Министерство образования и науки Российской Федерации

Южно-Уральский государственный университет

Кафедра системного программирования

**ОТЧЕТ**

**о производственной (преддипломной) практике**

Выполнил:

Н.Н. Демидов, группа КЭ-401

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель практики:

Преподаватель кафедры СП,

Барабанщикова Н.И.

**Введение**

***Актуальность темы***

На сегодняшний день платформа Android является одной из самых популярных мобильных платформ. Данная популярность обуславливается тем, что у Android есть несколько преимуществ по сравнению с другими платформами. Во-первых, порог вхождения в разработку мобильных приложений для Android достаточно низок. Чтобы начать разрабатывать необходимо иметь минимальные знания Java, а также скачать IDE и SDK. Во-вторых, у данной платформы огромное сообщество, благодаря которому появляется большое количество материалов – документаций, книг, онлайн-уроков.

***Цели и задачи***

Целью данной работы является разработка мобильного приложения для инвентаризации товаров путем сканирования штрих-кодов. Мобильное приложение позволит пользователям сканировать штрих-коды товаров и проводить инвентаризацию.

Для достижения поставленной цели были сформулированы следующие задачи:

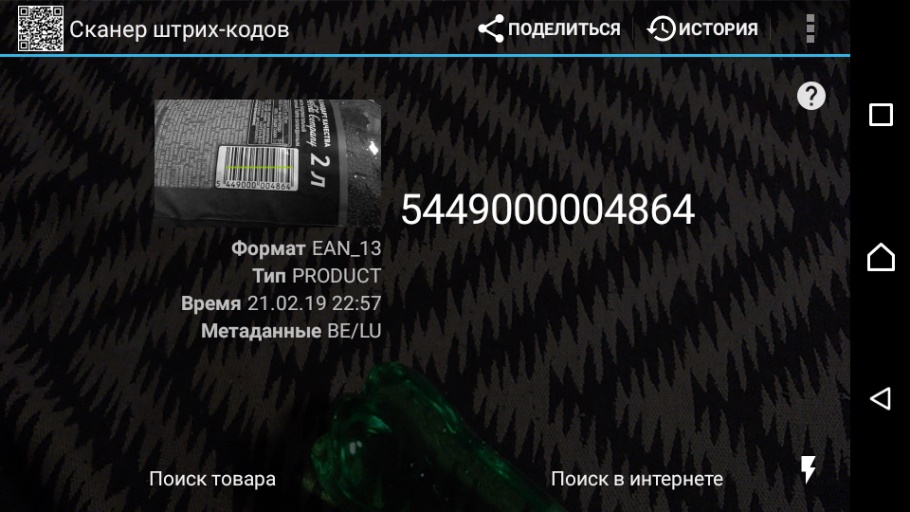
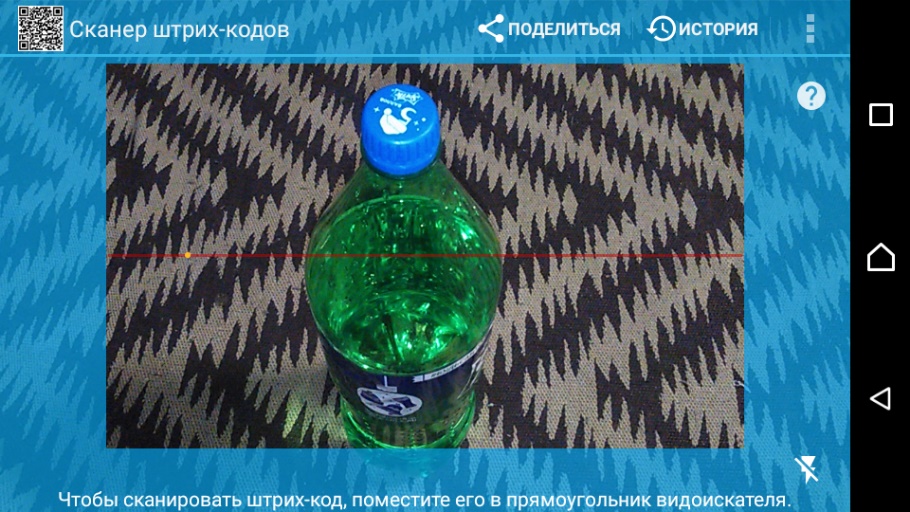
1. провести обзор аналогичных проектов;
2. выявить функциональные и нефункциональные требования, предъявляемые к приложению;
3. провести анализ существующих технологий для реализации приложения;
4. выполнить реализацию приложения;

**1.Обзор литературы**

**1.1. Анализ аналогичных проектов**

***Приложение «Barcode Scanner»***

На данный момент существует множество различных приложений для Android, позволяющих сканировать штрих-коды, а также QR-коды. Одним из таких приложений является “Barcode Scanner” (рис. 1). Особенностью данной программы является возможность просмотра формата штрих-кода, типа сканируемого объекта, а также страны, в которой был произведен сканируемый объект. Помимо этого, приложение предоставляет поиск в интернете сканируемого объекта. Достоинствами данного приложения являются возможность сканирования штрих-кода с изображения из альбома пользователя, быстрое сканирование, а также возможность создавать собственный штрих-код или QR-код.

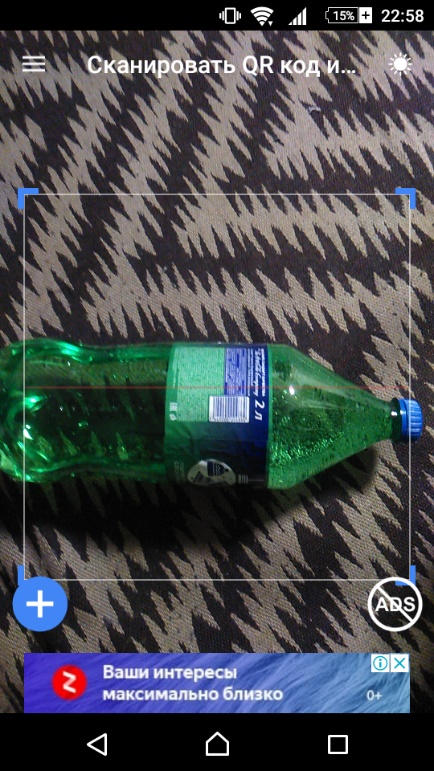
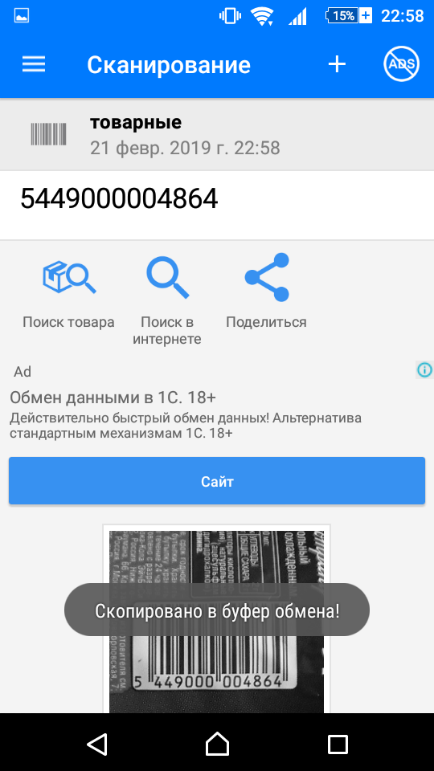


**Рис. 1.** Экран сканнера штрих-кодов в приложении «Barcode Scanner».

Недостатками данного приложения являются обилие рекламы, медленная фокусировка камеры в некоторых случаях и горизонтальная ориентация экрана, не позволяющая сканировать штрих-коды, держа девайс вертикально. Существует возможность приобрести платную версию данного приложения.

***Приложение «Сканер QR и штрих-кодов»***

Еще одним приложением для сканирования штрих-кодов и QR-кодов является “Сканер QR и штрих-кодов” (рис. 2). Данный сканер может считывать все типы штрих-кодов. После сканирования результат сканирования, а именно ссылка или номер штрих-кода автоматически копируется в буфер обмена, после чего пользователю предоставляется возможность найти отсканированный объект в интернете. По сравнению с рассмотренным ранее приложением “Barcode Scanner”, данная программа производит фокусировку камеры быстрее, однако распознавание штрих-кода происходит медленней.

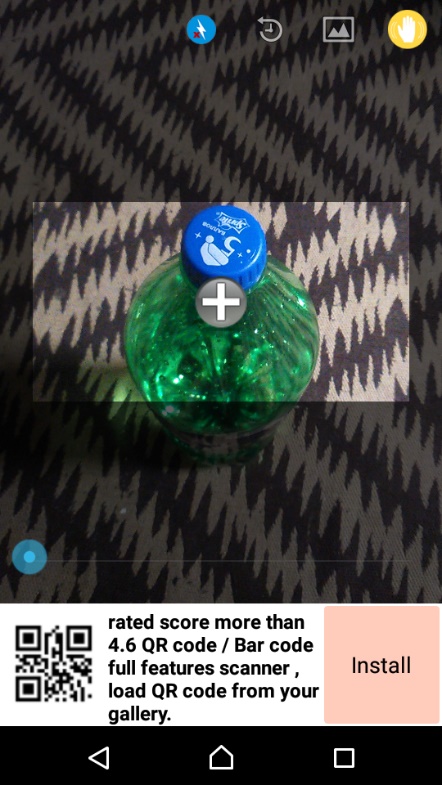
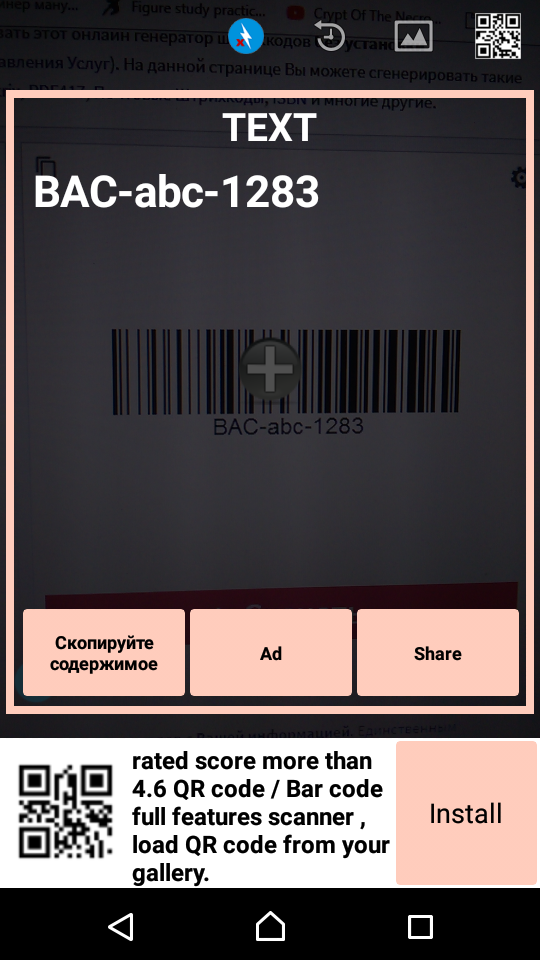
 

**Рис. 2.** Сканирование штрих-кодов в приложении «Сканер QR и штрих-кодов».

Достоинствами данного приложения являются возможность использования вспышки девайса для подсветки штрих-кода и поиск отсканированного объекта в интернете. Из недостатков можно выделить большое количество рекламы и относительно медленное сканирование штрих-кодов и QR-кодов.

***Приложение «Lightning QR»***

На рис. 3 представлено приложение, используемое для декодирования QR-кодов и штрих-кодов таких типов, как ISBN, EAN, UPC. Его достоинствами является возможность использования вспышки для освещения сканируемого штрих-кода, просмотр истории отсканированных товаров и возможность приближения камеры с помощью ползунка, отображенного на экране. К недостаткам можно отнести рекламу, которая появляется даже при отсутствии интернет соединения и низкая скорость считывания при плохой освещенности.

**Рис. 3.** Экран сканирования штрих-кодов в приложении **«**Lightning QR».

***Вывод***

Подводя итоги, можно сказать, что в целом эти приложения достаточно похожи. В работе данных приложений существенных недостатков не было обнаружено, кроме «подвисания» при подгрузке рекламы.

Существующие мобильные приложения для сканирования штрих-кодов не позволяют работать с базой 1С:Предприятие посредством интернет соединения. Мною было принято решение разработать подобное приложение, которое бы позволило сканировать только штрих-коды в коммерческих целях.

**1.2. Обзор используемых технологий.**

Для реализации приложения для воспроизведения прямых видеотрансляций можно использовать следующие технологии разработки мобильных приложений:

1. графические конструкторы для разработки под платформу Android (Nwicode, Flipabit);
2. мобильные ОС и языки программирования мобильных приложений (Kotlin, Swift);
3. среды разработки ПО под Android: IntelliJ IDEA, Android Studio;
4. библиотека ZXing;
5. программный продукт 1С:Предприятие;
6. концепция REST/HTTP Api;

*1.2.1. Графические конструкторы*

Графические конструкторы представляют собой сервисы, позволяющие создавать мобильные приложения для определенных нужд без использования языков программирования. Приложения создаются на базе заготовленных в конструкторе шаблонов с добавлением нужных виджетов и вариантов оформления. Как правило, подобные конструкторы позволяют создавать кроссплатформенные приложения.

Одним из примеров графических конструкторов для создания мобильных приложений является *Nwicode* – это лидер на рынке графических конструкторов приложений. На этой платформе можно создавать не только приложения для Android, iOS, но и Web приложения. Данный конструктор имеет открытый исходный код и стили, возможность интеграции с социальными сетями, а также предоставляет бесплатный хостинг для приложений на первый месяц обслуживания. Однако абонентская плата за использование платформы составит $9.99 ежемесячно.

Другим графическим конструктором мобильных приложений является *Flipabit.* С помощью данного визуального конструктора можно легко создавать и публиковать приложения не только для iOS и Android, но и для Windows и Mac. Flipabit предоставляет несколько шаблонов, с помощью которых можно сделать готовое приложение. Также существует большое количество разнообразных виджетов, которые могут понадобиться в создании приложения – изображения, видео, карты, веб-страницы, графики и диаграммы, таблицы, календари, база данных и прочее. Помимо виджетов в конструкторе есть возможность с помощью JavaScript и QML изменять поведение приложения. Имеется большое количество документации и форум, на котором можно обсудить вопросы, связанные с разработкой приложения. Существуют как бесплатная версия данного конструктора с урезанным функционалом, так и платные версии за $35 и $125 в месяц.

Достоинством любых графических конструкторов в сравнении со средами разработки является возможность написания программ без каких-либо знаний языков программирования, используя готовые шаблоны. Недостатком является отсутствие возможности создания гибкого приложения и ограниченная функциональность приложения, а также некоторые проблемы с безопасностью.

*1.2.2. Мобильные платформы и языки программирования*

На сегодняшний день популярными мобильными платформами являются Android, iOS, а также Windows Phone. Данные платформы пользуются спросом во всем мире. Они удобны в использовании и предоставляют пользователю ряд функций для работы, а также для развлечения.

С точки зрения надежности платформа iOS является закрытой. Мобильное устройство поставляется со всеми необходимыми заводскими настройками и избавляет пользователя от необходимости проводить настройку своего смартфона. Однако, с другой стороны, это и является минусом, так как нет возможности расширить память смартфона, нет дополнительных структурных компонентов. Платформа Android является открытой. Мобильное устройство перед использованием требует тщательной настройки, что, с одной стороны, требует времени, а с другой стороны, позволяет пользователю учесть все свои требования к смартфону и настроить его так, как необходимо пользователю. Windows занимает среднее положение между iOS и Android – настройки минимальны, но всё же существуют.

С точки зрения безопасности iOS и Windows имеют встроенную защиту от вирусов, так что автономные приложения этого действия ей не нужны. А вот с Android пользователю приходится сталкиваться с огромным количеством вирусов, так что без установки дополнительных приложений, отвечающих за безопасность, пользователю не обойтись.

Важным шагом в создании приложения является выбор языка программирования. В данной работе будет произведено сравнение двух современных популярных языков программирования, предназначенных для разработки мобильных приложений.

*Kotlin* — это статически типизированный язык программирования от компании JetBrains, работающий поверх JVM и созданный специально для разработки Android приложений [10]. Компилируется также в JavaScript и на другие платформы через инфраструктуру LLVM. Основными достоинствами данного языка программирования являются:

1. лаконичность языка;
2. язык поддерживается компанией Google;
3. полная совместимость с Java;

Недостатками языка являются:

1. меньшая скорость компиляции в сравнении с Java;
2. значительно меньшее комьюнити разработчиков в сравнении с Java;

*Swift* — это надёжный и интуитивно понятный язык программирования от Apple, при помощи которого можно создавать приложения для iOS, Mac, Apple TV и Apple Watch. Этот язык является заменой языку Objective-C. Достоинствами языка Swift являются:

1. для написания приложения требуется меньше кода;
2. удобен для чтения;
3. повышенная безопасность;

Недостатками данного языка являются:

1. минимальное знание Objective-C;
2. сбивающие с толку ошибки, выдаваемые компилятором;

1.2.3. Среды разработки мобильных приложений

Важнейшим элементом в процессе создания приложения в среде разработки IDE является уровень собственной подготовки. От него будет зависеть качество и быстродействие приложения.

*IntelliJ IDEA* — интегрированная среда разработки программного обеспечения для многих языков программирования, разработанная компанией JetBrains [6]. Достоинствами данной среды разработки являются:

1. множество языков программирования: Java, Kotlin, JavaScript, Python, Ruby, Groovy, Scala, PHP, C, C++;
2. интеграция с Git;

К недостаткам можно отнести относительно низкую производительность, долгое ожидание выполнения компиляции.

*Android Studio* — интегрированная среда разработки производства Google, с помощью которой разработчикам становятся доступны инструменты для создания приложений на платформе Android ОС [5]. Достоинства:

1. встроенный SDK;
2. удобный конструктор интерфейсов;

Основным недостатком данной среды можно выделить постоянную синхронизацию с Gradle, которая значительно замедляет компиляцию проектов.

*1.2.4. Библиотека ZXing*

*ZXing* – многоформатная библиотека обработки и распознавания изображений 1D/2D штрих-кодов с открытым исходным кодом, реализованная на Java и портированая на другие языки программирования, например: C++, Objective-C, ActionScript, Python, Delphi и множество других. При добавлении данной библиотеки к разрабатываемому проекту появляется возможность использовать функции, позволяющие изменять вид сканируемых штрих-кодов (штрих-коды или QR-коды), использовать звуковой сигнала после сканирования, изменять ориентацию экрана со сканером и сканировать штрих-код.

*1.2.5. 1С:Предприятие*

*1С:Предприятие* – программный продукт, предоставляющий инструменты, которые позволяют автоматизировать деятельность предприятий, разных компаний и направлений бизнеса. Данный продукт включает в себя саму платформу и программные продукты, разработанные на ее основе.

Гибкость платформы позволяет использовать систему программ 1С:Предприятие для автоматизации учета и управления на производственных предприятиях, в бюджетных и финансовых организациях, предприятиях оптовой и розничной торговли, сферы обслуживания, а также множество других направлений.

У данной платформы существуют встроенные инструменты формирования отчетов и печатных форм, обеспечивающие широкие возможности оформления и интерактивной работы:

* возможность формирования иерархических отчетов;
* различные типы диаграмм для графического представления экономической информации;
* сводные таблицы для анализа многомерных данных, динамическое изменение структуры отчета;

Масштабируемость системы 1С:Предприятие позволяет работать как в однопользовательском режиме, так и обеспечивает возможность параллельной работы большого количества пользователей. Помимо этого, данная система позволяет также эффективно работать при увеличении количества решаемых задач и объема обрабатываемых данных.

Также данная система дает возможность создавать собственные HTTP-сервисы и реализовывать обработку основных методов: GET, POST, DELETE, PUT. Таким образом нет необходимости создавать сторонний сервер-посредник между 1С:Предприятие и мобильным приложением.

*1.2.6. Концепция REST/HTTP Api*

*HTTP* – протокол, который описывает взаимодействие между двумя компьютерами (клиентом и сервером), построенное на базе сообщений, называемых запросом и ответом. Каждое сообщение состоит из трех частей: стартовая строка, заголовки и тело. При этом обязательной является только первая срока.

*REST* - это стиль архитектуры программного обеспечения для распределенных систем, таких как World Wide Web, который, как правило, используется для построения веб-служб. REST является очень простым интерфейсом управления информацией без использования каких-то дополнительных внутренних прослоек. Каждая единица информации однозначно определяется глобальным идентификатором, таким как URL. Каждая URL в свою очередь имеет строго заданный формат.

В итоге, совместив имеющуюся спецификацию HTTP и REST-подход наконец-то обретают смысл различные HTTP-методы. GET — возвращает ресурс, POST — создает новый, PUT — обновляет существующий, DELETE — удаляет.

**Вывод**

В результате анализа существующих решений, а также обзора средств разработки было принято решение разработать мобильное приложение на платформе Android в интегрированной среде разработки Android Studio на языке программирования Java. В качестве хранения данных и HTTP-сервиса был выбран программный продукт 1С:Предприятие, так как клиенты, для которых разрабатывается данное приложение используют 1С:Предприятие.

**2. Требования к системе.**

***Функциональные требования***

Функциональные требования определяют функциональность программного обеспечения, то есть описывают, что необходимо реализовать в продукте или системе, какие возможности должна предоставлять разрабатываемая система. Функциональные требования включают в себя бизнес-требования и пользовательские требования. Были выявлены следующие функциональные требования мобильного приложения для сканирования штрих-кодов:

1. Приложение должно предоставлять интерфейс для сканирования штрих-кода.
2. Приложение должно производить сканирование штрих-кода.
3. Приложение должно получить информацию о товаре из базы 1С:Предприятие.
4. Приложение должно добавлять отсканированный товар в список отсканированных товаров.
5. Приложение должно предоставлять интерфейс для отображения полученной информации о товаре.
6. Приложение должно предоставлять интерфейс для добавления новых товаров в базу данных.
7. Приложение должно предоставлять интерфейс для изменения количества единиц отсканированного товара.
8. Приложение должно производить удаление выбранного товара из списка отсканированных товаров.
9. Приложение должно формировать документ инвентаризации в 1С:Предприятие.
10. Приложение должно локально сохранять список отсканированных товаров;

***Нефункциональные требования***

Нефункциональные требования описывают свойства и ограничения, накладываемые на систему. Нефункциональные требования определяют, как должна работать система или программный продукт, и какими свойствами или характеристиками она должна обладать. Для реализации приложения сканирования штрих-кодов были сформулированы следующие нефункциональные требования:

1. Приложение должно быть разработано на ОС Android начиная с версии 4.2.
2. Приложение должно быть разработано в интегрированной среде разработке Android Studio.
3. Приложение должно быть разработано на языке программирования Java.
4. Приложение должно использовать базу 1С:Предприятие для хранения списка товаров.
5. Приложение должно использовать HTTP-сервис, реализованный в 1С:Предприятие.

**Варианты использования приложения**

Диаграмма вариантов использования применяется для моделирования функциональных требований к программному продукту при его проектировании и разработке.

На рис. 4 представлена диаграмма вариантов использования мобильного приложения для сканирования штрих-кодов. Диаграмма вариантов использования отражает отношение между актерами и прецедентами.

*Основные актеры, взаимодействующие с системой*

В данной системе существует единственный актер – *пользователь.* *Пользователь* может взаимодействовать с приложением: сканировать штрих-код, просматривать список отсканированных товаров, добавлять новые товары в базу, изменять количество единиц отсканированного товара, удалять товар из списка отсканированных товаров и формировать документ инвентаризации.



**Рис. 4.** Диаграмма вариантов использования.

*Описание вариантов использования системы*

На рис. 4 представлены следующие базовые варианты использования мобильного приложения:

1. *Сканировать штрих-код* – сканирование штрих-кода выбранного товара.
2. *Просмотреть список отсканированных товаров* – просмотр списка отсканированных товаров.
3. *Добавить новые товары в базу* – добавление отсканированного товара в базу 1С:Предприятие.
4. *Изменить количество единиц отсканированного товара* – изменение количества единиц отсканированного товара из списка отсканированных товаров.
5. *Удалить товар из списка отсканированных товаров* – удаление товара из списка отсканированных товаров.
6. *Сформировать документ инвентаризации* – формирование документа инвентаризации в 1С:Предприятие.

**3. Реализация**

**3.1. Схема БД в 1С:Предприятие**

Схема базы данных включает в себя описания содержания, структуры и ограничений целостности, используемые для создания и поддержки базы данных.



**Рис. 5**. Схема базы данных.

На рис. 5 изображена схема базы данных в 1С:Предприятие, которая используется для отображения таблиц, которые хранят номенклатуру, характеристики номенклатуры, партии номенклатуры и штрих-коды номенклатуры.

Справочник в 1С:Предприятие представляет собой прикладной объект конфигурации. Он позволяет хранить в информационной базе данные, имеющие одинаковую структуру и списочный характер. Это может быть, например, список контрагентов, перечень товаров, список поставщиков или покупателей. В данном случае справочник используется для хранения номенклатуры, характеристики номенклатуры и партий номенклатуры.

*Справочник.Номенклатура* – справочник, содержащий в себе название товара и ссылку на него в базе.

*Справочник.ХарактеристикиНоменклатуры* – справочник, который содержит поля НаименованиеДляПечати и ссылку на характеристику в базе.

*Справочник.ПартииНоменклатуры* – справочник, содержащий такие поля, как ВладелецПартии, являющийся в свою очередь ссылкой на запись из таблицы Контрагенты, Статус, являющийся ссылкой на Перечисление.СтатусыПартий. Поле Ссылка является ссылкой на запись в базе.

Регистры сведений – прикладные объекты конфигурации. Они позволяют хранить в прикладном решении произвольные данные в разрезе нескольких измерений. Например, в регистре сведений можно хранить курсы валют в разрезе валют, или цены предприятия в разрезе номенклатуры и типа цен.

Информация в регистре сведений хранится в виде записей, каждая из которых содержит значения измерений и соответствующие им значения ресурсов. Измерения регистра описывают разрезы, в которых хранится информация, а ресурсы регистра непосредственно содержат хранимую информацию.

*РегистрСведений.ШтрихкодыНоменклатуры* – регистр сведений, хранящий штрих-коды товаров, содержащий единственное измерение – *Штрихкод.* Ресурсами данного регистра являются *Номенклатура*, *ХарактеристикаНоменклатуры* и *Партия*.

**3.2. Реализация приложения**

Реализация мобильного приложения была разбита на следующие этапы:

1. Конфигурация приложения. Создание и настройка android-приложения в Android Studio, добавление запрашиваемых разрешений в файл манифеста.
2. Сканирование штрих-кода.
3. Работа с HTTP-сервисом.

- диаграмма классов

***3.2.1. Конфигурация приложения и структура проекта***

Для использования методов работы с интернет запросами в android необходимо получить разрешения "android.permission.INTERNET" и "android.permission.ACCESS\_WIFI\_STATE". Помимо этого были использованы разрешения "android.permission.VIBRATE" для использования вибрации девайса и "android.permission.CAMERA" для использования камеры Эти разрешения должны быть записаны в файл AndroidManifest.xml и запрошены у пользователя в момент установки приложения.

Для создания приложения использовался пустой шаблон, содержащий основной класс приложения MainActivity.java, а также основной файл разметки activity\_main.xml.

* + 1. ***Реализация сканирования штрих-кода***

В основном классе MainActivity были использованы методы и классы, реализованные в библиотеке ZXing (рис.6). Данный код создает объекта класса IntentIntegrator, выполняет блокирование ориентации экрана setOrientationLocked(true), а также использует предпочтительные типы штрих-кодов IntentIntegrator.PRODUCT\_CODE\_TYPES при инициализации сканирования. IntentIntegrator производит вызов новой Activity.

IntentIntegrator integrator = new IntentIntegrator(mActivity);

Integrator.setOrientationLocked(true);

integrator.initiateScan(IntentIntegrator.PRODUCT\_CODE\_TYPES);

**Рис.6**.Листинг использования класса IntentIntegrator для сканирования штрих-кодов.

Код, представленный на рис.6, был привязан к кнопке scanBtn(рис. 7) с использованием метода setOnClickListener.

scanBtn = (ImageButton) findViewById(R.id.scanButton);

scanBtn.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

}

});

**Рис. 7**. Листинг метода setOnClickListener().

Для обработки результата сканирования был реализован метод onActivityResult (рис. 8), так как сканирование происходит в другой Activity. Данный метод позволяет получить данные из другой Activity.

@Override

protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {

IntentResult result = IntentIntegrator.parseActivityResult(requestCode, resultCode, data);

if (result != null) {

if (result.getContents() != null) {

//Обработка результата сканирования

} else {

//Сканирование было отменено

}

} else {

super.onActivityResult(requestCode, resultCode, data);

}

}

**Рис.8**. Листинг метода onActivityResult.

* + 1. ***Работа с HTTP-сервисом***

Для отправки API запросов был реализован класс *JSONAsyncTask*, который расширяет абстрактный класс *AsyncTask*, который позволяет выполнять сложные задачи в фоновом потоке. На рис. 10. изображены основные методы, использующиеся в классе *JSONAsyncTask.* Это методы *doInBackground* и *onPostExecute*.

@Override

protected void onPostExecute(String s) {

super.onPostExecute(s);

if (s.equals("0")) {

Toast.makeText(MainActivity.this, "Отсутствует соединение с сервисом(", Toast.LENGTH\_LONG).show();

} else if (s != null) {

Object obj = null; // Object obj = new JSONParser().parse(new FileReader("JSONExample.json"));

try {

obj = new JSONParser().parse(s);

} catch (ParseException e) {

e.printStackTrace();

}

// Кастим obj в JSONObject

JSONObject jo = (JSONObject) obj;

String id = null;

String name = null;

String characteristic = null;

String party = null;

String error = (String) jo.get("error");

if (error.equals("false")) {

id = (String) jo.get("id");

name = (String) jo.get("name");

characteristic = (String) jo.get("characteristic");

party = (String) jo.get("party");

} else {

id = "Error";

name = (String) jo.get("msg");

}

codeView.setText(id);

nameView.setText(name);

charView.setText(characteristic);

partyView.setText(party);

}

}

@Override

protected String doInBackground(String... urls) {

URL url;

HttpURLConnection urlConnection = null;

String server\_response = null;

try {

url = new URL(urls[0]);

urlConnection = (HttpURLConnection) url.openConnection();

urlConnection.setRequestMethod("GET");

urlConnection.setRequestProperty("Authorization", "Basic username:password");

int responseCode = urlConnection.getResponseCode();

if (responseCode == HttpURLConnection.HTTP\_OK) {

server\_response = readStream(urlConnection.getInputStream());

Log.v("CatalogClient", server\_response);

}

} catch (MalformedURLException e) {

e.printStackTrace();

} catch (IOException e) {

server\_response = "0";

e.printStackTrace();

} finally {

if (urlConnection != null) {

urlConnection.disconnect();

}

}

return server\_response;

}

**Рис. 10**. Листинг методов *onPostExecute* и *doInBackground* класса *JSONAsyncTask*.

В методе *doInBackground* (рис. 9) происходит подключение к HTTP-сервису с помощью объекта класса *HttpURLConnection*. Выставляется метод запроса, а также дополнительное свойство – данные для авторизации. По стандарту строка «username:password» кодируется в формат Base64. После чего происходит проверка на подключение к сервису. Если проверка успешна, то в переменную server\_response считывается с помощью метода *readStream* (рис. 11) поток данных от HTTP-сервиса.

private String readStream(InputStream in) {

BufferedReader reader = null;

StringBuffer response = new StringBuffer();

try {

reader = new BufferedReader(new InputStreamReader(in));

String line = "";

while ((line = reader.readLine()) != null) {

response.append(line);

}

} catch (IOException e) {

e.printStackTrace();

} finally {

if (reader != null) {

try {

reader.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

return response.toString();

}

**Рис. 11**. Листинг метода *readStream*.

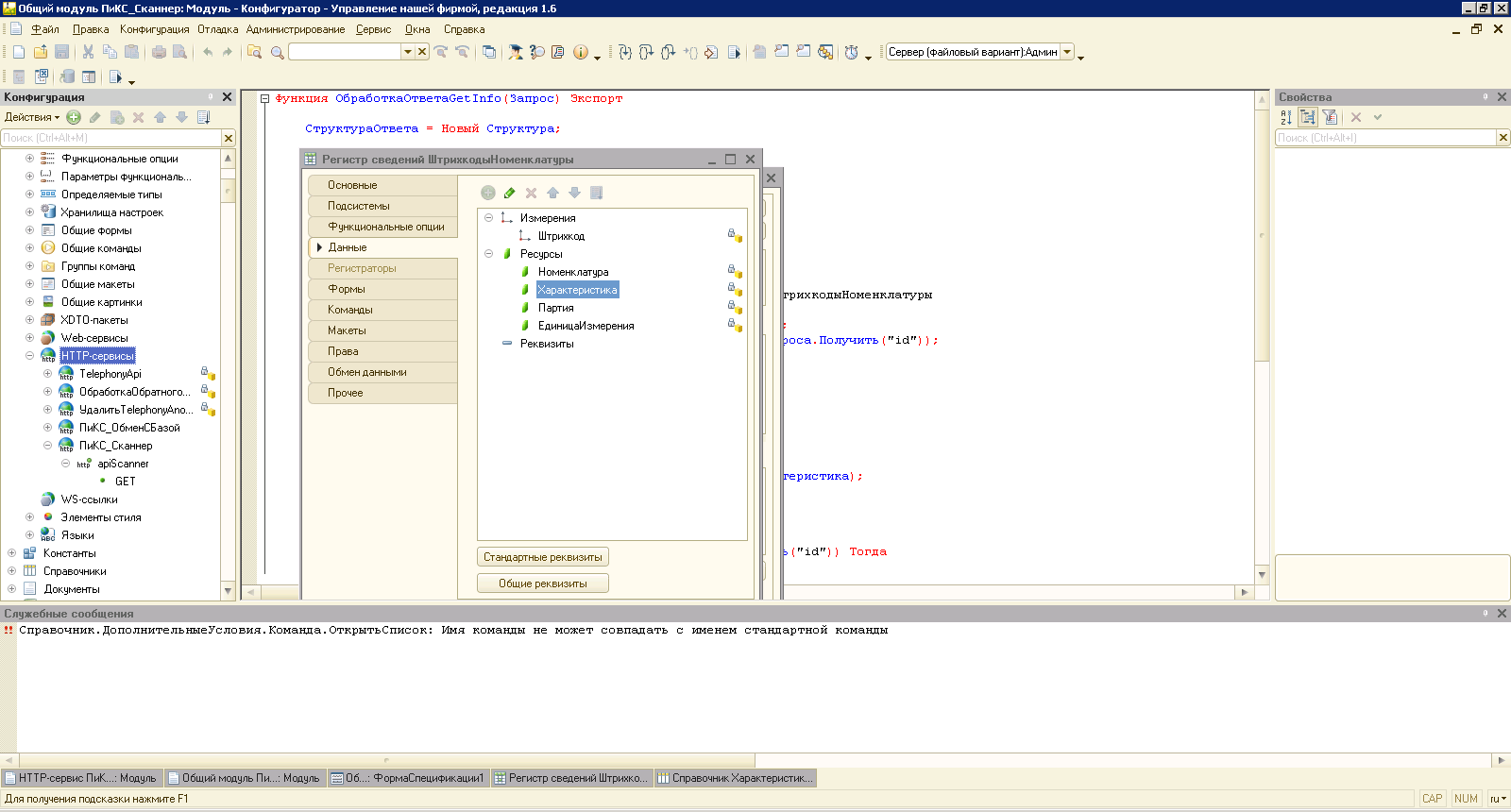
Метод, приведенный на рис. 11, позволяет считывать данные с входного потока данных и возвращает считанные данные в формате строки.

В методе *onPostExecute* (рис. 10) происходит парсинг полученной информации в формате JSON от сервиса, а также вывод этой информации на экран.

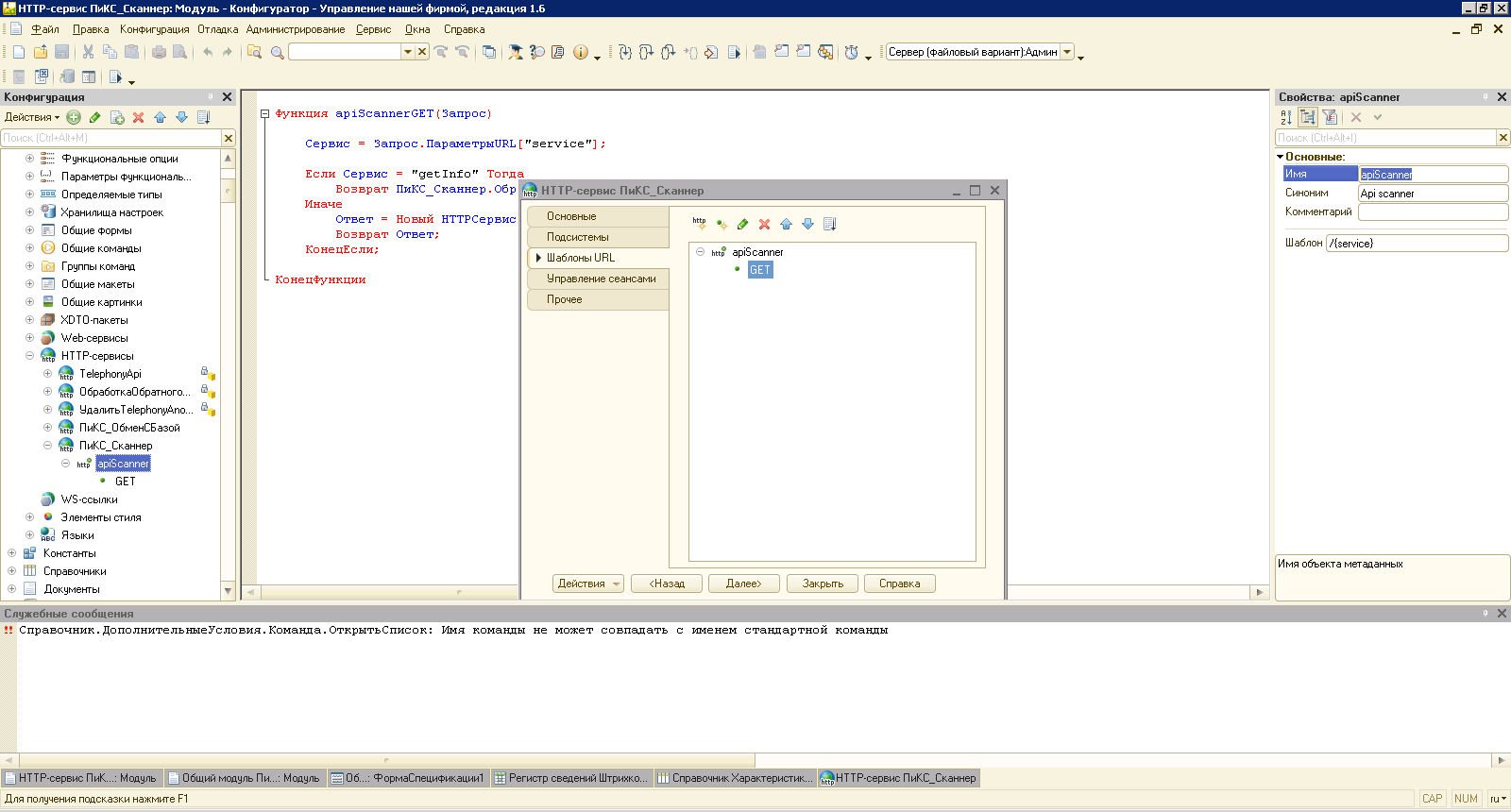
* 1. **Настройка 1С и реализация HTTP API**

В программе 1С:Предприятие была использована существующая тестовая база, в которой уже были созданы объекты конфигурации, описанные в разделе 3.1. Помимо объектов конфигурации, есть необходимость создать нового пользователя с правами администратора, под которым будет происходить авторизация при подключении к сервису

1С:Предприятие с помощью объекта конфигурации «HTTP-сервисы» был создан HTTP-сервис ПиКС\_Сканнер(рис. 12), корневым URL которого является apiScanner. Одним из свойств данного объекта является шаблон URL, который позволяет использовать несколько методов вместо одного. В данном случает был выбран шаблон “/{service}”, где *service* является ключевым словом, с помощью которого можно будет определять метод. В apiScanner был добавлен метод GET, который реализован в модуле ПиКС\_Сканнер (рис. 14).



**Рис. 12**. HTTP-сервисы в дереве конфигурации.



**Рис. 13**. Свойства apiScanner.

В функции *apiScannerGET* происходит проверка запроса на параметр service. Так как реализован только метод получения информации о сканируемом товаре, то в данной функции происходит проверка на равенство параметра и строки «getInfo». Если проверка успешна, то происходит обработка ответа, находящаяся в объекте конфигурации общий модуль «ПиКС\_Сканнер» (рис. 15).

Функция apiScannerGET(Запрос)

Сервис = Запрос.ПараметрыURL["service"];

Если Сервис = "getInfo" Тогда

Возврат ПиКС\_Сканнер.ОбработкаОтветаGetInfo(Запрос);

Иначе

Ответ = Новый HTTPСервисОтвет(200);

Возврат Ответ;

КонецЕсли;

КонецФункции

**Рис. 14**. Листинг функции apiScannerGET.

В функции *ОбработкаОтветаGetInfo* (рис. 15) происходит обработка ответа на запрос метода *getInfo*. В данной функции происходит запрос к базе данных с параметром, который является штрих-кодом. После чего формируется выборка, являющаяся результатом запроса. Далее происходит проверка на количество элементов в полученной выборке. Если количество элементов больше нуля, то формируется СтруктураОтвета, содержащая в себе данные о отсканированном товаре и являющаяся объектом типа Структура. Иначе в данную структуру записывается сообщение о том, что данный штрих-код не содержится в базе. После чего сформированная структура записывается в формате JSON и устанавливается в качестве ответа, который возвращает функция *ОбработкаОтветаGetInfo.*

Функция ОбработкаОтветаGetInfo(Запрос) Экспорт

СтруктураОтвета = Новый Структура;

ЗапросРегистр = Новый Запрос;

ЗапросРегистр.Текст =

"ВЫБРАТЬ

| ШтрихкодыНоменклатуры.Штрихкод,

| ШтрихкодыНоменклатуры.Номенклатура,

| ШтрихкодыНоменклатуры.Характеристика,

| ШтрихкодыНоменклатуры.Партия,

| ШтрихкодыНоменклатуры.ЕдиницаИзмерения

|ИЗ

| РегистрСведений.ШтрихкодыНоменклатуры КАК ШтрихкодыНоменклатуры

|ГДЕ

| ШтрихкодыНоменклатуры.Штрихкод = &Штрихкод";

ЗапросРегистр.УстановитьПараметр("Штрихкод",Запрос.ПараметрыЗапроса.Получить("id"));

Выборка = ЗапросРегистр.Выполнить().Выбрать();

ОтветJson = Новый ЗаписьJSON;

ОтветJson.УстановитьСтроку();

Если Выборка.Количество() > 0 Тогда

Выборка.Следующий();

СтруктураОтвета.Вставить("id", ""+Выборка.Штрихкод);

СтруктураОтвета.Вставить("name", ""+Выборка.Номенклатура);

СтруктураОтвета.Вставить("characteristic", ""+Выборка.Характеристика);

СтруктураОтвета.Вставить("party", ""+Выборка.Партия);

СтруктураОтвета.Вставить("error", "false");

СтруктураОтвета.Вставить("msg", "ok");

ЗаписатьJSON(ОтветJson, СтруктураОтвета);

Иначе

СтруктураОтвета.Вставить("error", "true");

СтруктураОтвета.Вставить("msg", "Нет товара с данным штрих-кодом: " + Запрос.ПараметрыЗапроса.Получить("id"));

ЗаписатьJSON(ОтветJson, СтруктураОтвета); КонецЕсли;

Ответ = Новый HTTPСервисОтвет(200);

Ответ.УстановитьТелоИзСтроки(ОтветJson.Закрыть());

Возврат Ответ;

КонецФункции

**Рис. 15**. Листинг функции ОбработкаОтветаGetInfo.

**Заключение**

Данная работа посвящена разработке мобильного приложения для сканирования штрих-кодов на платформе Android.

При выполнении работы были решены следующие задачи:

* Выполнен анализ существующих решений
* Выполнен обзор используемых инструментов
* Выполнена реализация HTTP-сервиса на стороне 1C:Предприятие
* Выполнена реализация приложения
* Проведено тестирование приложения

**Литература**