Федеральное государственное бюджетное образовательное

учреждение высшего образования

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

Кафедра информационных систем и программной инженерии

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

к выпускной квалификационной работе

«Информационная система барбершоп»

Выполнил:

Студент группы ИСсп-117

Раков А. Н.

Принял:

?

Владимир, 2021

**АННОТАЦИЯ**

Тема: Разработка информационной системы «Барбершоп».

Ключевые слова: информационная система, барбершоп, расписание, услуга.

В данной работе представлена реализация взаимодействия расписания мастеров барбершопа с конечным клиентом, написанная с использованием WEB технологий.

Выпускная квалификационная работа представлена на 66 страницах, рисунков – 15, использованных источников – 8, приложений – 3, таблиц - 3.

**ABSTRACT**

Subject: Development of information system «Barbershop».

Keywords: information system, barbershop, schedule, service.

This paper presents the implementation of the interaction of the barbershop master schedule with the end client, written using WEB technologies.

The course project is presented on 66 pages, figures-15, used sources-8, applications-3, tables-3.

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 2](#_Toc27480356)

[1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 3](#_Toc27480357)

[1.1 Описание предметной области 3](#_Toc27480358)

[1.2 Цели и задачи разработки 3](#_Toc27480359)

[1.3 Процессы предприятия 4](#_Toc27480360)

[1.4 Описание мобильной конфигурации 6](#_Toc27480361)

[2 РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ 9](#_Toc27480362)

[2.1 Составляющие информационной системы 9](#_Toc27480363)

[2.2. Описание взаимодействия конфигураций 9](#_Toc27480364)

[3. ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ ПРОЕКТА 14](#_Toc27480365)

[3.1 Расчет времени создания проекта 14](#_Toc27480366)

[3.2 Расчет себестоимости программного продукта 15](#_Toc27480367)

[3.3 Расчет экономии за счет увеличения производительности труда 16](#_Toc27480368)

[4. ЗАКЛЮЧЕНИЕ 18](#_Toc27480369)

[5. СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 19](#_Toc27480370)

[ПРИЛОЖЕНИЕ А ВЗАИМОДЕЙСТВИЕ ЭЛЕМЕНТОВ КОНФИГУРАЦИЙ 20](#_Toc27480371)

[ПРИЛОЖЕНИЕ Б КОД КОНФИГУРАЦИИ СЕРВЕРНОГО ПРИЛОЖЕНИЯ 22](#_Toc27480372)

[ПРИЛОЖЕНИЕ В КОД КОНФИГУРАЦИИ МОБИЛЬНОГО ПРИЛОЖЕНИЯ 27](#_Toc27480373)

# ВВЕДЕНИЕ

В барбершопе существует потребность в системах, обеспечивающих ведение учёта работы мастеров, информации о клиентах, хранение информации

обо всех услугах, а также возможность осуществлять запись без участия персонала. Особенность барбершопа подразумевает хранение информации об услугах и клиентах, а также о графике работы самих мастеров – барберах, ведь от этого зависит сможет ли конечный клиент получить необходимую услугу в необходимое время. Создание системы, которая позволила бы все это реализовать, обеспечило бы более качественную учетную деятельность барбершопа.

Эти спецификации применимы не только к барбершопу, но и к другим заведениям, где оказывают подобные услуги. Делается вывод, что данная предметная область актуальна, так как растет спрос на программное обеспечение для таких предприятий. Однако создание и содержание таких систем может быть очень дорогостоящим, что будет не выгодно для конечных предприятий.

Для удешевления и упрощения процесса управления барбершопом разумно создать расширяемую систему на базе WEB технологий.

В рамках данной работы рассмотрена разработка серверной и клиентской

части для возможности взаимодействовать с системой не только за компьютером, но и с использованием любого другого устройства, которое поддерживает выход в интернет и имеет на своём борту браузер.

# 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

## **1.1 Описание предметной области**

Барбершоп в своём классическом варианте - это парикмахерская для мужчин. Присутствие другой половины пола в таких местах исключено по определению. Так как барбершоп является предприятием, предоставляющее людям определённые услуги, то предприятие ведет расписание мастеров и что-нибудь дописать.

На предприятии происходят следующие процессы:

* Ведение расписания мастеров для получения услуг
* Ведение клиентской базы;
* Ведение базы сотрудников
* Может ещё что-то нужно

Исходя из рассмотренных процессов предприятия, в рамках выполнения работы реализован следующий функционал:

1. учет клиентов – персональная информация клиента для связи с ним (ФИО и телефон);
2. учёт мастеров и персонала
3. учёт расписания

**1.2 Цели и задачи разработки**

Цель данного проекта – автоматизация основополагающих процессов работы предприятия для сокращения затрат времени на выполнение посредством внедрения расширяемой информационной системы, включающую серверную и клиентскую конфигурации системы.

Основная задача реализации проекта – удобный процесс записи клиента на необходимую услугу по средствам динамического расписания. Процесс подготовки расписания заканчивается генерацией HTML-таблицы с временными ячейками и условно разделяется на следующие этапы:

1. Создание записей мастеров с информацией о биографии и личной информации.
2. Создание необходимых объектов сущностей для реализации проекта (календарные дни, временные промежутки, услуги и клиенты).
3. Создание определённого SQL запроса, который выполняет функцию сбора данных в один результирующий массив данных.
4. Генерация данных в HTML-таблицу с удобным пользовательским интерфейсом и стилизация для более удобного восприятия.

В итоге, необходимые данные должны быть синхронизированы между серверным и клиентскими приложениями, для упрощения некоторых процессов предприятия.

## **1.3 Процессы предприятия**

Рассмотрим некоторые процессы предприятия, функционал которых необходимо реализовать. В данный перечень входит:

1. Ведение складского учета материала рекламного агентства
2. Выписка товарно - кассового чека для клиента
3. Расчет финансовых оборотов предприятия

Автоматизация данных процессов в информационной системе позволит не только упростить сам процесс проверки данных, но и обеспечить снижение появление некорректных данных о состоянии финансов предприятии или количества материала.Рассмотрим перечисленные процессы в виде диаграмм последовательности до и после внедрения информационной системы на предприятие.

На рисунке 1 представлен процесс ведения складского учета до внедрения информационной системы на предприятие.

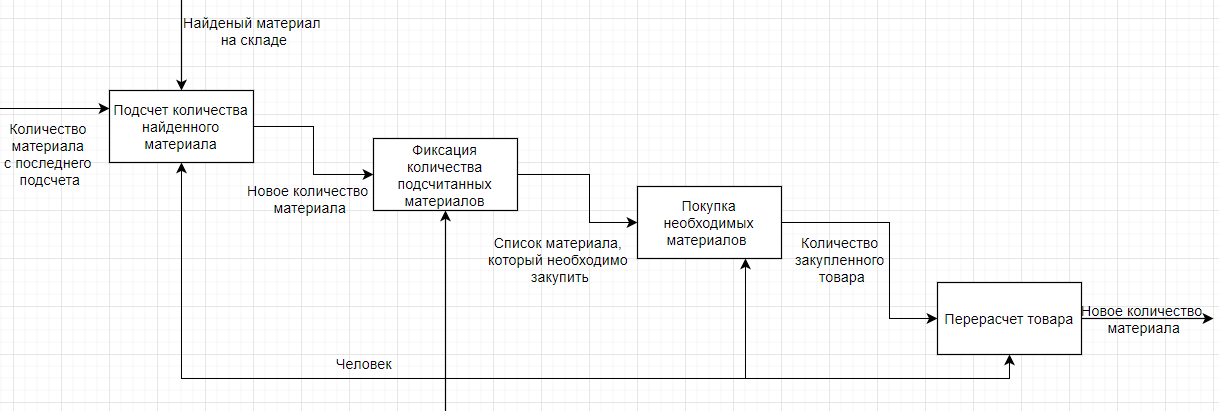


Рисунок 1. Ведение складского учета до внедрения информационной системы

На рисунке 2 представлен процесс ведения складского учета после внедрения информационной системы на предприятие.

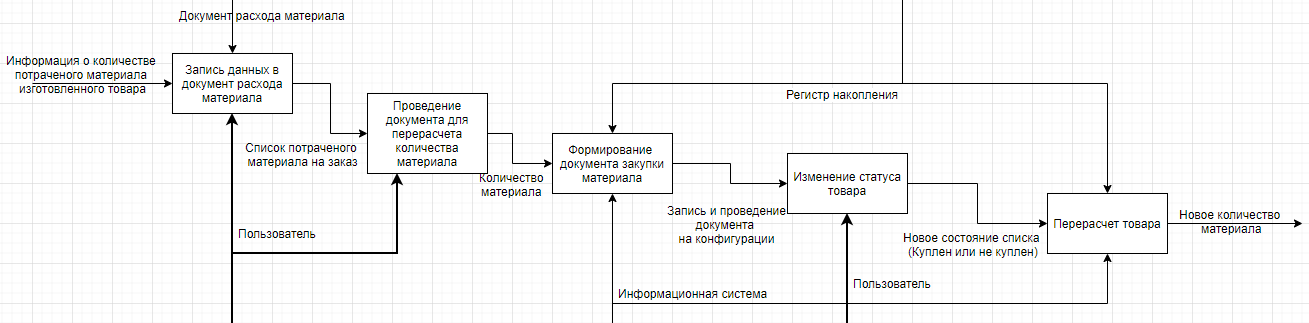


Рисунок 2. Ведение складского учета после внедрения системы

На рисунке 3 представлен процесс ведения учета денежных оборотов предприятия, а так же ведения заказов предприятия до внедрения информационной системы на предприятие.

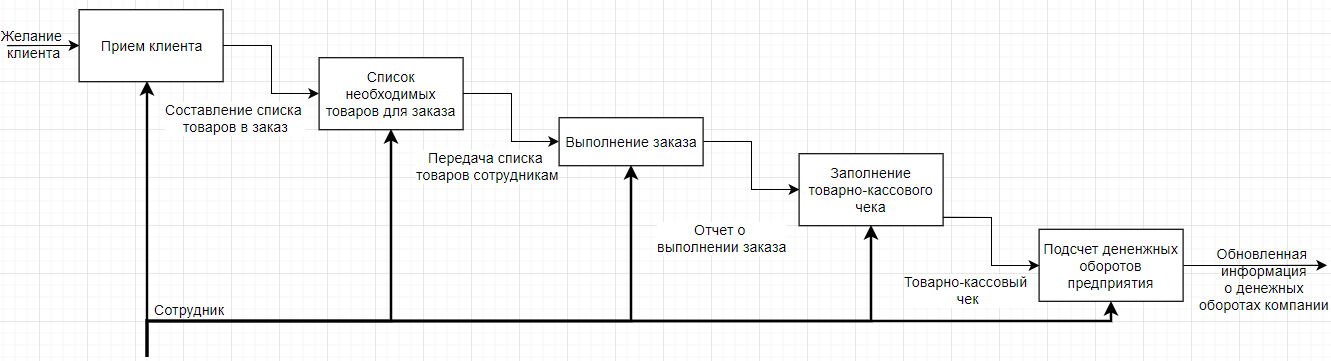


Рисунок 3. Ведение учета денежных средств до внедрения системы

На рисунке 4 представлен процесс ведения учета денежных оборотов предприятия, а так же автоматизация ведения заказов предприятия после внедрения информационной системы на предприятие.

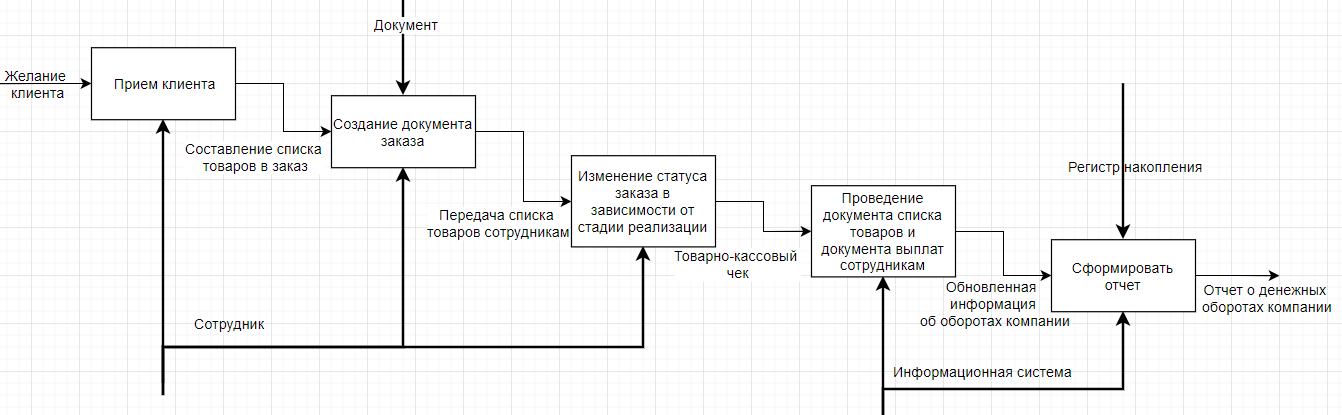


Рисунок 4.Ведение учета денежных средств после внедрения системы

## **1.4 Функциональные и нефункциональные требования**

Мобильное приложение, установленное на устройстве, представляет собой совокупность мобильной платформы и информационной базы. Информационная база на мобильном устройстве содержит аналог файловой базы данных и мобильное приложение. На рисунке 2 представлена компонентная схема мобильной платформы.

Основным назначением мобильных приложений является организация выполнения процессов предприятия, удаленно от рабочего места. Также мобильное приложение может быть рассчитано на работу одновременно с несколькими бэк-офисными приложениями. В том числе и с приложениями, созданными не на платформе 1С:Предприятие. На рисунке 5 представлена архитектура взаимодействия мобильного приложения и основной конфигурацией.

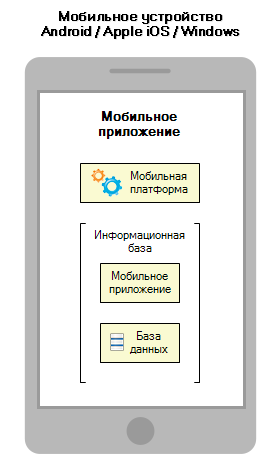


Рисунок 5.Компонентная схема мобильной плапформы

Разработка мобильных приложений ведется так же, как и разработка "обычных" конфигураций 1С, с той лишь разницей, что необходимо учитывать ограничения, которые накладывает мобильная платформа. Для конфигурации, разрабатываемой как мобильное приложение, следует устанавливать свойство «НазначениеИспользования» в значение «МобильноеУстройство». В этом случае система автоматически скроет возможности, недоступные для мобильной платформы, а рабочие инструменты будут настроены на работу именно с тем контекстом встроенного языка, который доступен в мобильной платформе.



Рисунок 6. Архитектура мобильной платформы 1С с одним серверо

Чтобы из конфигурации получить мобильное приложение, готовое для загрузки в магазин приложений, следует выгрузить мобильное приложение в XML файл. Затем, используя "Сборщик мобильных приложений", ,предложенный разработчиками «1С:Предприятие», из XML файла, дистрибутива мобильной платформы и дополнительных файлов собрать дистрибутивы мобильного приложения для операционной системы Android, iOS и Windows.

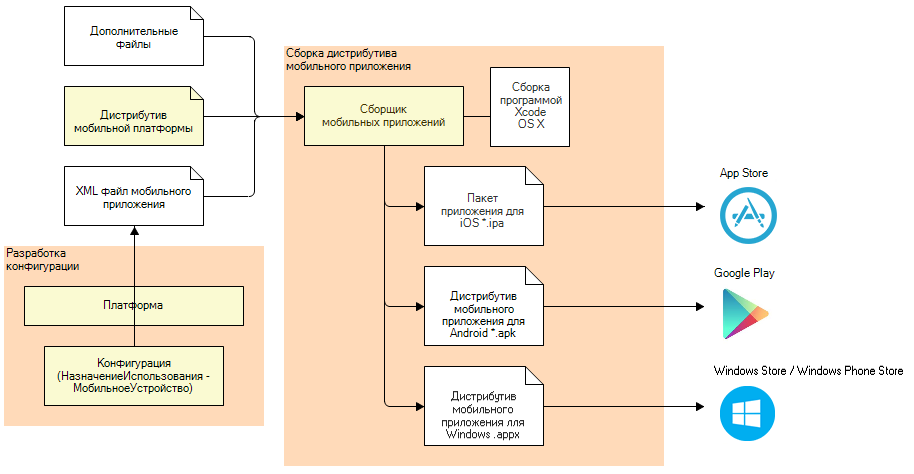


Рисунок 7. Формирование мобильного приложения 1С в зависимости от конечной платформ

# 2. РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ

## **2.1 Технологии и паттерны**

## **2.2 БД**

## **2.3** **Архитектура**

## **2.4 Логика и последовательность**

# 3. РЕАЛИЗАЦИЯ

## **3.1 Тестирование**

## **3.2 Развёртывание**

# 4. ЭКОНОМИКА

## **3.1 ?**

## **3.2 ?**

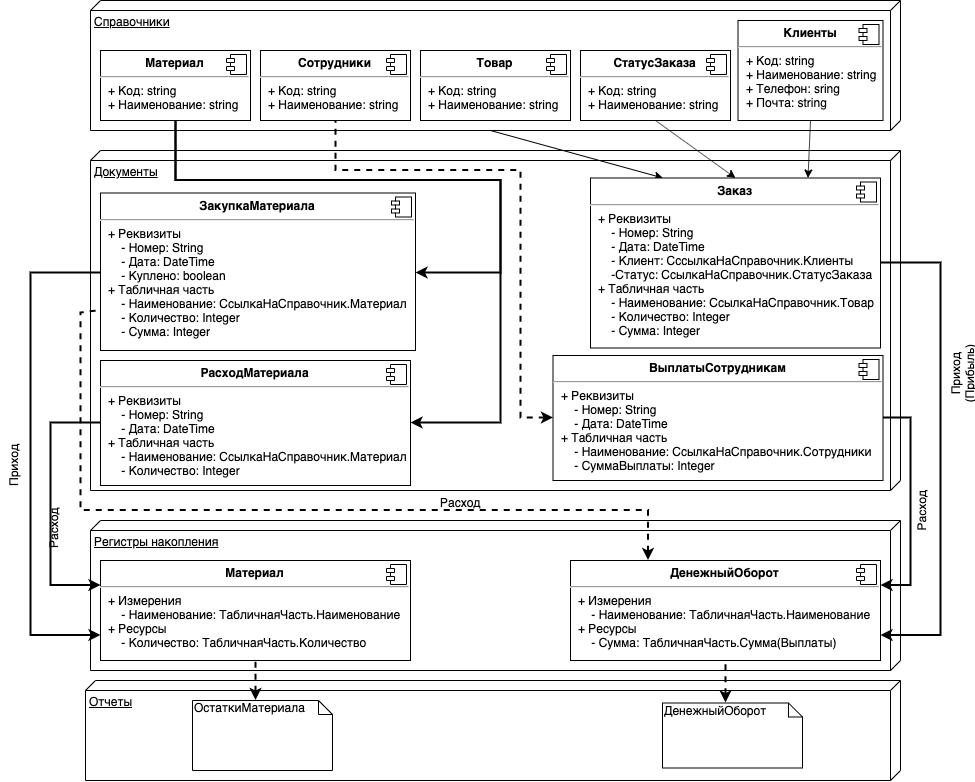
# ЗАКЛЮЧЕНИЕ

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

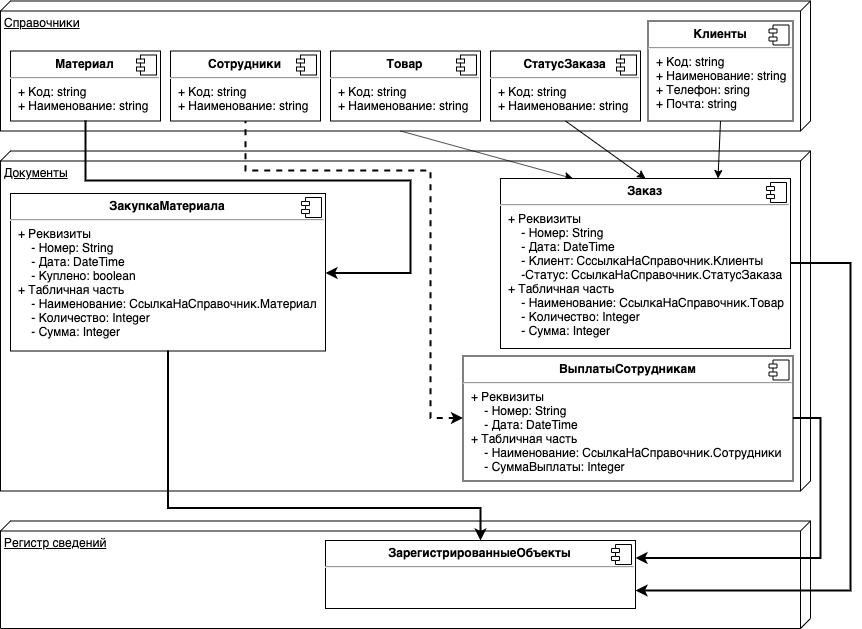
1. Экономические данные Росстата (Электронный ресурс). Режим доступа: https://www.rbc.ru/economics/05/02/2019/5c5948c59a794758389cfdf7 (Дата обращения: 12.12.19)

# ПРИЛОЖЕНИЕ А ВЗАИМОДЕЙСТВИЕ ЭЛЕМЕНТОВ КОНФИГУРАЦИЙ

А.1. Конфигурация серверного приложения



А.2. Конфигурация мобильного приложения



# ПРИЛОЖЕНИЕ Б КОД КОНФИГУРАЦИИ СЕРВЕРНОГО ПРИЛОЖЕНИЯ

Б.1. Код Web-сервисов

Функция AddClientList(Client)

Попытка

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| объекты.Ссылка КАК Ссылка

|ИЗ

| Справочник.Клиент КАК Объекты

|ГДЕ

| Объекты.Код = &Код";

Запрос.УстановитьПараметр("Код", Client.Код);

РезультатЗапроса = Запрос.Выполнить();

ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();

Если ВыборкаДетальныеЗаписи.Следующий() Тогда

объекты = ВыборкаДетальныеЗаписи.Ссылка.ПолучитьОбъект();

Иначе

объекты = Справочники.Клиент.СоздатьЭлемент();

объекты.Код = Client.Код;

КонецЕсли;

объекты.Наименование = Client.Наименование;

объекты.Телефон = Client.Телефон;

объекты.Почта = Client.Почта;

объекты.Записать();

Исключение

Возврат ОписаниеОшибки();

КонецПопытки;

Возврат "Успешно";

КонецФункции

Функция AddMaterial(Material)

Попытка

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| объекты.Ссылка КАК Ссылка

|ИЗ

| Справочник.Материал КАК Объекты

|ГДЕ

| Объекты.Код = &Код";

Запрос.УстановитьПараметр("Код", Material.Код);

РезультатЗапроса = Запрос.Выполнить();

ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();

Если ВыборкаДетальныеЗаписи.Следующий() Тогда

объекты = ВыборкаДетальныеЗаписи.Ссылка.ПолучитьОбъект();

Иначе

объекты = Справочники.Материал.СоздатьЭлемент();

объекты.Код = Material.Код;

КонецЕсли;

объекты.Наименование = Material.Наименование;

объекты.Записать();

Исключение

Возврат ОписаниеОшибки();

КонецПопытки;

Возврат "Успешно";

КонецФункции

Функция GetShoppingList(Code, List)

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| ЗакупкаМатериала.Ссылка КАК Ссылка

|ИЗ

| Документ.ЗакупкаМатериала КАК ЗакупкаМатериала

|ГДЕ

| ЗакупкаМатериала.Номер = &Номер";

Запрос.УстановитьПараметр("Номер", Code.Номер);

РезультатЗапроса = Запрос.Выполнить();

ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();

Если ВыборкаДетальныеЗаписи.Следующий() Тогда

лист1 = ВыборкаДетальныеЗаписи.Ссылка.ПолучитьОбъект();

СтатусДокумента = "найден";

Иначе

лист1 = Документы.ЗакупкаМатериала.СоздатьДокумент();

лист1.Номер = Code.Номер;

СтатусДокумента = "создан";

КонецЕсли;

лист1.Дата = Code.Дата;

лист1.Закуплено = Code.Закуплено;

ТЗ = Новый ТаблицаЗначений;

ТЗ.Колонки.Добавить("Наименование");

ТЗ.Колонки.Добавить("Количество");

ТЗ.Колонки.Добавить("Сумма");

Для ит = 0 по List.Наименование.Количество()-1 Цикл // Схождение кол-ва строчек

стр = тз.Добавить();

стр.Наименование = Справочники.Материал.НайтиПоКоду(List.Наименование[ит]);

стр.Количество = List.Количество[ит];

стр.Сумма = List.Сумма[ит];

КонецЦикла;

Для каждого эл из тз Цикл

стр = лист1.Сотрудники.Добавить();

стр.Наименование = эл.Наименование;

стр.Количество = эл.Количество;

стр.СуммаВыплаты = эл.СуммаВыплаты;

КонецЦикла;

Лист1.Записать(РежимЗаписиДокумента.Запись);

СтатусДокумента = "записан";

Если Code.Проведен Тогда

Лист1.Записать(РежимЗаписиДокумента.Проведение);

СтатусДокумента = "проведен";

Иначе

Лист1.Записать(РежимЗаписиДокумента.ОтменаПроведения);

СтатусДокумента = "не проведен";

КонецЕсли;

Возврат СтатусДокумента;

КонецФункции

Функция AddRab(Client)

Попытка

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| Сотрудники.Ссылка КАК Ссылка

|ИЗ

| Справочник.Сотрудники КАК Сотрудники

|ГДЕ

| Сотрудники.Код = &Код";

Запрос.УстановитьПараметр("Код", Client.Код);

РезультатЗапроса = Запрос.Выполнить();

ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();

Если ВыборкаДетальныеЗаписи.Следующий() Тогда

Сотрудники = ВыборкаДетальныеЗаписи.Ссылка.ПолучитьОбъект();

Иначе

Сотрудники = Справочники.Сотрудники.СоздатьЭлемент();

Сотрудники.Код = Client.Код;

КонецЕсли;

Сотрудники.Наименование = Client.Наименование;

Сотрудники.Записать();

Исключение

Возврат ОписаниеОшибки();

КонецПопытки;

Возврат "Успешно";

КонецФункции

Функция Gold(date, list)

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| ВыплатыСотрудникам.Ссылка КАК Ссылка

|ИЗ

| Документ.ВыплатыСотрудникам КАК ВыплатыСотрудникам

|ГДЕ

| ВыплатыСотрудникам.Номер = &Номер";

Запрос.УстановитьПараметр("Номер", date.Номер);

РезультатЗапроса = Запрос.Выполнить();

ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();

Если ВыборкаДетальныеЗаписи.Следующий() Тогда

лист1 = ВыборкаДетальныеЗаписи.Ссылка.ПолучитьОбъект();

СтатусДокумента = "найден";

Иначе

лист1 = Документы.ВыплатыСотрудникам.СоздатьДокумент();

лист1.Номер = date.Номер;

СтатусДокумента = "создан";

КонецЕсли;

лист1.Дата = date.Дата;

ТЗ = Новый ТаблицаЗначений;

ТЗ.Колонки.Добавить("Наименование");

ТЗ.Колонки.Добавить("СуммаВыплаты");

Для ит = 0 по list.Наименование.Количество()-1 Цикл // Схождение кол-ва строчек

стр = тз.Добавить();

стр.Наименование = Справочники.Сотрудники.НайтиПоКоду(list.Наименование[ит]);

стр.СуммаВыплаты = list.СуммаВыплаты[ит];

КонецЦикла;

Для каждого эл из тз Цикл

стр = лист1.Сотрудники.Добавить();

стр.Наименование = эл.Наименование;

стр.СуммаВыплаты = эл.СуммаВыплаты;

КонецЦикла;

Лист1.Записать(РежимЗаписиДокумента.Запись);

СтатусДокумента = "записан";

Если date.Проведен Тогда

Лист1.Записать(РежимЗаписиДокумента.Проведение);

СтатусДокумента = "проведен";

Иначе

Лист1.Записать(РежимЗаписиДокумента.ОтменаПроведения);

СтатусДокумента = "не проведен";

КонецЕсли;

Возврат СтатусДокумента;

КонецФункции

Функция GetOrders(Code, List)

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| Заказ.Ссылка КАК Ссылка

|ИЗ

| Документ.Заказ КАК Заказ

|ГДЕ

| Заказ.Номер = &Номер";

Запрос.УстановитьПараметр("Номер", Code.Номер);

РезультатЗапроса = Запрос.Выполнить();

ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();

Если ВыборкаДетальныеЗаписи.Следующий() Тогда

лист1 = ВыборкаДетальныеЗаписи.Ссылка.ПолучитьОбъект();

СтатусДокумента = "найден";

Иначе

лист1 = Документы.Заказ.СоздатьДокумент();

лист1.Номер = Code.Номер;

СтатусДокумента = "создан";

КонецЕсли;

лист1.Дата = Code.Дата;

лист1.Клиент = Справочники.Клиент.НайтиПоКоду(Code.Клиент);

лист1.Статус = Справочники.СтатусЗаказа.НайтиПоКоду(Code.Статус);

ТЗ = Новый ТаблицаЗначений;

ТЗ.Колонки.Добавить("Товар");

ТЗ.Колонки.Добавить("Количество");

ТЗ.Колонки.Добавить("Сумма");

Для ит = 0 по List.Наименование.Количество()-1 Цикл // Схождение кол-ва строчек

стр = тз.Добавить();

стр.Товар = Справочники.Товар.НайтиПоКоду(List.Товар[ит]);

стр.Количество = List.Количество[ит];

стр.Сумма = List.Сумма[ит];

КонецЦикла;

Для каждого эл из тз Цикл

стр = лист1.Товары.Добавить();

стр.Товар = эл.Товар;

стр.Количество = эл.Количество;

стр.Сумма = эл.Сумма;

КонецЦикла;

Лист1.Записать(РежимЗаписиДокумента.Запись);

СтатусДокумента = "записан";

Если Code.Проведен Тогда

Лист1.Записать(РежимЗаписиДокумента.Проведение);

СтатусДокумента = "проведен";

Иначе

Лист1.Записать(РежимЗаписиДокумента.ОтменаПроведения);

СтатусДокумента = "не проведен";

КонецЕсли;

Возврат СтатусДокумента;

КонецФункции

# ПРИЛОЖЕНИЕ В КОД КОНФИГУРАЦИИ МОБИЛЬНОГО ПРИЛОЖЕНИЯ

Б.1. Код общего модуля

Процедура зарегистрироватьКОбмену(Об) Экспорт

МенеджерЗаписи = РегистрыСведений.ЗарегистрированныеОбъекты.СоздатьМенеджерЗаписи();

МенеджерЗаписи.Объект = Об.Ссылка;

МенеджерЗаписи.Записать();

КонецПроцедуры

Процедура отменитьРегистрацию(Об) Экспорт

МенеджерЗаписи = РегистрыСведений.ЗарегистрированныеОбъекты.СоздатьМенеджерЗаписи();

МенеджерЗаписи.Объект = Об.Ссылка;

МенеджерЗаписи.Удалить();

КонецПроцедуры

Б.2. Справочник «Материал»

Код модуля объекта

Процедура ПриЗаписи(Отказ)

МодульОбмена.зарегистрироватьКОбмену(ЭтотОбъект);

КонецПроцедуры

Процедура ПриОбменеВС() Экспорт

Попытка

прокси = WSСсылки.WSСсылка1.СоздатьWSПрокси("localhost","AppService","AppServiceSoap");

операция = прокси.ТочкаПодключения.Интерфейс.Операции.Получить("AddMaterial");

параметр = прокси.ФабрикаXDTO.Создать(операция.параметры.Получить("Material").Тип); параметр.Код = ЭтотОбъект.Код;

параметр.Наименование = ЭтотОбъект.Наименование;

параметр.Проверить();

ответ = прокси.AddMaterial(параметр);

Сообщить(ответ);

МодульОбмена.отменитьРегистрацию(ЭтотОбъект);

Исключение

Сообщить(ОписаниеОшибки());

КонецПопытки;

КонецПроцедуры

Код формы списка

&НаКлиенте

Процедура ЗарегистрированныеОбъекты(Команда)

ОткрытьФорму("РегистрСведений.ЗарегистрированныеОбъекты.ФормаСписка");

КонецПроцедуры

&НаКлиенте

Процедура ОтправитьДанные(Команда)

ОтправитьДанныеНаСервере();

КонецПроцедуры

&НаСервере

Процедура ОтправитьДанныеНаСервере()

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ

| ЗарегистрированныеОбъекты.Объект КАК Объект

|ИЗ

| РегистрСведений.ЗарегистрированныеОбъекты КАК ЗарегистрированныеОбъекты";

РезультатЗапроса = Запрос.Выполнить();

ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();

Пока ВыборкаДетальныеЗаписи.Следующий() Цикл

ВыборкаДетальныеЗаписи.Объект.ПолучитьОбъект().ПриОбменеВС();

КонецЦикла;

КонецПроцедуры