ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК

КАФЕДРА МАТЕМАТИЧЕСКОГО И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

СИСТЕМА УЧЕТА СКЛАДСКОЙ ПРОДУКЦИИ НА ПРЕДПРИЯТИИ ООО «ЛИДЕР»

Выпускная квалификационная работы
студента очной формы обучения
направления подготовки 02.03.03
Математическое обеспечение и администрирование информационных систем
4 курса группы 07001302
Масычева Александра Сергеевича

Научный руководитель
к.т.н доцент
Муромцев В.В

БЕЛГОРОД 2017

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

В настоящее время мобильные устройства заняли достаточно прочное место в повседневной жизни людей. Начали они свой путь в качестве обычных телефонов, впоследствии приобретая все больше и больше функций, превратившись, в конце концов, а по настоящему, полезных помощников. Более того, с развитием интернета мобильные устройства стали средством получения доступа к разным мировым информационным ресурсам, из любой точки мира. Если учесть, что информация в современном обществе является одним из наиболее востребованных ресурсов, а также обратить внимание на развитие технологий, сказывающееся на цене, качестве и технических характеристиках мобильных устройств, то совершенно естественным следствием является очень высокий спрос на мобильные устройства в современном мире.

В наше время все больше и больше смартфонов, коммуникаторов, планшетных персональных компьютеров и других видов устройств, достаточно удобных для использования, как в обычной жизни, так и в заграничных поездках, выпускаются на базе операционной системы Android. Каковы же причины распространения данной операционной системы?

Android поддерживает достаточное количество устройств совершенно разных производителей. Также, Android характеризуется высокой доступностью средств разработки. Средства разработки для данной платформы бесплатны, в то время как разработка, скажем под iPhone (от компании Apple) требует немалых начальных финансовых вложений. Помимо всего вышеперечисленного, большим преимуществом операционной системы Android является наличие бесплатных библиотек для работы сторонними ресурсами(к

примеру YandexMapKit, GoogleMap API, и другими), в то время как например в WindowsPhoneMobile такие возможности не распространены.

В рамках данной дипломной работы будет описано создание складской клиентской системы на базе операционной системы Android, ориентированной в частности, на организацию ООО “ЛИДЕР”. Указанные выше преимущества оправдывают создание приложения на данной платформе и обслуживание ее в дальнейшем времени. Ориентированность на складской учет и средства для складского учета объясняется тем, что организации тратят огромные на приобретение средств для работы сотрудников склада в виде терминала сбора данных (сокр.ТСД), а также разработку программного обеспечения под них, что довольно затратно. Поэтому мной было решено разработать приложение, которое во-первых будет работать намного быстрее чем уже готовое решение на “ТСД” и самое главное дешевле как минимум в 15 раз.

. Давайте попробуем сформулировать основные аспекты складского учета с точки зрения использования информационных технологий

Двадцать первый век, по праву считается веком информационных технологий. Согласимся, что в настоящее время во многих сферах жизни без них не обойтись. Сфера работы склада не является исключением. Давайте на время перенесемся в прошлые времена, где вместо удобных решений связанных с информационными технологиями, использовались старые добрые журналы, куча бумаг, для заполнения которых тратилось огромное количество времени и главное сил. В наше же время все решается работающим сервером, базой данных и клиентскими приложениями для сотрудников склада. Задавшись вопросом, а какие основные потребности у сотрудников склада? Конечно же скорость и качество работы. Ведь крупные точки продаж продукции не могут ждать долгое время, как и клиенты, например купившие кровать, не будут же они ждать 2-3 дня, пока сотрудники ищут нужную кровать на складе. Для этого и были придуманы различные системы упрощения работы сотрудников склада.

Целью данной выпускной квалификационной работы является разработка приложения, функционирующее на платформе Android, которое будет представлять из себя клиентское приложение для системы учета складской продукции ООО “ЛИДЕР”.

Задачами данной выпускной квалификационной работы являются:

1)Выполнить анализ деятельности предприятия ООО “ЛИДЕР”, в том числе управление складским учетом.

2)Разработать проект автоматизации складского учета.

3)Разработать модуль интеграции 1С Предприятия с приложением Android.. Провести тестирование разработанного модуля на реальных данных.

4) Разработать приложение на платформе Android для складского учета на предприятии

5) Внедрить разработанное приложение в организации

ГЛАВА 1.РЕИНЖИНИРИНГ СИСТЕМЫ АВТОМАТИЗАЦИИ СКЛАДСКОГО УЧЕТА ОРГАНИЗАЦИИ ООО ЛИДЕР

1.1 Современное состояние информационного обеспечения в Организации

Далеко не секрет, что приложений для учета складской продукции множество, поэтому было решено проанализировать данные приложения и разбить их на несколько категорий, так как приложений с одинаковой функциональностью очень много.

В первую категорию было решено отнести все приложения, для которых необходимо наличие root-прав. К сожалению, наличие данных прав, очень часто неблагоприятно сказывается на устройстве, особенно в руках неопытных пользователей. К тому же, наличие таких прав лишает пользователя устройства гарантийного обслуживания.

Более подробно углубляться во все нюансы данной категории мы не будем, так как изначально было принято решение не использовать root-права при разработке приложения.

Вторая категория приложений - это приложения, которые предназначены только для работы со своей личной базой данных а не удаленной базой данных 1С. Как правило данные приложения требуют много усилий, чтобы привести их в рабочее состояния, так как база данных организации очень весомый объект, в случае ООО “ЛИДЕР” место которое занимает база данных на диске равняется 150 гигабайтам памяти, что вряд ли получится реализовать в интернет решениях, то есть когда база находится на удаленном ресурсе не расположеном в пределах организации или не находящаяся во внутренней сети организации.

Кроме того заметим, что есть масса приложений, которые позиционируют себя как системы учета складской продукции, но таковыми не являются и просто являются мошенническим инструментом.

В ходе анализа мобильных приложений по данной тематике, не было найдено ни одного приложения, которое будет соответствовать скорости работы решения описанного в данной дипломной работе, а это значит, что путь который был проделан для реализации данного задания был не напрасен.

Прежде чем приступить к разработке приложения для ООО “ЛИДЕР” я изучил то, что используется в данный момент, и в довесок еще очень немаловажный фактор, я изучил схему соединения и работы с базой данных терминала сбора данных. Принцип оказался довольно простым, и это в принципе устраивает организацию. Давайте разберем ситуацию как происходит работа сотрудника склада с точки зрения информационных технологий по пунктам:

1. Сотрудник склада берет Терминал Сбора Данных (ТСД) показан на рисунке 1.1. На котором, установлена ОС Windows SE, и “логинится” на терминальный сервер с помощью RDP соединения.

Рис. 1.1.ТСДMotorolaMC3190

2. Далее, когда сотрудник попадает на удаленный рабочий стол терминального сервера, и ему необходимо приступить к работе запустив программу 1С Предприятие.

3. После запуска и авторизации в программе 1С, сотрудник приступает к работе.

В 1С написана обработка для всего предприятия, начиная с бухгалтерии и заканчивая сотрудниками склада. Соответственно, можно сказать, что организация зависит от данного программного обеспечения, но не в случае склада. Да, кладовщики работают на ТСД, громозких устройствах, у которых единственный плюс, это сканер штрих-кодов. Но почему бы не заменить их на обычные смартфоны на Android.

Возвращаясь к теме 1С Предприятия, замечу, что абсолютно вся работа кладовщиков построена на нем, для каждого действия есть своя обработка, для приема, сбора, перемещения, подпитки(для водителей погрузчика). И придумать что-то новое, то есть заменить, мало у кого получится. Тем самым, глядя на современное состояние информационного обеспечения организации ООО “ЛИДЕР” можно смело сказать, что 1С заняло монопольную позицию в этих вопросах, говорить о удобстве и сравнивать с чем-то, практически не представляется возможным. Тем самым анализируя данный вопрос, я пришел к выводу, что необходимо что-то менять, например заменить те же Motorola MC3190 на обычный смартфон средней цены пять тысяч рублей, к примеру. Но появились еще масса вопросов по поводу работы данного приложения, ведь вся база, а ее размер 150 гигабайт, хранится на сервере. И причем база, тесно связана с 1С.

Но давайте изучим аппаратную часть ООО “ЛИДЕР”. Из аппаратуры, а именно серверного оборудования, имеется 4 сервера. Давайте разберем подробнее, как они связаны и для чего.

1. Сервер DellPowerEdge R210, показан на рис 1.2, на данном сервере, расположенаweb-версия обработок 1С, необходимая для работы и функционирования моего приложения.

Рис.1.2. Сервер DellPowerEdge R210

2. Сервер DellPowerEdge R470, на нем расположена система гипервизора ESXi. Это система виртуализации. На нем находится около 7 виртуальных машин, предназначенных под разные нужды предприятия, включая сервера для связи торговых представителей и системы ЕГАИС, Данон и масса других средств.

3. Сервер DellPowerEdge R720xd, это,пожалуй главный сервер предприятия. На нем установлена операционная система WindowsServer R2 Enterprise. Он поддерживает терминальную систему работы на нем, для сотрудников предприятия. На нем так же работают и кладовщики, и менеджеры склада, и бухгалтерия и в принципе все сотрудники. На нем находится 1С Предприятие 8.2, в котором ведется основная работа кладовщиков, бухгалтерии, транспортного отдела, ну и само собой рабочее место системного администратора. Данный сервер показан на рис 1.3.

Рис. 1.3. Сервер DellPowerEdge R720xd

4. Четвертый, пожалуй один из главных серверов, на котором находится база данных предприятия - DellPowerEdge R710, показан на рисунке 1.4. На его жестком диске имеется более 170 гигабайт данных, а именно базы данных. Соответственно 1С обращается к данному серверу, для того чтобы работать с базой данных. База данных предприятия хорошо защищена от взлома, и просто попасть на сервер с правами администратора не достаточно. Необходимо само собой знать путь, где она находится, а так же знать код доступа к данной директории. В виде ключа флеш-карты. После чего дается доступ с правами редактирования, удаления, изменения.

Рис.1.4. Сервер с базой данных DellPowerEdge R710

Проанализировав серверное оборудование, с которым будет тесно функционировать мое приложение, я пришел к выводу, что это более чем удовлетворяет абсолютно любые системные требования разрабатываемого проекта. Тем самым современное информационное обеспечение данной

организации является вполне себе передовым и отвечающим последним современным решениям в сфере данного бизнеса.

Если затрагивать позиции достигнутые данной организации на рынке услуг, то это одна из крупнейших организаций в области, охватывающая такую обширную территорию продаж продукции. Поставки ведутся в более чем 1800 точек реализаций продуктов. Тем самым это еще раз подчеркивает и показывает, что эффективность работы предприятия напрямую зависит от его информационного обеспечения.

Но пожалуй самое главное, что совокупность всех средств находящихся в данной организации, говорит о том, что при таких вычислительных мощностях и объеме к примеру той же базы данных, работа приложения будет очень быстрой, что является самой главной вещью в организациях подобного типа.

1.2 Современные технологии применяемые при складском учете в организациях различного профиля

Проанализировав информационное состояние организации ООО “ЛИДЕР”, само собой нужно было изучить другие реализации, других организаций. Больше всего конечно, интересует работа склада в организациях различного профиля. Не обязательно продуктов питания, алкогольной продукции и так далее.

По долгу работы системным администратором, поддерживаю связь связь с коллегами в различных организациях, тем самым интересуясь как это реализовано у них. На территории где расположено здание организации ООО “ЛИДЕР”, находятся еще три склада компаний “ПивСтар” - занимается поставками алкогольной продукции, “Сфера” - реализация строительных материалов, “Хадо-Сфера” - продажа, поставка автомобильный масел.

Анализируя и сравнивая работу сотрудников склада данных организаций и ООО “ЛИДЕР”, пришел к выводу и удостоверился, что реализация складского учета “ЛИДЕР”, намного продуктивнее, более качественно реализованнее и стабильнее, нежели в организациях граничащих с ней. К примеру в “Пивстар”, схема работы сотрудников склада сильно похожа со схемой в ООО “ЛИДЕР”, конечно там используется более новое оборудование терминала сбора данных, но от того, что устройство современнее, оно все равно выполняет одну функцию, грубо говоря - сканер штрих-кода.

В наше время штрих-код, это основной инструмент для работы склада, он был изобретен еще в 1951 году и прочно укоренился в современном обществе. Схема работы практически всех складских организаций связана с использованиемштрих-кодов и терминалов сбора данных.

Давайте рассмотрим какие средства используют организации.

1. Сканер штрих-кодов - это достаточно компактное устройство, главной функцией которого является считывание информации с этикеток товара и передача ее в ВМС. Сканер штрих-кода используется при сборе товара, при его поступлении и реализации. Показан на рисунке 1.5.

Рис.1.5. Пример сканера штрих-кодов

2. ТСД. Терминал сбора данных, это довольно специализированное устройство, представляющее из себя портативный компьютер со встроенным сканером штрих-кода. Он предназначен в первую очередь для быстрого сбора, обработки и передачи информации о товаре, а также при поступлении товара и сборке товара и последующей отгрузки клиенту. К сожалению стоимость данного аппарата варьируется от 20 тысяч рублей до 250 тысяч рублей, что согласитесь огромные суммы. Но самое забавное, что обычно эффективность не зависит от цены. Показан на рисунке 1.5.

3. Принтер этикеток - устройство, с помощью которого изображение штрих-кода, наносится на этикетку. Непосредственно в УС формируется изображение, которое наносится на этикетку с помощью принтера. Обычно данные этикетки печатают для товара, который не имеет штрих-кода, или для водителей. Разновидность данных принтеров огромная, как правило это термальные принтеры, не использующие краску. Показан на рисунке 1.6.

Рис.1.6. Пример принтера этикеток

Что касается автоматизации складских процессов в организациях, то условно работу склада можно разбить на три процесса:

1. Поступление товара - оно включает в себя следующие операции: оприходование излишков, поступление товаров

2. Хранение и учет товара - подразумевает проведение инвентаризаций товаров, перемещение между складами и помещениям.

Выдача товара - включает различные операции расходования товаров: списание на внутренние нужды, списание порчи товаров, отгрузка клиенту.

3. Давайте рассмотрим схему поступления товара на склад от поставщика на рисунке 7.

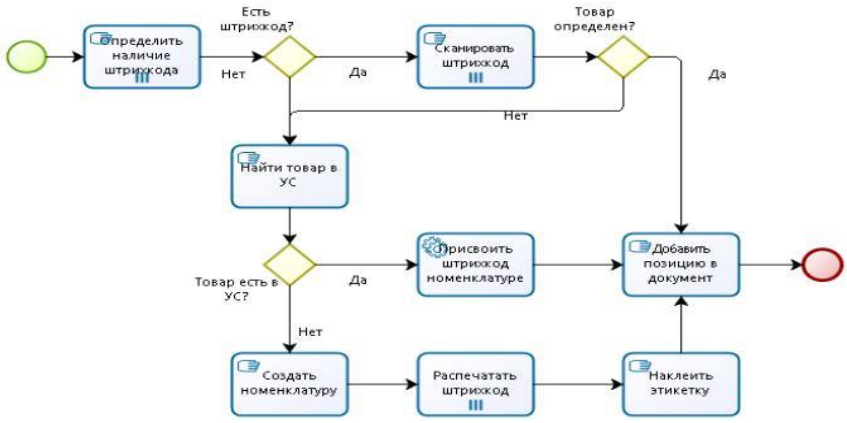


Рис.1.7. Схема поступления товара на склад

Как происходит операция приема товаров от поставщика?

1. Создается заказ поставщику, в котором указывается необходимый товар

2. Поставщик привозит товар

3. Считывается один за другим штрих-коды товаров сканером, и вводятся их в базу данных в документ, к примеру - Поступление товаров

Самое интересное, что принцип данной реализации используется в любой нормальной организации. И менять его нет смысла, так как это достаточно современное и автоматизированное решение. Исключения конечно же присутствуют, в различных организациях по сегодняшний день используются журналы и учет без каких либо автоматизированных систем.

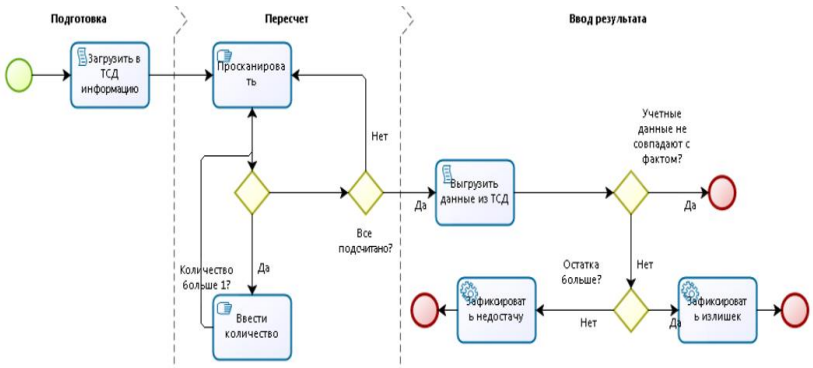


Рис. 1.8. Хранение и учет товара на складе

Теперь давайте перейдем к хранению и учету товара. Хранение и учет предполагает проведение инвентаризаций и перемещение товаров между складами и помещениями. Более подробно можно наблюдать на схеме рис1.8.

Разберем подробнее инвентаризацию в организациях различного профиля склада. Что такое инвентаризация - это проверка наличия товаров на определенную дату путем сравнения фактических данных с данными СУ. Как правило инвентаризацию проводят сотрудники склада, а именно менеджеры, те кто отвечает за работу кладовщиков. Проводится она как правило с помощью терминала сбора данных. Запускается определенная обработка в 1С или отдельно написанное приложение для Windows среды и проверяется досконально по пунктам. Ну и чтобы в системе постоянно была актуальная информация, об остатках, необходимо сделать следующее:

1. Списать недостачи;
2. Оприходовать излишки;
3. Проверить наличие нужного товара на складе;
4. Следить за количеством продуктов.

Разберем выдачу товара. Обычно схема реализации товаров выглядит так:

1. Менеджер по продажам оформляет заказ клиента
2. После оплаты менеджер оформляет документ реализации товара
3. Склад собирает и отпускает товар, заказанный клиентом.

Как правило данная система реализована везде, поэтому ничего особенногоздесь услышать нельзя. Более подробно можно увидеть на рисунке 9.

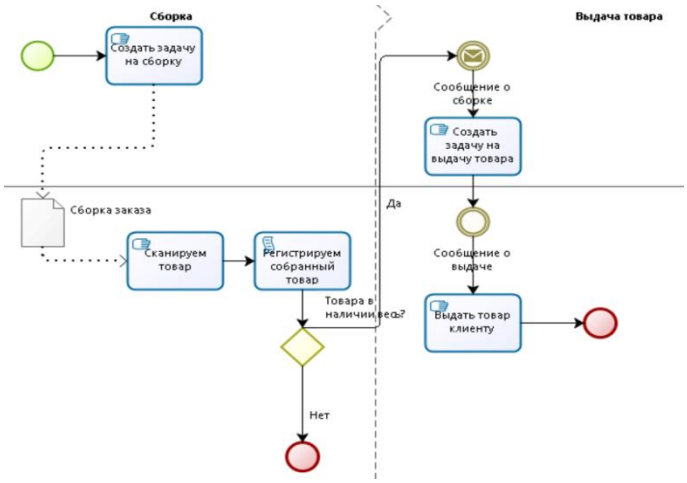


Рис. 1.9 .Выдача товаров

Тем самым, как-то суммируя все разновидности ведения склада, придем к выводу, что схема сбора, выдачи, перемещений, инвентаризаций практически одинакова во всех организациях которые применяются при складском учете различного профиля. Если говорить о современном оборудовании, то здесь тоже достаточно небольшие различия. Конечно мы не сравниваем все со складами Amazon, где этим всем занимаются роботы, а рассматриваем организации приближенные к ООО “ЛИДЕР”. Не важно насколько современно оборудование например ТСД, так как функции они выполняют одни и те же - сканер штрих-кода.

1.3 Реинжиниринг информационного обеспечения деятельности ООО ЛИДЕР, цели и задачи работы

Приступая к решению об оптимизации информационного обеспечения деятельности ООО Лидер я задался вопросом актуальности. К чему нарушать работу уже готового решения работы сотрудников склада? Я долго думал, анализировал, взвешивал все за и против и пришел к выводу, что попробовать стоит. Так я приступил к разработке своего мобильного приложения для складского учета для организации “ЛИДЕР”. Я прекрасно понимал, что решение с Терминалами Сбора Данных, слишком затратно в будущем. Что я под этим подразумеваю? Это ремонт, обслуживание и также цена на них. Чтобы избежать этого, я начал продумывать пути оптимизации данных решений. И нашел выход в Android приложении.

Ставя перед собой цели я старался советоваться в пожеланиях руководителя организации, дабы в конечном итоге наши представления не разошлись. Давайте постараемся рассмотреть каждую задачу подробно.

1.Информативность – это одно из главных требований от руководителя от организации. Проблема ТСД, в том что у них слишком маленький дисплей, само собой сотрудникам во-первых сложно увидеть некоторый текст, во-вторых для отображения большого количества текста необходимо пролистывать экран, что в условиях качества работы сенсора терминала дико неудобно, и само собой слишком затратно по времени. Тем самым необходимо провести анализ уже готового решения и максимально точно и информативно передать это на дисплее смартфона.

2. Эргономичный дизайн - также немаловажным следует сделать довольно простой и эргономичный дизайн приложения. Почему простой? Потому что в работе склада главное скорость, а красивые прорисовки приложения это второстепенное, или даже не значимое новшество. Поэтому была поставлена цель, сделать качественно(в плане удобного расположения

элементов управления на экране), и просто. Требования вполне себе обоснованные, так как экранов в приложении будет много, соответственно объем занимаемый во внутренней памяти смартфона будет большой, и лишние картинки и украшательства приведут все к не очень хорошим результатам.

3. Система идентификации сотрудников по штрих-коду или QR-коду - эта система уже реализована на терминале сбора данных в данной организации и в принципе она достаточно удобна, сотруднику необходимо просто отсканировать штрих-код, после чего система автоматически проверяет логин и пароль пользователя и тем самым его идентифицирует в системе. Реализовано это в 1С обработке, но требования организации было в том, чтобы данная система была в оффлайн версии приложения, то есть без соединения с сервером и доступом, даже если нет соединения с сервером во внутренней сети организации. К чему это реализовано на данном уровне мне не известно, но было предложена и согласована система логина путем соединения с удаленной базой данных, так как сотрудники меняются и каждый раз обновлять приложение как минимум глупо.

4. Работа приложения с возможностью доступа к web-интерфейсу обработки 1С. Так как рабочая система уже реализована на терминале сбора данных, было решено сделать вначале клиентское приложение на основе оконного режима обработки 1С на веб-сервисе предприятия. Приложение устанавливает браузерное соединение с нужным IP адресом, на котором установлена обработка в веб-режиме и тем самым работает. Тем самым обеспечивая наивысшую скорость работы приложения на платформе Android. Так как оно не выполняет никаких объемных вычислений. Что так же упрощает покупку устройств и их сопровождение.

5. Сетевая защищенность приложения - это, пожалуй, самая интересная цель в данной работе. Разберем все подробно. Так как внутренние IP адреса предприятия работают во внутренней сети, то есть доступ к ним возможен если человек подключен к сети по WiFi или Ethernet соединению внутри организации, или исключительно через VPN(virtualprivatenetwork), и то для того, чтобы удаленно подключится во внутреннюю сеть предприятия нужны сертификаты пользователя, а для этого системные администраторы должны выписать сертификат на шлюзе, предназначенном для подключения по VPN. Соответственно имея на руках просто приложение, без доступа во внутреннюю сеть предприятия, будет невозможным работа, а точнее полный функционал приложения. Также я не исключаю удаленную работу, но при условии того, что на Android устройстве будет настроено VPN соединение в сеть предприятия, что вряд ли так как необходимости в этом нет. Но в таком случае ставим новый вопрос, а что если человек подключился к сети предприятия и запустил приложение и даже залогинился, сможет ли он начать работу? Ответ: нет. Так как после анализа сетевой безопасности было принято решение, присвоить всем в будущем рабочим Android устройствам статический IP адрес и привязку по MAC адресу, чтобы никакие другие устройства, будь то персональные компьютеры, другие смартфоны не могли получить доступ к главному, веб обработке 1С. Доступ к ней будет разграничен исключительно по IP и MAC адресам, что уже говорит о стойкой и безопасной системе.

6. Удобный и практичный интерфейс приложения. Не секрет, что долгая работа сотрудников в одной системе, которая не меняется годами, заставляет полностью привыкнуть и не изучать нововведения. В принципе было решено практически скопировать интерфейс уже готового решения, которое используется на терминале сбора данных на разрабатываемое Android приложение. Тем самым это облегчает процесс обучения сотрудников работы в данном приложении. То есть нет необходимости объяснять какие-то новшества

введенные в системе, пользователю будет интуитивно понятно как и что работает. Тем самым откинув практически полностью трату времени на то, чтобы сотрудник освоил данную систему и приступил к работе. Считаю данную цель правильной и практически применимой.

В итоге поставлена цель создать то приложение, которое будет соответствовать задачам организации, ее пожеланиям и будет способствовать решению проблем выше упомянутой. Тем самым повысив показатели, скорость работы сотрудников.и пожалуй самое главное это материальную зависимость от уже готового решения.

Тем самым я приступил к подробному разбору работы уже готового решения организации - терминалу сбора данных. Так как нужно понять, получится ли реализовать подобное в условиях платформы Android. Разбирая и анализируя работу сотрудников организации, было правильным, спросить пожелания кладовщиков и менеджеров склада, чтобы они хотели видеть в приложении. Так как предназначается оно исключительно для них, а не для административно-управленческого персонала. Пожелания были практически схожими с задачами руководства, но только в сегменте интерфейса и оформления, но все же. Сотрудники попросили оформить приложение в темных тонах, так как солнечных лучей на склад проникает немного, из-за того, что там хранятся продукты. На складе имеется довольно хорошее искусственное освещение, от которого нет бликов на дисплее смартфона, в итоге нет необходимости делать светлые тона приложения, а именно белые-черные цвета, а сделать то, что будет удобно для глаз. Путем быстрого голосования было выбрано серо-белое оформление приложения. На мой взгляд, это вполне обоснованный выбор для оформления приложения. Вопрос о двух вариантах оформления приложения не поднимался, так как освещение на складе работает всегда, тем самым дневную-ночную версию приложения делать нет необходимости.

Соответственно были так же поставлены задачи от организации:

1) анализ уже существующих решений;

2) интеграция приложения с готовой базой данных организации;

3) анализ возможных улучшений работы базы данных с клиентским приложением;

4) анализ механизмов, необходимых для осуществления цели;

5) разработка приложения

6) внедрение и тестирование непосредственно на базе организации ООО “ЛИДЕР” для полной замены уже готового решения.

ГЛАВА 2: ПРОЕКТИРОВАНИЕ СИСТЕМЫ АВТОМАТИЗАЦИИ

2.1. Анализ требований и инфологическое моделирование системы автоматизации

Приступая к анализу требований от организации, я приступил к подробному разбору работы уже имеющегося решения с базой данных предприятия 1С. Процесс работы управляется обработками 1С. Будь то «приемка», инвентаризация, перемещения. Соответственно необходимо досконально изучить данный вопрос, прежде чем приступать к разработке системы для автоматизации на платформе Android. Первым делом я стал разрабатывать базу данных для системы автоматизации, которая разрабатывается в данной выпускной квалификационной работе. Но прежде чем приступать я обратился к требованиям, поставленным организацией. Одно из них было, чтобы сотрудники проходили идентификацию с подключением к удаленной базе данных. Именно этот вопрос я и начал прорабатывать. Первым делом было решено собрать данные по всем сотрудникам, относящимся к складу, это кладовщики, менеджеры склада и водители погрузчиков. Получив, данные результаты я приступил к разработке непосредственно самой базы данных.

Предварительно рассортировав каждого сотрудника по должностям, начал заполнять базу данных. Всего получилось около трех таблиц. Приступая к их составлению, я учел пожелания организации о том, чтобы доступ к ним можно было иметь через сеть интернет, для оперативного редактирования, в случае увольнения или приема сотрудников. Тем самым упростив мероприятия, проводимые для реализации какого либо действия.

И так первая таблица в базе данных это менеджеры склада, всего их 4 человека.Результаты сведены в таблицу 2.1

Таблица 2.1

Менеджеры склада

ID сотруд.	ФИО	Логин	Пароль
1	Мишина Т.В	M0622	M1d31
2	Жирков И.Д	J0633	M2d43
3	Потапов А.Н	P0422	M25d6
4	Жилюкова О.Ю	J0634	M34d7

За основу логина было принято взять первую букву фамилия сотрудника и последующий «рандомный» набор цифр. Что касаемо пароля, то здесь первая заглавная буква «М» подразумевая слово «менеджер», и последующий неопределенный набор символов с разным регистром. Так же каждому сотруднику присваивается свой «ID» в системе. Чтобы клиентскому приложению было проще найти нужного сотрудника и передать данные непосредственно обработке 1С. Далее я продолжил формирование базы данных системы авторизации и приступил к заполнению таблицы уже кладовщиков. Их в организации насчитывается 15 человек, 8 из которых работают в дневную смену, а в ночную непосредственно в основном складе и «холодильнике» 7 человек. Из ночной смены 3 человека работают в холодильнике, а остальные 4 в основном складе. Было решено поделить их на две разные группы, тем самым добавив еще одну таблицу к нашей базе данных, данные результаты показаны в таблице 2.2.

Таблица 2.2

Кладовщики – «холодильник»

ID	ФИО	Логин	Пароль
1	Киселев Ю.Г	hK001	K1535k
2	Гребников А.Ю	hG002	K1637g
3	Калешина О.В	hK003	K1742k

Заполнив таблицу 2.2, я приступил к заполнению следующей таблицы для ночной смены, только уже сотрудников основного склада.

Таблица 2.3

Ночная смена – основной склад

ID	ФИО	Логин	Пароль
1	Маслиев А.Ю	oM001	K1843m
2	Мерзлякина А.С	oM002	K1945m
3	Долгатов Е.В	oD001	K1967d
4	Иванов Г.П	oI001	K3675i

После заполнения данных таблиц, я приступил к заполнению следующих.

Далее встал вопрос о схеме логина, как же сотрудники будут идентифицироваться в Androidприложении? Для этого я приступил к разработке «UML» - диаграммы, для точного схематического описания работы с базой данных. Давайте попробуем подробнее разобраться, как будет,происходит

логин сотрудника в системе. Первоначально планируется экран логина с возможностью идентификации по штрих-коду, то есть достаточно просто отсканировать его, и приложение обратится к базе данных для поиска нужного сотрудника по его ID, сверив его логин и пароль. Соответственно нам необходимо построить две диаграммы, для логина в текстовом варианте. То есть когда сотрудник вводит данные с клавиатуры устройства и без нее, путем сканирования штрих-кода.

И так давайте приступим к построению схемы логина сотрудника путем ввода данных с клавиатуры устройства. Первоначально, в меню приложения будет стартовый экран с выбором должности сотрудника склада, далее непосредственно логин выше упомянутого. И так на экране авторизации находится два поля для ввода логина и пароля, в первый соответственно вводится логин для идентификации, во второй пароль. И далее происходит запрос в базу данных, для проверки и последующего допуска в систему учета. Соответственно если проходит авторизация, то сотрудника перебрасывает на следующий экран общей панели управления. Где он дальше начинает работу в системе и приступает к выполнению заявок по ней. Построив более подробную диаграмму, картина становится более отчетливой. Результат вы можете наблюдать на рисунке 2.1.

Продолжив работу с базами данных. Я приступил к схеме работы авторизации сотрудника с помощью штрих-кода. И так давайте подробнее ее рассмотрим. Схема логина практически одинаковая с выше упомянутым. Первым делом сотрудник выбирает свою должность, далее он нажимает на кнопку штрих кода, которая показана на рисунке 2.2. После чего на экране смартфона появляется экран с работающей камерой, которой нужно указать на штрих-код, в котором имеется информация в виде кода сотрудника

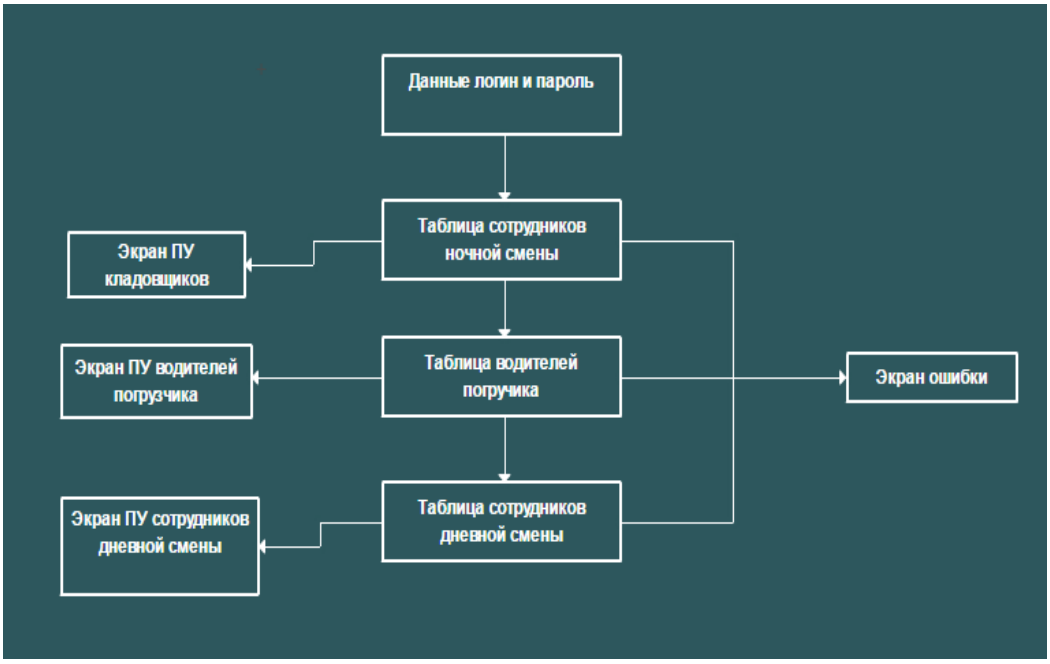


Рис. 2.1. Схема работы с базой данных

После сканирования штрих-кода, система проводит поиск по базе данных, после чего в случае совпадения, сотруднику дается доступ в главную панель управления, для продолжения работы на складе. В случае если данные с отсканированного штрих-кода не прошли проверку по базе данных, пользователя перенаправляет на экран с ошибкой, где ему пишется о том, что код идентификатора не подошел.

Рис. 2.2. Кнопка для логина сотрудника по штрих-коду

Тем самым подводя итоги по базам данным, получилось вместо планировавшихся трех – четырех таблиц, получилось пять таблиц:

- Таблица сотрудников дневной смены
- Таблица сотрудников ночной смены холодильника
- Таблица сотрудников ночной смены основного склада
- Таблица водителей погрузчиков
- Таблица для идентификации сотрудника по штрих-коду

Сперва кажется, что количество таблиц слишком велико, но это не так. Это вынужденная мера для удобства управления базой данных в дальнейшем при приеме или увольнении сотрудника. Тем самым составив более подробную схему работы с базой данных. Можно утверждать, что система логина будет работать без нареканий, как того и требует организация.

Реализовано,удаленное управления выше упомянутой с помощью веб-интерфейса. Упорядочены сотрудники по отделам, чего не наблюдается даже в нынешней базе данных организации. Тем самым можно спокойно формировать актуальные на сегодняшний день списки сотрудников, не боясь упустить какой либо пункт.

2.2.Модернизация системы автоматизации

Приступая к разработке системы автоматизации, я провел глубокий анализ уже действующей системы автоматизации в организации, ООО «Лидер». Досконально изучив все тонкости работы складских сотрудников на предприятии, было решено внести некоторые изменения в привычную, сферу их деятельности.

Первым делом необходимо было понять, алгоритм некоторых обработок в программе 1С, которые связанны с работой склада. Это касается приемки, возврата, перемещения и других процессов. Было решено изменить привычные интерфейсы, в которых работали сотрудники на совершенно другие. Давайте

подробнее разберем схему работы сотрудников склада, скажем, при отборе товара и его последующей отправки в точки продаж. Весь товар находится на стеллажах, стеллажи в свою очередь имеют ячейки, а каждая ячейка «пронумерована» штрих-кодом, в который заложен код ячейки и стеллажа. К примеру – ячейка «09.20.02.01» где «09» это номер ряда, «20» - это номер стеллажа, «02» - ряд верхней или нижнее зоны стеллажа, который доступен без применения технических средств, то есть когда кладовщик может достать рукой до нужного товара, а «01» это отсек в котором находится нужный товар. Пример вы можете видеть на рисунке 2.3. Подробный список возможных действий кладовщика вы можете наблюдать на списке ниже:

- Приемка
- Отбор
- Отгрузка

Сотрудник находит нужную ячейку, сканирует штрих-код, далее данные передаются в обработку 1С, где дается ответ, к той ли ячейке обращается сотрудник, если нет, то приложение выдаст ошибку, с советом поискать лучше.

Если сотрудник нашел нужную ячейку, и программа дала удовлетворительный ответ, он смотрит информацию по товару. А именно, его наименование, вес, количество. Соответственно на каждом товаре имеется свой штрих-код, который занесен в базу данных при приеме товара на склад. Кладовщик сканирует товар, данные отправляются снова в 1С, где проходит проверка соответствия заявки отбора, к штрих-коду товара. Если проверка прошла успешно, сотрудник приступает к сбору товара, проверяя его количество и складывая в корзину или на рохлю.

Рис. 2.3. Стеллажи склада

После того как кладовщик все отгрузил, ему необходимо напечатать этикетки груза, когда, в какое время и в каком количестве он был укомплектован. Для этого сотрудник подходит к принтеру этикеток и сканирует штрих-код, который активирует обработку 1С, которая отвечает за автоматическую печать этикеток согласно данным из заявки на отбор. После печати этикеток, кладовщику необходимо наклеить их на упакованный в коробки товар. Далее на дисплее появляется информация, что сотруднику необходимо выбрать стеллаж, на который будет отгружен товар, до его приема водителем и последующей доставки в точку продажи. Собственно по такой схеме проходит процесс отбора товара.

После разбора схемы работы кладовщика, можно перейти к работе менеджера склада. Менеджер склада, отвечает на наличие товара, исправление ошибок позиций товара и так далее. Работают они в программе

WarehouseManagementSystem, где они формируют заявки на отбор, прием, перемещения.

То есть могут управлять складом с монитора компьютера. Но так же у каждого из менеджеров имеется терминал сбора данных, с помощью которого он может выполнять роль кладовщика, водителя погрузчика и так далее. Но в основном он необходим, либо для приема товара, либо для перемещения. Менеджер склада обязан присутствовать при приеме товара на склад обязательно. Контролируя и занося в базу данных координаты будущих позиций прибывшего товара. Менеджер склада обладает всеми привилегиями в программе для управления складом. Давайте рассмотрим список возможностей менеджера склада:

- Прием товара от поставщика
- Инвентаризация
- Отбор
- Перемещения
- Отгрузка
- Размещение
- Подпитка
- Остатки
- Приемка возвратной тары

Если рассматривать работу водителя погрузчика склада, то здесь достаточно несколько примеров. Это перемещение и подпитка. Давайте разберемк примеру перемещение. Представим ситуацию, что после приема

товара от поставщика, его нужно разместить на стеллаже. Для этого и необходим погрузчик, так как высота стеллажей примерно 6-7 метров, то подручными средствами здесь не справится, а необходима помощь специализированной техники. Давайте обрисуем схему работы при перемещении. Каждый прибывший товар на склад, обязательно заносится в базу данных склада и требует размещения. Водитель погрузчика с помощью терминала сбора данных сканирует штрих-код товара и определяет, в какой ряд и какую ячейку его переместить. Так как каждый ряд и сторона, определены под разные товары по категориям. После перемещения товара, сотрудник фиксирует это в терминале сбора данных и передает информацию на сервер, для последующего занесения в базу данных.

Что касаемо «подпитки», то этим тоже занимается водитель погрузчика. В обработке формируется заказ на подпитку определенного товара, к примеру – чая. На нижних рядах стеллажа заканчивается чай, соответственно кладовщикам нечего грузить на отправку. Водитель погрузчика находит нужный товар на верхних рядах стеллажа и спускает часть на нижние ряды, тем самым добавляя количество товара. Более подробный список работ, выполняемых водителем погрузчика вы можете наблюдать в списке ниже:

- Перемещения
- Отгрузка
- Размещение
- Подпитка
- Остатки
- Приемка возвратной тары

Соответственно стоит вопрос о модернизации системы в плане не только аппаратной части, но и в части программного обеспечения, а так же смены платформы на котором будет работать приложение. В качестве платформы была выбрана операционная система Android. Прежде чем приступить к разработке необходимо составить примерный план, того как оно будет выглядеть, более подробно это опишется в следующем подпункте. А пока вернемся к формированию экранов, для работы сотрудников склада. В данный момент панель управления каждого сотрудника выглядит так, как показано на рисунке 2.4.

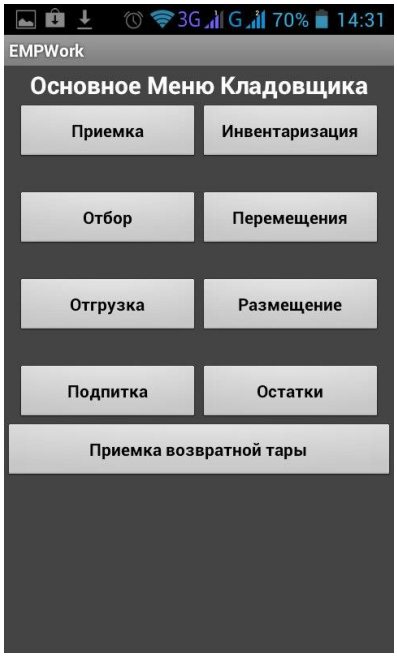


Рис. 2.4. Панель управления работников склада

Было решено полностью портировать данный интерфейс в мое приложение, но стоял самый главный вопрос. Как приложение будет взаимодействовать с базой данных 1С. Было рассмотрено два варианта, первый из них это размещение обработки платформы на веб-интерфейсе и второй, пожалуй, самый сложный вариант это отображение с помощью стандартных средств Android. Предстояло протестировать, скорость работы двух данных решений. Дабы понять, как будет выглядеть, и как будет лучше работать схема

соединения с базой данных. Итоги тестирования будут описаны в главе Тестирование и развертывание.

2.3. Программирование интерфейсов для системы учета складской продукции

Приступая к программированию приложения на платформе Android,я поставил вопрос, какой язык программирования использовать. Более хорошо я знал C++, и в принципе он пригоден для разработки приложений на Android, но устоявшимся и показавшим результаты в разработке на Androidявлялся язык Java. Проведя анализ скорости работы, по готовым приложениям. Было решено использовать язык Javaдля последующей разработки приложения.

Давайте подробнее разберем язык Java. Согласно данным Википедии – Javaэто сильно типизированный объектно-ориентированный язык программирования, разработанный компанией SunMicrosystems(в последующем приобретенной компанией Oracle). Приложения Java обычно транслируется в специальный байт-код, поэтому они могут работать на любой компьютерной архитектуре, с помощью виртуальной Java-машины. Дата официального выпуска – 23 мая 1995 года.Понятность и простота языка Java, усиленная обширной библиотекой классов Android, превращает Android вконкурентоспособную платформу для написания программ. На рисунке 2.5. дано схематическое представление программного стека Android. Принцип выбора платформы Android, объясняется тем, что во первых очень много устройств, достаточно дешевых, достаточный быстрых в плане вычислительной мощности. Что к примеру не видно например на iPhone, работающем на iOS. Тем самым плюсы выбора Androidочевидны. Давайте приступим к непосредственному планированию интерфейсов, которые будут выполнять работу с алгоритмом складского учета.

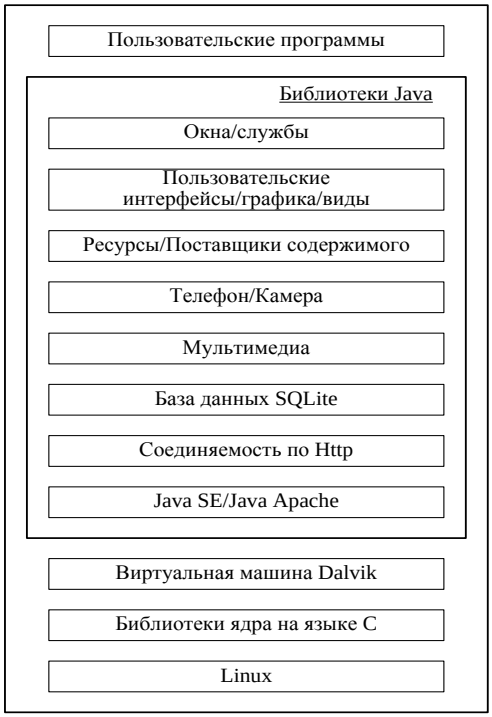


Рис. 2.5. Схематическое представление стека Android.

Так же не мало важную роль играет среда, в которой разрабатывается приложение. Мой выбор пал на AndroidStudio. AndroidStudioэто интегрированная среда разработки для работы с платформой Android, которая была анонсирована 16 мая 2013 года на конференции Google. Она основана на программном обеспечении IntelliJIDEAот компании JetBrains, официальное средство для разработки приложений на Android. Данная среда разработки доступна для Windows, OSX и самое главное Linux. К преимуществам этой программы относится огромный спектр возможностей, для редактирования кода, точного построения приложений, и дальнейшей стабильной работы. В AndroidStudiotак же имеется встроенный эмулятор платформ Android. Более 30 видов разновидностей систем, на которых приложение можно протестировать. Что соответственно облегчает разработку приложения. Так как нет необходимости, иметь в руках нужный смартфон с нужной системой, то само собой экономит время на разработку, на тестирование и оптимизацию.

И так первым делом необходимо продумать, как будет выглядеть приложение на экране смартфона пользователя. Так как в задачах было указано, что приложение должно иметь темные тона, то выбор остановился на сером цвете фона и светло-серым цветом кнопок. Первым делом нужно было распланировать количество экранов на первых шагах. Решено было сделать 11 экранов.

- Главный экран
- Экраны для менеджера склада, кладовщиков и водителя погрузчика
- Экран ошибки
- Три экрана идентификации для каждого сотрудника
- Три главных экрана для основной работы всех сотрудников

Далее я приступил к проектированию интерфейса. Как будут располагаться кнопки, какого размера они будут, и само собой как они будут называться. Ведь самое главное, необходимо первым делом сделать систему понятной для пользователей в первую очередь, которые будут в ней работать, конечно для разработчика это не так важно. Ведь алгоритм действия ему будет понятен. И опять же, с помощью понятного интерфейса, будет легче сопровождать приложение на пути работы, что сказывается на экономии средств по обслуживанию, и экономии времени сотрудников, а значит выским показателям скорости и качества работы. На рисунке 2.6 вы можете наблюдать главный, основной экран приложения.

Рис. 2.6. Стартовый экран приложения

На нем находятся панели управления кладовщика, менеджера склада, и водителя погрузчика. При переходе, то есть нажатии кнопки, пользователя переносит на другой экран – экран авторизации. Для каждого сотрудника экран авторизации выглядит одинаково. Но это визуально, если обратится к коду. То для экономной работы времени обработки данных, приложение на каждом экране обращается к определенной базе данных, которая относится к должности каждого сотрудника. Тем самым экономится время и увеличивается быстродействие системы.

Давайте подробнее рассмотрим экран авторизации для кладовщика. Он показан на рисунке 2.7.

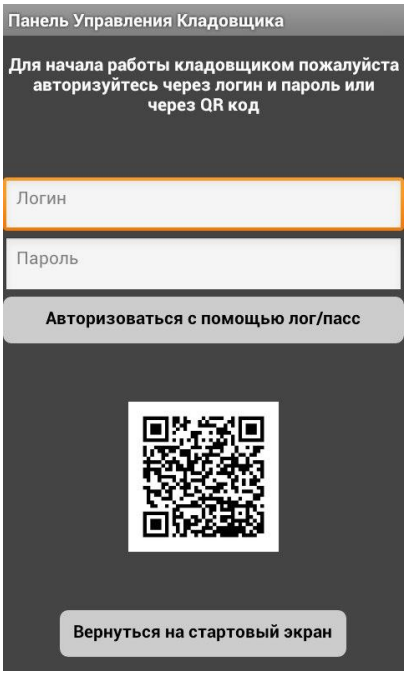


Рис. 2.7. Экран авторизации кладовщика

Как можно наблюдать на рисунке 2.7, на экране присутствуют поля для текстового ввода – Логин, Пароль. А так же основная кнопка длязапуски проверки и идентификации пользователя в системе под названием «Авторизоваться с помощью лог/пасс». Ниже находится кнопка, оформленная в виде QRкода, при нажатии на которую, камера смартфона приводится в активность для распознавания штрих или QRкода.

Система логина с помощью штрих-кода проводится практически по тому же принципу, что и логин и пароль. Только, информация зашифрованная в штрих-коде представляет один код из набора символов, вместо двух различных полей. Так же при неправильном вводе логина или пароля или несовпадения кода из отсканированного штрих-кода. Приложение выдаст ошибку с возможностью повторного ввода данных. Пример экрана ошибки вы можете наблюдать на рисунке 2.8.

Рис. 2.8. Ошибка при идентификации

Сотрудника предоставляется информация об ошибке и дается возможность авторизоваться в системе снова, проверив правильность ввода логина и пароля. Само собой в целях безопасности приложение не уведомляет, где было совпадение и данными, а где была указана ошибка. Что конечно не очень удобно, в том случае если сотрудник забудет что-то из данных для идентификации.

По итогам тестирования данной системы на несанкционированный доступ, ошибок допущено не было. Устойчивость данной системы крайне велика. Что соответственно говорит о том, что система более чем подходит для работы на складском предприятии, как очень подходящая замена уже устоявшемуся решению.

И так после успешной идентификации в приложении, сотруднику дается доступ в основное меню для работы. Его вы можете наблюдать на рисунке 2.9.

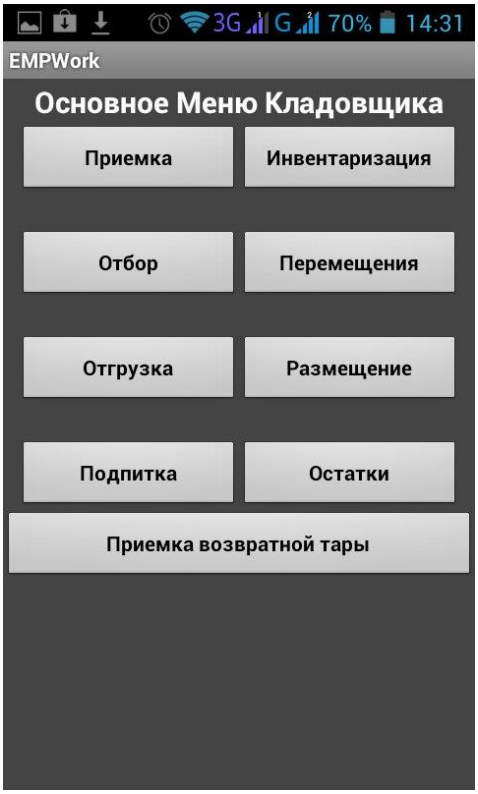


Рис. 2.9. Основное меню

На рисунке 2.9 вы можете наблюдать основное меню для сотрудников. А так же самые главные функции для работы – приемка, инвентаризаций, отбор, перемещения, отгрузка, размещение, подпитка, остатки и приемка возвратной тары. Для каждого сотрудника имеется разный доступ к функциям данного экрана. То есть кладовщику доступно только отбор, приемка и отгрузка. И перейдем к пожалуй самому главному экрану. Это экран, который связан с базой данных 1С и взаимодействие с обработками вышеупомянутой. Наблюдать его вы можете на рисунке 2.10. Где подробно показана работа с заявками и инструкциями для кладовщиков. Интерфейс полностью портирован с уже готового решения используемого на терминале сбора данных. Тем самым сотрудника не придется заново обучаться работе с данными приложения и изменять какие-либо правила, устоявшиеся с момента появления на работе. Тем самым, исключая большинство вопросов по работе приложения и по данным, которое оно показывает.

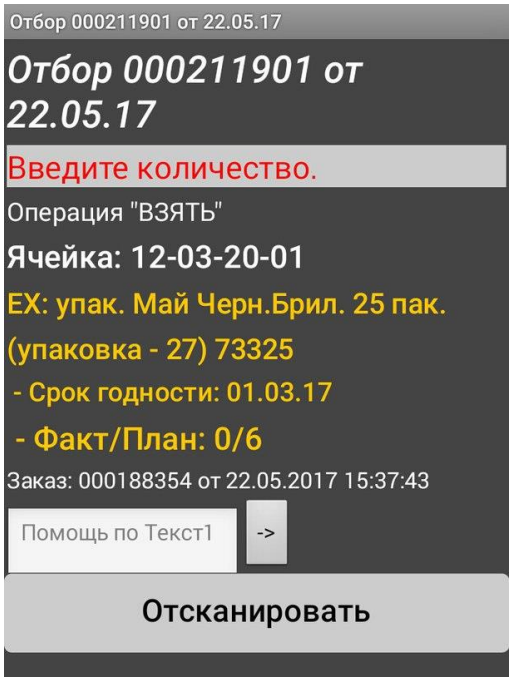


Рис. 2.10 Заявка на отгрузку

По итогам работы я разработал интерфейс для системы автоматизации складского учета для предприятия ООО «Лидер» в Androidприложении именуемом в дальнейшем - LiderStorageManager.

ГЛАВА 3: ТЕСТИРОВАНИЕ И РАЗВЕРТЫВАНИЕ

3.1 Тестирование

Приступая к тестированию приложения для организации, я обратился за помощью к сотрудникам, а именно менеджерам склада и кладовщикам. Огромным плюсом для организации является то, что в ее распоряжении есть более 70 смартфонов на платформе Android. А это значит, что взяв к примеру 2-3 телефона, я могу установить на них свою разработку, чтобы протестировать так сказать на «горячую». Установив приложение на смартфоны, я выдал 1 телефон менеджеру склада, чтобы был мониторинг за работой сотрудников и 2 телефона кладовщикам. Тем самым мы имеем более чем отличный шанс протестировать приложение непосредственно со стороны пользователя, а именно сотрудника склада. Предварительно проведя инструктаж для сотрудников, которым я выдал смартфоны. Я начал наблюдать за работой, попутно отвечая на вопросы связанные с приложением. Тем самым провелось живое тестирования, в результате которого обнаружился ряд проблем, а именно с некорректностью отображения данных о заявке, что связанно с ошибкой работы с базой данных 1С. Пытаясь устранить данную проблему, я столкнулся с тем, что часть обработок попросту не работала в 1С, то есть на терминале сбора данных ошибок не было, а после связи моего приложения с базой данных они появились. После мне пришлось изъять телефоны у сотрудников, для устранения данных ошибок.

И так, углубимся в тестирование. Рассмотрим контроль (verification) – попытка найти ошибку, выполняя программу в тестовой или моделируемой среде, то есть создать к примеру тестовую заявку, и самому

разработчикупопробовать выполнить ее на устройстве, то есть примерить на себя работу кладовщика. Далее использовалось испытание (validation) – попытка найти ошибки, выполняя программу в заданной реальной среде, то есть к примеру сотрудником организации, который знает, что где, и как расположено. Если к примеру у разработчика не получилось найти что-то, то для того, кто работает с этим не первый год, ошибки будут очевидны. И самый главный пункт тестирования, это отладка (debugging), хотя по сути отдладка не является разновидностью тестирования. Тестирование это деятельность направленная на обнаружение ошибок, а отладка направленна на установление точной природы известной ошибки, а затем на исправление. Эти два действия связанны, результаты тестирования являются исходными данными для отладки. Более подробную зависимость вы можете видеть на рисунке 3.1.

Рис. 3.1. Диаграмма зависимости этапов тестирования

Устранив данные ошибки, я решил повторить эксперимент с тестированием, и снова выдал устройства сотрудникам. В этот раз

тестирование прошло отлично и за 6 часов работы, не возникло никаких проблем с приложением, с Wi-Ficoединением и соединением с базой данных.

Соответственно приложение в принципе готово к внедрению в работу организации. Но появились некоторые дополнительные требования к приложению. Ведь количество версий Androidдостаточно большое, соответственно необходимо протестировать на разных версиях операционной системе.

Основное тестирование проводилось на телефоне FLYIQ4502 на операционной системе Android 5.1, в принципе основное устройство, которое будет использоваться в организации для работы торговых представителей и соответственно в будущем сотрудников склада. Дополнительно я провел анализ работы приложения в разных версиях систем а именно:

1. Android Google API 2.3
2. Android Google API 2.4.1
3. Android Google API 3.6
4. Android Google API 4.0 (IceCream)

Доступ к данным системам был получен с помощью эмулятора Androidв программе AndroidStudio. Вид эмулятора вы можете видеть на рисунке 3.19. Благо возможностей у данного эмулятора хватает, можно сказать это полноценный Androidсмартфон. Конечно, были проблемы со связью с базой данных, ведь приложение работает только во внутренней сети организации, без доступа в интернет для него, так как это в принципе не считается нужным. Но проблема решилась установкой VPNсоединения с главным шлюзом организации.

И по итогам тестирования в данном режиме, приложение показало работоспособность 100%.

Соответственно осталось протестировать его работу на экранах с разным разрешением. Я решил взять наиболее популярные разрешения, и протестировать приложение на них

Рис. 3.2.АндроидэмуляторпрограммыAndroidStudio

1) 320x280
2) 400x240
3) 800x420
4) 800x480
5) 1024x600

По итогам тестирования работы на всех разрешениях экрана стабильно работает, так как это было предусмотрено на стадии разработки. Добавлено адаптирование под разные разрешения, а так же элементарный «scroll» экрана, то есть листание.Соответственно приложение отлично функционирует во всех разновидностях разрешениях.

Так же в процессе разработки приложения производилось поэтапное тестирование с целью выявления программных и системных ошибок и несоответствий техническому заданию. Для тестирования отдельных модулей работы с базой данных в текст программы были внесены специальные функции, которые позволяют анализировать базу данных и, при подозрении на ошибку, выводят сообщение в системный лог файл.

1. Каждая активность была подвергнута юнит-тестированию с целью выявления ошибок, вызванных несоответствием ожидаемых и полученных параметров.

2. В базу данных намеренно вносились недопустимые значения и данные в соответствующие поля, которые могли быть неверно интерпретированы приложением и базой данных 1С.

3. После завершения цикла разработки, программный продукт тестировался на реальных устройствах.

Далее я приступил к тестированию базы данных 1С расположенной на сервере предприятия

Подводя итоги тестирования на реальных устройствах, и непосредственно на складе организациями, действующими сотрудниками, приложение прошло проверку на ошибки, найденные ошибки были исправлены и в данный момент приложение функционирует без каких либо сбоев.

Что естественно полностью устраивает организацию, так как согласитесь, что кому нужен частично рабочий продукт, который невозможно использовать в современных реалиях.

3.2 Развертывание системы

Развертывание системы – в нашем случае это совокупность нескольких моментов, а именно внедрение аппаратной части и развертывание программного обеспечения на базе организации, ООО «ЛИДЕР».

Первым делом необходимо составить план внедрения:

1. Развертывание программной части приложение, установка статических путей для связи с базой данных и сервером.

2. Подбор Android устройства, для корректной работы приложения и подходящим для интенсивной работы, а главное, чтобы устройство было энергоемким

3. Установка приложения и последующая настройка на Android смартфоны

4. Вводный инструктаж сотрудников организации и сопровождение работы приложения на протяжении полутора месяца, для пресечения каких либо проблем в работе организации.

Разберем каждый пункт отдельно. И так развертывание программной части. Первым делом необходимо дать полный доступ приложению к базе данных организации, а затем настроить в приложении статические пути для соединения с базой банных и с сервером. Чтобы в случае неполадок сети или каких-то других обстоятельств, после возобновления работы сети, пути для соединения не были нарушены. Далее передо мной стоял вопрос выбора устройства для работы приложения, но так как в организации есть порядка 70 смартфонов FLYIQ4502ERAEnergy 1 Quad, решено использовать именно их, так как они подходили по всем параметрам, а именно большой объем батареи и экран размером 5 дюймов и разрешением экрана 854x480. Так же не мало важным фактором является качество съемки камеры, так как работа смартфона тесно

связана со сканированием штрих-кодов. Соответственно, от того, как качество снимает камера, будет зависеть скорость сканирования кода. Но самое главное это автономность устройства, ведь согласитесь, какая будет польза, от устройства, которое разряжается после 5-10 часов работы, конечно проблему можно решить сменными аккумуляторами, что в принципе будет, но для них нужны зарядные станции, что лишние растраты, так что самый оптимальный вариант, найти такую грань, при которой у смартфона большой экран, и большой объем батареи.И решено было остановить свой выбора на FLYIQ4502. Тем самым необходимости закупки нового оборудования у организации нет.

Соответственно два пункта были выполнены, остались 3 и 4 пункты. А именно установка приложений и инструктаж сотрудников. Количество сотрудников на складе каждый раз меняется, и чтобы предотвратить дальнейшие манипуляции со смартфонами, то есть заново переустанавливать приложения и так далее, было решено выделить на склад 25 смартфонов, которые привязаны к каждому сотруднику.. Необходимо было провести инструктаж сотрудников предприятия. План инструктажа вы можете наблюдать ниже:

1. Введение в теоритическую часть пользования Android устройством

2. Инструктаж о правилах пользования смартфоном и технике безопасности

3. Инструктаж по работе приложения, без углубления в техническую составляющую данного.

4. Закрепление полученных знаний, непосредственно на складе предприятия и под контролем системных администраторов.

5. Мониторинг работы склада

Самая главная часть внедрения, это серверная часть. Так как настройка бесперебойной работы сервера и шлюза, довольно проблематично. Так как шлюзом для связи приложения и серверной части предприятия является обычный стационарный персональный компьютер, который управляется операционной системой Zentyal 3.2, она Linux-о подобна. Но тем не менее является отдельной частью, хотя в некоторых случаях есть просто дополнения к любой версии linux. Соединение проходит по VPN, то есть на каждом телефоне настроено VPNсоединение.

Если рассматривать количество времени, затраченное на разработку и внедрение приложения, то со всеми особенностями разработки и изучением языка программирования java, обща цифра подходит под полгода. Так же плюс полтора месяца сопровождения тестирование. На рисунке 3.3 вы можете наблюдать все затраты в течении жизненного цикла разработки и внедрения системы и эффект от ее использования, в котором сосредоточиваются основные, функциональные критерии качества, отражающие значение, область применения и качественные характеристики.

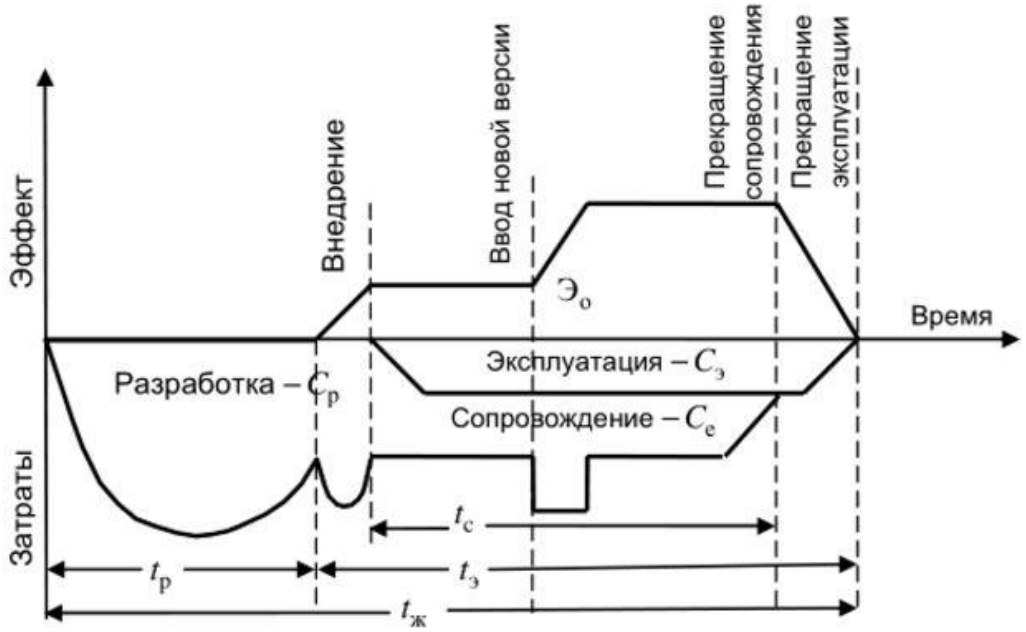


Рис. 3.3. Затраты в течении всего жизненного цикла

На протяжении полутора месяцев я и начальник ITотдела организации занимались мониторингом работы сотрудников склада и работы приложения. На протяжении данного промежутка времени сотрудники склада естественно обращались, с вопросами по поводу работы приложения, но вопросы не были связанны с технической частью приложения, а касались только рабочих вопросов. Соответственно работа предприятия не была нарушена, что является безусловным успехом внедрения и разработки приложения. Тем самым достигнув удовлетворения требований организации, по итогам было проведено отличное внедрение новой системы.

Так как организация является филиалом главного склада расположенного в городе Курск, то и там заинтересовались данной разработкой. С организацией было заключено соглашение на дальнейшее внедрение данной разработки на складе предприятия в городе Курск.

3.3 Анализ модернизированной системы и пути ее развития

Приступая к анализу модернизированной системы, было проведено совещание с руководством организации о выполненной работе, по результатам беседы был получен одобрительный отзыв о внедрении и в целом моей разработке. Соответственно необходимо провести анализ моего решения.

Что касаемо экономической составляющей, то давайте разберем все подробно. В данный момент стоимость терминала сбора данных, Motorola MC3190 составляет приблизительно 86 тысяч рублей. Но весь потенциал данного терминала в организации не используется, соответственно, зачем он нужен за такую сумму, ведь он просто выступает средством для обычной RDP сессии. А к примеру мое решение, использует смартфон ценой не более 8 тысяч рублей, это в рамках этой организации, FLYIQ4502. То есть стоимость смартфона, грубо говоря, в 10 раз меньше чем терминала сбора данных, соответственно экономия просто на лицо. Если взять, к примеру, сколько

средств было сэкономлено в данном случае, то подсчеты просты. В организации имеется 23 терминала сбора данных, то есть это уже сумма в практически 2 миллиона рублей, а стоимость смартфонов будет составлять 184 тысячи рублей. Согласитесь, разница колоссальна. Что говорит о том, что если бы организация не закупала терминалы сбора данных, она бы сэкономила по примерным расчетам 1 миллион 816 тысяч рублей.

Исходя от экономической составляющей, картина которая показывает эффективность работы является не хуже. Как вы понимаете, терминалы сбора данных довольно громоздкие, а смартфоны не больше ладони. Соответственно есть победа в эргономичности, что уже является большим плюсом. Далее сканеры используемые в терминале сбора данных, требуют технического обслуживания, что является довольно большой тратой средств организации, а в случае повреждения камеры телефона, она заменяется как отдельный маленький модуль, которых в организации достаточно. Соответственно особых дополнительных расходов у организации не будет.

Говоря о дальнейшем пути развития, представляется картина совершенно нового и более современного приложения, которое так же будет выполнять вычисления на сервере организации, которая будет им пользоваться или по VPNсоединению на удаленном сервере, к примеру расположенному в каком-нибудь data-центре. Соответственно управление приложением, обновлениями, обслуживанием будет удаленным. Соответственно у меня масса планов дальнейшей модернизации моей разработки, а именно:

1) Модернизация интерфейса приложения

2) Модернизация оформления, возможность смены темы

3) Работа либо на сервере организации, либо на удаленном сервере с помощью VPN соединения

4) Разработка единой системы не только для организации ООО Лидер, а для всех складских предприятий.

5) При успешном выполнении модернизаций и создании мощного инструмента, попытка внедрить мое решение в разные организации города Белгорода, а в дальнейшем России

ЗАКЛЮЧЕНИЕ

В процессе выполнения дипломного проекта была рассмотрена существующая в ООО “ЛИДЕР” система складского учета, обоснована необходимость его автоматизации. Достигнута поставленная цель дипломного проекта, усовершенствована система учета товаров на складе ООО “ЛИДЕР”. Были решены поставленные задачи.

1)Выполнен анализ деятельности предприятия ООО “ЛИДЕР”, в том числе управление складским учетом.

2)Разработан и обоснован проект автоматизации складского учета.

3)Разработан модуль интеграция 1С Предприятия с приложением Android.. Проведено тестирование разработанного модуля на реальных данных.

4) Разработано приложение на платформе Android

5)Разработанное приложение успешно внедрено в работу организации

В результате проделанной работы было автоматизировано несколько рабочих мест. Стало возможным снижение затрат на оборудование. Значительно уменьшилось количество допускаемых ошибок при проведении стандартных операций складского учета. Модернизирован процесс обработки информации, при этом повысилась степень достоверности информации и степень ее защиты.

В ООО “ЛИДЕР” произведена апробация созданного Androidприложения «LiderStorageManager», которая подтвердила актуальность и практическую значимость раскрытой в дипломном проекте темы (соответствующий акт прилагается).

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Хашими, С. Разработка приложений для Android / С. Хашими, С. Коматинени, Д. Маклин - СПб.: Питер, 2011 - 736 с.

2. Эккель, Б. Философия Java. Библиотека программиста / Б. Эккель - СПб.: Питер, 2009. - 640 с.

3. Дейтел П., Дейтел Х., Уолд А. Д27 Android для разработчиков. 3-е изд. — СПб.: Питер, 2016. — 512 с.: ил. — (Серия «Библиотека программиста»).

4. Голощапов, А.Л. GoogleAndroid: программирование для мобильных устройств / А.Л. Голощапов - СПб.: БХВ-Петербург, 2011. - 448 с.

5. Емельянова, И.А. Технико-экономическое обоснование в дипломных проектах. Пособие для студентов электротехнического факультета / И.А. Емельянова - Гомель: БелГУТ, 2004. - 50 с.

6. Описание среды разработки Java [Электронный ресурс]. – Режим доступа:https://ru.wikipedia.org/wiki/java

7. Статья по разработке Androidприложений [Электронный ресурс]. – Режим доступа:https://habrahabr.ru/post/146632/

8. Руководство по разработке Android приложений, курсы [Электронный ресурс]. – Режим доступа:http://developer.alexanderklimov.ru/android/

9. Разработка под Android [Электронный ресурс]. – Режим доступа: https://habrahabr.ru/hub/android_dev/

10. Организация работы склада. Описание [Электронный ресурс]. – Режим доступа: https://habrahabr.ru/company/trinion/blog/296718/

11. Сайт о программировании [Электронный ресурс]. – Режим доступа: https://metanit.com/java/android/1.1.php

12. Сканеры штрих кода в автоматизации торговли [Электронный ресурс]. – Режим доступа: https://habrahabr.ru/company/scancode/blog/243145/

13. Распознавание BarcodeAndroid. Разработка в AndroidStudio [Электронный ресурс]. – Режим доступа: https://habrahabr.ru/post/199258/

ПРИЛОЖЕНИЕ

Код приложения Android

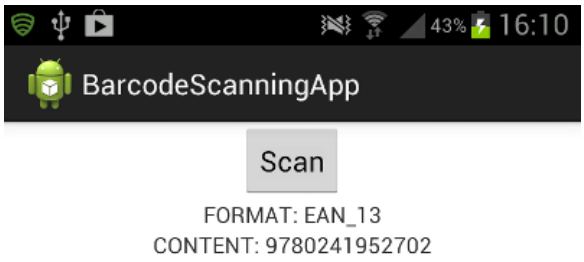
BarcodeScanner

```
<RelativeLayoutxmlns:android="http://schemas.android.com/apk/res/android"xmlns:tools="htt
p://schemas.android.com/tools"android:layout_width="match_parent"android:layout_height="m
atch_parent"><Buttonandroid:id="@+id/scan_button"android:layout_width="wrap_content"andro
id:layout_height="wrap_content"android:layout_centerHorizontal="true"android:text="@strin
g/scan" /></RelativeLayout>
```

```
<TextViewandroid:id="@+id/scan_format"android:layout_width="wrap_content"android:layout_h
eight="wrap_content"android:textIsSelectable="true"android:layout_centerHorizontal="true"
android:layout_below="@id/scan_button"
/><TextViewandroid:id="@+id/scan_content"android:layout_width="wrap_content"android:layou
t_height="wrap_content"android:textIsSelectable="true"android:layout_centerHorizontal="tr
ue"android:layout_below="@id/scan_format" />
```

```
importcom.google.zxing.integration.android.IntentIntegrator;
importcom.google.zxing.integration.android.IntentResult;

importandroid.os.Bundle; importandroid.app.Activity; importandroid.content.Intent;
importandroid.view.View; importandroid.view.View.OnClickListener;
importandroid.widget.Button; importandroid.widget.TextView; importandroid.widget.Toast;

protectedvoidonCreate(Bundle savedInstanceState){ super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main); }

private Button scanBtn; privateTextViewformatTxt, contentTxt;

scanBtn = (Button)findViewById(R.id.scan_button); formatTxt =
(TextView)findViewById(R.id.scan_format); contentTxt =
(TextView)findViewById(R.id.scan_content);

scanBtn.setOnClickListener(this);

publicclassMainActivityextendsActivityimplementsOnClickListener

publicvoidonClick(View v){ //respond to clicks }

if(v.getId()==R.id.scan_button){ //scan }

IntentIntegratorscanIntegrator = newIntentIntegrator(this);

scanIntegrator.initiateScan();

publicvoidonActivityResult(intrequestCode, intresultCode, Intent intent){ //retrieve scan
result }

IntentResultscanningResult = IntentIntegrator.parseActivityResult(requestCode,
```

```
resultCode, intent);

if (scanningResult != null) { //we have a result }

else{ Toast toast = Toast.makeText(getApplicationContext(), "No scan data received!",
Toast.LENGTH_SHORT); toast.show(); }

formatTxt.setText("FORMAT: " + scanFormat); contentTxt.setText("CONTENT: " +
scanContent);
```



Android manifest

```
<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:"http://schemas.android.com/apk/res/android" android:versionCode="1" android:versionName="1.0"
package="appinventor.ai_mitabishi.LiderS" platformBuildVersionCode="22" platformBuildVersionName="5.1.1-1819727">

<uses-permission android:name="android.permission.CAMERA" />

<uses-permission android:name="android.permission.INTERNET" />

<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />

<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

<uses-sdkandroid:minSdkVersion="4" />

<application android:label="Lider Storage Manager" android:icon="@drawable/ya"
android:name="com.google.appinventor.components.runtime.multidex.MultiDexApplication" android:debuggable="false">

<activity android:name=".Screen1" android:configChanges="keyboard|keyboardHidden|orientation"
android:windowSoftInputMode="2">

<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>
```

```
</activity>

<activity android:name="appinventor.ai_mitabishi.LiderS.CARWork"
android:configChanges="keyboard|keyboardHidden|orientation" android:windowSoftInputMode="2">

<intent-filter>

<action android:name="android.intent.action.MAIN" />

</intent-filter>

</activity>

<activity android:name="appinventor.ai_mitabishi.LiderS.EmployScreen"
android:configChanges="keyboard|keyboardHidden|orientation" android:windowSoftInputMode="2">

<intent-filter>

<action android:name="android.intent.action.MAIN" />

</intent-filter>

</activity>

<activity android:name="appinventor.ai_mitabishi.LiderS.ManagerScreen"
android:configChanges="keyboard|keyboardHidden|orientation" android:windowSoftInputMode="2">

<intent-filter>

<action android:name="android.intent.action.MAIN" />

</intent-filter>

</activity>

<activity android:name="appinventor.ai_mitabishi.LiderS.CarScreen"
android:configChanges="keyboard|keyboardHidden|orientation" android:windowSoftInputMode="2">

<intent-filter>

<action android:name="android.intent.action.MAIN" />

</intent-filter>

</activity>

<activity android:name="appinventor.ai_mitabishi.LiderS.MWork" android:configChanges="keyboard|keyboardHidden|orientation"
android:windowSoftInputMode="2">

<intent-filter>

<action android:name="android.intent.action.MAIN" />

</intent-filter>

</activity>

<activity android:name="appinventor.ai_mitabishi.LiderS.EMPWork"
android:configChanges="keyboard|keyboardHidden|orientation" android:windowSoftInputMode="2">
```

```
<intent-filter>

<action android:name="android.intent.action.MAIN" />

</intent-filter>

</activity>

<activity android:name="appinventor.ai_mitabishi.LiderS.Error" android:configChanges="keyboard|keyboardHidden|orientation"
android:windowSoftInputMode="2">

<intent-filter>

<action android:name="android.intent.action.MAIN" />

</intent-filter>

</activity>

<activity android:theme="@*android:style/Theme.NoTitleBar.Fullscreen"
android:name="com.google.zxing.client.android.AppInvCaptureActivity" android:stateNotNeeded="true"
android:screenOrientation="landscape" android:configChanges="keyboardHidden|orientation" android:windowSoftInputMode="3" />

</application>

</manifest>
```

Carscreen

```
/* compiled from: CarScreen.yail */
   public class frame extends ModuleBody {
CarScreen $main;

       public int match1(ModuleMethodmoduleMethod, Object obj, CallContextcallContext) {
          switch (moduleMethod.selector) {
             case ParseFormat.SEEN_MINUS /*1*/:
                callContext.value1 = obj;
callContext.proc = moduleMethod;
callContext.pc = 1;
                return 0;
             case XDataType.ANY_ATOMIC_TYPE_CODE /*3*/:
                if (!(objinstanceof Symbol)) {
                   return -786431;
                }
                callContext.value1 = obj;
callContext.proc = moduleMethod;
callContext.pc = 1;
                return 0;
             case ArithOp.DIVIDE_INEXACT /*5*/:
                if (!(objinstanceof Symbol)) {
                   return -786431;
                }
                callContext.value1 = obj;
```

```
callContext.proc = moduleMethod;
callContext.pc = 1;
                return 0;
            case ArithOp.ASHIFT_LEFT /*10*/:
                callContext.value1 = obj;
callContext.proc = moduleMethod;
callContext.pc = 1;
                return 0;
            case ArithOp.ASHIFT_RIGHT /*11*/:
                callContext.value1 = obj;
callContext.proc = moduleMethod;
callContext.pc = 1;
                return 0;
            case ArithOp.LSHIFT_RIGHT /*12*/:
                if (!(objinstanceofCarScreen)) {
                    return -786431;
                }
                callContext.value1 = obj;
callContext.proc = moduleMethod;
callContext.pc = 1;
                return 0;
            case Sequence.DOCUMENT_VALUE /*34*/:
                callContext.value1 = obj;
callContext.proc = moduleMethod;
callContext.pc = 1;
                return 0;
            default:
                return super.match1(moduleMethod, obj, callContext);
        }
    }

    public int match2(ModuleMethodmoduleMethod, Object obj, Object obj2, CallContextcallContext) {
        switch (moduleMethod.selector) {
            case SetExp.DEFINING_FLAG /*2*/:
                if (!(objinstanceof Symbol)) {
                    return -786431;
                }
                callContext.value1 = obj;
                callContext.value2 = obj2;
callContext.proc = moduleMethod;
callContext.pc = 2;
                return 0;
            case XDataType.ANY_ATOMIC_TYPE_CODE /*3*/:
                if (!(objinstanceof Symbol)) {
                    return -786431;
                }
                callContext.value1 = obj;
                callContext.value2 = obj2;
callContext.proc = moduleMethod;
callContext.pc = 2;
                return 0;
            case ArithOp.QUOTIENT /*6*/:
                if (!(objinstanceof Symbol)) {
                    return -786431;
                }
                callContext.value1 = obj;
```

```
                callContext.value2 = obj2;
callContext.proc = moduleMethod;
callContext.pc = 2;
                return 0;
            case ArithOp.QUOTIENT_EXACT /*7*/:
                callContext.value1 = obj;
                callContext.value2 = obj2;
callContext.proc = moduleMethod;
callContext.pc = 2;
                return 0;
            case ArithOp.ASHIFT_GENERAL /*9*/:
                callContext.value1 = obj;
                callContext.value2 = obj2;
callContext.proc = moduleMethod;
callContext.pc = 2;
                return 0;
            case ArithOp.IOR /*14*/:
                callContext.value1 = obj;
                callContext.value2 = obj2;
callContext.proc = moduleMethod;
callContext.pc = 2;
                return 0;
            default:
                return super.match2(moduleMethod, obj, obj2, callContext);
        }
    }

    public int match4(ModuleMethodmoduleMethod, Object obj, Object obj2, Object obj3, Object obj4, CallContextcallContext) {
        switch (moduleMethod.selector) {
            case SetExp.PREFER_BINDING2 /*8*/:
                callContext.value1 = obj;
                callContext.value2 = obj2;
                callContext.value3 = obj3;
                callContext.value4 = obj4;
callContext.proc = moduleMethod;
callContext.pc = 4;
                return 0;
            case ArithOp.AND /*13*/:
                if (!(objinstanceofCarScreen)) {
                    return -786431;
                }
                callContext.value1 = obj;
                if (!(obj2 instanceof Component)) {
                    return -786430;
                }
                callContext.value2 = obj2;
                if (!(obj3 instanceof String)) {
                    return -786429;
                }
                callContext.value3 = obj3;
                if (!(obj4 instanceof String)) {
                    return -786428;
                }
                callContext.value4 = obj4;
callContext.proc = moduleMethod;
callContext.pc = 4;
```

```
                return 0;
            default:
                return super.match4(moduleMethod, obj, obj2, obj3, obj4, callContext);
        }
    }

    public Object apply1(ModuleMethodmoduleMethod, Object obj) {
        switch (moduleMethod.selector) {
            case ParseFormat.SEEN_MINUS /*1*/:
                this.$main.androidLogForm(obj);
                return Values.empty;
            case XDataType.ANY_ATOMIC_TYPE_CODE /*3*/:
                try {
                    return this.$main.lookupInFormEnvironment((Symbol) obj);
                } catch (ClassCastException e) {
                    throw new WrongType(e, "lookup-in-form-environment", 1, obj);
                }
            case ArithOp.DIVIDE_INEXACT /*5*/:
                try {
return this.$main.isBoundInFormEnvironment((Symbol) obj) ? Boolean.TRUE : Boolean.FALSE;
                } catch (ClassCastException e2) {
                    throw new WrongType(e2, "is-bound-in-form-environment", 1, obj);
                }
            case ArithOp.ASHIFT_LEFT /*10*/:
                this.$main.addToFormDoAfterCreation(obj);
                return Values.empty;
            case ArithOp.ASHIFT_RIGHT /*11*/:
                this.$main.sendError(obj);
                return Values.empty;
            case ArithOp.LSHIFT_RIGHT /*12*/:
                this.$main.processException(obj);
                return Values.empty;
            case Sequence.DOCUMENT_VALUE /*34*/:
                return this.$main.BarcodeScanner1$AfterScan(obj);
            default:
                return super.apply1(moduleMethod, obj);
        }
    }

    public Object apply4(ModuleMethodmoduleMethod, Object obj, Object obj2, Object obj3, Object obj4) {
        switch (moduleMethod.selector) {
            case SetExp.PREFER_BINDING2 /*8*/:
                this.$main.addToComponents(obj, obj2, obj3, obj4);
                return Values.empty;
            case ArithOp.AND /*13*/:
                try {
                    try {
                        try {
                            try {
                                try {
return this.$main.dispatchEvent((Component) obj, (String) obj2, (String) obj3, (Object[]) obj4) ? Boolean.TRUE : Boolean.FALSE;
                                } catch (ClassCastException e) {
                                    throw new WrongType(e, "dispatchEvent", 4, obj4);
                                }
                            } catch (ClassCastException e2) {
                                throw new WrongType(e2, "dispatchEvent", 3, obj3);
                            }
```

```
                } catch (ClassCastException e22) {
                    throw new WrongType(e22, "dispatchEvent", 2, obj2);
                }
            } catch (ClassCastException e222) {
                throw new WrongType(e222, "dispatchEvent", 1, obj);
            }
        default:
            return super.apply4(moduleMethod, obj, obj2, obj3, obj4);
    }
  }

  public Object apply2(ModuleMethodmoduleMethod, Object obj, Object obj2) {
    switch (moduleMethod.selector) {
      case SetExp.DEFINING_FLAG /*2*/:
        try {
          this.$main.addToFormEnvironment((Symbol) obj, obj2);
          return Values.empty;
        } catch (ClassCastException e) {
          throw new WrongType(e, "add-to-form-environment", 1, obj);
        }
      case XDataType.ANY_ATOMIC_TYPE_CODE /*3*/:
        try {
          return this.$main.lookupInFormEnvironment((Symbol) obj, obj2);
        } catch (ClassCastException e2) {
          throw new WrongType(e2, "lookup-in-form-environment", 1, obj);
        }
      case ArithOp.QUOTIENT /*6*/:
        try {
          this.$main.addToGlobalVarEnvironment((Symbol) obj, obj2);
          return Values.empty;
        } catch (ClassCastException e22) {
          throw new WrongType(e22, "add-to-global-var-environment", 1, obj);
        }
      case ArithOp.QUOTIENT_EXACT /*7*/:
        this.$main.addToEvents(obj, obj2);
        return Values.empty;
      case ArithOp.ASHIFT_GENERAL /*9*/:
        this.$main.addToGlobalVars(obj, obj2);
        return Values.empty;
      case ArithOp.IOR /*14*/:
        return this.$main.lookupHandler(obj, obj2);
      default:
        return super.apply2(moduleMethod, obj, obj2);
    }
  }

  public Object apply0(ModuleMethodmoduleMethod) {
    switch (moduleMethod.selector) {
      case ArithOp.XOR /*15*/:
        return CarScreen.lambda2();
      case SetExp.PROCEDURE /*16*/:
this.$main.$define();
        return Values.empty;
      case Sequence.INT_U8_VALUE /*17*/:
        return CarScreen.lambda3();
      case Sequence.INT_S8_VALUE /*18*/:
```

```
            return CarScreen.lambda4();
          case Sequence.INT_U16_VALUE /*19*/:
            return CarScreen.lambda5();
          case Sequence.INT_S16_VALUE /*20*/:
            return CarScreen.lambda6();
          case Sequence.INT_U32_VALUE /*21*/:
            return CarScreen.lambda7();
          case Sequence.INT_S32_VALUE /*22*/:
            return this.$main.TextBox1$LostFocus();
          case Sequence.INT_U64_VALUE /*23*/:
            return CarScreen.lambda8();
          case Sequence.INT_S64_VALUE /*24*/:
            return CarScreen.lambda9();
          case Sequence.FLOAT_VALUE /*25*/:
            return CarScreen.lambda10();
          case Sequence.DOUBLE_VALUE /*26*/:
            return CarScreen.lambda11();
          case Sequence.BOOLEAN_VALUE /*27*/:
            return this.$main.Button1$Click();
          case Sequence.TEXT_BYTE_VALUE /*28*/:
            return CarScreen.lambda12();
          case Sequence.CHAR_VALUE /*29*/:
            return CarScreen.lambda13();
          case XDataType.DAY_TIME_DURATION_TYPE_CODE /*30*/:
            return this.$main.Button2$Click();
          case Sequence.CDATA_VALUE /*31*/:
            return CarScreen.lambda14();
          case SetExp.SET_IF_UNBOUND /*32*/:
            return CarScreen.lambda15();
          case Sequence.ELEMENT_VALUE /*33*/:
            return this.$main.Button3$Click();
          default:
            return super.apply0(moduleMethod);
        }
      }
```