

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

**ЛЕСОСИБИРСКИЙ ПЕДАГОГИЧЕСКИЙ ИНСТИТУТ –
филиал Сибирского федерального университета**

Высшей математики, информатики, экономики и естествознания
кафедра

УТВЕРЖДАЮ

Заведующий кафедрой

 Л.Н. Храмова
подпись инициалы, фамилия


« 13 » 06 2023 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.02 Информационные системы и технологии
код-наименование направления

РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ «ИНВЕНТАРИЗАЦИЯ»


Руководитель

 13.06.2023
подпись, дата

доцент, канд. пед. наук
должность, ученая степень


А. В. Фирер
инициалы, фамилия

Выпускник

 13.06.2023
подпись, дата

Е. П. Елизарова
инициалы, фамилия

Нормоконтролер

 13.06.2023
подпись, дата

Е. В. Киргизова
инициалы, фамилия

Лесосибирск 2023

РЕФЕРАТ

Выпускная квалификационная работа по теме «Разработка мобильного приложения «Инвентаризация»» содержит 60 страниц текстового документа, 13 иллюстраций, 6 таблиц, 41 использованный источник.

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ, ANDROID, МОБИЛЬНОЕ ПРИЛОЖЕНИЕ, МОБИЛЬНАЯ РАЗРАБОТКА, KOTLIN.

Цель работы – теоретически обосновать и разработать мобильное приложение для инвентаризации с использованием QR-кодов.

Объект исследования – процесс инвентаризации в организации.

Предмет исследования – процесс разработки мобильного приложения для инвентаризации с использованием QR-кодов.

Задачи исследования:

- провести анализ предметной области и сделать обзор существующих мобильных решений;

- выделить функциональные требования к разрабатываемому мобильному приложению;

- разработать мобильное приложение для инвентаризации под операционную систему Android;

- провести апробацию и тестирование приложения.

Во время разработки мобильного приложения изучены основы языка программирования Kotlin. Подробно изучены скрипты Google, а также взаимодействие клиента приложения с Android API.

В ходе написания выпускной квалификационной работы автор принимал участие в конференциях и конкурсах. Так же опубликовано 3 статьи.

В результате выполнения выпускной квалификационной работы разработано мобильное приложение для проведения инвентаризации, которое было апробировано и протестировано.

СОДЕРЖАНИЕ

Введение.....	4
1 Проектирование мобильного приложения	7
1.1 Анализ предметной области	7
1.2 Обозначение требований к мобильному приложению и анализ существующих мобильных решений.....	12
1.3 Структурная схема мобильного приложения	18
2 Разработка и тестирование мобильного приложения	21
2.1 Выбор средств разработки	21
2.2 Разработка модулей мобильного приложения.....	25
2.2.1 Модуль сохранения объекта.....	26
2.2.2 Модуль чтения объекта.....	29
2.2.3 Модуль сканирования объекта.....	31
2.3 Тестирование мобильного приложения.....	33
2.4 Руководство пользователя.....	39
2.4.1 Установка и запуск	39
2.4.2 Работа с приложением.....	39
Заключение	43
Список использованных источников	45
Приложение А Организационная схема ЛПИ – филиала СФУ.....	49
Приложение Б Листинг модуля чтения	50
Приложение В Листинг модуля сохранения	53
Приложение Г Листинг модуля сканирования	58

ВВЕДЕНИЕ

В настоящее время мобильные технологии прочно установились в производственном цикле любого предприятия, позволяют облегчить труд, усовершенствовать процесс обработки информации, ее накопление и хранение.

«Инвентаризация – это проверка наличия имущества организации и состояния её финансовых обязательств на определённую дату путём сличения фактических данных с данными бухгалтерского учёта» [10].

Облегчить процесс инвентаризации можно с помощью внедрения на предприятии компьютерных программ. В настоящее время большинство обыденных процессов можно автоматизировать. Инвентаризация так же не стала исключением в процессе автоматизации. Поэтому очень важно в процессе автоматизации использовать мобильные средства (телефон или планшет). Это повысит скорость проведения инвентаризации, точность, а также добавит возможность использовать удаленный доступ к хранилищу (таблице) с любого устройства.

Цель работы – теоретически обосновать и разработать мобильное приложение для инвентаризации с использованием QR-кодов.

Объект исследования – процесс инвентаризации в организации.

Предмет исследования – процесс разработки мобильного приложения для инвентаризации с использованием QR-кодов.

Мобильное приложение позволит пользователям сканировать QR-коды имущества организации и проводить инвентаризацию.

Цель, объект и предмет исследования обусловили постановку и решение следующих задач:

- провести анализ предметной области и сделать обзор существующих мобильных решений;

- выделить функциональные требования к разрабатываемому мобильному приложению;

– разработать мобильное приложение для инвентаризации под операционную систему Android;

– провести апробацию и тестирование приложения.

Методы исследования:

– теоретические: анализ учебной и научно-технической литературы по теме исследования; обобщение; сравнительный анализ;

– эмпирические: наблюдение; беседа; моделирование; тестирование программного продукта.

Экспериментальная база исследования: Лесосибирский педагогический институт – филиал Сибирского федерального университета (ЛПИ – филиал СФУ).

Практическая значимость мобильного приложения для инвентаризации заключается в том, что разработанное мобильное приложение для инвентаризации может быть использовано сотрудниками любой организации, являющимися материально ответственными лицами.

Результаты исследования представлены на следующих научных мероприятиях:

– VI Всероссийская научно-практическая конференция преподавателей, учителей, студентов и молодых ученых «Актуальные проблемы преподавания дисциплин естественнонаучного цикла» (г. Лесосибирск, ЛПИ – филиал СФУ, 7–12 ноября 2022 г., участие).

– VII Международная научно-практическая конференция «Молодёжь, наука, образование: Актуальные вопросы, достижения и инновации» (г. Пенза, МЦНС «Наука и Просвещение», 12 апреля 2023 г., участие).

– Всероссийский молодежный научный форум «Современное педагогическое образование: теоретический и прикладной аспекты» (г. Лесосибирск, ЛПИ – филиал СФУ, 11 апреля 2023 г., I место).

– II Всероссийский конкурс научных работ «Молодёжный научный потенциал» (г. Лесосибирск, ЛПИ – филиал СФУ, 12 апреля 2023 г., участие).

По результатам исследования опубликованы статьи:

1. Елизарова, Е. П. Анализ мобильных приложений для инвентаризации / Е. П. Елизарова // Актуальные проблемы преподавания дисциплин естественнонаучного цикла: тезисы докладов VI Всероссийской научно-практической конференции преподавателей, учителей, студентов и молодых ученых, Лесосибирск, 14–15 ноября 2022 года / Сибирский федеральный университет. – Красноярск, 2022. – С. 12–14.

2. Елизарова, Е. П. Актуальные средства разработки мобильных приложений / Е. П. Елизарова // Современное педагогическое образование: Теоретический и прикладной аспекты: сборник научных статей II Всероссийский молодёжный научный форум, студентов и молодых ученых, Лесосибирск, 10–15 апреля 2023 года / Сибирский федеральный университет. – Лесосибирск – Красноярск, 2023. – С. 58-60.

3. Елизарова, Е. П. Разработка мобильных приложений на Kotlin / Е. П. Елизарова // Молодёжь, наука, образование: Актуальные вопросы, достижения и инновации, Пенза, 12 апреля 2022 года / МЦНС «Наука и просвещение». – Пенза, 2023. – С. 55–57.

Структура работы – работа состоит из введения, двух глав, заключения, списка литературы, включающего 41 наименование. Результаты работы представлены в 6 таблицах, 13 рисунках. В 4 приложениях представлены организационная структура организации и листинги модулей приложения. Общий объем работы – 60 печатных листов.

1 Проектирование мобильного приложения

1.1 Анализ предметной области

Для того чтобы начать проектирование мобильного приложения необходимо определиться с предметной областью. В качестве предметной области был выбран процесс инвентаризации в Лесосибирском педагогическом институте – филиале Сибирского федерального университета (ЛПИ – филиал СФУ).

Основным видом деятельности в ЛПИ – филиале СФУ является обучение студентов по нескольким направлениям и специальностям. Каждый год на каждое направление/специальность на основе конкурсного отбора по результатам вступительных экзаменов зачисляются студенты. На каждого студента заводится личное дело, которое хранится в течение всего срока обучения, а затем передается в архив. Для правильной организации процесса обучения, студентов делят на группы. В каждой группе назначается староста.

В институте инвентаризация проводится с целью точного определения и учета всех существующих ресурсов, таких как здания, оборудование, компьютеры, мебель, учебные пособия и другие материальные ценности. Это позволяет управляющим органам института иметь полное представление о своих активах и эффективно планировать их использование.

Организационная структура института [19] представлена в Приложении А.

Инвентаризация представляет собой процесс учета основных средств предприятия и товарно-материальных ценностей [10]. Для целей проведения инвентаризации собирается инвентаризационная комиссия, во главе которой находится главный бухгалтер организации. Инвентаризация регулируется Федеральным законом «О бухгалтерском учете» Федеральный закон от 06.12.2011 N 402-ФЗ (ред. от 18.07.2017) статьей 11 [24].

Целью проведения инвентаризации является фактическое выявление наличия или отсутствия того или иного имущества в организации в соответствии с бухгалтерскими регистрами учета [22].

Порядок проведения инвентаризации, ее сроки определяются самим предприятием. Исключение составляют случаи обязательного проведения инвентаризации, которые устанавливаются Российскими отраслевыми стандартами.

Изучив приказ Минфина РФ от 13.06.1995 N 49 (ред. от 08.11.2010) «Об утверждении Методических указаний по инвентаризации имущества и финансовых обязательств» [22] выделим обязательные случаи для проведения инвентаризации.

Инвентаризация проводится обязательно в случаях:

- если имущество предприятия подлежит передаче в аренду или подлежит продаже, выкупе и при реорганизации предприятия;
- перед составлением годовой бухгалтерской отчетности (кроме имущества, инвентаризация которого проводилась не ранее 1 октября отчетного года);
- при смене материально ответственных лиц;
- при выявлении фактов хищения, злоупотребления или порчи имущества;
- в случае стихийного бедствия, пожара или других чрезвычайных ситуаций, вызванных экстремальными условиями;
- при ликвидации (реорганизации) организации перед составлением ликвидационного (разделительного) баланса и в других случаях, предусмотряемых законодательством Российской Федерации или нормативными актами Министерства финансов Российской Федерации.

Выявленные расхождения подлежат регистрации в бухгалтерском учете в том отчетном периоде, к которому относится дата проведения инвентаризации.

По результатам инвентаризации, инвентаризационная комиссия подписывает опись.

На данный момент процесс инвентаризации протекает следующим образом:

- ручное распечатывание на бумаге всего перечня номенклатуры;
- непосредственно на рабочих местах происходит визуальный поиск в распечатанном перечне и проставление отметки о наличии номенклатуры;
- ручное создание и заполнение документов инвентаризации, последующая сверка полученных результатов.

Процесс инвентаризации в институте представим в виде диаграммы IDEF0. IDF0, или Integration Definition for Function Modeling, представляет собой метод моделирования, который используется для отображения процессов, процедур и действий внутри организации [14]. Он включает в себя графический язык, который служит для передачи специфической информации и обеспечивает понимание и анализ процессов.

Диаграмма IDEF0 «Инвентаризация» представлена ниже на рисунке 1.

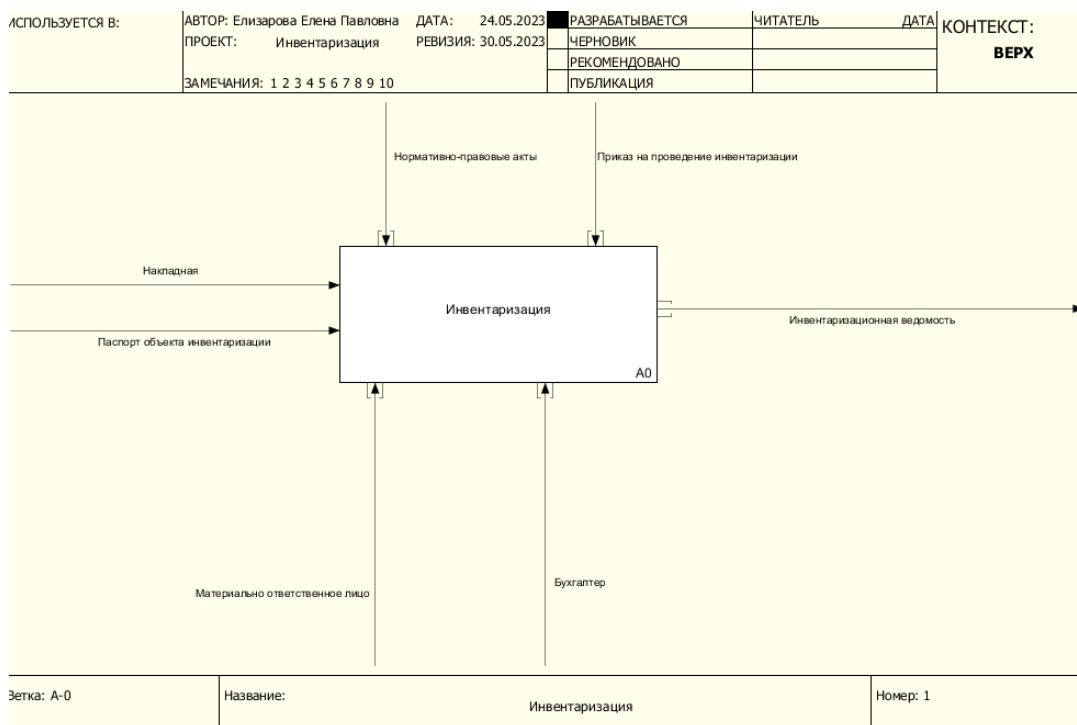


Рисунок 1 – Бизнес-процесс инвентаризации (базовый уровень)

Проведем декомпозицию процесса инвентаризации. Декомпозиция представлена ниже на рисунке 2.

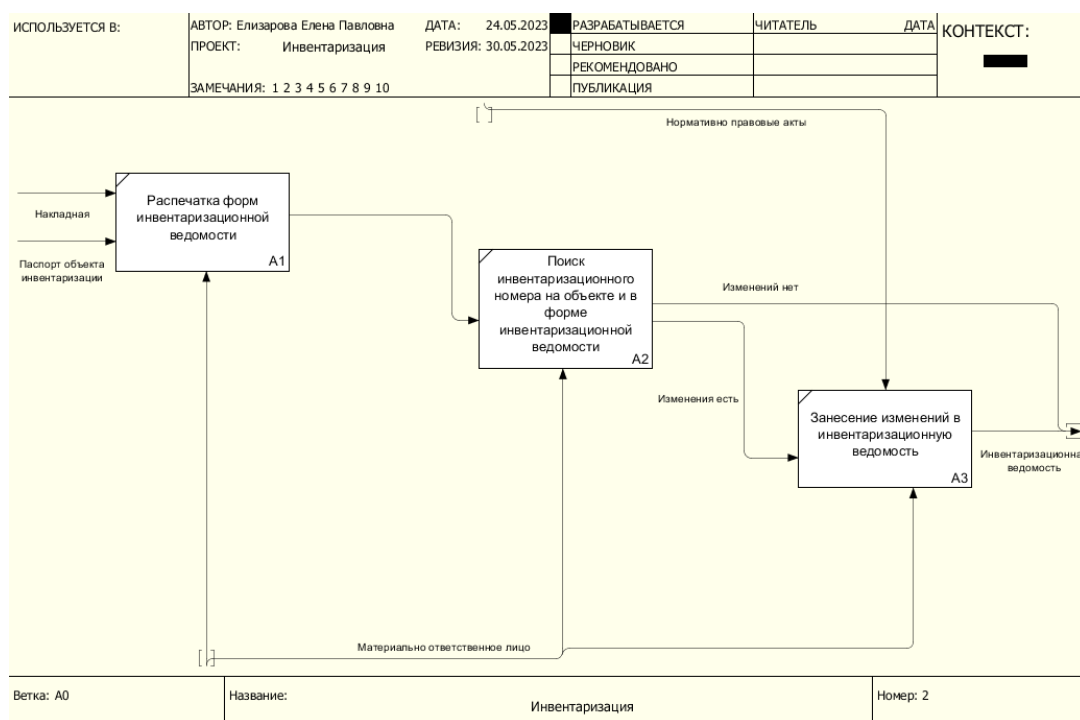


Рисунок 2 – Декомпозиция процесса инвентаризация

Таким образом мы разбили процесс инвентаризации на блоки, что позволит нам выделить процессы, которые хотим оптимизировать. Для нашего проекта нас интересует только блоки «Поиск инвентаризационного номера на объекте и в форме» и блок «Занесение инвентарного номера в форму».

Так же необходимо выбрать модель проектирования мобильного приложения.

Мы выбрали инкрементную модель проектирования [12] для разработки нашего мобильного приложения для инвентаризации. Этот подход предполагает разбиение процесса разработки на небольшие инкременты или итерации. Каждый инкремент включает в себя полный цикл разработки, начиная от анализа требований и проектирования, до разработки, тестирования и внедрения конкретной части функциональности приложения.

Инкрементная модель обладает рядом преимуществ [12; 13]:

– быстрое применение: благодаря разделению разработки на небольшие части, каждый инкремент может быть внедрен быстрее. Это позволяет получать обратную связь от пользователей раньше и оперативно вносить изменения.

– гибкость: инкрементная модель позволяет гибко реагировать на изменения требований или пожелания пользователей. Если возникает необходимость в изменении функциональности, это можно легко внести в следующий инкремент.

– лучшая прослеживаемость: каждый инкремент имеет свои требования и функциональность, что облегчает прослеживание и управление процессом разработки. Каждый этап разработки фокусируется на конкретной части приложения.

– раннее обнаружение проблем: благодаря частым итерациям и тестированию каждого инкремента можно быстро выявить и исправить проблемы или ошибки в функциональности.

В случае мобильного приложения для инвентаризации каждый инкремент может быть посвящен разработке конкретных модулей или функций.

Теперь можно перейти к выбору мобильной платформы для будущего приложения.

В настоящее время в мире имеется огромный выбор языков программирования для того, чтобы, разрабатывать мобильное программное обеспечение. Это связано с тем, что для различных мобильных устройств необходимо применять разные SDK (SDK – комплекты для разработки программного обеспечения) [7]. Как правило, это зависит от операционной системы устройства. Наиболее популярными сейчас являются IOS, Android.

Данные платформы пользуются спросом во всем мире. Они удобны в использовании и предоставляют пользователю ряд функций для работы.

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. 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. С точки зрения безопасности iOS имеет 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.

Таким образом, для разработки мобильного приложения была выбрана операционная система Android.

1.2 Обозначение требований к мобильному приложению и анализ существующих мобильных решений

Для анализа имеющихся мобильных приложений для инвентаризации выделим наиболее существенные требования к мобильному приложению для инвентаризации.

Существенным достоинством мобильного приложения является его доступность для неограниченного количества пользователей и экономия денежных средств при его использовании.

Другим достоинством мобильного приложения является то, что для сверки не нужны сторонние приборы для инвентаризации (сканеры и т. п.).

Кроме того, приложение должно содержать широкий функционал возможностей.

Таким образом, сформулируем следующие требования к мобильному приложению:

- 1) программное обеспечение должно быть свободно-распространяемым;

2) в качестве базы данных должна использоваться Google Sheets (таблица Google);

3) в приложении присутствует возможность поиска данных по Google Sheets (таблица Google);

4) в качестве сканера QR-кодов используется камера мобильного телефона;

5) в программном продукте должна быть возможность просматривать ранее отсканированные данные.

Анализ программных решений проведем в двух направлениях: приложения, реализующие использование QR-кодов и приложения, непосредственно касающиеся автоматизации процесса инвентаризации.

Рассмотрим приложения, относящиеся к первому направлению.

1. Приложение «Barcode Scanner» [27]

На сегодняшний день существует множество различных приложений для Android, позволяющих сканировать QR-коды. Одним из таких приложений является «Barcode Scanner». Особенностью данной программы является возможность просмотра отсканированного QR-кода. Помимо этого, приложение предоставляет поиск в интернете сканируемого объекта, а также возможность поделиться содержимым QR-кода. Недостатками данного приложения являются обилие рекламы, медленная фокусировка камеры. Так же в этом приложении нет базы данных, в которую сохраняются отсканированные данные. Скриншот главного экрана приложения представлен на рисунке 3.

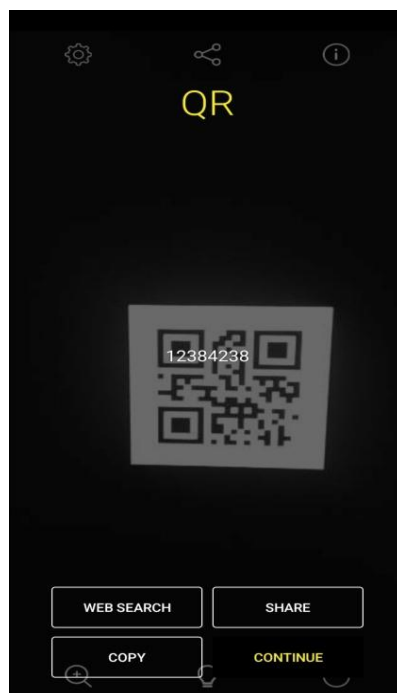


Рисунок 3 – Экран приложения «Barcode Scanner»

2. Приложение «Сканер QR и штрихкодов» [29]

Еще одно приложение для сканирования штрих-кодов и QR-кодов «Сканер QR и 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.

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. Рабочая зона приложения «Сканер QR и штрихкодов» предоставлена на рисунке 4.

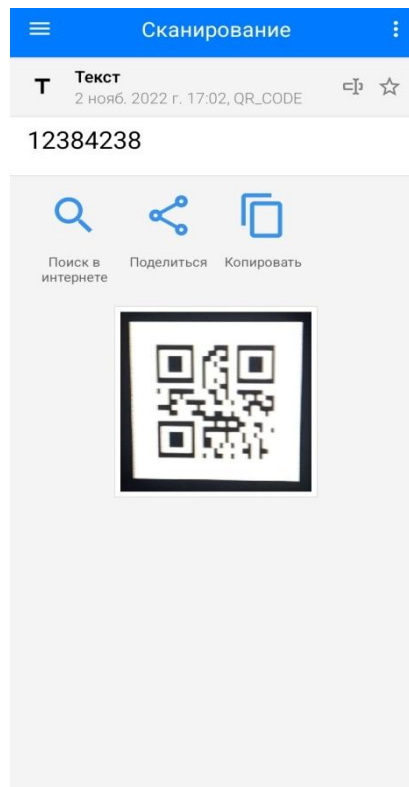


Рисунок 4 – Экран приложения «Сканер QR и штрихкодов»

После того как мы рассмотрели приложения, реализующие использование QR-кодов, перейдем ко второму типу приложений, относящихся к процессу инвентаризации.

3. Приложение «Штрих-коды и инвентарь и Excel» [32]

Перейдем к анализу приложения «Штрих-коды и инвентарь и Excel», которое уже используется под задачу инвентаризации. После сканирования QR-кода можно выбрать некоторые параметры: сколько предметов в наличии, место нахождения, цена. В приложении есть возможность поиска данных в Excel таблице. В целом это приложение хорошее, но главным недостатком является, то, что оно платное (бесплатный пробный период 60 минут, дальше придется платить). Экраны приложения «Штрих-коды и инвентарь и Excel» представлены ниже на рисунке 5.

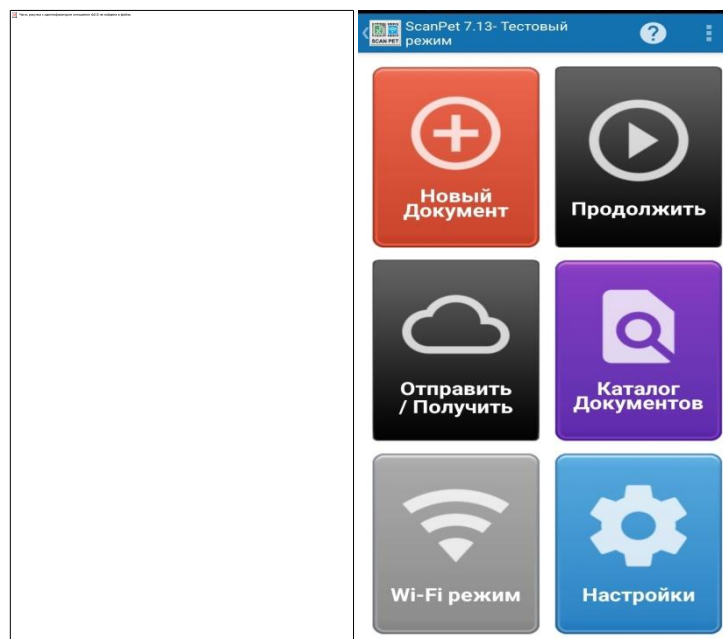


Рисунок 5 – Экраны приложения «Штрих-коды и инвентарь и Excel»

4. Приложение «Переучет и инвентаризация» [38]

Приложение «Переучет и инвентаризация» похоже на предыдущее, единственным отличием является, то, что для проведения инвентаризации придется подключать сканер. Так же, есть возможность сканировать и выбирать некоторые параметры. Это приложение так же является платным. Экраны приложения «Переучет и инвентаризация» представлены ниже на рисунке 6.

Анализ существующих решений показал, что рассмотренные данные приложения не соответствуют выделенным требованиям. После проведенного анализа было принято решение о создании собственного мобильного приложения, в котором учитываются перечисленные выше требования.

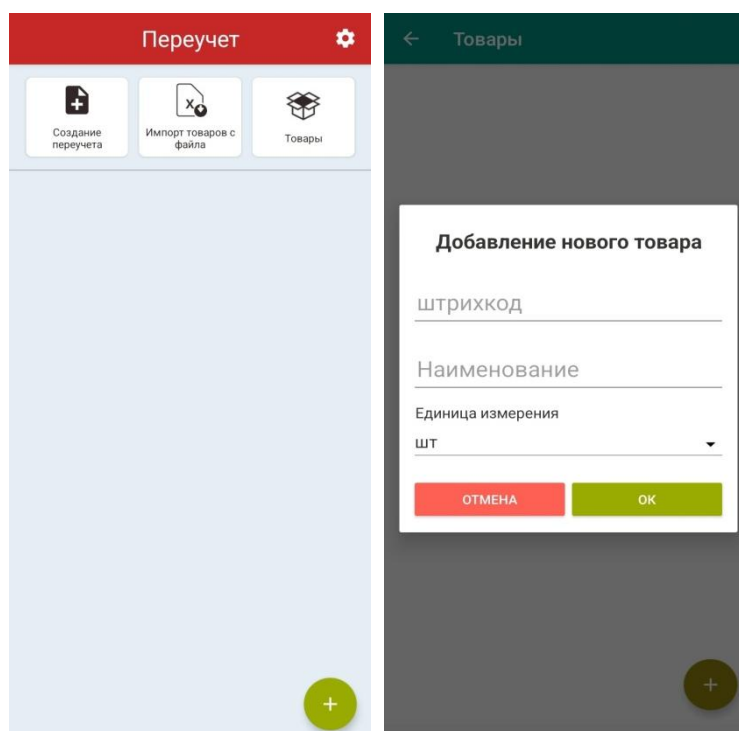


Рисунок 6 – Экран приложения «Переучет и инвентаризация»

На основе анализа источников [31], [34], [36] и выделенных выше требований [8] опишем следующие функциональные возможности разрабатываемого приложения «Инвентаризация»:

1) Основной функционал:

- возможность сканировать QR-код;
- возможность добавлять данные вручную;
- возможность просматривать данные.

2) Дополнительный функционал (не обязательный для реализации):

- экспорт данных в файл;
- возможность поиска отсканированных данных.

Так же необходимо учитывать следующие требования к пользовательскому интерфейсу:

- элементы должны отображаться корректно;
- пользовательский интерфейс должен адаптироваться к разным разрешениям экрана;

– приложение должно быть только на русском языке.

Программное решение должно выдавать корректную работу на различных устройствах под операционной системой Android, начиная с версии 5.0 и выше.

1.3 Структурная схема мобильного приложения

Структурная схема – это тип графической модели, которая изображает набор основных компонентов и отношения между ними.

Структурная схема имеет важное значение в мобильном приложении для инвентаризации по следующим причинам:

– понимание структуры данных – структурная схема помогает разработчикам и пользователям приложения понять, как данные инвентаря организованы и как они связаны друг с другом. Это позволяет эффективно работать с данными и обеспечивает понятность для пользователей приложения.

– идентификация компонентов – структурная схема помогает выделить основные компоненты инвентаря и понять их взаимосвязь. Например, можно выделить категории товаров, отдельные элементы инвентаря и связи между ними. Это позволяет легче ориентироваться в данных и управлять ими.

Основные функции, которые необходимо реализовать в приложении, вытекают из требований к разрабатываемому приложению. На рисунке 7 показана структурная схема программы, изображенная в виде ключевых модулей и взаимосвязей между ними. Эта диаграмма дает четкое представление о том, как вышеупомянутые требования реализованы в программе.

Для дальнейшей разработки необходимо построить диаграмму компонентов.

Диаграмма компонентов мобильного приложения представлена на рисунке 8 и состоит из следующих компонентов:

– мобильное приложение;

- api;
- google sheets.

База данных представлена в виде таблицы Google Sheets, которая устанавливает связь с мобильным приложением посредством API [5].

В мобильной разработке API (Application Programming Interface) является набором функций, методов и протоколов, предоставляемых операционной системой (iOS, Android и т. д.) или сторонними сервисами, которые позволяют разработчикам создавать приложения для мобильных устройств и взаимодействовать с различными функциональными возможностями устройства или внешними сервисами.

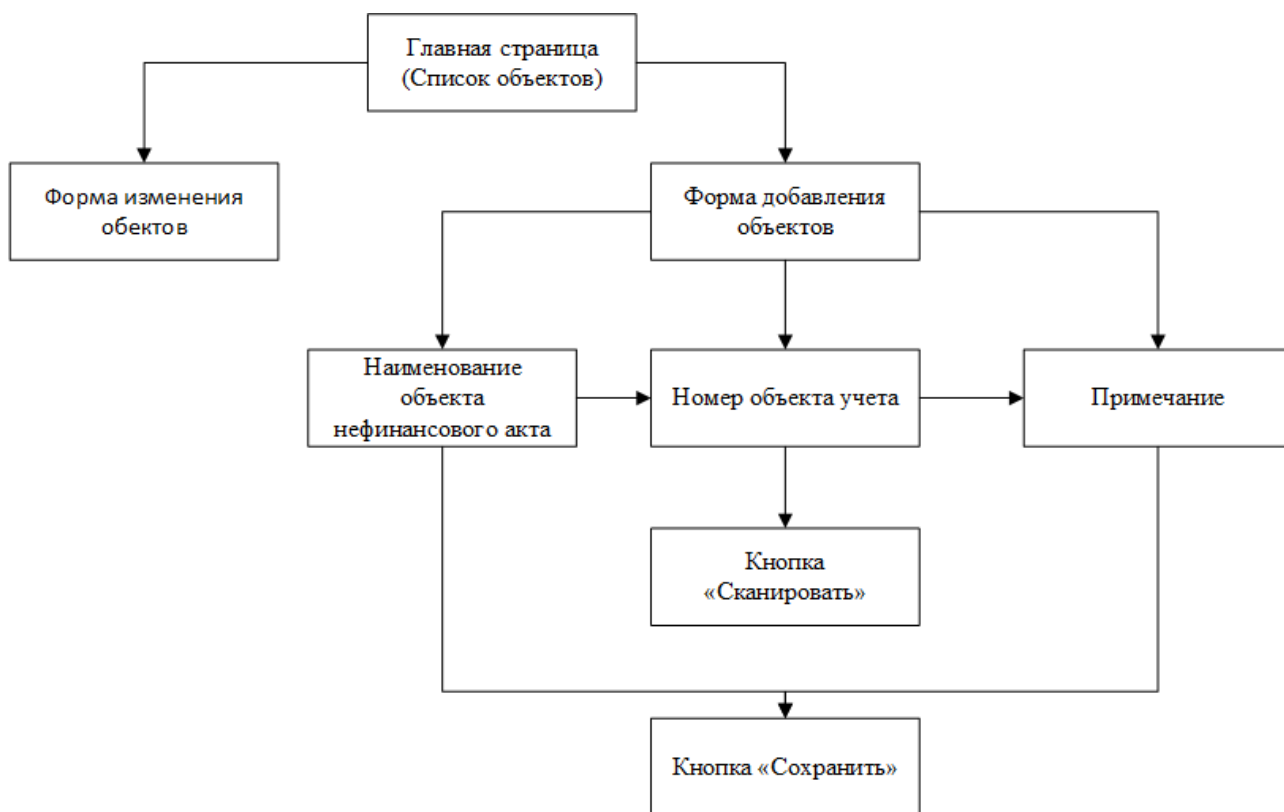


Рисунок 7 – Структурная схема приложения

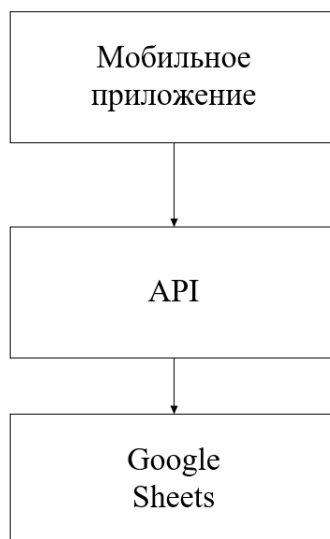


Рисунок 8 – Диаграмма компонентов

Google Sheets API работает на основе протокола HTTP и позволяет взаимодействовать с таблицами Google Sheets и их данными. Приложение может отправлять HTTP-запросы к API для выполнения различных операций, таких как чтение, запись или обновление данных в таблицах.

В результате изучения существующих мобильных платформ, для реализации была выбрана система Android. Помимо этого, проведя анализ аналогов был выделен основные требования к функционалу мобильного приложения. Так же был выделен бизнес-процесс «Инвентаризация», он был декомпозирован. Кроме того, была построена структурная схема приложения и диаграмма компонентов, которые помогут при дальнейшей разработке мобильного приложения.

2 Разработка и тестирование мобильного приложения

2.1 Выбор средств разработки

Для реализации задуманного мобильного приложения необходимо определиться со средой разработки.

1. Android Studio [25]

Android Studio – это официальная интегрированная среда разработки для создания приложений для Android, разработанная компанией Google. Она основана на популярной среде IntelliJ IDEA Java IDE и предоставляет собой полный набор инструментов и функций, помогающих разработчикам создавать высококачественные, производительные и многофункциональные приложения для Android.

Android Studio имеет удобный интерфейс с функциями редактирования макета и отладки с помощью перетаскивания, что помогает разработчикам быстро создавать и тестировать свои приложения. Она также поддерживает различные языки программирования, включая Java и Kotlin. Это позволяет разработчикам интегрироваться с другими инструментами, например такими как GitHub для контроля версий.

Одним из основных преимуществ Android Studio является наличие большого и активного сообщества разработчиков, которое предоставляет обширную документацию, учебные пособия и поддержку для разработчиков. Это облегчает разработчикам изучение и использование платформы, а также получение помощи при необходимости [7].

В целом Android Studio – это мощная и гибкая интегрированная среда разработки, которая идеально подходит для разработки приложений для Android любого размера и уровня сложности, от простых прототипов до крупномасштабных корпоративных приложений.

2. Xamarin [41]

Xamarin – это кроссплатформенная среда разработки приложений, которая позволяет разработчикам использовать единую кодовую базу для создания собственных мобильных приложений для платформ Android, iOS и Windows. Он был приобретен Microsoft в 2016 году и с тех пор интегрирован в среду разработки Visual Studio.

Одним из ключевых преимуществ Xamarin является то, что он позволяет разработчикам писать приложения на C#, популярном языке программирования, который широко используется в корпоративной разработке. Это позволяет разработчикам использовать имеющиеся у них знания о библиотеках C# и .NET для создания мобильных приложений, что может привести к сокращению времени разработки и снижению затрат.

Xamarin обладает дополнительным преимуществом, заключающимся в доступе к разнообразным API-интерфейсам и функциям для конкретных платформ. Это позволяет разработчикам создавать мобильные приложения, полностью соответствующие стандартам и внешнему виду для каждой платформы. Xamarin также предоставляет надежный набор инструментов для отладки, тестирования и развертывания приложений.

Одним из потенциальных недостатков Xamarin является то, что его может быть сложнее настроить и использовать, чем другие кроссплатформенные среды и для некоторых проектов может потребоваться дополнительная настройка. Кроме того, выполнение кода C# на мобильных устройствах может привести к снижению производительности, что в некоторых случаях может повлиять на производительность приложения.

В целом, Xamarin – это мощная и гибкая платформа, которая хорошо подходит для разработки кроссплатформенных мобильных приложений.

3. Appcelerator Titanium [26]

Appcelerator Titanium – это кроссплатформенная среда разработки приложений, которая позволяет разработчикам создавать нативные мобильные

приложения для платформ Android и iOS с использованием таких технологий веб-разработки, как HTML, CSS и JavaScript. Он предоставляет широкий спектр инструментов и функций, помогающих разработчикам быстро и эффективно создавать высококачественные и многофункциональные мобильные приложения.

Одним из ключевых преимуществ Appcelerator Titanium является то, что он позволяет разработчикам создавать кроссплатформенные мобильные приложения с единой кодовой базой, что может сэкономить время и снизить затраты на разработку. Titanium также предоставляет доступ к большой библиотеке готовых модулей и плагинов, которые разработчики могут использовать для добавления функциональности в свои приложения.

Еще одним преимуществом Titanium является то, что он предоставляет визуальный интерфейс для создания пользовательских интерфейсов, что позволяет разработчикам быстро и легко создавать собственные элементы пользовательского интерфейса. Он также предоставляет широкий спектр инструментов отладки и тестирования, которые помогают разработчикам обеспечивать качество и надежность своих приложений.

Одним из потенциальных недостатков Appcelerator Titanium является то, что он может быть более сложным в использовании, чем другие кроссплатформенные среды, особенно для разработчиков, которые не знакомы с такими технологиями веб-разработки, как HTML и CSS. Кроме того, некоторые разработчики могут обнаружить, что производительность приложений Titanium не такая высокая или отзывчивая, как у полностью нативных приложений.

В целом Appcelerator Titanium – это мощная и гибкая платформа, которая хорошо подходит для разработки кроссплатформенных мобильных приложений для различных вариантов использования, особенно для разработчиков, знакомых с технологиями веб-разработки и желающих использовать эти навыки для создания мобильных приложений.

4.Unity [37]

Unity – популярный кроссплатформенный игровой движок, который также можно использовать для разработки мобильных приложений. Он предоставляет мощную и гибкую среду разработки, которая позволяет разработчикам создавать высококачественные мобильные приложения с улучшенной 3D-графикой и физикой.

Одним из ключевых преимуществ использования Unity для разработки мобильных приложений является поддержка широкого спектра платформ, включая iOS, Android и Windows. Это позволяет разработчикам создавать приложения, которые могут работать на нескольких устройствах с единой кодовой базой, что может сэкономить время и снизить затраты на разработку.

Еще одним преимуществом Unity является большое и активное сообщество разработчиков, которое предоставляет обширную документацию, учебные пособия и поддержку. Это облегчает разработчикам изучение и использование платформы, а также получение помощи при необходимости.

Кроме того, Unity предоставляет ряд функций и инструментов, специально разработанных для мобильных приложений, таких как поддержка сенсорного ввода, ввода с помощью акселерометра и оптимизация графики и производительности для мобильных устройств. Это позволяет создавать мобильные приложения, оптимизированные для уникальных функций и требований мобильных устройств.

Одним из потенциальных недостатков использования Unity для разработки мобильных приложений является то, что его может быть сложнее настроить и использовать, чем другие платформы, особенно для разработчиков, которые не знакомы с концепциями разработки игр. Кроме того, приложения Unity могут иметь больший размер файла, чем другие мобильные приложения, что может повлиять на время загрузки и требования к хранилищу.

В целом Unity – это мощная и гибкая платформа для разработки мобильных приложений, особенно для разработчиков, которые хотят создавать захватывающие 3D-приложения или уже знакомы с концепциями разработки игр.

Резюмируя, стоит отметить, что Android Studio – универсальная среда разработки, так как позволяет оптимизировать работу будущих приложения для работы не только на смартфонах, но и на планшетах, портативных персональных компьютерах, которые работают на основе рассматриваемой операционной системы .

Таким образом, Android Studio была выбрана для написания мобильного приложения по нескольким причинам:

- небольшой опыт в написании мобильных приложений на языках Java/Kotlin;
- гибкость среды разработки;
- большое количество обучающих ресурсов;
- поддержка компании Google.

2.2 Разработка модулей мобильного приложения

Приложение состоит из нескольких страниц, написанных на Kotlin с использованием Google Apps Script. Пример структуры страницы приложения «Инвентаризация» представлен на рисунке 9. Основными файлами являются ReadActivity и WriteActivity, код для каждого из файлов находится в приложениях Б, В соответственно.

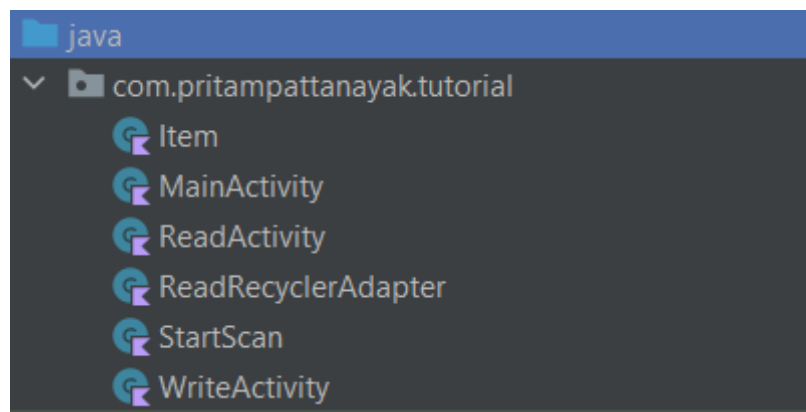


Рисунок 9 – Страница «Главная страница»

Остановимся на каждом модуле подробнее.

2.2.1 Модуль сохранения объекта

Модуль предназначен для сохранения данных об объекте инвентаризации с использованием скриптов Google Apps Script.

Модуль сохранения объектов в Google Apps Script – это функция, которая позволяет пользователям хранить и извлекать структурированные данные в своих таблицах Google с помощью редактора сценариев в Google Sheets или любом другом продукте Google Apps. Этот модуль предоставляет удобный способ сохранения и загрузки объектов или сложных структур данных без необходимости ручного разбора и форматирования. Подробнее с этим модулем можно ознакомиться в видеоуроке [30].

Модуль использует встроенные методы, предоставляемые скриптом Google Apps, такие как `PropertiesService` и `SpreadsheetApp`, для обработки хранения и извлечения данных. Он следует парному подходу «ключ-значение», когда объект или структура данных сериализуются в формате JSON, а затем сохраняются как значение, связанное с определенным ключом.

Чтобы использовать модуль сохранения объектов, необходимо включить скрипты в проект Apps Script. Опишем основные скрипты, необходимые для нашего проекта в рассматриваемом модуле.

1. Function doGet.

Function doGet в скрипте Google Apps – это обработчик событий, который автоматически выполняется, когда к URL-адресу веб-приложения отправляется запрос GET. Эта функция позволяет обрабатывать входящий запрос GET и генерировать ответ для отправки обратно запрашивающей стороне.

Function doGet принимает один параметр, request, который представляет собой HTTP-запрос, сделанный веб-приложению. Этот параметр представляет собой объект, который содержит различные свойства и методы для доступа к информации о запросе, такой как заголовки запроса, параметры запроса и IP-адрес отправителя запроса.

Внутри данного скрипта можно выполнять такие задачи, как извлечение данных из внешних источников, манипулирование данными, создание динамического контента или взаимодействие с другими службами Google. Также имеется возможность настроить логику в соответствии с вашим конкретным вариантом использования и требованиями.

Структура листинга Function doGet для нашего проекта:

```
function doGet(e){  
  post_value(e);  
}
```

2. Function post_value

Предоставленная post_value функция как таковая не имеет какой-либо определенной функциональности или реализации. Его цель и поведение будут зависеть от того, как вы собираетесь его использовать, и от конкретной логики, которую вы реализуете в функции.

Как правило, именованная функция post_value, которая принимает request параметр, обычно используется для обработки запроса POST в веб-приложении

или конечной точке API. Обычно данные, отправленные в теле запроса, обрабатываются, и выполняются соответствующие операции на основе этих данных.

В данном случае функция выполняет следующие действия:

1. Открывает электронную таблицу Google Sheets на основе предоставленного URL-адреса.
2. Извлекает лист с именем «barcode» из электронной таблицы.
3. Получает текущую дату и время и разбивает их на отдельные переменные.
4. Извлекает значения barcode из параметров запроса.
5. Добавляет новую строку к листу «barcode» с текущей датой, временем, штрих-кодом и другими параметрами.
6. Подготавливает объект ответа со строкой JSON».
7. Возвращает ответ, используя `ContentService.createTextOutput` [30].

Таким образом, этот код получает запрос POST с параметрами `barcode` и `country`. Затем он добавляет эти значения вместе с текущей датой и временем в указанную электронную таблицу Google Sheets. Наконец, он отвечает объектом JSON, указывающим на успех операции.

Листинг Function `post_value` для нашего проекта:

```
function post_value(request){  
  //Edit this link to your google sheets  
  const  
  ss=SpreadsheetApp.openByUrl("https://docs.google.com/spreadsheets/d/1QGqd6sQg  
  XKZqzAvXhS4MLtu4M2UbUuULXd9ATmDqzj8/edit?usp=sharing");  
  const sheet = ss.getSheetByName("barcode");  
  const d = new Date();  
  const currentTimeArray = d.toLocaleString().split(", ");  
  const date = currentTimeArray[0]  
  const time = currentTimeArray[1]
```

```

const barcode = request.parameter.barcode;
sheet.appendRow([date,time,barcode,country]);
result = JSON.stringify({
  "result": "OK"
});
return ContentService
.createTextOutput("(" + result + ")")
.setMimeType(ContentService.MimeType.JAVASCRIPT);
}

```

2.2.2 Модуль чтения объекта

Модуль предназначен для чтения данных об объекте инвентаризации с использование скрипов Google Apps Script.

Модуль чтения объекта с помощью скрипта Google Apps из Google Sheets позволяет получать данные из этой оболочки или внешней ячейки в Google Sheets и вставлять эти данные в приложение. Для этого необходимо использовать методы, доступные в объекте Range класса Sheet в Google Apps Script.

– Function doGet

GET-запрос (HTTP GET request) – это один из методов HTTP-протокола, который используется для получения данных с сервера. В контексте Google Apps Script и Google Sheets, функция doGet() является обработчиком GET-запросов и позволяет взаимодействовать с данными в таблице Google Sheets при получении GET-запроса.

В данном случае функция выполняет следующие действия:

1. Код объявляет переменную date и описывает ее дату и время.

2. Код создает переменную `id`, которая представляет собой реализующий элемент идентификатора. Он является следствием включения значения «`item`» и последней строки в листе с помощью метода `getLastRow()`.

3. Далее код получает значение параметра `Name` из объекта `e.parameter` и сохраняет его в настройках `itemName`.

4. Аналогичным образом код получает значения `ID` и `Prices` сохраняет их в `itemid` и `itemprices` соответственно.

5. Код использует метод `appendRow()` объекта `sheet` для добавления новой строки в таблицу.

6. В конце создается и возвращается значение объекта `ContentService.createTextOutput()`.

Таким образом, функция `doPost()` принимает POST-запрос, из набора параметров запроса, сохраняет их в таблице Google Sheets.

– Function `doPost`

В данном случае функция выполняет следующие действия:

1. Код объявляет переменную `record` для хранения записей (данных) в виде объекта.

2. Затем код получает диапазон значений из листа. Он использует методы `getRange()` для определения диапазона ячеек и `getValues()` для получения значений этого диапазона. Полученные значения сохраняются в переменной `rows`.

3. Далее код итерируется по каждой строке в переменной `rows` и формирует объект `record` для каждой строки. Значения из определенных столбцов сохраняются в свойства объекта `record` с соответствующими именами, такими как `'itemName'`, `'itemid'`, `'itemprimech'`.

4. Сформированные объекты `record` добавляются в массив `data`.

5. Затем код создает свойство `bookList` в объекте `record` и присваивает ему значение массива `data`, содержащего записи.

6. Переменная `result` содержит JSON-строку, полученную из объекта `record` с помощью метода `JSON.stringify()`.

7. В конце создается и возвращается значение объекта `ContentService.createTextOutput()` содержащий JSON-строку `result`.

Таким образом, функция `doGet()` получает GET-запрос, читает данные из таблицы Google Sheets, формирует объект JSON, содержащий эти данные, и возвращает его в виде JSON-строки.

2.2.3 Модуль сканирования объекта

Модуль сканирования объекта представляет собой часть программного обеспечения, которая позволяет сканировать и получать информацию об объектах с использованием специализированных устройств или технологий. В контексте Google Apps Script модуль сканирования объекта может использоваться для сканирования и считывания информации с помощью штрихкодов, QR-кодов или других типов кодированных меток.

Модуль сканирования объекта на основе Google Apps Script может включать следующие функциональности:

1. Интеграция с устройствами сканирования: модуль может взаимодействовать с физическими сканерами или камерами устройств для получения изображений или данных с объектов.

2. Обработка кодированных меток: модуль может анализировать полученные данные и распознавать кодированные метки, такие как штрихкоды или QR-коды. Это может включать декодирование информации, содержащейся в кодированных метках.

3. Интеграция с базой данных: модуль может использовать данные, полученные из сканирования объектов, и связывать их с соответствующими записями в базе данных или таблице Google Sheets. Например, можно сохранять

информацию о сканированных объектах, их характеристиках или местоположении.

4. Обработка и анализ данных: модуль может предоставлять функциональность для обработки и анализа полученных данных. Например, можно выполнять поиск объектов по определенным критериям или агрегировать информацию о сканированных объектах.

5. Интеграция с другими приложениями: модуль сканирования объекта может интегрироваться с другими приложениями или сервисами, такими как Google Sheets, для автоматического обновления данных или выполнения определенных действий на основе результатов сканирования.

Общий функционал модуля сканирования объекта может сильно различаться в зависимости от конкретных требований и возможностей разработанного решения на основе Google Apps Script. С листингом модуля сканирования можно ознакомиться в приложении Г.

Таким образом, мы разработали мобильное приложение для облегчения проведения инвентаризации, которое позволяет пользователю проводить сканирование QR-кодов для учета и отслеживания инвентаря. Приложение предоставляет возможность использовать камеру мобильного устройства для сканирования QR-кодов, привязанных к каждому предмету инвентаря. Отметим, что приложение работает не только с QR-кодами, но и любой текстовой информацией, которую можно ввести в поле «Наименование объекта нефинансового акта». После сканирования QR-кода, приложение сохраняет информацию о предмете, такую как его идентификатор, название или другие дополнительные данные, связанные с этим предметом.

Мы разработали приложение, удобное для использования на предприятиях, или в любой ситуации, где требуется провести инвентаризацию и отслеживание предметов. Наше приложение позволяет автоматизировать и упростить процесс инвентаризации, сократить время и устранить возможность ошибок при ручном учете.

2.3 Тестирование мобильного приложения

Тестирование мобильного приложения для инвентаризации с использованием сканирования QR-кода является важной частью разработки. По мнению А. Красивской [17] и А. Коротеевой [17], тестирование позволяет убедиться в функциональности, надежности и соответствии требованиям приложения. Вот несколько шагов, которые можно выполнить при тестировании нашего приложения:

1. Функциональное тестирование: необходимо убедиться, что основные функции приложения работают должным образом. Необходимо проверить следующие аспекты:

- сканирование QR-кода с использованием камеры мобильного устройства;

- чтение и распознавание информации из QR-кода;

- сохранение информации об инвентаре, полученной из сканированного QR-кода;

- отображение сохраненной информации о предметах инвентаря.

2. Тестирование различных типов QR-кодов: проверить, как приложение обрабатывает различные типы QR-кодов. Необходимо удостовериться, что оно может справиться с различными форматами QR-кодов и корректно распознает информацию из них.

3. Тестирование ввода данных: проверить, как приложение обрабатывает различные варианты ввода данных. Необходимо протестировать ввод недопустимых символов, длинных строк, специальных символов и других возможных ситуаций, чтобы убедиться, что приложение корректно обрабатывает их и не вызывает сбоев.

4. Тестирование стабильности: проверить стабильность и производительность приложения. Запустить приложение в различных условиях сети и проверьте его реакцию на низкую скорость интернета или отсутствие

сети. Также проверить, как приложение работает при большом объеме данных и множественных сканированиях.

5. Тестирование совместимости: убедиться, что приложение работает корректно на различных моделях мобильных устройств и операционных системах. Протестировать его на разных версиях Android и на различных разрешениях экрана, чтобы убедиться, что пользователи с разными устройствами могут использовать его без проблем.

6. Тестирование пользовательского интерфейса: оценить пользовательский интерфейс приложения, проверить, что он интуитивно понятен и легко используется. Протестировать навигацию, кнопки, визуальное оформление и отзывчивость интерфейса при взаимодействии с приложением.

В таблице 1 представлены результаты проведенных функциональных тестов мобильного приложения «Инвентаризация».

Таблица 1 – Результаты функциональных тестов

Тест	Описание	Результат
Тест сканирования QR-кода	Приложение корректно сканирует QR-коды с помощью камеры мобильного устройства. Запустить приложение, навести камеру на QR-код и убедиться, что содержимое QR-кода успешно распознается и отображается на экране.	Тест пройден без ошибок
Тест сохранения информации об инвентаре	Проверить, что информация о предмете инвентаря корректно сохраняется после сканирования QR-кода. Отсканировать QR-код, содержащий информацию о предмете, и убедитесь, что соответствующие данные, такие как название, описание идентификатор, сохраняются в базе данных.	Тест пройден без ошибок

Окончание таблицы 1

Тест	Описание	Результат
Тест редактирования информации об объекте инвентаризации:	Проверить возможность редактирования информации о объекте инвентаризации. Запустить приложение, отсканировать QR-код, а затем изменить некоторые данные, такие как название или описание, и сохраните изменения. Убедиться, что измененная информация корректно сохраняется и отображается при последующем сканировании того же QR-кода.	Тест пройден без ошибок
Тест обработки некорректных QR-кодов:	Проверить, что приложение обрабатывает некорректные или поврежденные QR-коды. Попробовать сканировать QR-код с ошибкой или искаженным изображением и убедиться, что приложение показывает соответствующее сообщение об ошибке или неудачном сканировании.	Тест пройден без ошибок
Тест интеграции с базой данных	Приложение использует базу данных для хранения информации об объектах инвентаризации, протестировать интеграцию с базой данных. Убедиться, что данные корректно сохраняются, извлекаются и обновляются в базе данных после сканирования QR-кода.	Тест пройден без ошибок

В таблице 2 представлены результаты проведенных тестов на чтение различных типов QR-кодов.

Таблица 2 – Результаты тестов на чтение различных типов QR-кодов

Тест	Результат
Тест на стандартный текст	Тест пройден без ошибок
Тест на URL-адрес	Тест пройден без ошибок
Тест на серийный номер	Тест пройден без ошибок
Тест на информацию об объекте инвентаризации	Тест пройден без ошибок

В таблице 3 представлены результаты проведенных тестов на ввод данных.

Таблица 3 – Результаты тестов ввода данных

Тест	Описание	Результат
Тест на ввод корректных данных	Проверить, что приложение корректно обрабатывает ввод правильных данных. Ввести корректные данные, такие как название предмета, описание, идентификатор и другую необходимую информацию. Убедиться, что приложение сохраняет и отображает введенные данные без ошибок	Тест пройден без ошибок
Тест на ввод обязательных данных	Проверить, что приложение проверяет и обрабатывает обязательные поля данных. Попробовать ввести только часть необходимых данных и убедиться, что приложение предупреждает о незаполненных полях и не позволяет сохранить объект инвентаря без всех обязательных данных	Тест пройден без ошибок
Тест на ввод специальных символов	Проверить, как приложение обрабатывает ввод специальных символов, таких как кавычки, косые черты или знаки препинания. Убедиться, что приложение правильно экранирует или обрабатывает специальные символы, чтобы избежать проблем с сохранением и отображением данных	Тест пройден без ошибок
Тест на ввод повторяющихся данных	Проверить, как приложение обрабатывает ввод повторяющихся данных, таких как идентификаторы предметов инвентаря. Попробовать сохранить несколько объектов с одинаковыми идентификаторами и убедиться, что приложение предупреждает о конфликте и не позволяет сохранить повторяющиеся данные	Тест пройден без ошибок

Окончание таблицы 3

Тест	Описание	Результат
Тест на ввод некорректных данных	Проверить, что приложение обрабатывает некорректный ввод данных. Ввести данные с ошибками, такие как некорректный формат идентификатора или недопустимые символы в полях данных. Убедиться, что приложение отображает соответствующие сообщения об ошибке и не позволяет сохранить некорректные данные	Тест пройден без ошибок

В таблице 4 представлены результаты проведенных тестов проверки стабильности мобильного приложения.

Таблица 4 – Результаты тестов проверки стабильности мобильного приложения

Тест	Описание	Результат
Тест на длительную непрерывную работу	Запустить приложение и отсканировать несколько QR-кодов подряд без перезапуска приложения. Убедиться, что приложение остается стабильным и продолжает корректно обрабатывать сканирование QR-кодов даже после длительного использования	Тест пройден без ошибок
Тест на переключение между сетями	Проверить стабильность приложения при переключении между различными сетями, такими как Wi-Fi и мобильная сеть. Переключится между сетями во время сканирования QR-кодов и убедиться, что приложение сохраняет стабильное соединение и продолжает работать надежно	Тест пройден без ошибок

Окончание таблицы 4

Тест	Описание	Результат
Тест на многопоточность	Проверить стабильность приложения при одновременном сканировании нескольких QR-кодов. Запустить несколько экземпляров приложения на разных устройствах или эмуляторах и отсканировать QR-коды одновременно. Убедиться, что все экземпляры приложения работают стабильно и корректно обрабатывают сканирование	Тест пройден без ошибок

В таблице 5 представлены результаты проведенных тестов на совместимость.

Таблица 5 – Результаты тестов проверки совместимости мобильного приложения

Тест	Результат
Тестирование на разных версиях операционной системы начиная от версии Android 5.0	Тест пройден без ошибок
Тестирование на разных разрешениях экрана начиная от 640×1136	Тест пройден без ошибок
Тестирование на разных устройствах	Тест пройден без ошибок

В таблице 6 представлены результаты проведенных тестов проверки пользовательского интерфейса мобильного приложения.

Таблица 6 – Результаты тестов проверки пользовательского интерфейса мобильного приложения

Тест	Результат
Тест на отображение и компоновку элементов	Тест пройден без ошибок
Тест на адаптивность и разрешение экрана	Тест пройден без ошибок
Тест на локализацию	Тест пройден без ошибок

2.4 Руководство пользователя

2.4.1 Установка и запуск

Для использования приложения необходимо установить и запустить *apk*-файл на мобильном устройстве. Нужно отметить, что версия операционной системы должна быть Android 5.0 или старше.

Для того, чтобы запустить приложение у себя на телефоне, требуется просто нажать на установленную иконку.

2.4.2 Работа с приложением

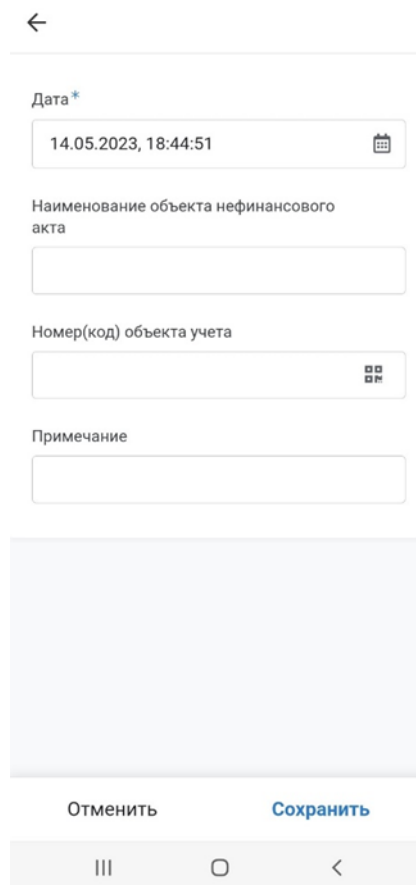
На стартовой странице, которая представлена на рисунке 10 представлены уже внесенные в Google Sheets объекты инвентаризации.



1 – кнопка поиска объектов инвентаризации; 2 –кнопка обновления списка объектов инвентаризации; 3 – кнопка добавления объекта инвентаризации

Рисунок 10 – Главная страница приложения

Для того чтобы добавить новый объект инвентаризации необходимо нажать на кнопку «+». Далее откроется форма внесения нового объекта инвентаризации в нашу базу данных. На рисунке 11 представлена форма для нового объекта.



The screenshot shows a mobile application interface for adding a new inventory object. At the top left, there is a back arrow icon. Below it, the form contains the following fields:

- Дата***: A date and time field with the value "14.05.2023, 18:44:51" and a calendar icon on the right.
- Наименование объекта нефинансового акта**: A text input field.
- Номер(код) объекта учета**: A text input field with a QR code icon on the right.
- Примечание**: A text input field.

At the bottom of the form, there are two buttons: "Отменить" (Cancel) and "Сохранить" (Save). Below the buttons is a standard Android navigation bar with three icons: a home button (three vertical lines), a back button (a circle), and a forward button (a left-pointing arrow).

Рисунок 11 – Верхнее меню приложения

Далее заносим данные и сохраняем объект инвентаризации. Проверяем, что наш объект сохранился и в приложении, и в нашей базе данных. Рисунок 12 скриншот только что добавленного объекта инвентаризации в приложении.

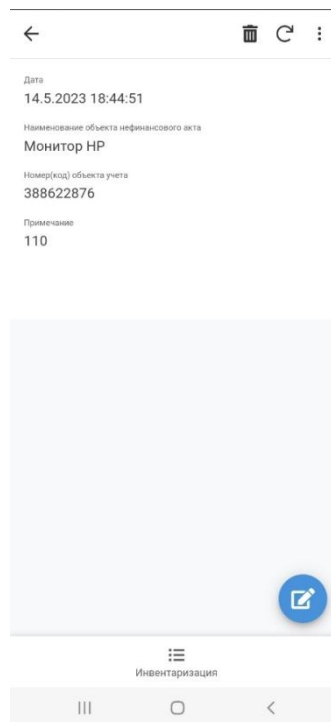


Рисунок 12 – Отображение нового объекта инвентаризации в приложении

На рисунке 13 представлен фрагмент таблицы с добавленным объекта инвентаризации в базе данных Google Sheets.

	A	B	C	D
1				
2				
3				
4				
	Дата	Наименование объекта нефинансового акта	Номер(код) объекта учета	Примечание
5				
6	09.11.2022 17:2	Видеорегистратор цифровой триплексный 4-х канальный	41013400003471	212
7	09.11.2022 17:2	Источник бесперебойного питания	21013400007717	101
8	13.11.2022 14:2	Источник бесперебойного питания	21013400007719	102
9	13.11.2022 14:3	Источник бесперебойного питания	21013400007710	103
10	13.11.2022 14:3	Источник бесперебойного питания	21013400007715	104
11	13.11.2022 14:3	Источник бесперебойного питания	21013400007716	105
12	13.11.2022 14:3	Источник бесперебойного питания	21013400007711	106

Рисунок 13 – Отображение нового объекта инвентаризации в таблице Google

Реализованное приложение обладает следующим функционалом:

- возможность сканировать QR-код;
- возможность добавлять данные вручную;
- возможность удалять данные;
- возможность поиска данных;
- возможность просматривать данные.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы были реализованы все поставленные задачи, а именно: проведен анализ предметной области, были изучены и проанализированы аналоги приложений для инвентаризации, выявлены их недостатки, за счет этого были сформулированы требования к разрабатываемому продукту.

Для разработки приложения под Android был выбран оптимальный для данного случая набор инструментов, в который вошли: язык программирования Kotlin, Google App Script, база данных Google Sheets, а также сторонние библиотеки для Android.

Во время разработки мобильного приложения были изучены основы языка программирования Kotlin, подробно изучены скрипты Google, а также взаимодействие клиента приложения с Android API.

Так же были выделены функциональные требования к мобильному приложению.

В ходе написания выпускной квалификационной работы были участия в конференциях и конкурсах. Так же опубликовано несколько статей, представленных на следующих конференциях и конкурсах:

– VI Всероссийская научно-практическая конференция преподавателей, учителей, студентов и молодых ученых «Актуальные проблемы преподавания дисциплин естественнонаучного цикла» (г. Лесосибирск, ЛПИ – филиал СФУ, 7–12 ноября 2022 г., участие).

– VII Международная научно-практическая конференция «Молодёжь, наука, образование: Актуальные вопросы, достижения и инновации» (г. Пенза, МЦНС «Наука и Просвещение», 12 апреля 2023 г., участие).

– Всероссийский молодежный научный форум «Современное педагогическое образование: теоретический и прикладной аспекты» (г. Лесосибирск, ЛПИ – филиал СФУ, 11 апреля 2023 г., I место).

– II Всероссийский конкурс научных работ «Молодёжный научный потенциал» (г. Лесосибирск, ЛПИ – филиал СФУ, 12 апреля 2023 г., участие).

Разработанное приложение было апробировано в организации ЛПИ - филиале СФУ, где показало себя отлично. В дальнейшем предполагается выгрузка приложения в Google Play.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Белоусова, С. Жизненный цикл программных систем / С. Белоусова. – 2022. – URL: <https://clck.ru/34f9rf> (дата обращения: 25.04.2023).
2. Бурнет, Э. Привет, Android! Разработка мобильных приложений / Э. Бурнет. – Санкт-Петербург: Питер – 2012. – 253 с.
3. Гайдукова, Е. Базовые знания по IDEF0 / Е. Гайдукова. // Comidware. – 2022. – URL: <https://clck.ru/34f2Gv> (дата обращения: 04.04.2023)
4. Гриффитс, Д. Head First. Kotlin / Д. Гриффитс. – Санкт-Петербург: Питер – 2020. – 646 с.
5. Гриффитс, Д. Head First. Программирование для Android / Д. Гриффитс. – Санкт-Петербург: Питер – 2018. – 912 с.
6. Дарвин, Я. Ф. Android. Сборник рецептов: задачи и решения для разработчиков приложений / Я. Ф. Дарвин. – Москва: Диалектика, 2019. – 768 с.
7. Дейтел, П. Android для разработчиков / П. Дейтел. – Санкт-Петербург: Питер, 2016. – 560 с.
8. Елизарова, Е. П. Анализ мобильных приложений для инвентаризации / Е. П. Елизарова. // Актуальные проблемы преподавания дисциплин естественнонаучного цикла: тезисы докладов VI Всероссийской научно-практической конференции преподавателей, учителей, студентов и молодых ученых, Лесосибирск, 14–15 ноября 2022 года / Сибирский федеральный университет – Красноярск, 2022. – С. 12–14.
9. Елизарова, Е. П. Разработка мобильных приложений на Kotlin / Е. П. Елизарова. // Молодёжь, наука, образование: Актуальные вопросы, достижения и инновации, Пенза, 12 апреля 2022 года / МЦНС «Наука и просвещение» – Пенза, 2023. – С. 55–57.
10. Инвентаризация // «Авдеев и Ко»: аудиторские и бухгалтерские услуги [сайт]. – 2023. – URL: <https://clck.ru/GaToD> (дата обращения: 06.04.2023)
11. Инвентаризация для руководства // ФИН-АУДИТ [сайт]. – 2019. – URL: <https://clck.ru/34iCrX> (дата обращения: 06.04.2023)

12. Инкрементная модель // Байтэкс – 2017. – URL: <https://clck.ru/34cBJ5>
(дата обращения: 08.04.2023)
13. Инкрементная модель // JavaTPoint: [сайт]. – 2018. – URL: <https://clck.ru/34eztP> (дата обращения: 08.04.2023)
14. Кинзябулатов, Р. IDEF0. Знакомство с нотацией и пример использования / Р. Кинзябулатов // Тринион: [сайт]. – 2022. – URL: <https://clck.ru/ec648> (дата обращения: 09.04.2023)
15. Кинзябулатов, Р. В чём различия нотаций IDEF0, DFD и BPMN2.0 / Р. Кинзябулатов. / Тринион : [сайт]. – 2023. – URL: <https://clck.ru/ec648> (дата обращения: 04.04.2023)
16. Колисниченко, Д.Н. Самоучитель. Программирование для Android / Д. Н. Колисниченко. – Москва: БХВ, 2021. – 288 с.
17. Красивская, А., Коротеева, А. Тестирование мобильных приложений инструкция для начинающих / А. Красивская, А. Коротеева. // Блог Яндекс.Практикума – 2022. – URL: <https://clck.ru/34f2cM> (дата обращения: 25.04.2023).
18. Куликов, С. С. Тестирование программного обеспечения. Базовый курс / С. С. Куликов. – Беларусь: Минск, 2020. – 314с.
19. Лесосибирский педагогический институт – филиал СФУ: официальный сайт. – Лесосибирск, 2023. – URL: <https://ipi.sfu-kras.ru/sveden/struct> (дата обращения: 25.04.2023).
20. Машнин, Т. С. Разработка Android – приложений в деталях / Т. С. Машнин. – Екатеринбург: Издательские решения, 2021. – 400 с.
21. Мейкл, Х. Инвентаризация с системой сканирования штрих-кода в Google Sheets/ Х. Мейкл. // SheetgoBlog – 2019. – URL: <https://clck.ru/34f9FP>
(дата обращения: 07.04.2023)
22. Министерство финансов Российской Федерации приказ: Об утверждении Методических указаний по инвентаризации имущества и финансовых обязательств № 49: [принят Министерством финансов Российской Федерации]

Федерации 13 сентября 1995 года: редактирован Министерством финансов Российской Федерации 8 ноября 2010 года]. – Москва: Финансовая газета.

23. Нотация IDEF0 // BisnessStudio – 2019. – URL: <https://clck.ru/ec648> (дата обращения: 18.04.2023)

24. Федеральные законы и акты Правительства Федеральным Законом: «О бухгалтерском учете»: [Принят Государственной Думой 22 ноября 2011 года: Одобрен Советом Федерации 29 ноября 2011 года]

25. Android Studio: официальный сайт. – 2023. – URL: <https://clck.ru/Gjs9Y> (дата обращения: 25.04.2023).

26. Appcelerator Titanium — новая RIA платформа / Хабр [сайт]. – 2023. – URL: <https://clck.ru/34jSyH> (дата обращения: 3.05.2023).

27. Cognex Corporation Приложение «Barcode Scanner» // Cognex – URL: <https://clck.ru/34iHmg> (дата обращения: 1.05.2023).

28. Flutter: официальный сайт. – 2023. – URL: <https://flutter.dev/learn> (дата обращения: 1.05.2023).

29. Gamma Play Приложение «Сканер QR и штрих-кодов» // Gamma Play – URL: <https://clck.ru/Em3N7> (дата обращения: 1.05.2023).

30. Pattanayak, P. Google Drive (Google sheet) as Realtime Database as Firebase | Android Project| Kotlin // P. Pattanayak. – 2020. – URL: <https://clck.ru/34f9eH> (дата обращения: 1.05.2023).

31. OrcaScan How to scan barcodes into Google Sheets // OrcaScan – 2020. – URL: <https://clck.ru/34f9MG> (дата обращения: 1.05.2023).

32. Redondo, M. Приложение «Штрих-коды и инвентарь и Excel» / M. Redondo. – 2019. – URL: <https://clck.ru/34iJ8f> (дата обращения: 1.05.2023).

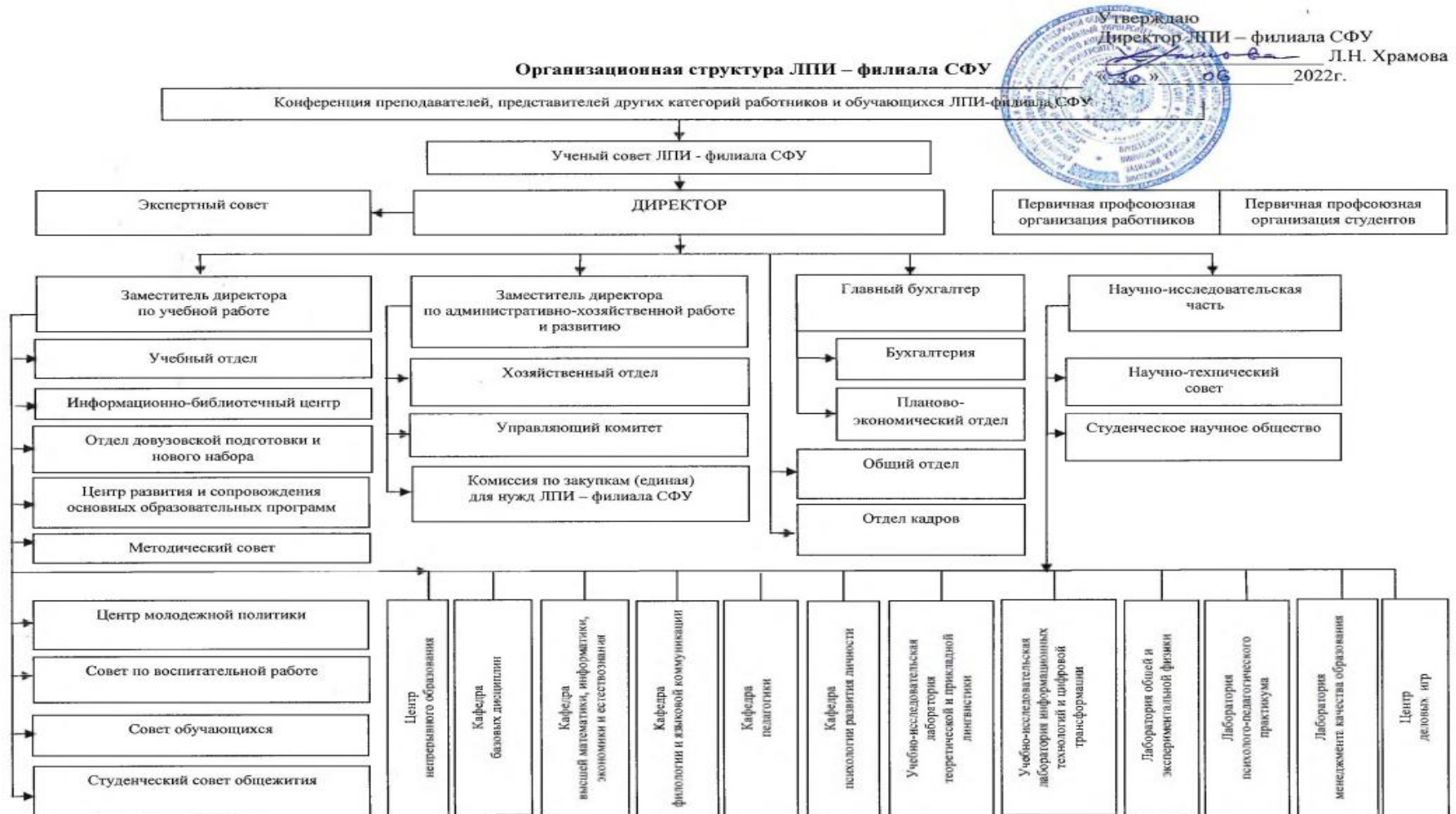
33. KB CODER QR Code Scanner in Android Studio | Kotlin | Android Tutorials / KB CODER – 2021. – URL: <https://clck.ru/34f9gv> (дата обращения: 1.05.2023).

34. QR code into a line in Google sheets – 2020. – URL: <https://clck.ru/34f9Nc> (дата обращения: 3.05.2023).

35. React Native: официальный сайт. – 2023. – URL: <https://reactnative.dev/docs/getting-started> (дата обращения: 3.05.2023).
36. StartAndroid [сайт]. – 2023. – URL: <https://startandroid.ru/ru/> (дата обращения: 3.05.2023).
37. Unity: официальный сайт. – 2023. – URL: <https://clck.ru/Tf2J> (дата обращения: 3.05.2023).
38. Horodetskyi, V. Приложение «Переучет и инвентаризация» / V. Horodetskyi. – 2020. – URL: <https://clck.ru/34iKHr> (дата обращения: 3.05.2023).
39. Visual Studio официальный сайт. – 2023. – URL: <https://clck.ru/34jT5S> (дата обращения: 3.05.2023).
40. XCode официальный сайт. – 2023. – URL: <https://clck.ru/34jT5w> (дата обращения: 3.05.2023).
41. Xamarin официальный сайт. – 2023. – URL: <https://clck.ru/34WKDM> (дата обращения: 3.05.2023).

ПРИЛОЖЕНИЕ А

Организационная схема ЛПИ – филиала СФУ



ПРИЛОЖЕНИЕ Б

Листинг модуля чтения

```
package com.pritampattanayak.tutorial
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import android.widget.ProgressBar
import android.widget.RelativeLayout
import android.widget.Toast
import androidx.recyclerview.widget.LinearLayoutManager
import androidx.recyclerview.widget.RecyclerView
import com.android.volley.Request
import com.android.volley.Response
import com.android.volley.toolbox.JsonObjectRequest
import com.android.volley.toolbox.Volley
class ReadActivity : AppCompatActivity() {

    lateinit var readProgressLayout:RelativeLayout
    lateinit var readProgressBar:ProgressBar
    lateinit var recyclerView:RecyclerView
    lateinit var layoutManager: LinearLayoutManager
    lateinit var recyclerAdapter: ReadRecyclerAdapter
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_read)
        readProgressLayout=findViewById(R.id.readProgressLayout)
        readProgressBar=findViewById(R.id.readProgressBar)
```

```

recyclerView=findViewById(R.id.recyclerView)
layoutManager= LinearLayoutManager(this)
val bookList= arrayListOf<Item>()
val queue=Volley.newRequestQueue(this)
val
url="https://script.google.com/macros/s/AKfycbz2fZnhZ7PE8IsMcMITAifob2Unqfx
07nyBMqy2GA/exec"
val jsonObjectRequest=object :JsonObjectRequest(
    Request.Method.GET,url,null,
    Response.Listener {
        readProgressLayout.visibility=View.GONE
        readProgressBar.visibility=View.GONE
        val data=it.getJSONArray("bookList")
        for(i in 0 until data.length()){
            val bookJsonObject=data.getJSONObject(i)
            val bookObject=Item(
                bookJsonObject.getString("itemName"),
                bookJsonObject.getString("itemAuthor"),
                bookJsonObject.getInt("itemPrice")
                //bookJsonObject.getString("itemRating")
            )
            bookList.add(bookObject)
        }
        recyclerViewAdapter= ReadRecyclerAdapter(this,bookList)
        recyclerView.adapter=recyclerViewAdapter
        recyclerView.layoutManager=layoutManager
    },Response.ErrorListener {
        readProgressLayout.visibility=View.GONE
        readProgressBar.visibility=View.GONE
    }
)

```

```

        Toast.makeText(this@ReadActivity, it.toString(),
Toast.LENGTH_SHORT).show()
    }
){
    override fun getHeaders(): MutableMap<String, String> {
        return super.getHeaders()
    }
}
queue.add(jsonObjectRequest)

}
fun main() {
    val qrCodeModule = QRCodeModule()
    val imagePath = "path_to_qr_code_image.png"
    val result = qrCodeModule.decodeQRCode(imagePath)
    if (result != null) {
        println("Decoded QR Code: $result")
    } else {
        println("QR Code not found or cannot be decoded.")
    }
}
}
}

```

ПРИЛОЖЕНИЕ В

Листинг модуля сохранения

```
package com.pritampattanayak.tutorial
import android.Manifest
import android.content.Intent
import android.content.pm.PackageManager
import android.os.Bundle
import android.os.Message
import android.view.View
import android.widget.*
import androidx.appcompat.app.AlertDialog
import androidx.appcompat.app.AppCompatActivity
import com.android.volley.AuthFailureError
import com.android.volley.Request
import com.android.volley.Response
import com.android.volley.toolbox.StringRequest
import com.android.volley.toolbox.Volley
import com.budiyev.android.codescanner.AutoFocusMode
import com.budiyev.android.codescanner.CodeScanner
import com.budiyev.android.codescanner.CodeScannerView
import com.budiyev.android.codescanner.ScanMode
import kotlinx.android.synthetic.main.activity_write.*
import java.util.*
import kotlin.collections.HashMap
@Suppress("IMPLICIT_CAST_TO_ANY")
class WriteActivity : AppCompatActivity() {
```

```

lateinit var writeProgressLayout:RelativeLayout
lateinit var writeProgressBar:ProgressBar
lateinit var edtBookName:EditText
lateinit var edtBookAuthor:EditText
lateinit var edtBookPrice:EditText
lateinit var btnSaveToDrive:Button
lateinit var btnScan:Button
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_write)
    writeProgressLayout=findViewById(R.id.writeProgressLayout)
    writeProgressBar=findViewById(R.id.writeProgressBar)
    edtBookName=findViewById(R.id.edtBookName)
    edtBookAuthor=findViewById(R.id.edtBookAuthor)
    edtBookPrice=findViewById(R.id.edtBookPrice)
    btnSaveToDrive=findViewById(R.id.btnSaveToDrive)
    btnScan = findViewById(R.id.btnScan)
    writeProgressLayout.visibility=View.GONE
    writeProgressBar.visibility=View.GONE
/*
    btnScan.setOnClickListener(View.OnClickListener { scanner() })
    fun scanner() {
        val options = ScanOptions()
        options.setPrompt("Volume up to Flash on")
        options.setBeepEnabled(true)
        options.setOrientationLocked(true)
        options.captureActivity = StartScan::class.java
        launcher.launch(options)

```

```

    }

    var launcher = registerForActivityResult(ScanContract()) { result:
ScanIntentResult ->

        if (result.contents != null) {
            val builder = AlertDialog.Builder(this@WriteActivity)
            builder.setTitle("QR-SCANNER RESULT")
            builder.setMessage(result.contents)
            builder.setPositiveButton("Oke") { dialog, which -> dialog.dismiss()
}.show()
        }
    }*/

    btnSaveToDrive.setOnClickListener {
        val any = if (edtBookName.text.toString().isEmpty() or
edtBookAuthor.text.toString()
            .isEmpty() or
            edtBookPrice.text.toString().isEmpty()
        ) {
            Toast.makeText(this@WriteActivity, "Enter All Data",
Toast.LENGTH_SHORT).show()
        } else {

            writeProgressLayout.visibility = View.VISIBLE
            writeProgressBar.visibility = View.VISIBLE

            val url =
"https://script.google.com/macros/s/AKfycbz2fZnhZ7PE8IsMcMITAifob2Unqfx07n
yBMqy2GA/exec"

            val stringRequest = object : StringRequest(Request.Method.POST,
url,

                Response.Listener {

```

```

        Toast.makeText(this@WriteActivity, it.toString(),
Toast.LENGTH_SHORT).show()

        writeProgressLayout.visibility = View.GONE
        writeProgressBar.visibility = View.GONE
    },
    Response.ErrorListener {
        Toast.makeText(this@WriteActivity, it.toString(),
Toast.LENGTH_SHORT).show()

        writeProgressLayout.visibility = View.GONE
        writeProgressBar.visibility = View.GONE
    }) {
    @Throws(AuthFailureError::class)
    override fun getParams(): MutableMap<String, String>{
        val params=HashMap<String, String>()
        params["bookName"]=edtBookName.text.toString()
        params["bookAuthor"]=edtBookAuthor.text.toString()
        params["bookPrice"]=edtBookPrice.text.toString()

        return params
    }
}

val queue = Volley.newRequestQueue(this@WriteActivity)
queue.add(stringRequest)

private fun getService(): Sheets {
    val credential = authorize()
    return Sheets.Builder(HTTP_TRANSPORT, JSON_FACTORY,
credential)

        .setApplicationName(APPLICATION_NAME)
        .build()
}

```



```
}
```

```
private fun authorize(): Credential {
```

```
    val credentialsStream = FileInputStream(credentialsFilePath)
```

```
    val clientSecrets = GoogleCredential.fromStream(credentialsStream)
```

```
    val credential = GoogleCredential.Builder()
```

```
        .setTransport(HTTP_TRANSPORT)
```

```
        .setJsonFactory(JSON_FACTORY)
```

```
        .setServiceAccountId(clientSecrets.serviceAccountId)
```

```
    .setServiceAccountPrivateKey(clientSecrets.serviceAccountPrivateKey)
```

```
        .setServiceAccountScopes(SCOPES)
```

```
        .build()
```

```
    credentialsStream.close()
```

```
    return credential
```

```
}
```

```
fun readSpreadsheet(spreadsheetId: String, range: String): List<List<Any>>
```

```
{
```

```
    val service = getService()
```

```
    val response = service.spreadsheets().values()
```

```
        .get(spreadsheetId, range)
```

```
        .execute()
```

```
    val values = response.getValues()
```

```
    return if (values != null && values.isNotEmpty()) {
```

```
        values
```

```
    } else {
```

```
        emptyList()
```

```
}
```

ПРИЛОЖЕНИЕ Г

Листинг модуля сканирования

```
import android.Manifest
import android.content.pm.PackageManager
import android.os.Bundle
import android.util.Log
import androidx.appcompat.app.AppCompatActivity
import androidx.core.app.ActivityCompat
import androidx.core.content.ContextCompat
import com.google.zxing.Result
import me.dm7.barcodescanner.zxing.ZXingScannerView

class MainActivity : AppCompatActivity(), ZXingScannerView.ResultHandler
{
    private lateinit var scannerView: ZXingScannerView
    companion object {
        private const val CAMERA_PERMISSION_REQUEST_CODE = 100
    }
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        scannerView = ZXingScannerView(this)
        setContentView(scannerView)
    }

    override fun onResume() {
        super.onResume()

        if (ContextCompat.checkSelfPermission(this,
Manifest.permission.CAMERA) == PackageManager.PERMISSION_GRANTED) {
            startCameraScanner()
        } else {
            requestCameraPermission()
        }
    }
    override fun onPause() {
        super.onPause()
        scannerView.stopCamera()
    }
}
```

```

    }
    private fun requestCameraPermission() {
        ActivityCompat.requestPermissions(
            this,
            arrayOf(Manifest.permission.CAMERA),
            CAMERA_PERMISSION_REQUEST_CODE
        )
    }
    private fun startCameraScanner() {
        scannerView.setResultHandler(this)
        scannerView.startCamera()
    }
    override fun handleResult(result: Result?) {
        result?.let {
            Log.d("QR Code", "Содержимое QR-кода: ${it.text}")
        }
        scannerView.resumeCameraPreview(this)
    }
    override fun onRequestPermissionsResult(requestCode: Int, permissions:
Array<out String>, grantResults: IntArray) {
        when (requestCode) {
            CAMERA_PERMISSION_REQUEST_CODE -> {
                if (grantResults.isNotEmpty() && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
                    startCameraScanner()
                } else {
                }
                return
            }
        }
        super.onRequestPermissionsResult(requestCode, permissions,
grantResults)
    }
    fun main() {

        val qrCodeImage = generateQRCodeImage(qrCodeText, 200, 200)
        saveQRCodeImage(qrCodeImage, qrCodeFilePath)
        println("QR code сохранен как $qrCodeFilePath")

        // Чтение QR-кода
        val decodedText = readQRCode(qrCodeFilePath)

```

```

        println("Распознанный текст: $decodedText")
    }

fun generateQRCodeImage(text: String, width: Int, height: Int): BufferedImage
{
    val qrCodeWriter = QRCodeWriter()
    val bitMatrix = qrCodeWriter.encode(text, BarcodeFormat.QR_CODE,
width, height)
    return MatrixToImageWriter.toBufferedImage(bitMatrix)
}

fun saveQRCodeImage(image: BufferedImage, filePath: String) {
    ImageIO.write(image, "PNG", File(filePath))
}

val qrCodeModule = QRCodeModule()
val data = "New QR Code Data"
val filePath = "path_to_output_qr_code.png"
qrCodeModule.generateQRCode(data, filePath)
println("QR Code with updated data generated successfully.")

fun readQRCode(filePath: String): String {
    val qrCodeBufferedImage = ImageIO.read(File(filePath))
    val source = BufferedImageLuminanceSource(qrCodeBufferedImage)
    val bitmap = BinaryBitmap(HybridBinarizer(source))
    val reader = QRCodeReader()
    val result = reader.decode(bitmap)
    return result.text
}
}
}

```