type="radio"/> HTTP Порт: <input type="text" value="5000"/>	
Период (дни): <input type="text" value="208"/>	Адрес запроса: <input type="text" value="/predict-optimize"/>	

Настройки

Оптимизация

Данные результата

Еще ▾

Номенклатура	Прогноз закупок
Сок апельсиновый 1 л	182
Сок сливовый 1 л	152
Сок сливовый 2 л	118
Батон нарезной	330
Хлеб дарницкий	184
Плюшка московская	244
Булочка с маком	282
Мука гречневая	577
Вентилятор BINATONE 200 Вт, напольный, оконный	296
Вентилятор настольный, Модель 901	613
Шкаф кухонный	5
Шкаф 3-дверный	15
Тумбочка прикроватная	11
Туалетный столик	20
Лимонад	2 900

Рис. 6. Результат оптимизации

даваемых товаров, второй элемент — объект `days` содержит временной промежуток в днях, для которого были собраны данные о товарах, третий элемент — объект `data` содержащий массив объектов товаров, в каждом объекте массива будет три поля:

1. Поле `key` — уникальный идентификатор товара.
2. Поле `turnover` — данные оборота товара за назначенный период.
3. Поле `stock` — количество товара на складе на конец назначенного периода.

Данные от сервера к клиенту будут также приходить в формате JSON, но в несколько другой структуре, так как «гонять» туда-обратно данные о периоде и количестве товара на складе не имеет смысла (так как клиенту уже известна эта информация). Поэтому от сервера к клиенту поступит обратно только массив с оптимизированными данными о закупках товаров, в каждом объекте товаров будет содержаться два поля:

1. Поле `key` — уникальный идентификатор товара
2. Поле `value` — спрогнозированное и/или оптимизированное количество товара.

Теперь можно перейти к разработке основных алгоритмов вычисления (прогнозирование и оптимизации) закупок. В качестве «ядра» вычислений будет использоваться библиотека `ruLp`, она предоставляет большой набор возможностей линейного программирования.

Сначала для каждого товара, с помощью библиотеки `ruLp` будет вычислен прогноз закупок на следующий временной период, прогнозирование будет осуществлено с помощью симплекс — метода, более подробно описанного в главе «Общее описание программы».

После прогнозирования закупок, данные будут оптимизированы, также с помощью библиотеки `ruLp` и симплекс — метода, математически процесс оптимизации описан в главе «Общее описание программы».

После прохождения всех этапов (прогнозирования и оптимизации) благодаря спроектированным структу-

рам данных, автоматически конвертируются в приемлемый вид, и отправляются на сервер.

Реализация клиентского приложения достаточно тривиально, в основном она сводится к написанию алгоритмов сбора данных из соответствующих источников 1С.

Для разработки этого приложения используется модуль «Настройки» (представленный на рисунке 3), позволяющий выбрать источники данных.

На рисунке 6 представлена основная часть окна настроек, при выборе соответствующих параметров, требуется нажать кнопку «Установить» под определенным параметром. Например, при выборе в списке источников данных (правая колонка на рисунке 5) регистра содержащего данные о товарах на складах, нужно нажать кнопку «Установить склад». Далее в списке реквизитов (левая колонка на рисунке 5) появятся список измерений, ресурсов и реквизитов выбранного источника данных, выбрав атрибут номенклатура склада следует нажать кнопку «Установить номенклатуру склада» и так далее.

После настройки источников данных формируется запрос на выборку из базы данных на основе заполненных полей, из выборки формируется JSON и отправляется на сервер.

Перед началом тестирования следует задать требуемые параметры, основным параметром является диапазон дат, по которому будет выполняться выборка из базы данных и последующие прогнозирование и оптимизация.

На рисунке 6 можно увидеть результат оптимизации на период в 208 дней.

Результаты работы программного комплекса корректно проводят оптимизацию складских запасов и позволяют пользователю эффективно ими управлять. Указанные мероприятия позволяют снизить количество товара на складе компании и высвободить оборотные средства.

ЛИТЕРАТУРА

1. Прусс Б.Н., Романов В.А. Проектирование интерфейса АРМ коменданта общежития // Наука и образование: сохраняя прошлое, создаём будущее. Сборник статей XII международной научно-практической конференции. — Пенза: МЦНС «Наука и Просвещение», 2017. — Ч.1. — С.36–38.
2. Федоров Д.Ю. Программирование на языке высокого уровня Python: учебное пособие для прикладного бакалавриата / Д.Ю. Федоров. — М.: Издательство Юрайт, 2019. — 161 с. — URL: <https://urait.ru/bcode/437489> (дата обращения: 14.06.2025).

© Прусс Борис Наумович (prussbor@gmail.com); Башмаков Алексей Геннадьевич (alexbs007@rambler.ru);

Плескачева Ольга Юрьевна (pleskacheva@inbox.ru)

Журнал «Современная наука: актуальные проблемы теории и практики»