Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Факультет математики и информатики
Кафедра информационных и управляющих систем
Направление подготовки 09.04.04 – Программная инженерия
Направленность (профиль) образовательной программы Управление разработкой программного обеспечения

ДОПУСТИТЬ К ЗАЩИТЕ
Зав. кафедрой
_____ А.В. Бушманов
«_____»_____2021 г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

на тему: Разработка Android-приложения для применения в медицине

Исполнитель
студент группы 957-ом _____ А.Б. Храпов
(подпись, дата)

Руководитель
доцент, канд. тех. наук _____ А.В. Бушманов
(подпись, дата)

Руководитель научного
содержания программы
магистратуры
профессор, доктор техн. наук _____ И.Е. Еремин
(подпись, дата)

Нормоконтроль
Инженер кафедры _____ В.Н. Адаменко
(подпись, дата)

Рецензент
Генеральный директор
агентства «Z-LABS» _____ И.С. Вирта
(подпись, дата)

Благовещенск 2021

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Факультет математики и информатики
Кафедра информационных и управляющих систем

УТВЕРЖДАЮ
Зав. кафедрой
_____ А.В. Бушманов
подпись
«_____»_____2021 г.

З. А Д А Н И Е

К магистерской диссертации студента группы 957-ом

Храпова Алексея Борисовича

1. Тема магистерской диссертации: Разработка Android-приложения для применения в медицине

(Утверждено приказом от 17.05.2021 № 931-уч)

2. Срок сдачи студентом законченной работы (проекта) 23.06.2021

3. Исходные данные к магистерской диссертации: Интернет ресурсы, отчёт по практике.

4. Содержание магистерской диссертации (перечень подлежащих разработке вопросов): анализ предметной области проводимого исследования; архитектура Andoid-приложений.

5. Рецензент магистерской диссертации: И.С. Вирта

6. Дата выдачи задания 25.02.2021

Руководитель выпускной квалификационной работы: А.В. Бушманов,

доцент, канд. техн. наук

(фамилия, имя, отчество, должность, уч.степень, уч.звание)

Задание принял к исполнению (25.02.2021) _____

РЕФЕРАТ

Магистерская диссертация содержит 95 с., 37 рисунок, 1 таблицу, 64 источника.

ANDROID, ANDROID SDK, JAVA, МОБИЛЬНОЕ ПРИЛОЖЕНИЕ.

Целью данной выпускной квалификационной работы является разработка и внедрение Android – приложений для применения в медицине.

В перечень задач данной выпускной квалификационной работы входит:

- изучить архитектурные особенности платформы Android для реализации исходных требований;
- спроектировать приложения в области медицины;
- реализовать приложение;
- протестировать приложение и установить для использования на персональные телефоны с платформой Android.;

СОДЕРЖАНИЕ

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

ANDROID – операционная система;

ANDROID SDK – комплект для разработки программного обеспечения;

JAVA – язык программирования и платформу вычислений;

МП – МОБИЛЬНОЕ ПРИЛОЖЕНИЕ.

ВВЕДЕНИЕ

В наши дни мобильные устройства заняли прочное место в повседневной жизни людей. Они начали свой путь в качестве телефонов, впоследствии обретая всё больше и больше функций, превратившись, в конце концов, в полезных помощников. Более того, с развитием интернета мобильные устройства стали средством получения доступа к 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. Очень важной частью мобильного устройства является программное обеспечение. Оно может способствовать решению различных бизнес- задач, служить для обучения, для хранения важной информации, для развлечения, помогать в работе, собирать информацию, а также выполнять множество других функций. Рынок программного обеспечения для мобильных устройств развивается очень бурно в связи с высоким спросом на мобильные устройства и, что естественно, на различного рода мобильные приложения. Предприниматели не могли обойти вниманием преимущества, предоставляемые мобильными устройствами. Все больше и больше компаний разрабатывают или заказывают разработку приложений для удовлетворения собственных нужд.

Целью выпускной квалификационной работы является разработка и внедрение Android – приложений для применения в медицине.

Задачи исследования:

– изучить архитектурные особенности платформы Android для реализации исходных требований;

– спроектировать приложение в области медицины;

– реализовать приложение;

– протестировать приложение и установить для использования на персональные телефоны с платформой Android.

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Общие сведения

Архитектура ОС Android

В архитектуре операционной системы Android нас интересует уровень каркаса приложений, так как в нем находятся основные службы Android для управления жизненным циклом приложений, пакетами, ресурсами и т. д.

Программист имеет полный доступ ко всем компонентам уровня каркаса приложений, что упрощает реализацию приложений [1]. Компоненты уровня каркаса приложений представлены соответственно на рисунке. 1.

Архитектуру ОС Android принято делить на четыре уровня:

– уровень ядра (Linux Kernel);

– уровень библиотек и среды выполнения (Libraries and Android Runtime);

– уровень каркаса приложений (Application Framework);

– уровень приложений (Applications).

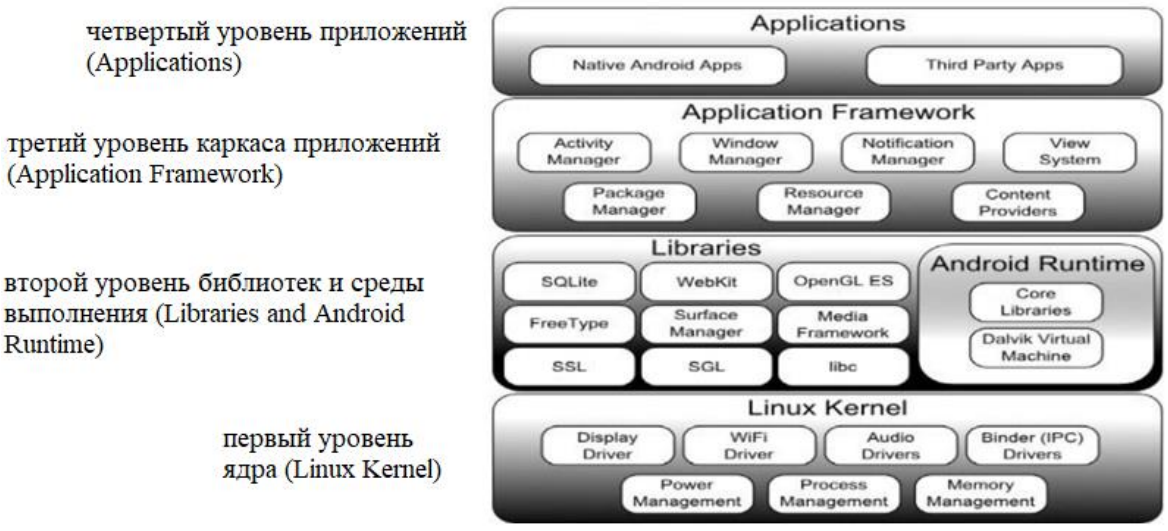


Рисунок 1 — Архитектура ОС Android

При реализации приложения понадобятся следующие компоненты:

— Resource Manager – предназначена для доступа к строковым, графическим и другими типами ресурсов.

При разработке Android – приложений ресурсы принято хранить в специально предназначенных для этого папках. Таким образом, их легче изменять и использовать. Это упрощает локализацию приложений.

— View System – система с расширяемой функциональностью, которая служит для создания внешнего вида приложений, включающего такие компоненты, как списки, таблицы, поля ввода, кнопки и многое другое.

— Notification Manager – позволяет любому приложению отображать пользовательские уведомления в строке состояния.

Пример отправки уведомления представлен на диаграмме последовательности соответственно рисунок 2.

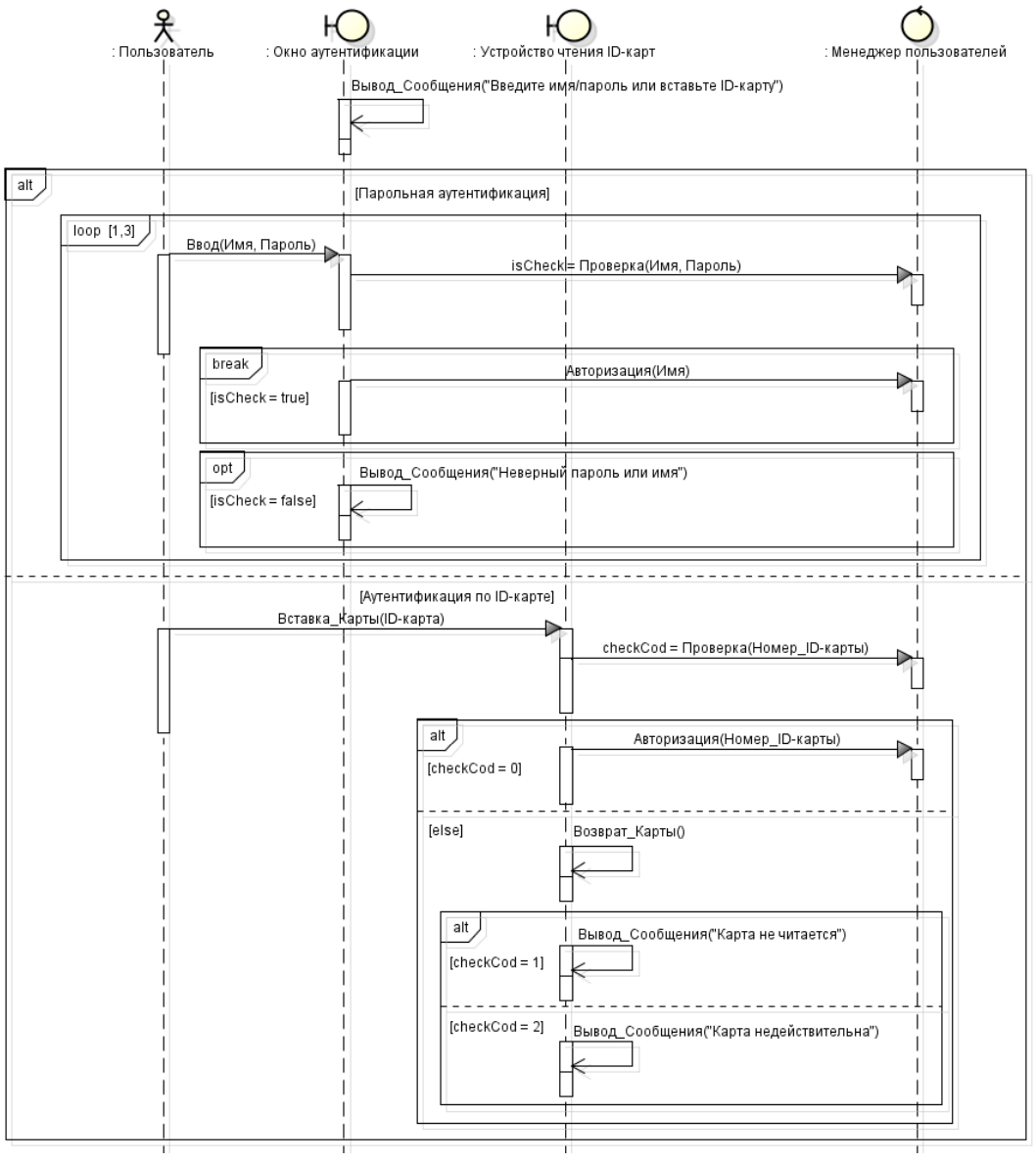


Рисунок. 2 — Диаграмма последовательности отправки уведомления

Ключевые особенности Android

Android является распространенной операционной системой (ОС) для мобильных устройств – телефонов и планшетов. Данная система имеет множество отличительных черт, которые делают ее узнаваемой и привлекательной для большого количества пользователей по всему миру [2].

Операционная система Android является нетребовательной и способна работать на разных конфигурациях. Именно поэтому большинство мировых производителей оснащают свои устройства данной ОС, поскольку другие программные продукты предназначены для отдельных аппаратов, соответствующих определенной спецификации. Такая гибкость Android связана с тем, что система построена на ядре Linux, имеющей открытый программный код, что дает неограниченные возможности разработчикам. Android может быть запущен на устройствах, имеющих объем оперативной памяти менее 256 Мб. Наиболее новые версии системы требуют 512 Мб оперативки, что также является небольшим значением для современных аппаратов. Система не требует наличия высокопроизводительного процессора и может работать на устройствах, оснащенных ядром с частотой 600 МГц [3]. Операционная система дает возможность установки приложений с официального репозитория Google, который предоставляет самую большую в мире базу программ. Это связано с тем, что каждый разработчик может самостоятельно написать любую программу для аппарата и разместить ее в магазине. Возможность также реализована благодаря открытости операционной системы. Стоит отметить, что приложения на устройства под управлением Android могут быть установлены как непосредственно с телефона или планшета, так и через компьютер путем загрузки файла .apk и его последующей установки на аппарате. Отличительной особенностью Android является его интегрированность с сервисами Google, Gmail, Hangouts, Voice Search и т.п. На Android официально реализована поддержка Chrome, что позволяет синхронизировать открываемые в браузере вкладки на смартфоне с компьютерным браузером. Например, вы можете начать просмотр страниц с вашего телефона и при желании продолжить изучать информацию, открыв эту

же вкладку на компьютере, не прибегая к помощи повторного поиска. «Андроид» имеет достаточно простой и интуитивно понятный интерфейс. Все нужные приложения размещаются одновременно на главном экране и в меню аппарата, которое вызывается нажатием на центральную сенсорную клавишу или соответствующую кнопку на экране [4]. Все настройки располагаются в секции «Настройки», а каждое действие пользователя поясняется комментариями и подсказками при первом запуске аппарата. Операционная система быстро реагирует на нажатия пользователя и производит установку и скачивание нужных программ и файлов со скоростью, которая не проигрывает другим современным мобильным ОС.

Общая схема работы приложения Android

Приложения для Android в своей работе используют окна (аналогично

Windows), однако в данной системе вышеуказанные окна носят иное название – Activity. Как и в Windows, каждое окно имеет свой жизненный цикл и свои особенности. При создании нового окна вызывается метод onCreate(), при разработке данный метод переопределяется и в нем происходит инициализация приложения и его компонентов. Далее вызываются методы onStart() и onResume(). Оба метода вызываются перед отображением окна при его создании, либо восстановлении (при переключении из другого приложения, при разворачивании свернутого приложения и тп). При сворачивании вызываются методы onPause() и onStop(). При закрытии приложения и окна вызывается onDestory(), в данном методе можно сохранить пользовательские данные и параметры [5]. Полное описание и последовательность вызова методов можно найти на официальном сайте. Общая схема жизненного цикла приложения для Android представлена соответственно на рисунке. 3.

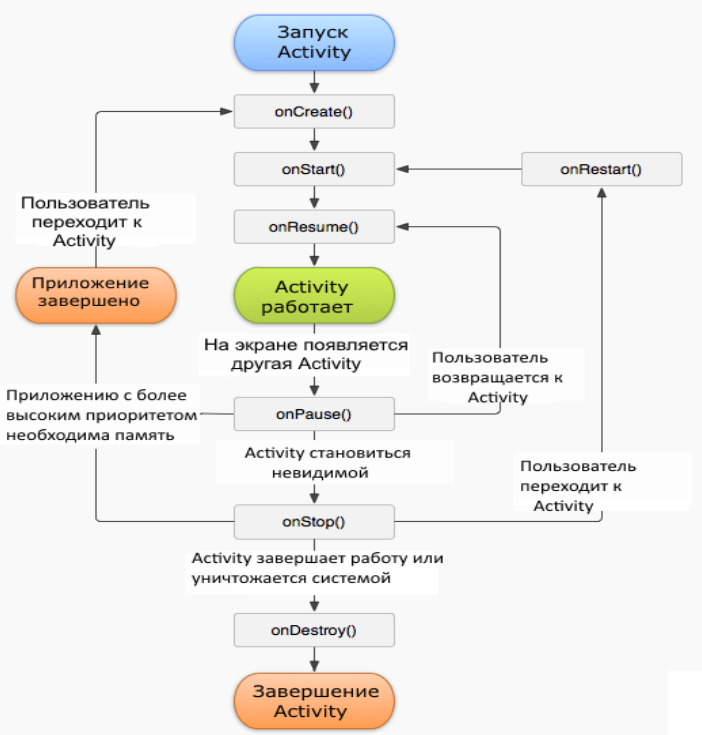


Рисунок 3 — Общая схема жизненного цикла приложения для Android

onCreate – первый метод, с которого начинается выполнение activity. В этом методе activity переходит в состояние Created. Этот метод обязательно должен быть определен в классе activity. В нем производится первоначальная настройка activity. В частности, создаются объекты визуального интерфейса. Этот метод получает объект Bundle, который содержит прежнее состояние activity, если оно было сохранено. Если activity заново создается, то данный объект имеет значение null. Если же activity уже ранее была создана, но находилась в приостановленном состоянии, то bundle содержит связанную с activity информацию [6].

После того, как метод onCreate() завершил выполнение, activity переходит в состояние Started, и и система вызывает метод onStart().

В методе onStart() осуществляется подготовка к выводу activity на экран устройства. Как правило, этот метод не требует переопределения, а всю работу производит встроенный код. После завершения работы метода activity отображается на экране, вызывается метод onResume, а activity переходит в состояние Resumed.

При вызове метода onResume activity переходит в состояние Resumed и отображается на экране устройства, и пользователь может с ней взаимодействовать. Activity остается в этом состоянии, пока она не потеряет фокус, например, вследствии переключения на другую activity или просто из-за выключения экрана устройства.

Если пользователь решит перейти к другой activity, то система вызывает метод onPause, а activity переходит в состояние Paused. В этом методе можно освобождать используемые ресурсы, приостанавливать процессы, например, воспроизведение аудио, анимаций, останавливать работу камеры (если она используется) и т.д., чтобы они меньше сказывались на производительность системы [7].

Но надо учитывать, что в этот состоянии activity по-прежнему остается видимой на экране. На работу данного метода отводится очень мало времени, поэтому не стоит здесь сохранять какие-то данные, особенно если при этом требуется обращение к сети, например, отправка данных по интернету, или обращение к базе данных - подобные действия лучше выполнять в методе onStop().

После выполнения этого метода activity становится невидимой, не отображается на экране, но она все еще активна. И если пользователь решит вернуться к этой activity, то система вызовет снова метод onResume, и activity снова появится на экране.

Другой вариант работы может возникнуть, если вдруг система видит, что для работы активных приложений необходимо больше памяти. И система может сама завершить полностью работу activity, которая невидима и находится в фоне. Либо пользователь может нажать на кнопку Back (Назад). В этом случае у activity вызывается метод onStop [8].

В этом методе activity переходит в состояние Stopped. В этом состоянии activity полностью невидима. В методе onStop следует особождать используемые ресурсы, которые не нужны пользователю, когда он не взаимодействует с activity. Здесь также можно сохранять данные, например, в базу данных.

При этом во время состояния Stopped activity остается в памяти устройства, сохраняется состояние всех элементов интерфейса. К примеру, если в текстовое поле EditText был введен какой-то текст, то после возобновления работы activity и перехода ее в состояние Resumed мы вновь увидим в текстовом поле ранее введенный текст.

Если после вызова метода onStop пользователь решит вернуться к прежней activity, тогда система вызовет метод onRestart. Если же activity вовсе завершила свою работу, например, из-за закрытия приложения, то вызывается метод onDestroy() [9].

Завершается работа activity вызовом метода onDestroy, который возникает либо, если система решит убить activity в силу конфигурационных причин (например, поворот экрана или при многоконном режиме), либо при вызове метода finish ().

Следует отметить, что при изменении ориентации экрана система завершает activity и затем создает ее заново, вызывая метод onCreate.

Более подробно эти методы жизненного цикла перечислены в таблице 1, в которой описан каждый метод обратного вызова и указано его место в жизненном цикле операции в целом, включая сведения о том, может ли система завершить операцию по завершении метода обратного вызова [10].

Таблица 1— Сводные сведения о методах обратного вызова жизненного цикла операции.

Метод	Описание	Завершаемый	Следующий
onCreate()	Вызывается при первом создании операции. Здесь необходимо настроить все обычные статические элементы – создать представления, привязать данные и т. д. Этот метод передает объект Bundle, содержащий предыдущее состояние операции.	Нет	onStart()

Продолжение таблицы 1

onRestart()	Вызывается после остановки операции непосредственно перед ее повторным запуском. За ним всегда следует метод onStart().	Нет	onStart()
onStart()	Вызывается непосредственно перед тем, как операция становится видимой для пользователя. За ним следует метод onResume(), если операция переходит на передний план, или метод onStop(), если она становится скрытой.	Нет	onResume() или onStop()
onResume()	Вызывается непосредственно перед тем, как операция начинает взаимодействие с пользователем. На этом этапе операция находится в самом верху стека операций, и в нее поступают данные, вводимые пользователем. За ним всегда следует метод onPause().	Нет	onPause()
onStop()	Вызывается в случае, когда операция больше не отображается для пользователя. Это может произойти по причине того, что операция уничтожена, или ввиду возобновления поверх нее другой операции (существующей или новой). За ним следует либо метод onRestart(), если операция возобновляет взаимодействие с пользователем, либо метод onDestroy(), если операция переходит в фоновый режим.	Да	onRestart() или onDestroy()
onDestroy()	Вызывается перед тем, как операция будет уничтожена. Это финальный вызов, который получает операция. Его можно вызвать либо по причине завершения операции (вызов метода finish()), либо ввиду временного уничтожения системой этого экземпляра операции с целью освободить место. Чтобы различить эти два сценария, используется метод isFinishing().	Да	Ничего

В столбце «Завершаемый?» указывается, может ли система в любое время завершить процесс, содержащий операцию, после возвращения метода без выполнения другой строки кода операции. Для трех методов в этом столбце указано «Да»: (onPause(), onStop() и onDestroy()). Поскольку метод onPause() является первым из этих трех после создания операции, метод onPause() является последним, который гарантированно будет вызван перед тем, как процесс можно будет завершить. Если системе потребуется срочно восстановить память в случае аварийной ситуации, то методы onStop() и onDestroy() вызвать не удастся. Поэтому следует воспользоваться onPause(), чтобы записать критически важные данные (такие как правки пользователя) в хранилище постоянных данных [11].

Однако следует внимательно подходить к выбору информации, которую необходимо сохранить во время выполнения метода onPause(), поскольку любая блокировка процедур в этом методе может вызвать блокирование перехода к следующей операции и тормозить работу пользователя. Методы, для которых в столбце Завершаемый? указано «Нет», защищают процесс, содержащий операцию, от завершения сразу с момента их вызова. Поэтому завершить операцию можно в период между возвратом onPause() и вызовом on Resume(). Его снова не удастся завершить, пока снова не будет вызван и возвращен onPause().Сохранение состояния операции [12].

В обзорных сведениях об управлении жизненным циклом операции кратко упоминается, что в случае приостановки или полной остановки операции ее состояние сохраняется. Это действительно так, поскольку объект Activity при этом по-прежнему находится в памяти, и вся информация о ее элементах и текущем состоянии по-прежнему активна. Поэтому любые вносимые пользователем в операции изменения сохраняются, и когда операция возвращается на передний план (когда она «возобновляется»), эти изменения остаются в этом объекте.

Однако, когда система уничтожает операцию в целях восстановления памяти, объект Activity уничтожается, в результате чего системе не удается про-

сто восстановить состояние операции для взаимодействия с ней. Вместо этого системе необходимо повторно создать объект Activity, если пользователь возвращается к нему. Но пользователю неизвестно, что система уже уничтожила операцию и создала ее повторно, поэтому, возможно, он ожидает, что операция осталась прежней. В этой ситуации можно обеспечить сохранение важной информации о состоянии операции путем реализации дополнительного метода обратного вызова – onSaveInstanceState(), который позволяет сохранить информацию о вашей операции. Прежде чем сделать операцию доступной для уничтожения, система вызывает метод onSaveInstanceState(). Система передает в этот метод объект Bundle, в котором можно сохранить информацию о состоянии операции в виде пар «имя – значение», используя для этого такие методы, как putString() и putInt(). Затем, если система завершает процесс вашего приложения и пользователь возвращается к вашей операции, система повторно создает операцию и передает объект Bundle в оба метода: onCreate() и onRestoreInstanceState(). С помощью любого из этих методов можно извлечь из объекта Bundle сохраненную информацию о состоянии операции и восстановить ее [13].

Если такая информация отсутствует, то объект Bundle передается с нулевым значением (это происходит в случае, когда операция создается в первый раз).

На рисунке 4.сотвественно продемонстрировано два способа возврата операции к отображению для пользователя в неизмененном состоянии: уничтожение операции с последующим ее повторным созданием, когда операция должна восстановить свое ранее сохраненное состояние, или остановка операции и ее последующее восстановление в неизмененном состоянии.

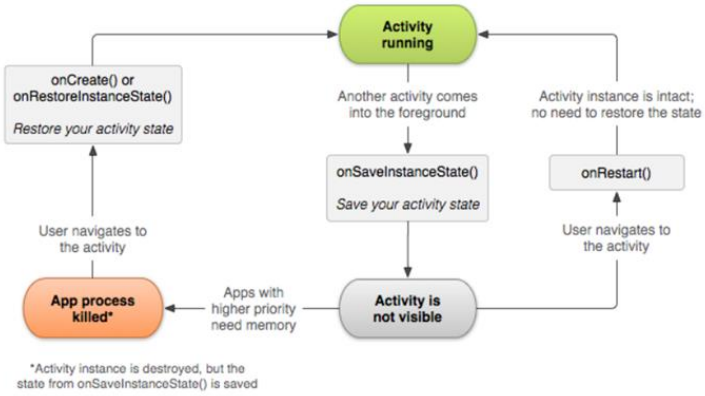


Рисунок 4 — Два способа возврата операции к отображению для пользователя в неизмененном состоянии

Однако даже если вы ничего не предпринимаете и не реализуете метод on Save Instance State(), часть состояния операции восстанавливается реализацией по умолчанию метода on Save Instance State() класса Activity. В частности, реализация по умолчанию вызывает соответствующий метод on Save Instance State() для каждого объекта View в макете, благодаря чему каждое представление может предоставлять ту информацию о себе, которую следует сохранить [15].

Почти каждый виджет в платформе Android реализует этот метод необходимым для себя способом так, что любые видимые изменения в пользовательском интерфейсе автоматически сохраняются и восстанавливаются при повторном создании операции. Например, виджет EditText сохраняет любой текст, введенный пользователем, а виджет CheckBox сохраняет информацию о том, был ли установлен флажок. От вас требуется лишь указать уникальный идентификатор (с атрибутом android:id) для каждого виджета, состояние которого необходимо сохранить. Если виджету не присвоен идентификатор, то системе не удастся сохранить его состояние [16].

Вы также можете явно отключить сохранение информации о состоянии представления в макете. Для этого задайте для атрибута android:saveEnabled значение "false" или вызовите метод set Save Enabled(). Обычно отключать сохранение такой информации не требуется, однако это может потребоваться в

случаях, когда восстановить состояние пользовательского интерфейса операции необходимо другим образом.

Несмотря на то, что реализация метода on Save Instance State() по умолчанию позволяет сохранить полезную информацию о пользовательском интерфейсе вашей операции, вам по-прежнему может потребоваться переопределить ее для сохранения дополнительной информации. Например, может потребоваться сохранить значения элементов, которые изменялись в течение жизненного цикла операции (которые могут коррелировать со значениями, восстановленными в пользовательском интерфейсе, однако элементы, содержащие эти значения пользовательского интерфейса, по умолчанию не были восстановлены). Поскольку реализация метода on Save Instance State() по умолчанию позволяет сохранить состояние пользовательского интерфейса, в случае , если вы переопределите метод с целью сохранить дополнительную информацию о состоянии, перед выполнением каких-либо действий вы всегда можете вызвать реализацию суперкласса для метода on Save Instance State().

Точно так же реализацию суперкласса on Restore Instance State следует вызывать в случае ее переопределения, чтобы реализация по умолчанию могла сохранить состояния представлений.

Отличный способ проверить возможность вашего приложения восстанавливать свое состояние – это просто повернуть устройство для изменения ориентации экрана. При изменении ориентации экрана система уничтожает и повторно создает операцию, чтобы применить альтернативные ресурсы, которые могут быть доступны для новой конфигурации экрана. Только по одной этой причине крайне важно, чтобы ваша операция могла полностью восстанавливать свое состояние при ее повторном создании, поскольку пользователи постоянно работают с приложениями в разных ориентациях экрана [17].

Обработка изменений в конфигурации

Некоторые конфигурации устройств могут изменяться в режиме выполнения (например, ориентация экрана, доступность клавиатуры и язык). В таких

случаях Android повторно создает выполняющуюся операцию (система сначала вызывает метод on Destroy(), а затем сразу же вызывает метод on Create()).

Такое поведение позволяет приложению учитывать новые конфигурации путем автоматической перезагрузки в приложение альтернативных ресурсов, которые вы предоставили (например, различные макеты для разных ориентаций и экранов разных размеров). Если операция разработана должным образом и должным образом поддерживает перезапуск после изменения ориентации экрана и восстановление своего состояния, как описано выше, ваше приложение можно считать более устойчивым к другим непредвиденным событиям в жизненном цикле операции. Лучший способ обработки такого перезапуска – сохранить и восстановить состояние операции с помощью методов on Save Instance State() и on Restore Instance State() (или on Create()), как описано в предыдущем разделе. Дополнительные сведения об изменениях конфигурации, происходящих в режиме выполнения, и о способах их обработки представлены в руководстве Обработка изменений в режиме выполнения.

Согласование операций

Когда одна операция запускает другую, в жизненных циклах обеих из них происходит переход из одного состояния в другое. Первая операция приостанавливается и завершается (однако она не будет остановлена, если она по-прежнему видима на фоне), а вторая операция создается [18].

В случае, если эти операции обмениваются данным, сохраненными на диске или в другом месте, важно понимать, что первая операция не останавливается полностью до тех пор, пока не будет создана вторая операция. Наоборот, процесс запуска второй операции накладывается на процесс остановки первой операции.

Порядок обратных вызовов жизненного цикла четко определен, в частности, когда в одном и том же процессе находятся две операции, и одна из них запускает другую. Ниже представлен порядок выполнения действий в случае, когда операция А запускает операцию Б. Выполняется метод on Pause() операции А. Последовательно выполняются методы on Create(), on Start() и on Resume()

операции Б. (Теперь для пользователя отображается операция Б.) Затем, если операция А больше не отображается на экране, выполняется ее метод on Stop(). Такая предсказуемая последовательность выполнения обратных вызовов жизненного цикла позволяет управлять переходом информации из одной операции в другую. Например, если после остановки первой операции требуется выполнить запись в базу данных, чтобы следующая операция могла считать их, то запись в базу данных следует выполнить во время выполнения метода on Pause(), а не во время выполнения метода on Stop() [19].

1.2 Архитектура Android-приложений

MVC в Android – приложениях прежде чем переходить к рассмотрению MVC в Android-приложениях необходимо рассмотреть подробнее следующие компоненты Android – приложения: layout и activity

Layout в Android – приложении пользовательский интерфейс представляет собой один или несколько макетов (layout), демонстрируемых поочерёдно на экране. Layout состоит из набора элементов-наследников класса View, организованных в древовидную структуру. Конкретными наследниками могут быть, к примеру, кнопки, поля для ввода текста, области экрана для демонстрации изображений и прочее. Особо следует выделить наследника класса View под названием View Group – с его помощью можно группировать другие элементы пользовательского интерфейса (к примеру, отображать их в виде списка).

Существует два способа создания layout:

— описание дерева элементов в xml-файле;

— программное создание дерева элементов.

При этом layout может являться частью другого layout.

В приложении Android контейнер компонентов имеет тип View Group. Существует несколько разновидностей классов, наследующих свойства View Group и определяющих структуру расположения компонентов в интерфейсе: Linear Layout, Relative Layout, Frame Layout, Table Layout, Constraint Layout и т.д.

Отличия этих классов связаны с упорядочиванием компонентов:

— Constraint Layout - позволяет привязывать компонент к границам экрана или к другим компонентам;

— Linear Layout - позволяет размещать View-компоненты в виде одной строки (horizontal) или одного столбца (vertical);

— Relative Layout - настраивает положение каждого компонента относительно других;

— Absolute Layout - использует для каждого компонента явную позицию на экране в системе координат X, Y;

— TableLayout - отображает элементы в виде таблицы, по строкам и столбцам.

Activity операция (Activity) представляет собой один экран с пользовательским интерфейсом. Например, в приложении для работы с электронной почтой одна операция может служить для отображения списка новых сообщений, другая – для составления сообщения и третья операция – для чтения сообщений.

Несмотря на то, что операции совместно формируют связное взаимодействие пользователя с приложением по работе с электронной почтой, каждая из них не зависит от других операций. Любые из этих операций могут быть запущены другим приложением (если это позволяет приложение по работе с электронной почтой). Например, приложение для камеры может запустить операцию в приложении по работе с электронной почтой, которая составляет новое сообщение, чтобы пользователь мог отослать фотографию [20].

Операция относится к подклассу класса Activity. Подробные сведения об этом можно найти в руководстве для разработчиков в статье Операции.

Activity – компонент приложения, связанный с одним layout. Отвечает за поведение демонстрируемого на экране layout-а: содержит описание обработчиков событий и прочей логики, требуемой для работы с layout.

Приложение состоит из одного или нескольких Activity, чаще всего слабо связанных друг с другом.

Activity – это компонент приложения, который выдает экран, и с которым пользователи могут взаимодействовать для выполнения каких-либо действий, например набрать номер телефона, сделать фото, отправить письмо или просмотреть карту. Каждой операции присваивается окно для прорисовки соответствующего пользовательского интерфейса. Обычно окно отображается во весь экран, однако его размер может быть меньше, и оно может размещаться поверх других окон.

Как правило, приложение состоит из нескольких операций, которые слабо связаны друг с другом. Обычно одна из операций в приложении обозначается как «основная», предлагаемая пользователю при первом запуске приложения.

В свою очередь, каждая операция может запустить другую операцию для выполнения различных действий. Каждый раз, когда запускается новая операция, предыдущая операция останавливается, однако система сохраняет ее в стеке («стек переходов назад»). При запуске новой операции она помещается в стек переходов назад, и новая операция отображается для пользователя.

Стек переходов назад работает по принципу «последним вошёл – первым вышел», поэтому после того, как пользователь завершил текущую операцию и нажал кнопку. Назад, текущая операция удаляется из стека (и уничтожается), и возобновляется предыдущая операция. (Подробные сведения о стеке переходов назад представлены в статье Задачи и стек переходов назад.) [21].

Когда операция останавливается по причине запуска новой операции, для уведомления об изменении ее состояния используются методы обратного вызова жизненного цикла операции. Существует несколько таких методов, которые может принимать операция вследствие изменения своего состояния – создание операции, ее остановка, возобновление или уничтожение системой; также каждый обратный вызов представляет возможность выполнить определенное действие, подходящее для соответствующего изменения состояния. Например, в случае остановки операция должна освободить любые крупные объекты, например, подключение к сети или базе данных. При возобновлении операции вы можете повторно получить необходимые ресурсы и возобновить выполне-

ние прерванных действий. Такие изменения состояния являются частью жизненного цикла операции [22].

Основы создания и использования операций, включая полное описание жизненного цикла операции, чтобы вы могли лучше понять, как следует управлять переходами между различными состояниями операции.

Чтобы создать операцию, сначала необходимо создать подкласс класса Activity (или воспользоваться существующим его подклассом). В таком подклассе необходимо реализовать методы обратного вызова, которые вызывает система при переходе операции из одного состояния своего жизненного цикла в другое, например при создании, остановке, возобновлении или уничтожении операции. Вот два наиболее важных метода обратного вызова:

Этот метод необходимо обязательно реализовать, поскольку система вызывает его при создании вашей операции. В своей реализации вам необходимо инициализировать ключевые компоненты операции. Наиболее важно именно здесь вызвать set Content View() для определения макета пользовательского интерфейса операции.

Система вызывает этот метод в качестве первого признака выхода пользователя из операции (однако это не всегда означает, что операция будет уничтожена). Обычно именно здесь необходимо применять любые изменения, которые должны быть сохранены помимо текущего сеанса работы пользователя (поскольку пользователь может не вернуться назад).

Существуют также и некоторые другие методы обратного вызова жизненного цикла, которые необходимо использовать для того, чтобы обеспечить плавный переход между операциями, а также для обработки непредвиденных нарушений в работе операции, в результате которых она может быть остановлена или даже уничтожена. Более подробное описание всех методов обратного вызова жизненного цикла представлены в разделе, посвященном управлению жизненным циклом операций [23].

Для реализации пользовательского интерфейса операции используется иерархия представлений – объектов, полученных из класса View. Каждое пред-

ставление отвечает за определенную область окна операции и может реагировать на действия пользователей. Например, представлением может быть кнопка, нажатие на которую приводит к выполнению определенного действия.

В Android предусмотрен набор уже готовых представлений, которые можно использовать для создания дизайна макета и его организации.

Виджеты – это представления с визуальными (и интерактивными) элементами, например, кнопками, текстовыми полями, чек боксами или просто изображениями. Макеты – это представления, полученные из класса View Group, обеспечивающие уникальную модель компоновки для своих дочерних представлений, таких как линейный макет, сетка или относительный макет. Также можно создать подкласс для классов View и View Group (или воспользоваться существующими подклассами), чтобы создать собственные виджеты и макеты, и затем применить их к макету своей операции.

Чаще всего для задания макета с помощью представлений используется XML – файл макета, сохраненный в ресурсах приложения. Таким образом вы можете хранить дизайн пользовательского интерфейса отдельно от исходного кода, который служит для задания поведения операции. Чтобы задать макет в качестве пользовательского интерфейса операции, можно использовать метод set Content View(), передав в него идентификатор ресурса для макета. Однако вы также можете создать новые View в коде вашей операции и создать иерархию представлений. Для этого вставьте View в View Group, а затем используйте этот макет, передав корневой объект View Group в метод set Content View.

MVC – это концепция проектирования приложений, основной целью которой является отделение логики приложения от пользовательского интерфейса с целью упрощения разработки приложений и их сопровождения.

Как упоминалось ранее, в Android – приложениях пользовательским интерфейсом является набор layoutov, каждый из которых определяет внешний вид одного демонстрируемого на экране окна.

Несмотря на возможность создания layout в коде Activity, рекомендуется создавать layout в виде xml – файла, хранящегося в репозитории ресурсов. Это

позволяет максимально отделить логику приложения от пользовательского интерфейса, благодаря чему естественным образом достигается основная цель концепции MVC. Контроллером в Android – приложении является Activity, связанная с одним layout и определяющая логику его поведения. Таким образом, ввиду отсутствия одного – единственного контроллера, отвечающего за обработку всех поступающих в приложение запросов, в Android – приложениях используется контроллер страниц. В качестве модели выступают созданные разработчиком иерархии классов, отражающие необходимые сущности предметной области. Singleton в Android – приложениях для создания activity необходимо создать наследника класса Activity, в котором реализовать методы для обработки изменения состояния activity в процессе работы [24].

При реализации паттерна Singleton в Android – приложениях следует помнить о том, что Activity может быть уничтожено в любой момент времени, вследствие чего все данные, хранящиеся в объекте Singleton, будут утеряны. Их можно сохранить при помощи штатных средств, предоставляемых ОС Android, но храниться они будут в виде пар «ключ – значение», доступных только одному Activity.

Одним из способов корректной реализации данного паттерна является использование класса Application в качестве родительского для класса Singleton. Экземпляр класса Application будет создан до создания любого другого класса приложения и гарантируется, что он будет один на протяжении всего времени жизни приложения.

Существуют и другие способы реализации паттерна Singleton, но среди разработчиков нет единого мнения насчѐт того, какой из них является наилучшим.

2 АЛГОРИТМИЧЕСКОЕ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ДЛЯ РЕШЕНИЯ ЗАДАЧИ СОЗДАНИЯ ANDROID ПРИЛОЖЕНИЯ

Для запуска компонента приложения системе Android необходимо знать, что компонент существует. Для этого она читает файл Android Manifest.xml приложения (файл манифеста). В этом файле, который должен находиться в корневой папке приложения, должны быть объявлены все компоненты приложения [25].

Помимо объявления компонентов приложения, манифест служит и для других целей, среди которых:

— указание всех полномочий пользователя, которые требуются приложению, например разрешения на доступ в Интернет или на чтение контактов пользователя;

— объявление минимального уровня, требуемого приложению, с учетом того, какие API – интерфейсы оно использует;

— объявление аппаратных и программных функций, которые нужны приложению или используются им, например камеры, службы Bluetooth или сенсорного экрана;

— указание библиотек API, с которыми необходимо связать приложение (отличные от API – интерфейсов платформы Android), например библиотеки Google Maps и многое другое.

Основная задача манифеста – это информировать систему о компонентах приложения. Например, файл манифеста может объявлять операцию следующим образом представлено соответственно на рисунке 5.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest ... >
    <application android:icon="@drawable/app_icon.png" ... >
        <activity android:name="com.example.project.ExampleActivity"
                  android:label="@string/example_label" ... >
        </activity>
        ...
    </application>
</manifest>
```

Рисунок 5 — Объявление файла манифеста

Атрибут android:icon в элементе <application> указывает на ресурсы для значка, который обозначает приложение.

Атрибут android:name в элементе <activity> указывает полное имя класса подкласса Activity, а атрибут android:label указывает строку, которую необходимо использовать в качестве метки операции, отображаемой для пользователя.

Все компоненты приложения необходимо объявлять следующим образом:

— элементы <activity> для операций;

— элементы <service> для служб;

— элементы <receiver> для приемников широковещательных сообщений;

— элементы <provider> для поставщиков контента.

Системе не видны операции, службы и поставщики контента, которые имеются в исходном коде, но не объявлены в манифесте, поэтому они не могут быть запущены. А вот приемники широковещательных сообщений можно либо объявить в манифесте, либо создать динамически в коде (как объекты Broad cast Receiver) и зарегистрировать в системе путем вызова register Receiver().

Подробные сведения о структуризации файла манифеста для приложения см. в документе Файл AndroidManifest.xml [26].

Объявление возможностей компонентов

Как уже говорилось в разделе Активация компонентов, с помощью объекта Intent можно запускать операции, службы и приемники широковещательных сообщений. Для этого в объекте Intent следует явно указать имя целевого компонента (с помощью имени класса компонента). Однако в полной мере возможности объектов Intent раскрываются при использовании концепции неявных Intent. В неявном сообщении Intent просто описывается тип действия, которое требуется выполнить (а также, хотя это и не обязательно, данные, в отношении которых вы бы хотели выполнить это действие). Системе же предоставляется возможности найти на устройстве компонент, который может выполнить это действие, и запустить его. При наличии нескольких компонентов, которые мо-

гут выполнить действие, описанное в сообщении Intent, пользователь выбирает, какой из них будет использоваться [27].

Система определяет компоненты, которые могут ответить на сообщение Intent, путем сравнения полученного сообщения Intent с фильтрами объектов Intent, указанными в файле манифеста других приложений, имеющихся на устройстве.

При объявлении операции в манифесте своего приложения по желанию можно указать фильтры объектов Intent, которые указывают возможности операции, с тем чтобы она могла реагировать на сообщения Intent от других приложений. Чтобы объявить фильтр Intent для своего компонента, необходимо добавить элемент <intent – filter> в качестве дочернего для элемента объявления компонента [28].

Например, если вы создали приложение для работы с электронной почтой с операцией составления нового сообщения, вы можете объявить фильтр для ответа на сообщения Intent типа "send" (для отправки нового сообщения электронной почты) представлено на рисунке 6.

```
<manifest ... >
    ...
    <application ... >
        <activity android:name="com.example.project.ComposeEmailActivity">
            <intent-filter>
                <action android:name="android.intent.action.SEND" />
                <data android:type="*/*" />
                <category android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Рисунок 6 — Объявление фильтра ответа

Затем, если другое приложение создаст объект Intent с действием ACTION_SEND и передаст его в start Activity(), система сможет запустить вашу

операцию, дав пользователю возможность написать и отправить сообщение электронной почты.

Подробные сведения о создании фильтров объектов Intent приведены в документе объекты Intent и фильтры объектов Intent.

Объявление требований приложения

Существует огромное количество устройств, работающих под управлением Android, и не все они имеют одинаковые функциональные возможности. Чтобы ваше приложение не могло быть установлено на устройствах, в которых отсутствуют функции, необходимые приложению, важно четко определить профиль для типов устройств, поддерживаемых вашим приложением, указав требования к аппаратному и программному обеспечению в файле манифеста. Эти объявления по большей части носят информационный характер, система их не читает. Однако их читают внешние службы, например Google Play, с целью обеспечения фильтрации для пользователей, которые ищут приложения для своих устройств [29].

Например, если вашему приложению требуется камера и оно использует API – интерфейсы из Android 2.1 (уровень API 7), эти параметры следует объявить в файле манифеста в качестве требований представлены на рисунке 7.

```
<manifest ... >
    <uses-feature android:name="android.hardware.camera.any"
                  android:required="true" />
    <uses-sdk android:minSdkVersion="7" android:targetSdkVersion="19" />
    ...
</manifest>
```

Рисунок 7 — Объявление требований в манифесте

Теперь ваше приложение нельзя будет установить из Google Play на устройствах, в которых нет камеры, а также на устройствах, работающих под управлением Android версии ниже 2.1.

Однако можно также объявить, что приложение использует камеру, но для его работы она не является непременно необходимой. В этом случае в приложении атрибуту required необходимо задать значение "false", а во время работы оно должно проверять, имеется ли на устройстве камера, и при необходимости отключать свои функции, которые используют камеру [30].

Более подробные сведения об управлении совместимостью своего приложения с различными устройствами приведены в документе Совместимость устройств.

Приложение Android состоит не только из кода – ему необходимы такие существующие отдельно от исходного кода ресурсы, как изображения, аудиофайлы и все, что связано с визуальным представлением приложения. Например, необходимо определять анимацию, меню, стили, цвета и макет пользовательских интерфейсов операций в файлах XML. Используя ресурсы приложения, можно без труда изменять его различные характеристики, не меняя код, а, кроме того – путем предоставления наборов альтернативных ресурсов – можно оптимизировать свое приложение для работы с различными конфигурациями устройств (например, для различных языков или размеров экрана) [31].

Для каждого ресурса, включаемого в проект Android, инструменты SDK задают уникальный целочисленный идентификатор, который может использоваться, чтобы сослаться на ресурс из кода приложения или из других ресурсов, определенных в XML. Например, если в вашем приложении имеется файл изображения с именем logo.png (сохраненный в папке res/drawable/), инструменты SDK сформируют идентификатор ресурса под именем R.drawable.logo, с помощью которого на изображение можно будет ссылаться и вставлять его в пользовательский интерфейс [32].

Один из наиболее важных аспектов предоставления ресурсов отдельно от исходного кода заключается в возможности использовать альтернативные ресурсы для различных конфигураций устройств. Например, определив строки пользовательского интерфейса в XML, вы сможете перевести их на другие языки и сохранить эти переводы в отдельных файлах. Затем по квалификатору язы-

ка, добавленному к имени каталога ресурса (скажем res/values-fr/ для строк на французском языке), и выбранному пользователем языку система Android применит к вашему пользовательскому интерфейсу строки на соответствующем языке [33].

Android поддерживает разные квалификаторы для соответствующих ресурсов. Квалификатор представляет собой короткую строку, которая включается в имена каталогов ресурсов с целью определения конфигурации устройства, для которой эти ресурсы следует использовать. В качестве другого примера можно сказать, что для своих операций следует создавать разные макеты, которые будут соответствовать размеру и ориентации экрана устройства. Например, когда экран устройства имеет книжную ориентацию (расположен вертикально), кнопки в макете можно также размещать по вертикали, а когда экран развернут горизонтально (альбомная ориентация), кнопки следует размещать по горизонтали. Чтобы при изменении ориентации экрана изменялся макет, можно определить два разных макета и применить соответствующий квалификатор к имени каталога каждого макета. После этого система будет автоматически применять соответствующий макет в зависимости от ориентации устройства [34].

Управление жизненным циклом операций

Управление жизненным циклом операций путем реализации методов обратного вызова имеет важное значение при разработке надежных и гибких приложений. На жизненный цикл операции напрямую влияют его связи с другими операциями, его задачами и стеком переходов назад [35].

Существует всего три состояния операции:

1. Возобновлена

Операция выполняется на переднем плане экрана и отображается для пользователя. (Это состояние также иногда называется «Выполняется».)

2. Приостановлена

На переднем фоне выполняется другая операция, которая отображается для пользователя, однако эта операция по – прежнему не скрыта. То есть поверх текущей операции отображается другая операция, частично прозрачная или не

занимающая полностью весь экран. Приостановленная операция полностью активна (объект Activity по – прежнему находится в памяти, в нем сохраняются все сведения о состоянии и информация об элементах, и он также остается связанным с диспетчером окон), однако в случае острой нехватки памяти система может завершить ее [36].

3. Остановлена

Операция полностью перекрывается другой операцией (теперь она выполняется в фоновом режиме). Остановленная операция по – прежнему активна (объект Activity по – прежнему находится в памяти, в нем сохраняются все сведения о состоянии и информация об элементах, но объект больше не связан с диспетчером окон). Однако операция больше не видна пользователю, и в случае нехватки памяти система может завершить ее. Если операция приостановлена или полностью остановлена, система может очистить ее из памяти путем завершения самой операции (с помощью метода finish()) или просто завершить ее процесс. В случае повторного открытия операции (после ее завершения) ее потребуется создать полностью [37].

При переходе операции из одного вышеописанного состояния в другое, уведомления об этом реализуются через различные методы обратного вызова. Все методы обратного вызова представляют собой привязки, которые можно переопределить для выполнения подходящего действия при изменении состояния операции. Указанная ниже соответственно на рисунке 8 базовая операция включает каждый из основных методов жизненного цикла [38].

```java
public class ExampleActivity extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        // The activity is being created.
    }
    @Override
    protected void onStart() {
        super.onStart();
        // The activity is about to become visible.
    }
    @Override
    protected void onResume() {
        super.onResume();
        // The activity has become visible (it is now "resumed").
    }
    @Override
    protected void onPause() {
        super.onPause();
        // Another activity is taking focus (this activity is about to be "paused").
    }
    @Override
    protected void onStop() {
        super.onStop();
        // The activity is no longer visible (it is now "stopped")
    }
    @Override
    protected void onDestroy() {
        super.onDestroy();
        // The activity is about to be destroyed.
    }
}
```

Рисунок 8 — Базовая операция

Вместе все эти методы определяют весь жизненный цикл операции. С помощью реализации этих методов можно отслеживать три вложенных цикла в жизненном цикле операции:

Весь жизненный цикл операции происходит между вызовом метода on Create() и вызовом метода on Destroy(). Ваша операция должна выполнить настройку «глобального» состояния (например, определение макета) в методе on Create(), а затем освободить все оставшиеся в on Destroy() ресурсы. Например, если в вашей операции имеется поток, выполняющийся в фоновом режиме, для загрузки данных по сети, операция может создать такой поток в методе on Create(), а затем остановить его в методе on Destroy() [39].

Видимый жизненный цикл операции происходит между вызовами методов on Start() и on Stop(). В течение этого времени операция отображается на экране, где пользователь может взаимодействовать с ней. Например, метод on Stop() вызывается в случае, когда запускается новая операция, а текущая больше не отображается. В промежутке между вызовами этих двух методов можно сохранить ресурсы, необходимые для отображения операции для пользователя. Например, можно зарегистрировать объект Broad cast Receiver в методе on Start() для отслеживания изменений, влияющих на пользовательский интерфейс,

а затем отменить его регистрацию в методе on Stop(), когда пользователь больше не видит отображаемого [40]. В течение всего жизненного цикла операции система может несколько раз вызывать методы on Start() и on Stop(), поскольку операция то отображается для пользователя, то скрывается от него. Жизненный цикл операции, выполняемый на переднем плане, происходит между вызовами методов on Resume() и on Pause(). В течение этого времени операция выполняется на фоне всех прочих операций и отображается для пользователя. Операция может часто уходить в фоновый режим и выходить из него – например, метод on Pause() вызывается при переходе устройства в спящий режим или при появлении диалогового окна. Поскольку переход в это состояние может выполняться довольно часто, код в этих двух методах должен быть легким, чтобы не допустить медленных переходов и не заставлять пользователя ждать [41].

2.1. Профильное техническое и программное обеспечение Android приложений

Существующее техническое обеспечение

По степени влияния на человека можно выделить три направления mHealth – фитнес, медицина и диагностика. Наиболее популярным является направление фитнеса. Объединяя лучшие стороны на стыке фитнеса и медицины, сейчас набирает популярность диагностическое направление [42].

Рынок «умных» медицинских устройств с каждым годом становится все разнообразнее по ассортименту и все больше по объему. Разрабатывая новые устройства, производители стараются улучшить их, добавляя в них дополнительные функции, что приводит к производству многофункциональных устройств. Благодаря такому прогрессу становится все сложнее отнести подобные устройства к какой-то определенной категории. С помощью таких устройств пользователь имеет возможность производить мониторинг своего физического состояния, а в случае необходимости – отправлять полученные данные врачу [43].

Повсеместное распространение концепции Интернета Вещей предполагает подключение медицинских устройств к сети Интернет для отправки данных

на облачные сервера. Ниже будут представлены «умные» медицинские устройства, условно разделенные на категории: Фитнес-трекеры.

Фитнес – трекер представляет собой носимый гаджет компактного размера, предназначенный для контроля физической активности человека представлены на рисунке 9. В настоящее время производством таких устройств занимается большое количество производителей электроники такие как: Jawbone, Garmin, Fitbit, Misfit и другие [44].

Рисунок 9 — Схема мониторинга физического состояния пациента

Основной задачей фитнес-трекера является мотивация своего владельца на активную деятельность и контроль получаемой нагрузки. Для выполнения поставленных задач в подобных устройствах установлены датчики, фиксирующие данные пульса, количества пройденных шагов, затраченных калорий, уровня стресса, качества сна, скорости перемещения и длины пройденного расстояния [45].

Полученная информация может передаваться на смартфон или компьютер, а использование специально разработанного приложения позволит произвести расчеты активности человека, изменение показателей здоровья и при необходимости даст рекомендации для успешного достижения, поставленной пользователем цели. Наиболее распространённой формой фитнес-трекеров является браслет.

Но существуют модели в виде клипс, очков и даже наушников. Существуют трекеры, способные сами распознать, что человек лег спать и те, которым нужно об этом сообщать, нажав соответствующую кнопку. Самые «умные» из трекеров умеют определять в глубокой или легкой фазе сна находится человек и корректируют срабатывание будильника именно на легкую фазу, когда человека легче разбудить [46].

Большинство умных часов может функционировать как в связке со смартфоном, так и отдельно от него, как самостоятельное устройство. В качестве операционной системы чаще всего используется Android Wear, watchOS и Tizen. Android Wear разработана компанией Google и используется в смарт-часах таких производителей, как: Motorola, Sony, LG, Asus, Huawei.

Ранние версии операционной системы могли работать только с Android-смартфонами. С 2015 года в Android Wear появилась поддержка Apple iPhone. Важные отличия у часов на Android Wear касаются только ёмкости батареи, дизайна и размера экрана.

У всех производителей часов на Android Wear интерфейс выглядит одинаково. Tizen OS используется только в смарт-часах компании Samsung. Другие производители смарт – часов данную операционную систему не используют. WatchOS разработана компанией Apple и используется только в смарт-часах компании.

Смарт – часы совместимы только с iPhone. Смарт-часы Pebble с установленной на них одноименной операционной системой не привязаны к какой-то одной мобильной платформе, а потому работают и со смартфонами на Android, Windows и Apple iPhone.

Умная одежда – одежда, интернирующаяся с современными информационными технологиями. В данный момент, умная одежда находит широкое применение в таких сферах, как медицина, спорт, военная сфера.

Умная одежда способна помочь людям, имеющим физиологические отклонения или заболевания, также позволяет следить за состоянием людей, работающих с опасными веществами.

Такая одежда позволяет производить контроль основных жизненных показателей таких как частота пульса, частота дыхания, температура тела и др. Умная одежда позволяет удаленно производить анализы и осуществлять дистанционное медицинское консультирование.

К тому же, умная одежда способна помочь выжить в экстремальных условиях. С её помощью возможно отслеживать состояние человека и его местоположение, проводить контроль уровня усталости водителя или лётчика.

Мониторинг с использованием умных носков и обуви используется для людей всех возрастных категорий, например: «умные» носки Owlet, разработанные для родителей детей, могут отслеживать частоту сердцебиения ребёнка, температуру кожи, уровень кислорода и то, как он спит. «Умные» стельки Smart Sole со встроенными датчиками GPS помогут передать местоположение человека его родственникам. Эти стельки незаменимы для пациентов, страдающих рассеянным склерозом. − канадские разработчики создали стельки с датчиками Orpyx.

Эти стельки разработаны для людей, страдающих периферической невропатией (осложнением диабета), которое приводит к нарушению чувствительности в конечностях. Задача стелек – оповестить лечащих врачей об избыточном давлении на стопу т. к. такое давление приводит к повреждениям конечностей, а их лечение у диабетиков крайне затруднительно.

Компания Oticon, являющаяся одним из лидеров на рынке технологий слуховых протезов, разработала первый в мире слуховой аппарат, подключённый к Интернету – Oticon Opn. С помощью нескольких технологических новшеств Oticon Opn позволяет сделать жизнь людей с нарушениями слуха более комфортной.

Хоть это и не первый подключающийся к iOS или Android смартфонам для управления аудио потоком слуховой аппарат, но зато первый, с возможностью подключаться к другим устройствам в Интернете – детекторам дыма, умным звонкам, и радио няням, чтобы пользователь смог лучше услышать важные оповещения. Таким образом, создается целая сеть устройств

Интернета вещей, между которыми можно прописывать сценарии взаимодействия, например, сбор данных с нательных сетей с использованием БПЛА.

Oticon Opn имеет малый физический размер, высокие показатели автономности работы, снабжена выходом в Интернет и использует систему связи TwinLink, позволяющую получать звуковые сигналы в оба уха.

Opn подключен к сервису IFTTT, способному связать множество действий на компьютерах и других устройствах с веб-сервисами. В ответ на произошедшие события пользователь может создавать алгоритмы для запуска определенных действий. При звонке в дверь умный дверного звонка отправит звуковое оповещение на слуховой аппарат.

Можно настроить текстовое оповещение членов семьи, когда слуховой аппарат разряжен. Утром, при включении слухового можно настроить автоматическое включение света и кофеварки.

Также, для Oticon Opn существует приложение Oticon ON App для переключения программ слухового аппарата, проверки уровня заряда батареи и регулировки уровня громкости одним касанием пальца. Ещё одним плюсом приложения является наличие функции «найти мой слуховой аппарат» в случае, если слуховой аппарат потерялся. Oticon Opn поддерживает работу с устройствами на операционной системе iOS iOS 9.3 и старше, а также с Apple Watch.

Устройства непрерывного мониторинга глюкозы, избавляющие от ежедневной процедуры прокола пальца диабетиков, набирают большую популярность и становятся удобнее для пользователей.

Давно работающая на рынке устройств для диабетиков компания Dexcom, разработала свою систему непрерывного мониторинга уровня сахара в крови Dexcom G5 Mobile.

Система использует носимый на теле человека датчик малых размеров, производящий измерения и беспроводным образом транслирующий данные в смартфон. Благодаря этому, пользователь избавился от необходимости дополнительно носить с собой отдельное принимающее устройство. G5 Mobile под-

держивает работу с Apple Watch, iPhone и устройствами, работающими на Android и Android Wear.

Тонометр является обязательным средством для профилактики и лечения артериальной гипертонии. Не так давно тонометры научились запоминать результаты прошлых измерений и представлять их в виде списков и таблиц.

Но настала очередь приборов, способных синхронизировать данные с приложениями и облачными сервисами для создания понятных пользователю отчетов. Автоматические тонометры можно разделить на тонометры с манжетой на плечо и тонометры с манжетой на запястье.

Оптимальными для самостоятельного измерения артериального давления, с точки зрения надежности получаемых результатов, являются автоматические осциллометрические аппараты с плечевой манжетой.

Умный ингалятор представляет собой обычный ингалятор способный пассивно отслеживать и беспроводным путем передавать данные об использовании лекарства с помощью, встроенной в него электроники. В него встроен сенсор, фиксирующий время и дату применения лекарства. Эти данные сохраняются во встроенной памяти ингалятора и с помощью Bluetooth способны передаваться на другие устройства.

Используя эксклюзивные для каждого ингалятора приложения, обратную связь с помощью Email или текстовых сообщений, пациенты имеют возможность подробно узнавать о своей болезни и о том, как правильно с ней бороться.

В свою очередь, врачи могут идентифицировать пациентов, более нуждающихся в помощи для контроля заболевания. Согласно данным Adherium, использование умного ингалятора повысило долю принимающих лекарства детей с астмой – на 180 %, а взрослых пациентов на 59 %.

Умные» весы – весы, взаимодействующие с приложением на смартфоне и отправляющие ему результаты измерений. Приложение построит график изменений веса, вычислит индекс массы тела и другие важные показатели, даст совет о том, что съесть на завтрак для достижения идеальной формы.

Существующее программное обеспечение

Так как задача мобильных приложений для применения в медицине не является сравнительно новой, то есть множество наработок в данной области. Рассмотрим несколько из них.

Medisafe – это удобное приложение, которое поможет вам всегда вовремя принимать лекарства согласно инструкции или предписанию врача. Не подвергайте себя и своих близких риску пропустить прием лекарств или принять больше, чем нужно.

Приложение по настроенному пользователем графику будет напоминать, что пора выпить таблетку. Чтобы понять, какую именно, не обязательно брать в руки смартфон: для каждого лекарства можно выбрать свою мелодию. Сервис также напомнит, что средство заканчивается и необходимо купить новую упаковку.

Кроме того, в Medisafe можно вносить данные о весе, давлении, уровне глюкозы в крови, а также сохранять отчёты о приёме лекарств в форматах PDF и Excel, чтобы отправить их врачу.

Достоинства:

— доступность;

— простота применения.

Недостатки:

— нет возможности сравнить предыдущие результаты;

— нет возможности генерации отчетов и их распечатки.

— отсутствие нормальной кастомизирования в приложение

Приложение – Напоминания о приёме лекарств. В приложении можно настраивать напоминания о необходимости выпить лекарство, вести журнал пропущенных и подтверждённых приёмов таблеток, отслеживать график изменений веса, давления.

Достоинства:

— большое количество разнообразных схем приема лекарств;

— простота применения.

Недостатки:

— нет возможности генерации отчетов и их распечатки;

— необходим доступ в интернет.

— некорректное добавление препарата

Приложение - Мои Таблетки Light (My Therapy). В приложении «Мои таблетки» доступен один активный курс. Можно настроить любую схему приёма лекарства, отложить или перенести приём очередной таблетки, сохранять завершённые курсы в архиве или повторять их по необходимости.

Достоинства:

— как напоминания о приёме лекарств;

— отслеживание перепадов настроения;

— журнал состояния здоровья;

— график изменения веса и многое другое.

Недостатки:

— нет возможности сравнить предыдущие результаты;

— является платной.

— не всегда срабатывает напоминание в нужное время

2.2. Характеристика выбранного программного обеспечения Android приложения

Существует несколько самых распространенных операционных систем для смартфонов, такие как, IOS, Android, Windows Phone, BlackBerry, Simbian.

Для создания моего приложения выбран Android, потому что:

— Android – операционная система с открытым исходным кодом.

— Распространенность ОС Android, указана на рисунке 10

— Доступ к разработке любому пользователю.

— Абсолютно бесплатная для разработки.

Рисунке 10 — Доля устройств на различных ОС

Для создания приложения на ОС Android, могут быть выбраны различные среды разработки, такие как, Eclipse, Embarcadero JBuilder, JDeveloper и т.д, но для работы выбрана Android Studio от компании Google. Причиной выбора в качестве среды для разработки приложения была выбрана программа Android Studio. Так как она основана на программном обеспечении IntelliJIDEA от компании JetBrains.

Android Studio – это интегрированная среда разработки (IDE) для работы с платформой Android.

IDE находилась в свободном доступе начиная с версии 0.1, опубликованной в мае 2013, а затем перешла в стадию бета – тестирования, начиная с версии 0.8, которая была выпущена в июне 2014 года. Первая стабильная версия 1.0 была выпущена в декабре 2014 года, тогда же прекратилась поддержка плагина Android Development Tools (ADT) для Eclipse. Android Studio, основана на программном обеспечении IntelliJ IDEA от компании Jet Brains, официальное средство разработки Android приложений. Данная среда разработки доступна для Windows, OS X и Linux.

За счет своей лаконичности по сравнению с XML, формат JSON может быть более подходящим для сериализации сложных структур. Если рассматривать веб – приложения, в таком ключе он уместен в задачах обмена данными как между браузером и сервером (AJAX), так и между самими серверами (программные HTTP-интерфейсы).

Поскольку формат JSON является подмножеством синтаксиса языка Java Script, то он может быть быстро десериализован встроенной функцией eval(). Кроме того, возможна вставка вполне работоспособных JavaScript – функций. В языке PHP, начиная с версии 5.2.0, поддержка JSON включена в ядро в виде функций json_decode() и json_encode(), которые сами преобразуют типы данных JSON в соответствующие типы PHP и наоборот.

В качестве значений в JSON могут быть использованы:

Объект – это неупорядоченное множество пар ключ: значение, заключенное в фигурные скобки «{}». Ключ описывается строкой, между ним и значением стоит символ «:». Пары ключ – значение отделяются друг от друга запятыми.

Массив (одномерный) – это упорядоченное множество значений. Массив заключается в квадратные скобки «[]». Значения разделяются запятыми.

Число. Литералы true, false и null.

Строка – это упорядоченное множество из нуля или более символов юникода, заключенное в двойные кавычки. Символы могут быть указаны с использованием escape – последовательностей, начинающихся с обратной косой черты «\» (поддерживаются варианты \", \\, \/, \t, \n, \r, \f и \b), или записаны шестнадцатеричным кодом в кодировке UTF-8 в виде \uFFFF.

Технология Java включает в себя язык программирования Java, средства трансляции исходного текста программы – исходного кода – в специальную форму, пригодную для исполнения компьютером, и средства исполнения Java – программ на различных платформах, то есть в различных операционных системах и на различном аппаратном обеспечении. Основная особенность JAVA-технологии в том, что преобразованная на этапе трансляции в специальный код

Java – программа полностью «машинно – независима". Если исполняемый код, полученный из программ на других распространенных языках, обычно не пригоден для исполнения компьютером "другой платформы", то к исполняемому коду Java такое ограничение не относится. Правда, необходимо, чтобы для "целевой платформы" имелась реализация так называемой Java-машины – среды исполнения JAVA – программ. JAVA – технологии, активно продвигаемые компанией SUN, получили широкое распространение (далеко не только в web – решения). А платформа независимость Java, позволившая интегрировать средства исполнения Java – программ в браузеры, работающие в самых разных операционных системах, определила распространение Java в качестве элемента Web – технологий. JAVA используется для создания сложных интерактивных элементов, связанных с web – сайтом. Например, на Java реализуются сложные инструменты для работы с базами данных, размещенными в Web. Или графические интерфейсы, требующие вывода сложных интерактивных элементов. И, конечно, многое другое, от сетевых шахматных программ до средств редактирования звуковых файлов.

Существует множество сред разработки, но данная была выбрана ввиду ее удобного графического интерфейса и средств отладки. Также основными особенностями Android Studio является возможность верстки в реальном времени, доступно множество вариантов размеров и разрешений экранов. Присутствует раздел справки, что намного облегчает работу в среде разработки. Встроены инструменты улучшения качества приложений и монетизации. Имеются инструменты для отслеживания эффективности рекламных объявлений. Добавлено средство взаимодействия с бета – тестерами и много другое. Вдобавок ко всему, новая среда разработки обладает функцией Google Cloud Messaging, которая позволяет посылать данные с сервера на Android – устройства через облако. Это отличный способ посылать push – уведомления в приложения. Еще одна полезная вещь в Android Studio – инструмент анализа производительности, который называется MemoryMonitor. Он выдает всю информацию об использовании памяти, что дает возможность оптимизировать работу

приложения. Android Studio – новая и полностью интегрированная среда разработки приложений, выпущенная компанией Google. Данный продукт призван снабдить разработчиков новыми инструментами для создания приложений, а также предоставить альтернативу устаревшей среде разработки Eclipse. Среда разработки поддерживает систему «Codeinjection», данная система позволяет редактировать фрагменты кода с привязкой к различным языкам. Новые функции появляются с каждой новой версией Android Studio.

На данный момент доступны следующие функции:

– Расширенный редактор макетов: WYSIWYG, способность работать с UI компонентами при помощи Drag-and-Drop, функция пред просмотром макета на нескольких конфигурациях экрана;

– Сборка приложений, основанная на Gradle;

– Различные виды сборок и генерация нескольких .apk файлов;

– Рефакторинг кода;

– Статический анализатор кода (Lint), позволяющий находить проблемы производительности, несовместимости версий и другое.

– Встроенный ProGuard и утилита для подписывания приложений;

– Шаблоны основных макетов и компонентов Android;

– Поддержка разработки приложений для Android Wear и Android TV;

– Встроенная поддержка Google Cloud Platform, которая включает в себя интеграцию с сервисами Google Cloud Messaging и App Engine.

– Android Studio 2.1 поддерживает Android N Preview SDK, а это значит, что разработчики смогут начать работу по созданию приложения для новой программной платформы;

– Новая версия Android Studio 2.1 способна работать с обновленным компилятором Jack, а также получила улучшенную поддержку Java 8 и усовершенствованную функцию Instant Run;

– Начиная с Platform-tools 23.1.0 для Linux исключительно 64- Разрядная;

– В Android Studio 3.0 будут по стандарту включены инструменты языка Kotlin основанные на JetBrains IDE.

Android SDK – универсальная оболочка для моделирования и разработки различных программных продуктов под операционную систему Android. Здесь можно как создать, так и протестировать разработанные приложения с использованием достаточно широкого набора встроенных инструментов [31]. Основанный на Linux, Android SDK использует виртуальное устройство для запуска приложений с поддержкой 3G, WiFi, GPS, сенсорных экранов, Bluetooth, компаса, акселерометра и других опций, которые являются сегодня неотъемлемой частью любого Android – устройства. Итогом использования этой среды станет качественно отлаженная и проверенная программа, готовая к публикации на Android Market, что также можно осуществить через данное приложение. Кроме всего перечисленного, нужно отметить полную поддержку мультимедийного аудио и видео контента самых разнообразных форматов, довольно качественное интегрирование с браузерами, работу с базами данных SQLite и многие другие полезные и удобные "штучки", которые станут отменными помощниками любого Android – разработчика.

Для хранения данных используют различные СУБД, такие как, SQLite,

MySQL, Oracle, Microsoft SQL сервер, PostgreSQL, MongoDB, MariaDB, DB2, SAP HANA, ЛИНТЕР, РЕД База Данных.

SQLite – легко встраиваемая в приложения база данных. Так как это система базируется на файлах, то она предоставляет довольно широкий набор инструментов для работы с ней, по сравнению с сетевыми СУБД. При работе с этой СУБД обращения происходят напрямую к файлам (в эти файлах хранятся данные), вместо портов и сокетов в сетевых СУБД. Именно поэтому SQLite очень быстрая, а также мощная благодаря технологиям обслуживающих библиотек.

Типы данных SQLite

— NULL – значение NULL

— INTEGER – знаковое целочисленное значение, использует (1, 2, 3, 4, 6, или 8) байт в зависимости от порядка числа

— REAL – число с плавающей точкой, занимает 8 байт для хранения числа в формате IEEE

— TEXT – текстовая строка, при хранении используются кодировки UTF-8, UTF-16BE или UTF-16LE

— BLOB – тип данных BLOB, массив двоичных данных (предназначенный, в первую очередь, для хранения изображений, аудио и видео).

Преимущества SQLite:

— Файловая структура – вся база данных состоит из одного файла, поэтому её очень легко переносить на разные машины;

— Используемые стандарты – хотя может показаться, что эта СУБД примитивная, но она использует SQL;

— Отличная при разработке и тестировании – в процессе разработки приложений часто появляется необходимость масштабирования. SQLite предлагает всё что необходимо для этих целей, так как состоит всего из одного файла и библиотеки, написанной на языке C.

Недостатки SQLite:

— отсутствие системы пользователей – более крупные СУБД включают в свой состав системы управления правами доступа пользователей. Обычно применения этой функции не так критично, так как эта СУБД используется в небольших приложениях;

— отсутствие возможности увеличения производительности – опять, исходя из проектирования, довольно сложно выжать что-то более производительное из этой СУБД.

MySQL – это самая распространенная полноценная серверная СУБД. MySQL очень функциональная, свободно распространяемая СУБД, которая успешно работает с различными сайтами и веб приложениями. Обучиться использованию этой СУБД довольно просто, так как на просторах интернета вы легко найдете большее количество информации.

Несмотря на то, что в ней не реализован весь SQL функционал, MySQL предлагает довольно много инструментов для разработки приложений. Так как

это серверная СУБД, приложения для доступа к данным, в отличии от SQLite работают со службами MySQL.

Типы данных MySQL:

— TINYINT – очень малые целочисленные значения;

— SMALLINT – малые целочисленные значения;

— MEDIUMINT – средние целочисленные значения;

— INT или INTEGER – стандартные целочисленные значения;

— BIGINT – большие целочисленные значения;

— FLOAT – маленькие значения с плавающей точкой (точность до одного значения после точки). Всегда знаковые значения;

— DOUBLE, BOUBLE PRECISION, REAL – Стандартные значения с плавающей точкой. Всегда знаковые;

— DECIMAL, NUMERIC – распакованное значение с плавающей точкой, всегда знаковое;

— DATE – дата;

— DATETIME – дата и время в одном значении;

— TIMESTAMP – временная отметка timestamp;

— TIME – время;

— YEAR – год, 2 или 4 числа (4 - по-умолчанию);

— CHAR – строковое значение фиксированной длины, справа всегда добавляются пробелы до указанной длины при сортировке;

— VARCHAR – строковое значение переменной длины;

— TINYBLOB, TINYTEXT – значение типа BLOB или TEXT,

255 (2^8 - 1) символов - максимальная длина;

— BLOB, TEXT – значение типа BLOB или TEXT, 65535 (2^16 - 1) символов - максимальная длина;

— MEDIUMBLOB, MEDIUMTEXT – значение типа BLOB или TEXT, 16777215 (2^24 - 1) символов - максимальная длина;

— LONGBLOB, LONGTEXT – значение типа BLOB или TEXT, 4294967296 (2^32 - 1) символов - максимальная длина;

— ENUM – перечисление;

— SET – множество.

Преимущества MySQL:

— Простота в работе – установить MySQL довольно просто. Дополнительные приложения, например GUI, позволяет довольно легко работать с БД;

— Богатый функционал – MySQL поддерживает большинство функционала SQL;

— Безопасность – большое количество функций, обеспечивающих безопасность, которые поддерживается по умолчанию;

— Масштабируемость – MySQL легко работает с большими объемами данных и легко масштабируется;

— Скорость – упрощение некоторых стандартов позволяет MySQL значительно увеличить производительность.

Недостатки MySQL:

— Известные ограничения – по задумке в MySQL заложены некоторые ограничения функционала, которые иногда необходимы в особо требовательных приложениях;

— Проблемы с надежностью – из-за некоторых способов обработки данных MySQL (связи, транзакции, аудиты) иногда уступает другим СУБД по надежности;

— Медленная разработка – Хотя MySQL технически открытое ПО, существуют жалобы на процесс разработки. Стоит заметить, что существуют другие довольно успешные СУБД, созданные на базе MySQL, например MariaDB.

Oracle Database – это объектно-реляционная СУБД (система управления базами данных), созданная компанией Oracle. В настоящее время она имеет множество разных версий и типов.

Достоинства:

— Самые свежие инновации и впечатляющий функционал уже внедрены в этом продукте, поскольку компания Oracle стремится держать планку даже на фоне других разработчиков СУБД;

— СУБД от Оракул является крайне надёжной, фактически это эталон надёжности среди подобных систем.

Недостатки:

— Стоимость Oracle может оказаться непомерно высокой, особенно для небольших организаций;

— Система может потребовать значительных ресурсов уже сразу после установки, поэтому возможно потребуется модернизировать оборудование для внедрения Oracle.

Идеально подходит для крупных организаций, которые работают с огромными базами данных и разнообразными функциями.

Microsoft SQL Server – система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов – Transact – SQL, создан совместно Microsoft и Sybase. Transact – SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия.

Microsoft SQL – сервер. Это система управления базами данных, движок которой работает на облачных серверах, а также локальных серверах, причем можно комбинировать типы применяемых серверов одновременно. Вскоре после выпуска Microsoft SQL сервер 2016, Microsoft адаптировала продукт для операционной системы Linux, а на Windows – платформе он работал изначально.

Одной из уникальных особенностей версии 2016 года является temporal data support (временная поддержка данных), которая позволяет отслеживать изменения данных с течением времени. Последняя версия Microsoft SQL – сервер поддерживает dynamic data masking (динамическую маскировку данных), кото-

рая гарантирует, что только авторизованные пользователи будут видеть конфиденциальные данные.

Достоинства:

— Продукт очень прост в использовании;

— Текущая версия работает быстро и стабильно;

— Движок предоставляет возможность регулировать и отслеживать уровни производительности, которые помогают снизить использование ресурсов;

— Вы сможете получить доступ к визуализации на мобильных устройствах;

— Он очень хорошо взаимодействует с другими продуктами Microsoft.

Недостатки:

— Цена для юридических лиц оказывается неприемлемой для большей части организаций;

— Даже при тщательной настройке производительности корпорация SQL Server способен занять все доступные ресурсы;

— Сообщается о проблемах с использованием службы интеграции для импорта файлов.

Идеально подходит для: крупных организаций, которые уже используют ряд продуктов Microsoft.

Postgre SQL является одним из нескольких бесплатных популярных вариантов СУБД, часто используется для ведения баз данных веб-сайтов. Это была одна из первых разработанных систем управления базами данных, поэтому в настоящее время она хорошо развита, и позволяет пользователям управлять как структурированными, так и неструктурированными данными. Может быть использован на большинстве основных платформ, включая Linux. Прекрасно справляется с задачами импорта информации из других типов баз данных с помощью собственного инструментария.

Движок БД может быть размещен в ряде сред, в том числе виртуальных, физических и облачных. Самая свежая версия, Postgre SQL 9.5, предлагает об-

работку больших объемов данных и увеличение числа одновременно работающих пользователей. Безопасность была улучшена благодаря поддержке DBMS_SESSION.

Достоинства:

— Является масштабируемым и способен обрабатывать терабайты данных;

— Поддерживает формат json;

— Существует множество предопределенных функций;

— Доступен ряд интерфейсов.

Недостатки:

— Документация туманна, поэтому, возможно, ответы на некоторые вопросы придется искать в интернете;

— Конфигурация может смутить неподготовленного пользователя;

— Скорость работы может падать во время проведения пакетных операций или выполнения запросов чтения.

Идеально подходит для организаций с ограниченным бюджетом, но квалифицированными специалистами, когда требуется возможность выбрать свой интерфейс и использовать json.

Mongo DB – еще одна бесплатная база данных, которая имеет коммерческую версию – Mongo DB, она предназначена для приложений, которые используют как структурированные, так и неструктурированные данные. Ядро является очень гибким и работает при подключении базы данных к приложениям через драйверы Mongo DB. Существует широкий выбор доступных драйверов, поэтому легко найти драйвер, который будет работать с требуемым языком программирования.

Поскольку изначально система Mongo DB не была разработана для обработки моделей реляционных данных (хотя может это выполнять), могут возникнуть проблемы производительности, если вы попытаетесь использовать её таким образом. Однако, движок предназначен для обработки различных дан-

ных, которые нельзя отнести к реляционным, и может хорошо справляться там, где другие движки работают медленно или бессильны.

Mongo DB 3.2 – это последняя версия, и она имеет новую подключаемую систему движков хранения. Документы могут быть проверены в процессе обновления или выполнения вставок, а функции текстового поиска были улучшены. Новая способность частичного индексирования может привести к более высокой производительности, уменьшая размер индексов.

Достоинства:

— Скорость и простота в использовании;

— Движок поддерживает json и другие традиционные документы NoSQL;

— Данные любой структуры могут быть сохранены/прочитаны быстро и легко.

Недостатки:

— SQL не используется в качестве языка запросов;

— Инструменты для перевода SQL-запросов в MongoDB доступны, но их следует рассматривать именно как дополнение;

— Программа установки может занять много времени.

Подходит для организаций, работающих с разнородными данными, которые тяжело поддаются классификации. Для внедрения потребуются высококлассные специалисты.

Maria DB – эта СУБД является бесплатной, но, как и многие другие бесплатные приложения, предлагает платные версии. Есть множество доступных плагинов расширений, пожалуй, это самая быстроразвивающаяся СУБД на данный момент.

Maria DB фактически – это ответвление от СУБД My SQL, разрабатываемое сообществом под лицензией GNU GPL. Разработку и поддержку Maria DB осуществляет компания Maria DB Corporation Ab и фонд Maria DB Foundation. Толчком к созданию стала необходимость обеспечения свободного статуса СУБД, в противовес политике лицензирования My SQL компанией

Oracle. Система лицензирования Maria DB обязывает участников, желающих добавить свой код в основную ветку СУБД, обмениваться своими авторскими правами с Maria DB Foundation для охраны лицензии и возможности создавать критические исправления для My SQL. Ядро базы данных позволяет делать выбор из нескольких систем хранения, и это делает использование ресурсов более оптимизированным, что повышает производительность запросов и обработки. В состав MariaDB включена подсистемы хранения данных XtraDB для возможности замены InnoDB, как основной подсистемы хранения. Также включены подсистемы Aria, PBXT и FederateX. Она полностью совместима с MySQL, и прекрасно подходит в качестве замены, т.к. полностью соответствует как набор команд, так и API. Многие разработчики MySQL были вовлечены в процесс разработки, а сейчас принимают участие в развитии.

Достоинства:

— Система работает быстро;

— Индикаторы дадут вам знать, как обрабатывается запрос;

— Расширяемая архитектура и плагины позволяют настраивать инструмент в соответствии с вашими потребностями;

— Шифрование доступно в сети, сервере и уровне приложения.

Недостатки:

— На данный момент стабильность ниже, чем у MySQL, поэтому даже на новых проектах можно рекомендовать устанавливать mysql;

— Движок довольно новый, поэтому пока нет никаких гарантий дальнейших обновлений;

— Как и во многих других бесплатных базах данных, вам придется платить за поддержку.

Идеальна как альтернатива MySQL, если MySQL не устраивает по каким-то причинам.

Созданная компанией IBM, DB2 представляет собой СУБД, которая имеет возможности NoSQL, и может читать JSON и XML-файлы. Ввиду того, что

система разрабатывалась для серверов компании IBM модельного ряда iSeries, логично, что система работает на Windows, Linux и Unix.

Диалект языка SQL, используемый в DB2 за редкими исключениями строго декларативен, система снабжена многофазовым оптимизатором, строящим по этим декларативным конструкциям план выполнения запроса. В диалекте SQL DB2 отсутствуют подсказки оптимизатору, мало развит (а долгое время вообще отсутствовал) язык хранимых процедур, и, таким образом, всё направлено на поддержание декларативного стиля написания запросов. Язык SQL DB2 при этом является вычислительно полным, то есть потенциально позволяет в декларативной форме определять любые вычислимые соответствия между исходными данными и результатом. Это достигается в том числе за счёт использования табличных выражений, рекурсии и других развитых механизмов манипулирования данными.

Оптимизатор DB2 широко использует статистику распределения данных в таблицах (если процесс её сбора был выполнен администратором базы данных), поэтому один и тот же запрос на языке SQL может быть оттранслирован в совершенно различные планы выполнения в зависимости от статистических характеристик данных, которые он обрабатывает.

В рамках концепции повышения уровня интеграции средств безопасности в компьютерной системе, DB2 не имеет собственных средств аутентификации пользователей, интегрируясь со средствами операционной системы или специализированными серверами безопасности. В рамках DB2 осуществляется только авторизация пользователей, аутентифицированных системой.

DB2 является единственной реляционной СУБД общего назначения, имеющей реализации на аппаратно-программном уровне (система IBM i; также в оборудовании мэйнфреймов IBM System z реализуются средства поддержки DB2).

Современные версии DB2 обеспечивают расширенную поддержку использования данных в формате XML, в том числе операции с отдельными элементами документов XML.

Текущая версия DB2 – это LUW 11.1, которая предлагает разнообразные улучшения и доработки. Одно из них, ускорение Blu, которое предназначено, для того чтобы сделать эту базу данных быстрее. Пропуск данных предназначен для повышения быстродействия системы с большим количеством данных, чем может она может вместить в себя. Последняя версия DB2 также обеспечивает усовершенствованные функции аварийного восстановления, совместимости и аналитики.

Достоинства:

— Blu Acceleration позволяет грамотно задействовать ресурсы для объёмных баз данных;

— Может быть размещена в облачном хранилище, на физическом сервере, или же и там, и там одновременно;

— Несколько задач могут выполняться одновременно с помощью планировщика задач;

— Коды ошибок и коды завершения позволяют легко отследить, какие задания выполняются или выполнились с помощью планировщика задач.

Недостатки:

— Цена за пределами бюджета многих физических лиц и небольших организаций;

— Сторонние приложения или дополнительное программное обеспечение требуется, для того чтобы заставить функционировать кластеры или несколько вторичных узлов;

— Базовая поддержка доступна только в течение трех лет; после этого, вы должны заплатить за это.

Подходит для: крупных организаций, которые планируют выжимать максимум из имеющихся ресурсов и обрабатывают большие БД.

Разработанная компанией SAP SE, SAP HANA – это СУБД, с движком ориентированным на работу со столбцами, работающая с родными данными SAP и чужими данными. Ядро ориентировано на сохранение и извлечение дан-

ных из приложений и других источников на нескольких уровнях хранения. Система может быть размещена на физических серверах или в облаке.

Достоинства:

— Она поддерживает SQL, OLTP и OLAP;

— Ядро снижает требования к ресурсам за счет использования сжатия;

— Данные хранятся в памяти, сокращая время доступа, в некоторых случаях, значительно;

— Отчеты формируются в реальном времени;

— Может взаимодействовать с рядом других приложений.

Недостатки:

— Поставляется только для работы на ограниченном наборе оборудования: производитель производит сертификацию определённых моделей серверных узлов с конкретной конфигурацией;

— Высокая стоимость лицензий даже если речь идёт о плате за программное обеспечение предприятия;

— Это всё ещё относительный новичок, требуются постоянные обновления.

Идеально подходит для: организаций, которые захватывают данные из приложений и при этом неограниченны в бюджете.

«Линтер» – российская СУБД, реализующая стандарт SQL:2003 (за исключением не скалярных типов данных и объектно-ориентированных возможностей) и поддерживающая большинство операционных систем, в том числе семейство Windows, различные версии UNIX, ОС реального времени (включая QNX).

К особенностям можно отнести защиту данных: 2 класс защиты данных от несанкционированного доступа и 2 уровень контроля отсутствия не декларированных возможностей. Мандатный контроль доступа к данным на уровне таблиц, столбцов записей и отдельных полей записей. Управление доступом к рабочим станциям и устройствам хранения информации. Контроль доступа к СУБД по расписанию. Управление протоколированием операций над БД

(аудит). Аутентификация пользователей через LDAP, Kerberos, средствами операционной системы. Хеширование паролей по алгоритму FIPS 180-2 SHA-224.

Репликация асинхронная (в том числе и двунаправленная), возможна репликация с другими БД через ODBC.

Имеет утилиты конвертации, работающие через ODBC и ADO.NET. Конвертер из DBF-формата. Конвертер модели данных (из ERwin в ЛИНТЕР).

Достоинства:

— Российская разработка;

— Она поддерживает SQL:2003;

— Облегчается конвертация при переходе с других СБУД;

— Рекомендована "Единым реестром российских программ".

Недостатки:

— Падение эффективности в случае высокой динамики изменений.

Идеально подходит для: отечественных организаций, которые работают с конфиденциальными и персональными данными.

«РЕД База Данных» — российская СУБД, работает на всех основных платформах и ОС (Windows, Linux, BSD Unix, IBM AIX, HP-UX, Sun Solaris и т.д.). Система модульная. Имеет открытый исходный код.

Возможность «горячего» резервного копирования и инкрементного резервного копирования. Сертифицирована ФСТЭК России. Соответствует отечественным требованиям по защите информации. Может использоваться при создании информационных систем до класса защищенности 1Г включительно и при создании информационных систем персональных данных до 1 класса включительно. Полное соответствие принципам атомарности, непротиворечивости, изоляции, долговечности (ACID).

Имеются модули сопряжения практически для всех используемых сред разработки (драйверы ODBC, JDBC, C/C++, C#, Java, Delphi, PHP, Python, Perl, VB, и т.д.), результатов тестов этих модулей и гарантия стабильной работы.

Возможность работы во «встроенном» в ПО (embedded) локальном режиме в виде библиотеки DLL без отдельной установки и настройки СУБД, в т.ч. поддержка встраивания в виртуальную машину Java.

Достоинства:

— Российская разработка;

— Соответствует отечественным требованиям по защите информации;

— Высокое быстродействие, сравнимое с лидерами рынка;

— Возможность хранения базы данных в одном отдельном файле.

Недостатки:

— Низкая распространённость.

Идеально подходит для: отечественных организаций (включая оборонные), которые работают с конфиденциальными и персональными данными. Идеально подходит для крупных организаций, которые работают с огромными базами данных и разнообразными функциями.

Поэтому для работы была выбрана SQ Lite, так как:

— поддерживается хостингом;

— гибкость, которая обеспечивается поддержкой большого количества типов таблиц;

— продукт класса Open Source (открытые исходные тексты), который

— можно получить бесплатно расписать более обширно.

3 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ РАЗРАБОТАННОГО АЛГОРИТМА РЕШЕНИЯ ЗАДАЧИ

3.1. Этапы практической разработки программного продукта

Приложение для ОС Android состоит из набора активностей, каждой из которых соответствует экран приложения. Каждая активность представлена в проекте классом, реализованном на языке Java, хранящемся в одноименном файле с расширением .java. Каждой активности соответствует xml файл-описание. В xml – файле описано в виде xml кода расположение визуализируемых объектов. При запуске активности система Android автоматически распознает размер экрана мобильного устройства и приводит выводимый контент в соответствие с разметкой, описанной в xml файле. Таким образом, одна и та же активность будет выглядеть одинаково независимо от диагонали используемого устройства. Также, для каждого приложения Android должен существовать xml файл, в котором в виде xml кода будут прописаны минимальные требования к системе, а также активность, вызываемая при запуске приложения.

На первом этапе разработки необходимо изучить архитектуру android и ее ключевые особенности.

На втором этапе разработки необходимо было проанализировать какие существуют аналогичные решения по данной задаче.

На третьем этапе разработки необходимо было выполнить написание технического задания согласно стандартам.

На четвертом этапе разработки выполняем, проектирование и разработку дизайна для данного продукта.

В результате выполнения четвертого этапа разработки получим следующее:

— карта экранов;

— статичный или интерактивный прототип приложения;

— отрисованные экраны и элементы интерфейса.

На пятом этапе разработки выполняем разработку, и написание программного кода, где будет происходить разработка клиентской часть сервиса так и серверной часть приложения.

На клиентской части сервиса будет выполняться, разработка интерфейса пользователя и бизнес-логика приложения.

На серверной части будет разработана система, которая будет отвечать за передачу данных между пользователями или ресурсами.

На шестом этапе разработки необходимо было выполнить тестирование и стабилизацию работы данного продукта, устранить все ошибки, возникшие при тестировании, (устранены с помощью стабилизации) чтобы в дальнейшем при эксплуатации избежать некорректной и нестабильной работы данного продукта.

3.2. Тестирование программного продукта мобильного приложения

Для теста был разработан основной сценарий функциональных тестов:

– Проверить корректность работы обязательных полей соответственно

Рисунок.11. Исходя из рисунков видно, что обязательные поля отображаются и корректно работают.

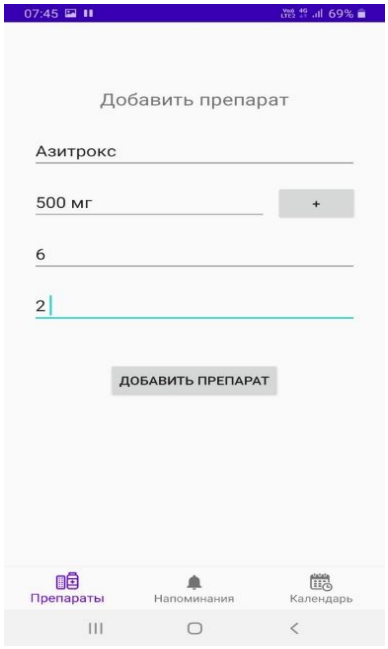


Рисунок 11 — Проверка корректности работы полей

Когда пользователь, указав все необходимые данные и добавил препарат в базу соответственно Рисунок. 12. Исходя из рисунков видно, что препарат был добавлен.

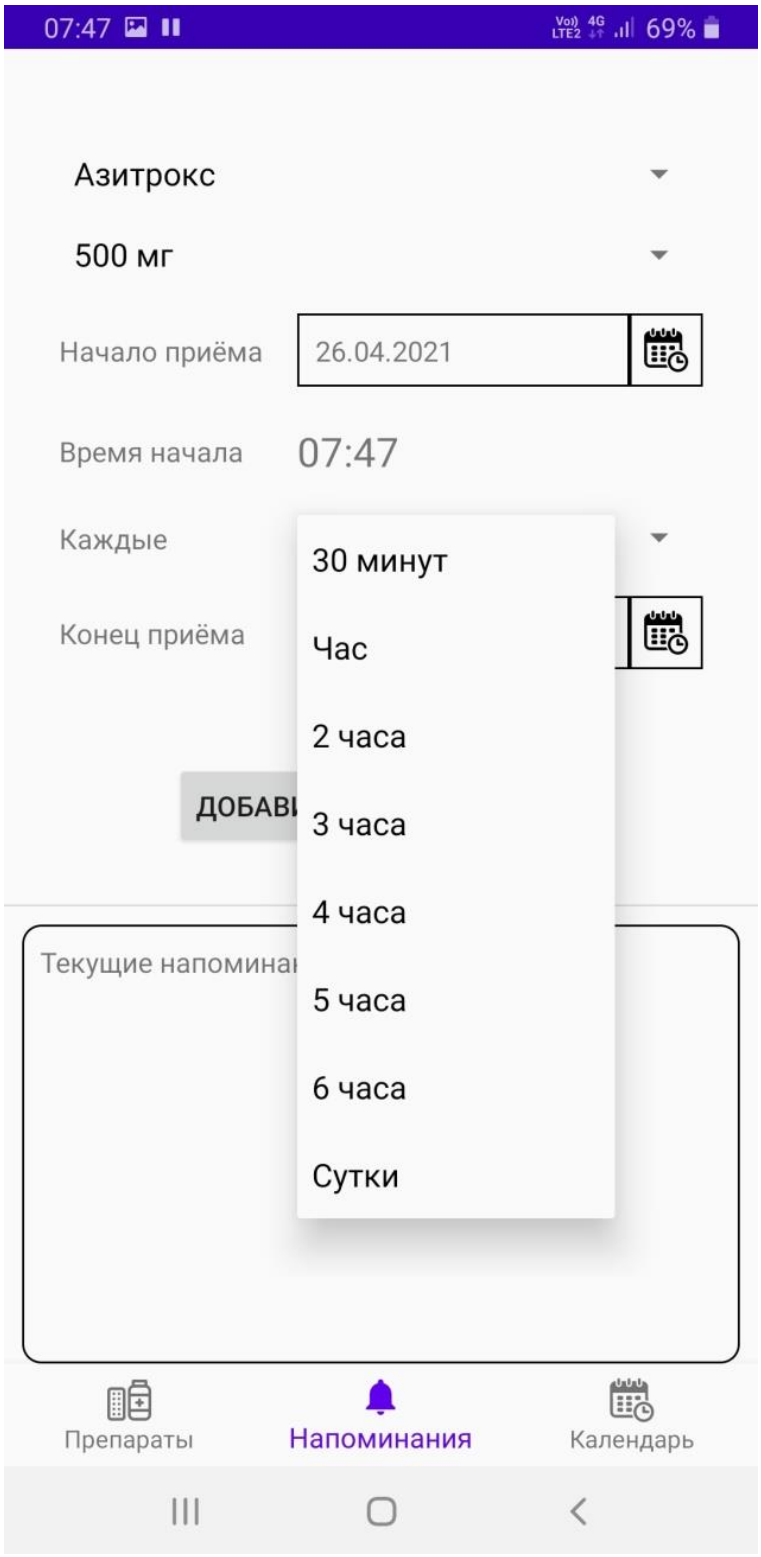


Рисунок 12 — Добавление препарата

Убедиться, что работа приложения во время запуска/выхода удовлетворяет основным требованиям. Соответственно Рисунок 13-14. Исходя из рисунков видно, что при запуске приложения и выходе из приложения удовлетворяет требованиям.

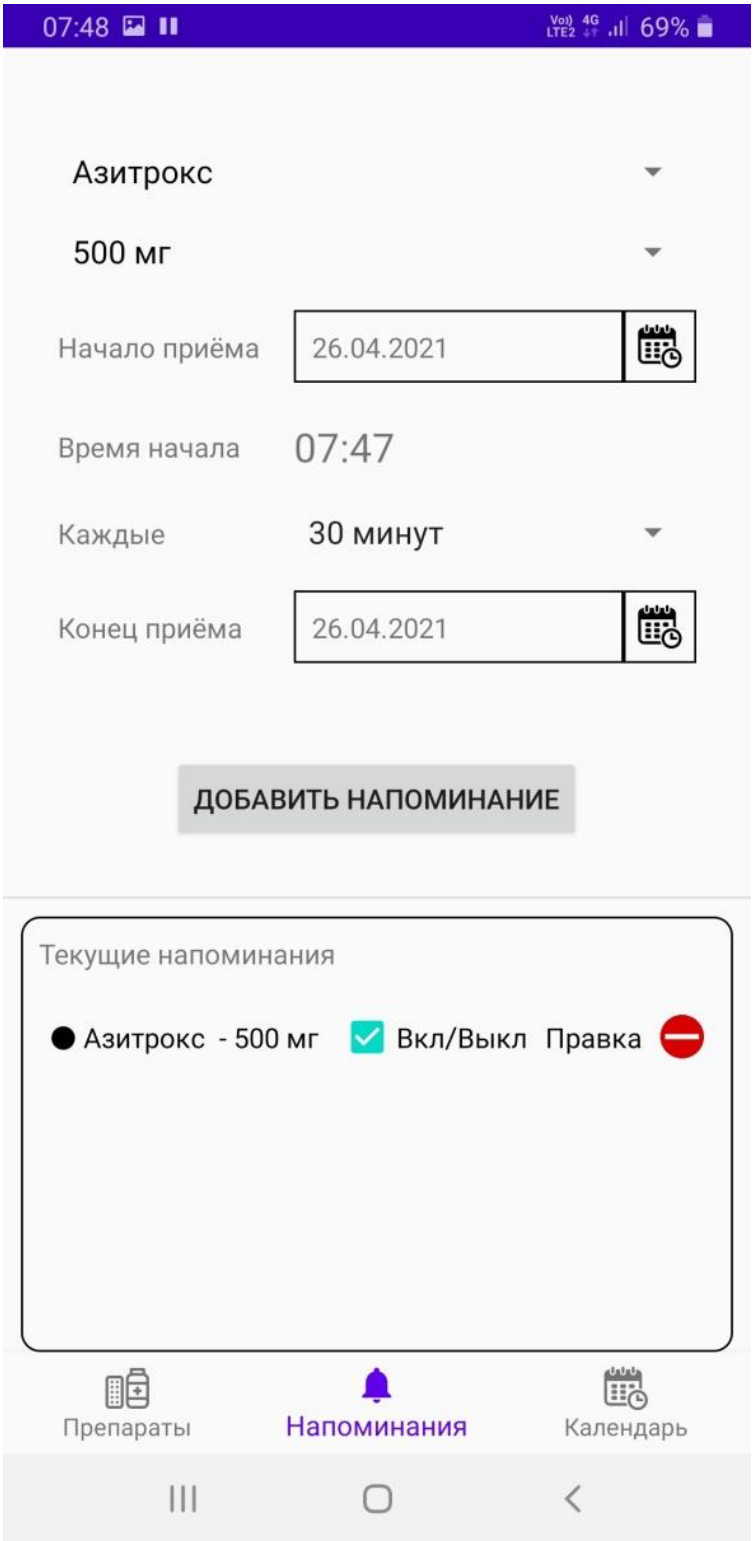


Рисунок 13 — Проверка запуска и выхода из приложения

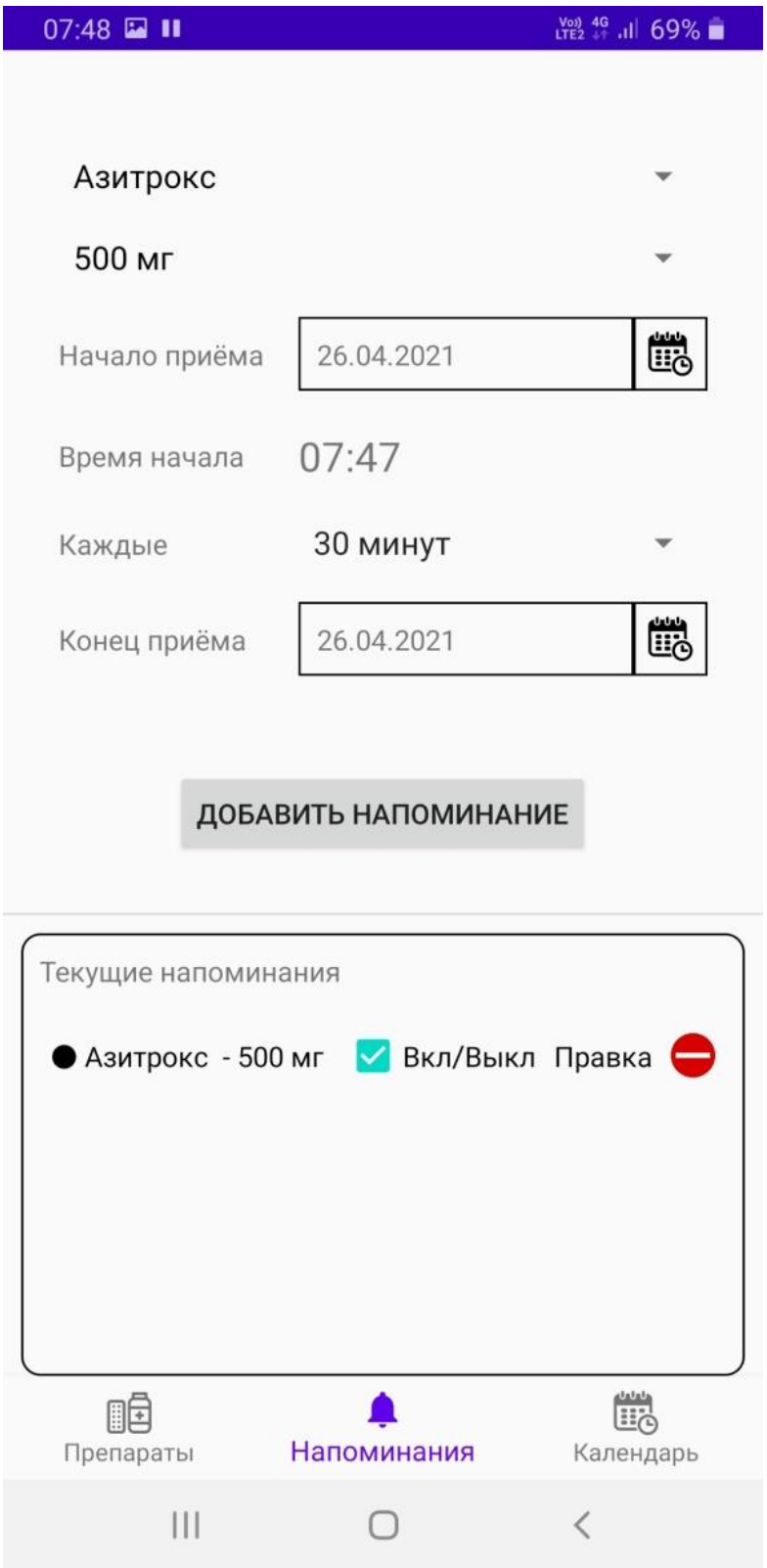


Рисунок 14 — Проверка данных

Убедиться, что устройство работает в многозадачном режиме, когда это необходимо. Соответственно Рисунок 15-16. Исходя из рисунков видно, что

при работе приложения в режиме многозадачности работает стабильно без каких-либо ошибок.

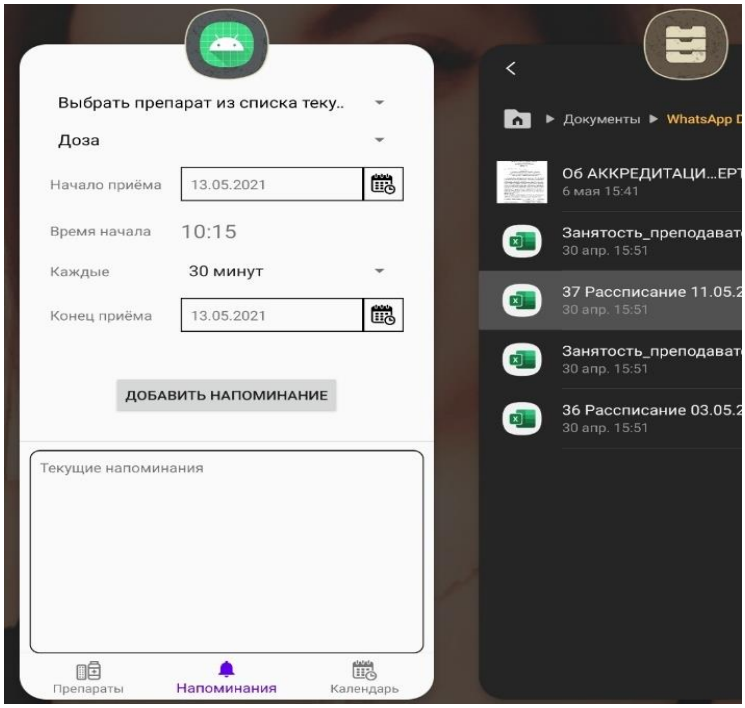


Рисунок 15 — Проверка многозадачности

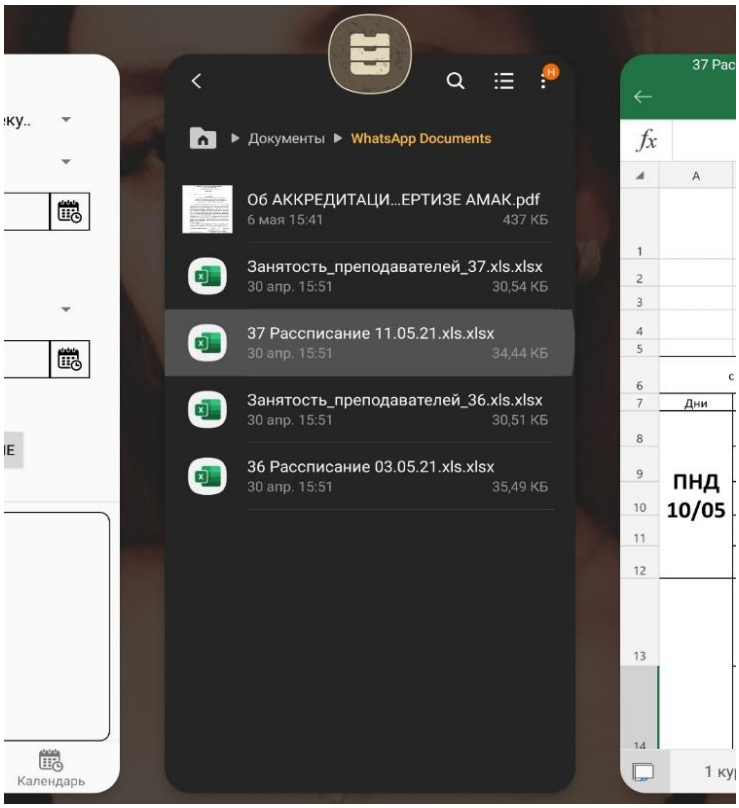


Рисунок 16 — Проверка многозадачности с переходом из одного приложения в другое

Проверить адекватность работы сценариев прокрутки страницы. Соответственно Рисунок 17-19. Исходя из рисунков видно, что при работе

Приложения при выполнении прокрутки страниц выполняется все плавно без каких-либо торможений приложения.

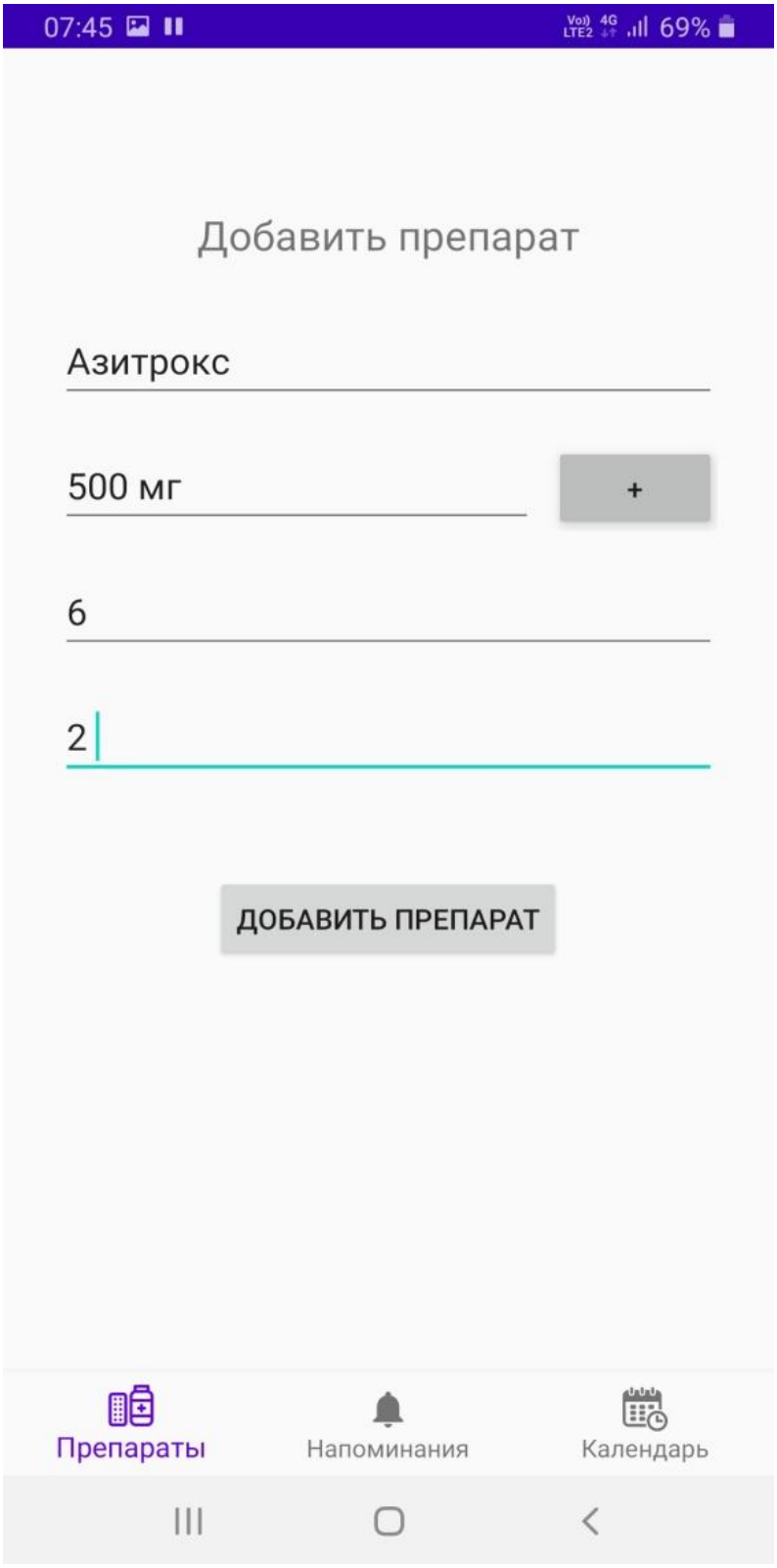


Рисунок 17 — Добавление препарата

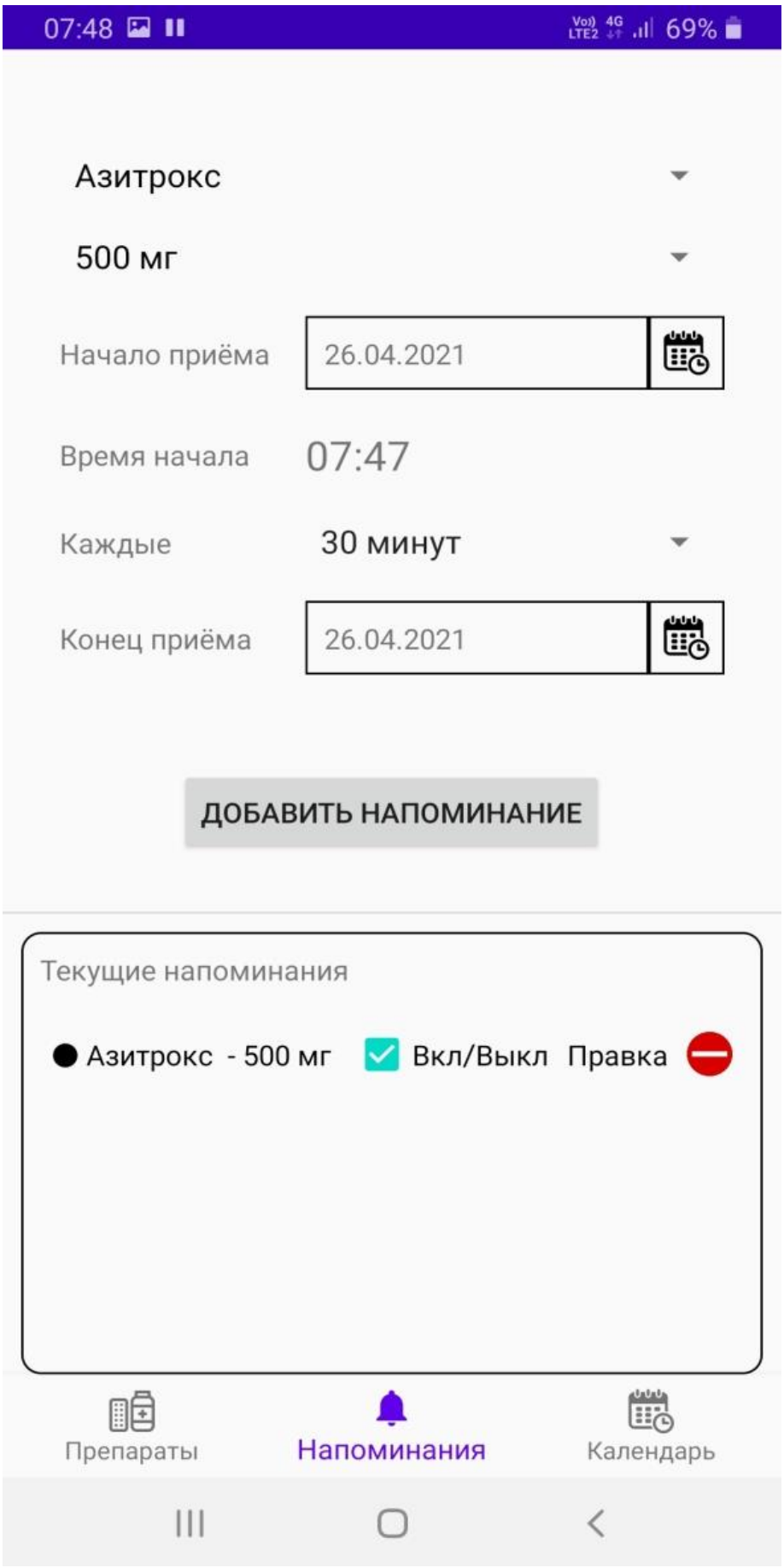


Рисунок 18 — Переход со страницы препараты на напоминания

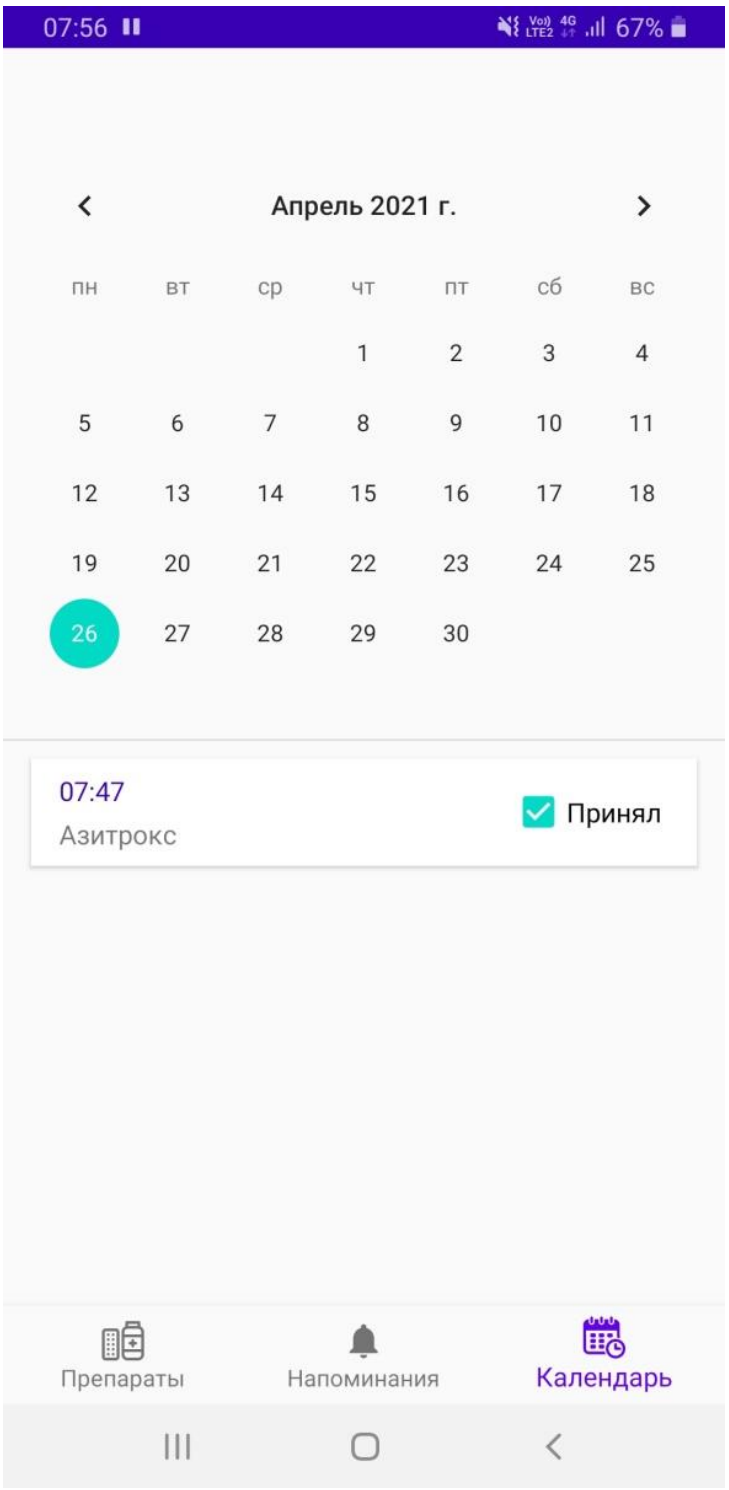


Рисунок 19 — Переход со страницы напоминания на календарь

Убедиться, что установленное приложение не препятствует нормальной работе других приложений и не съедает их память. Соответственно Рисунок 20-23. Исходя из рисунков видно, что при работе приложения не препятствует ра-

боте других приложений и не используется лишнюю память других приложений.

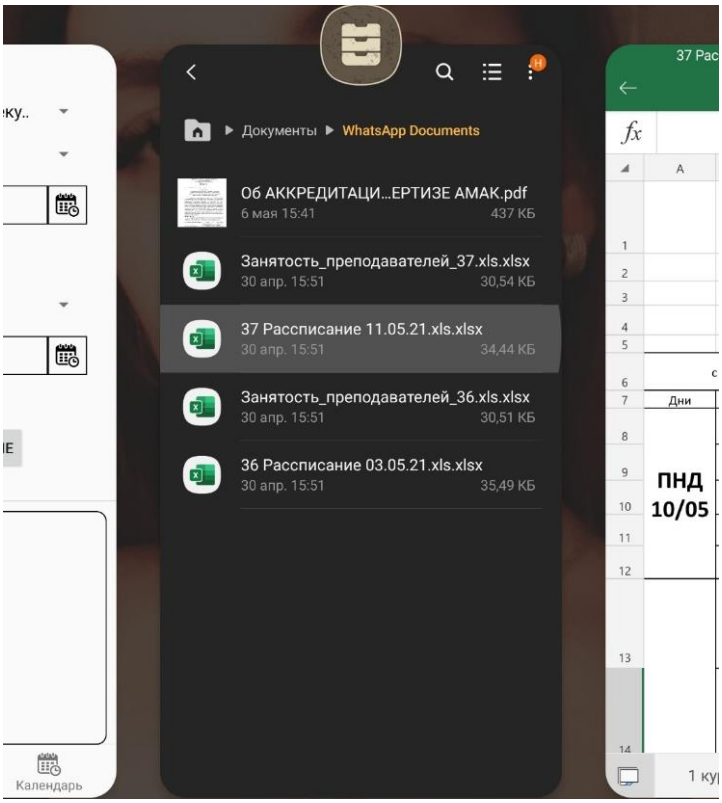


Рисунок 20 — Запуск нескольких приложений

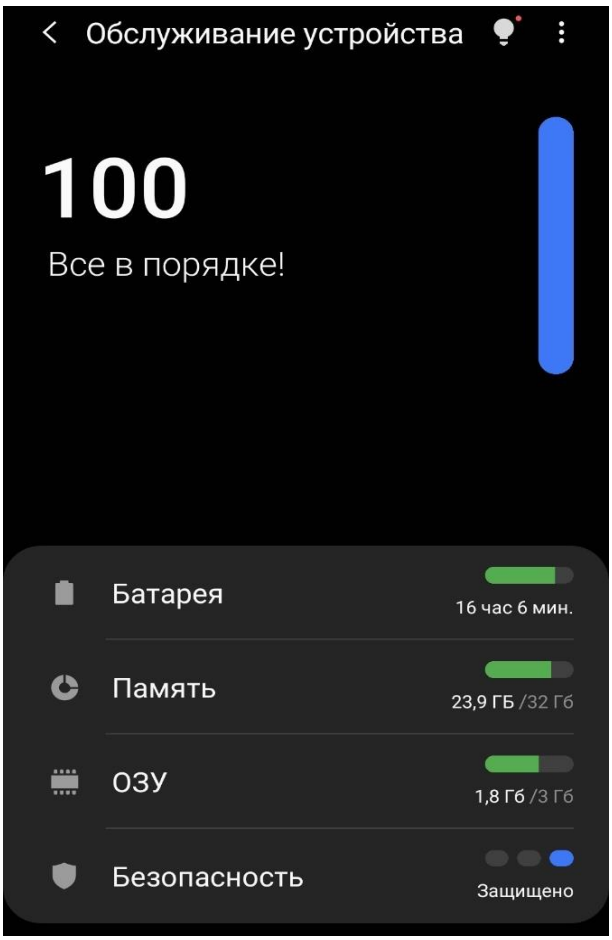


Рисунок 21 — Проверка устройства в обслуживание

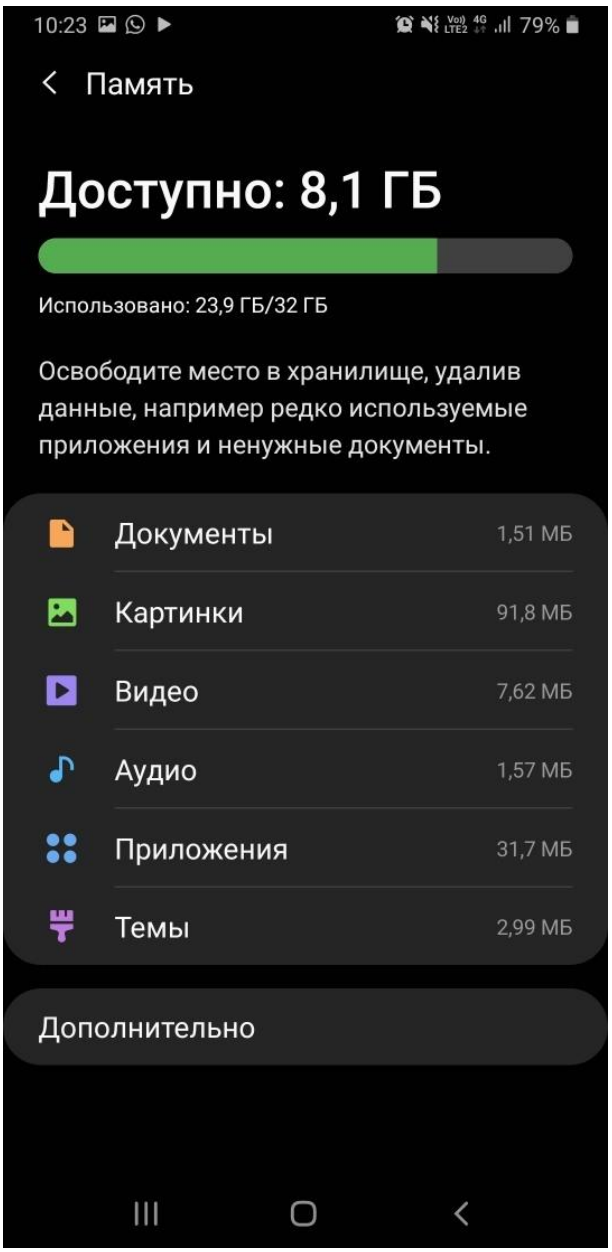


Рисунок 22 — Проверка устройства в памяти

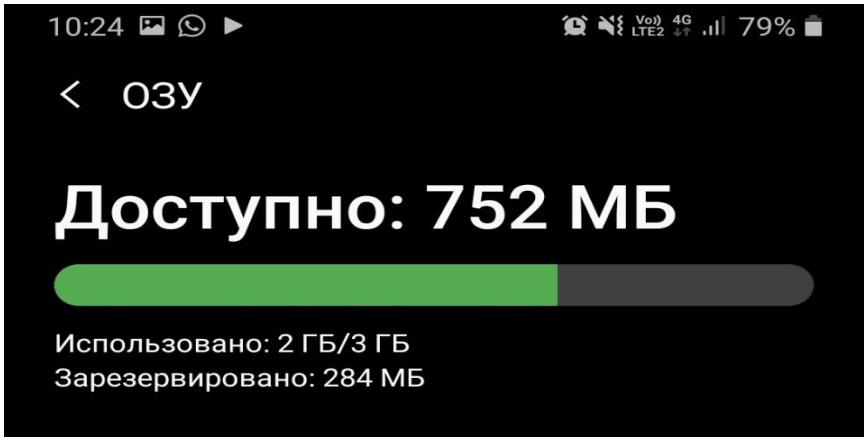


Рисунок 23 — Проверка ОЗУ в устройстве

Тестирование показало работоспособность программного продукта на указанном программном и аппаратном обеспечении, и корректность.

Руководство по установке программного продукта

Перед запуском установочного файла необходимо загрузить его на наше устройство после того, когда был загружен установочный файл на устройство, еще необходимо чтобы на устройстве был архиватор для открытия и установки нашего приложения т.к. не на всех устройствах есть приложения для того чтобы открывать файлы такого формата. Непосредственно переходим к его установки запускаем наш архиватор переходим во вкладку загрузки и находим наш файл, который имеет имя app-release2.1.apk. после запускаем наш установочный файл app-release2.1.apk. будет предложено на выбор с помощью чего открыть данное приложение выбираем установщик пакетов после чего начинается процесс установки нажимаем кнопку установить и приложение начинается устанавливаться по окончанию установки будет предложено на выбор открыть приложение или нажать готово. Соответственно на рисунке 24-25.

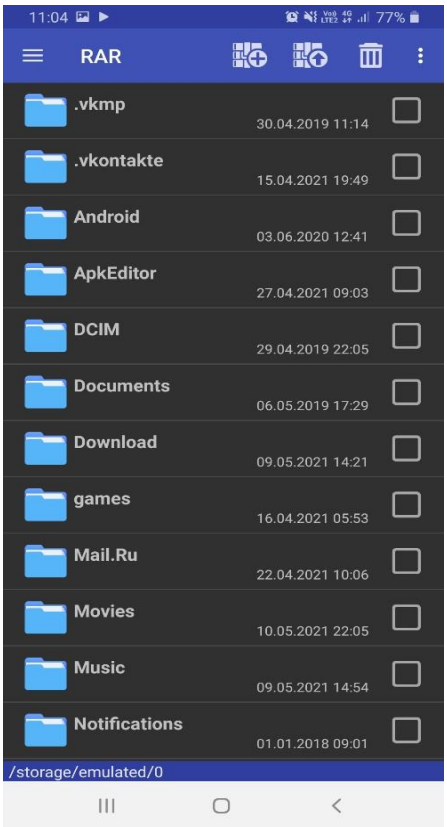


Рисунок 24 — Запуск архиватора WinRAR

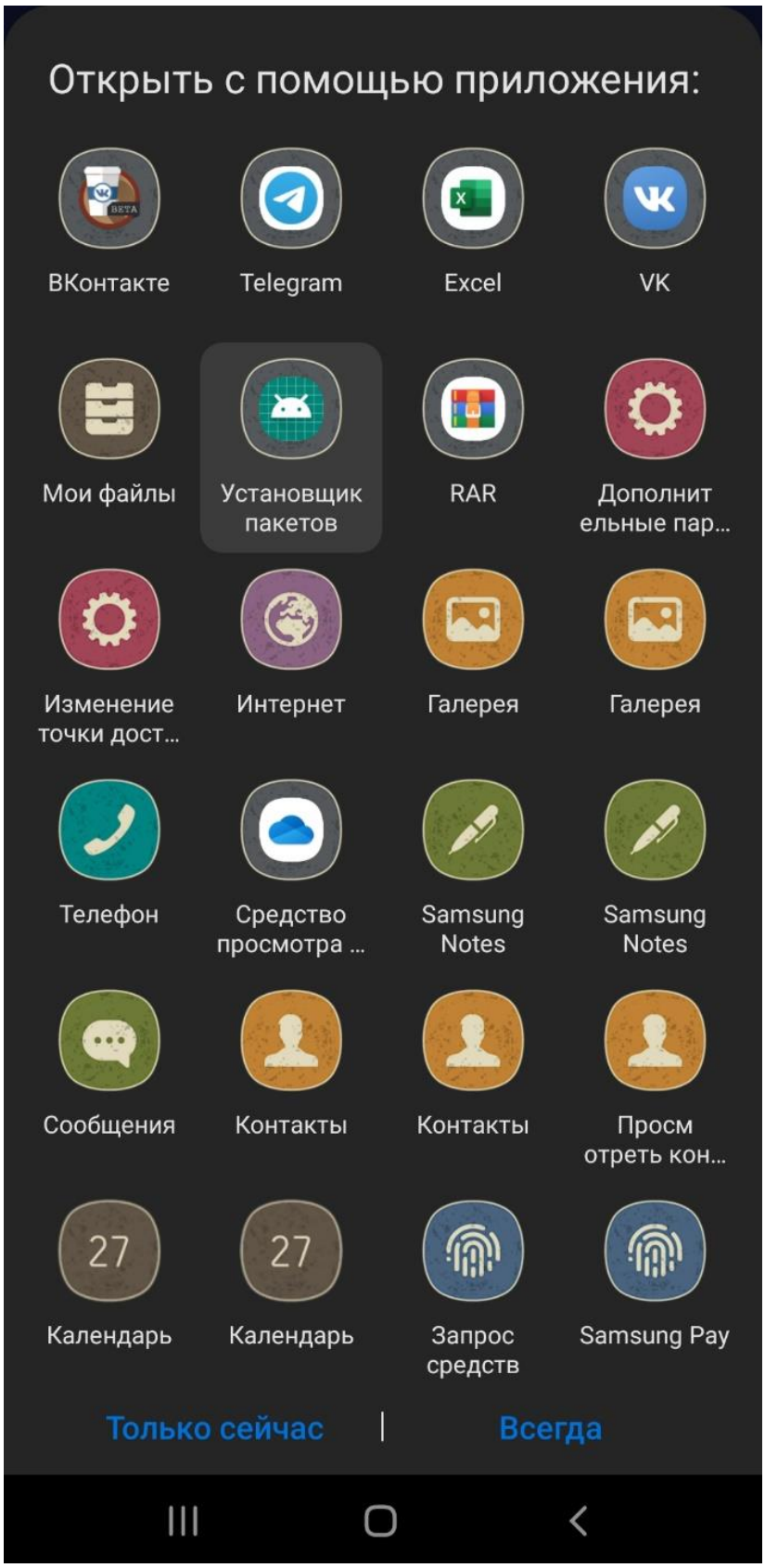


Рисунок 25 — Выбор приложения для открытия файла

После запуска установочного файла всплывает уведомление об установке приложения. Есть выбор установить либо отменить установку. Соответственно на рисунке 26.

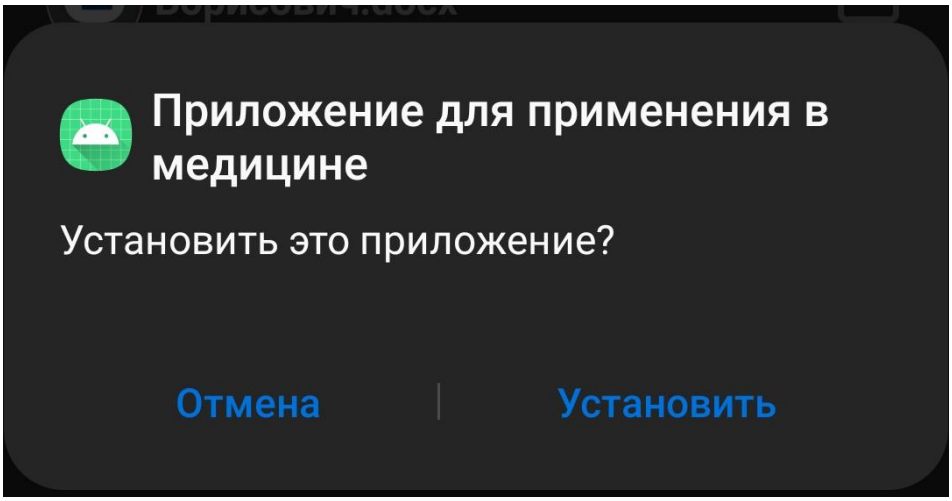


Рисунок 26 — Выбор установки

Нажимаем кнопку установить, и начинается процесс установки, после чего приложение будет установлено на наше устройство и им можно будет пользоваться. Соответственно на рисунках 27-28.

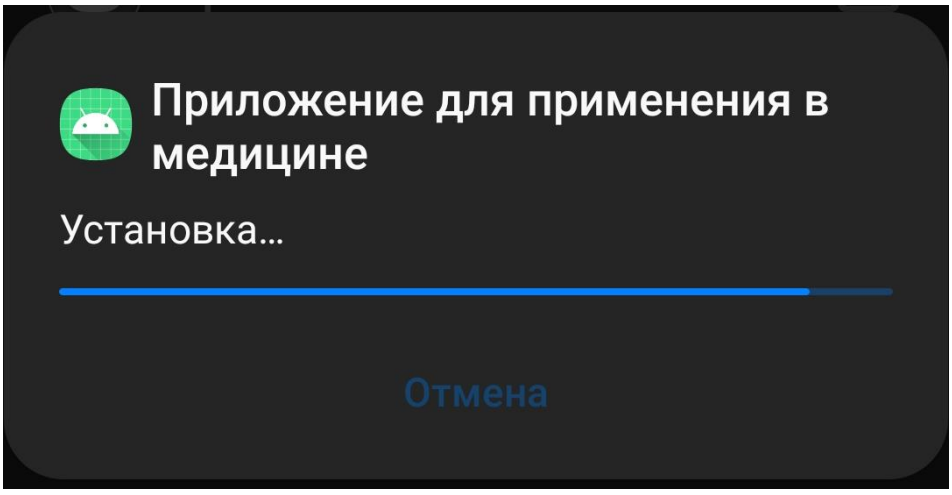


Рисунок 27 — Процесс установки

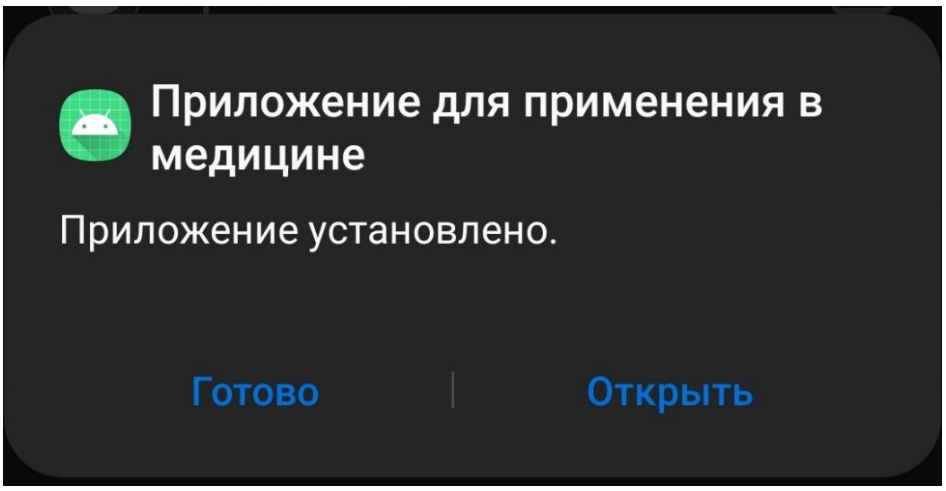


Рисунок 28 — Завершение установки

3.3. Руководство пользователя

Главная панель проектирования. После входа в мобильное приложение перед пользователем откроется раздел «главная», который соответственно на рисунке 29. Для начала пользователь вводит данные о препарате, после чего препарат будет добавлен. В этом разделе доступны опции: − Ввода данных названия о препарате. Нужно заполнить название, выбрать график приема, дозировку, количество таблеток в упаковке и срок годности; − Установка назначенной дозировки – эта опция поможет регулировать дозировку препарата (увеличение, уменьшение).

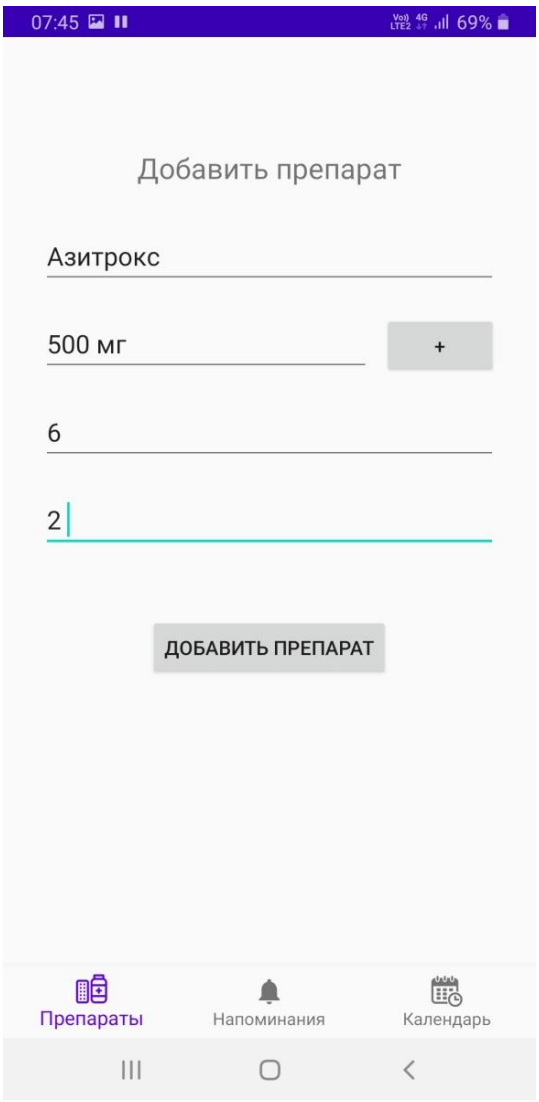
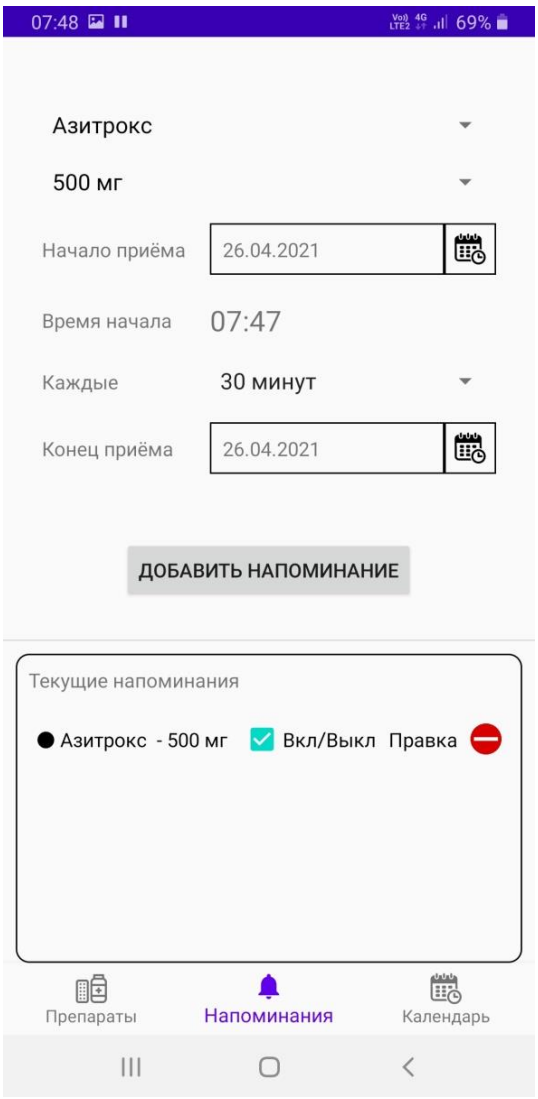


Рисунок 29 — Главная панель

После того как пользователь указал всю необходимую информацию и добавил все данные о препаратах, он попадает на последующую форму напоминания, где может внести все препараты, которые использует. Эта опция служит для того, чтобы при построении личного графика можно было просто и быстро выбрать их. Функционал данной опции наглядно представлен в соответствии на рисунке 30.



Рисунке 30 — Форма напоминания

На рисунке 31 и 32 представлена опция установки напоминания о приеме препарата. На данном этапе пользователь может выбрать препараты из текущего списка в соответствии с назначенной дозировкой. Указать дату начала и окончания приема. А также время, в которое необходимо принять тот или иной препарат. Если пользователь допустит ошибку или дозировку необходимо будет изменить, в приложении реализована опция редактирования. Также у пользователя при добавлении напоминания есть возможность выбора двух форм ввода времени. Пользователь самостоятельно выбирает вариант, который будет для него более удобней и практичней.

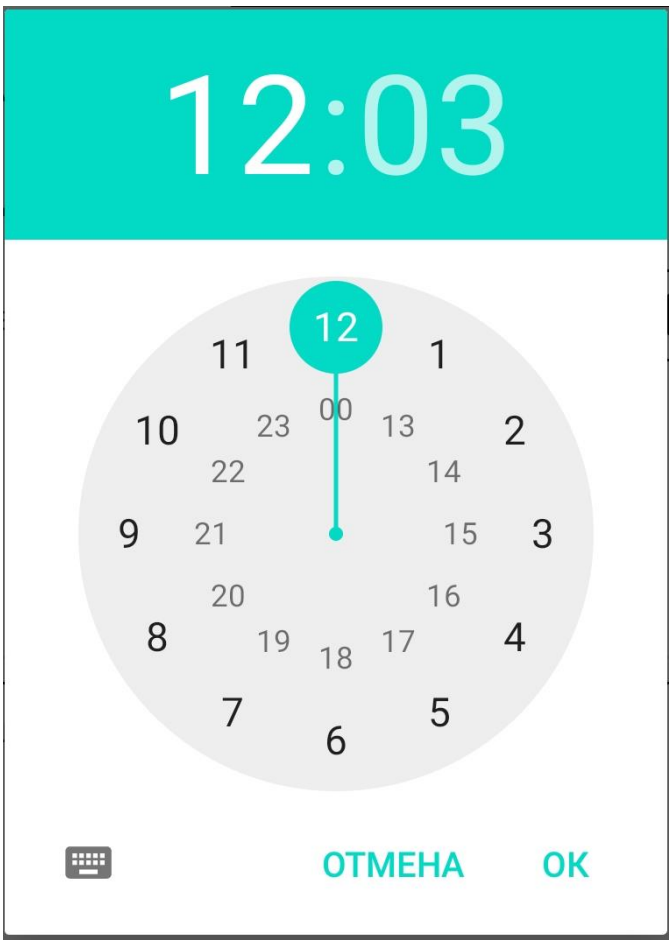


Рисунок 31 — Форма ввода времени

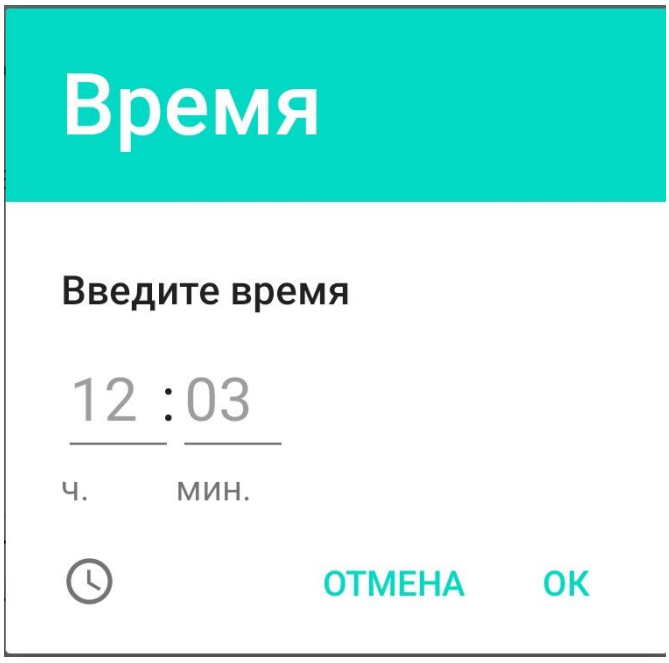
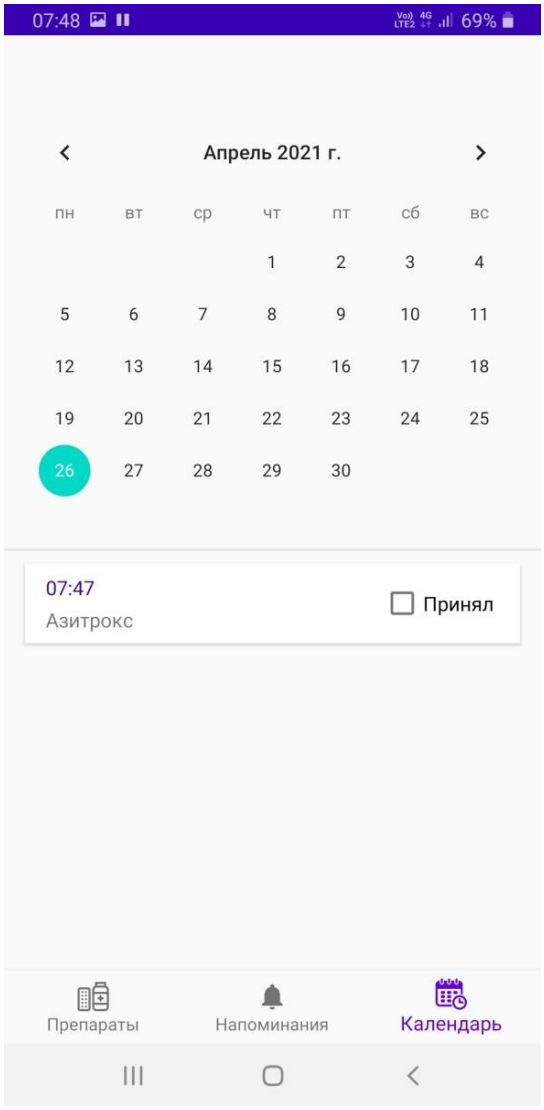


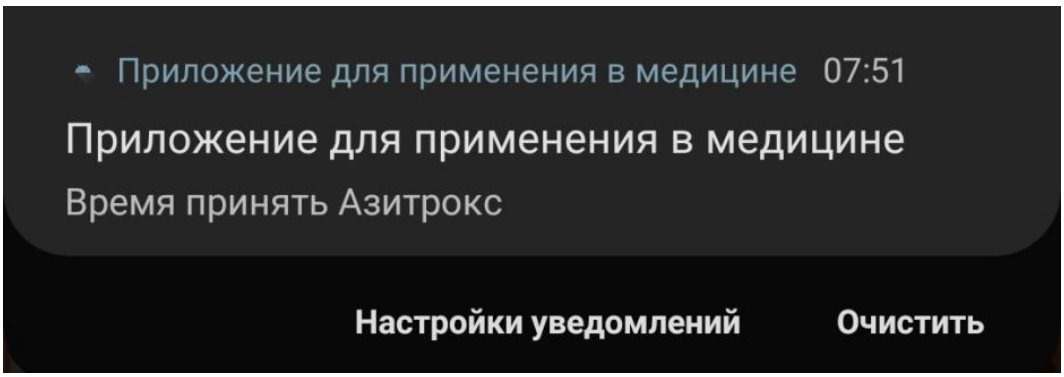
Рисунок 32 — Другая форма ввода времени

На форме календаря пользователь может просмотреть информацию, как на текущую дату, так и на другие даты, которые еще не наступили или же даты которые уже прошли. Форма календаря в соответствии представлена на рисунке 33.



Рисунке 33 — Форма календаря

После того, когда все препараты были добавлены и назначены напоминания то придет напоминание виде push-уведомления как представлено ниже на рисунке 34



Рисунке 34 — Форма push-уведомления

После чего нажимаем на наше уведомление открывается приложение и ставим галочку в напоминание, что препарат был принят, представлено на рисунке 35.

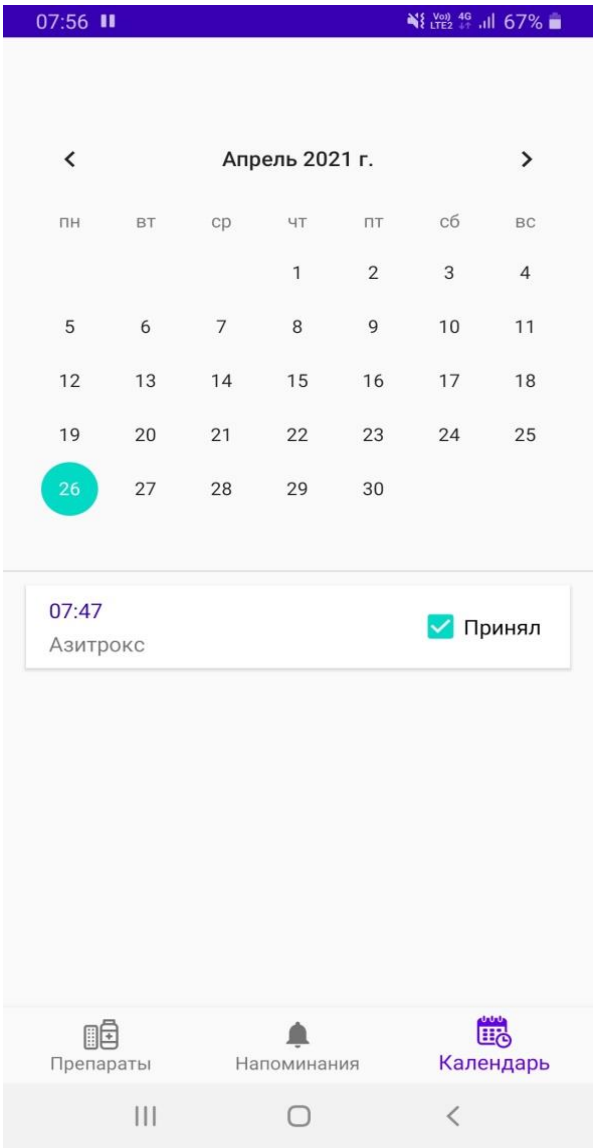


Рисунок 35 — Форма напоминания

Так же если у человека есть проблемы со зрением, то можно с помощью тройного касания по экрану и увеличиться область экрана, которая необходима, представлена на рисунках 36-37.

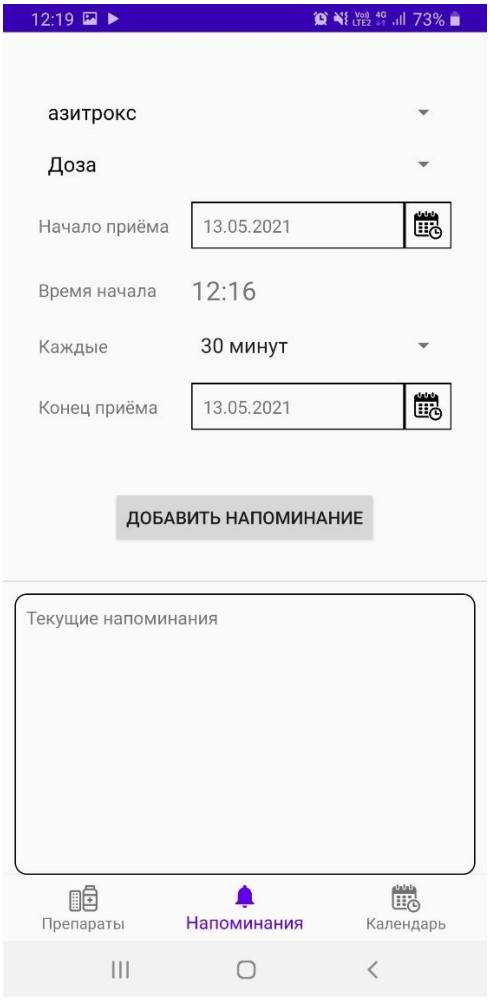


Рисунок 36 — Форма напоминания в обычном режиме.

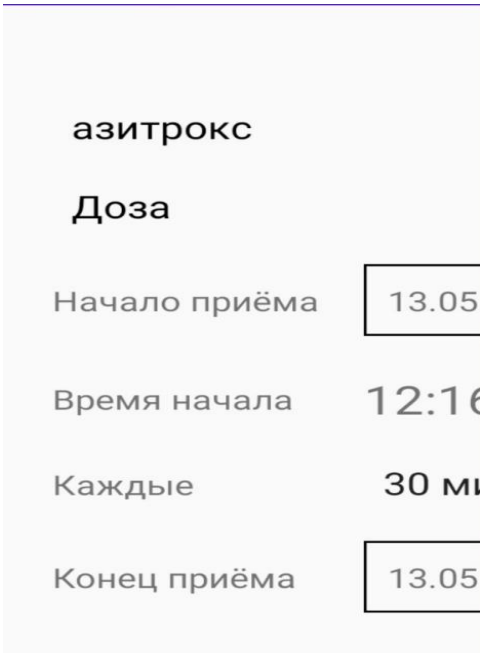


Рисунок 37 — Форма напоминания в режиме увеличения.

3.4. Анализ достоверности и практической значимости результатов

В наши дни мобильные устройства заняли прочное место в повседневной жизни людей. Они начали свой путь в качестве телефонов, впоследствии обретая всё больше и больше функций, превратившись, в конце концов, в полезных помощников. Более того, с развитием интернета мобильные устройства стали средством получения доступа к 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.

Данное программное обеспечение, позволит значительно сэкономить время врача по контролю и назначению приема лекарственных средств пациентом, за счет возможности передачи данных в режиме реального времени, например, пациент сможет переслать лечащему врачу скриншот с экрана приложения, на котором будет изображен электронный лист назначения. Мобильное приложение будет пользоваться популярностью среди людей, которым по

клиническим показаниям необходимо принимать относительно длинный список лекарственных препаратов при этом необходимо учесть: время приема, дозировку и тип лекарства соответственно, также приложение пригодится для людей, принимающих самостоятельно Бады, спортивное питание, витаминные комплексы и др.

Также плюсом будет являться то, что будет производиться сбор и надежное хранение данных с мобильных приложений. Это поможет тем самым отслеживать изменения в состояние здоровья пациента, и анализировать статистику, а также обеспечивать базу для дальнейших исследований.

Разработанный продукт прошел все тесты на корректность работы и готов к установке и использованию.

Целью диссертации является внедрение Android-приложения в современную медицину. Разработка программного Android-приложения облегчит работу клиенту, окажет помощь в составлении и добавлении нужных препаратов, которые необходимо принять в определенный день и время.

Для достижения поставленной цели были поставлены следующие задачи:

— изучить архитектурные особенности платформы Android для реализации исходных требований;
— спроектировать приложения;
— реализовать приложения;
— протестировать и внедрить.

Основным научным результатом выполнения данной работы является разработка Android-приложения в современной медицине. Мобильного приложения облегчит работу пользователю, а именно окажет помощь в составлении и добавлении необходимых препаратов, которые нужно принять в определенный день и время.

Практическая значимость исследования состоит в возможном применении разработанного мобильного приложения в современной медицине.

Анализ достоверность результатов исследования обусловлена строгой аргументацией полученных положений с использованием большего числа литературных источников и обобщения исследования отечественных и зарубежных ученых для проектирования систем управления корпоративной мобильностью. Достоверность включённых в исследование научных положений, теоретических выводов, практических рекомендаций обусловлена корректным применением используемых методов исследования и успешным практическим применением результатов диссертационной работы, а также подтверждается результатами проверки работоспособности и эффективности, применяемых в ходе экспериментального исследования.

ЗАКЛЮЧЕНИЕ

В процессе выполнения магистерской диссертационной работы в соответствии с целью и задачами было проведено исследование предметной области, а именно анализ предметной области, обзор и анализ существующих решений и непосредственно разработка программного обеспечения. Данное программное обеспечение, позволит значительно сэкономить время врача по контролю и назначению приема лекарственных средств пациентом, за счет возможности передачи данных в режиме реального времени, например, пациент сможет переслать лечащему врачу скриншот с экрана приложения, на котором будет изображен электронный лист назначения. Мобильное приложение будет пользоваться популярностью среди людей, которым по клиническим показаниям необходимо принимать относительно длинный список лекарственных препаратов, при этом необходимо учесть:

— время приема;

— дозировку и тип лекарства соответственно.

Также приложение пригодится для людей, принимающих самостоятельно Бады, спортивное питание, витаминные комплексы и др. Также плюсом будет являться то, что будет производиться сбор и надежное хранение данных с мобильных приложений. Это поможет тем самым отслеживать изменения в состояние здоровья пациента, и анализировать статистику, а также обеспечивать базу для дальнейших исследований. Разработанный продукт прошел все тесты на корректность работы и готов к установке и использованию.

БИБЛИОГРАФИЧЕСКИЕ ССЫЛКИ

1. Приложение Google I / O 2017 для Android [Электронный ресурс] - https://github.com/google/iosched

2. Официальная документация по Android [Электронный ресурс]. - https://developer.android.com/guide/

3. Основные этапы разработки мобильных приложений [Электронный ресурс] – Режим доступа: https://spark.ru/startup/componentix/blog/4499/ osnovnie-etapirazrabotki-mobilnih-prilozhenij

4. Новиков, Ю.В. Основы локальных сетей [Электронный ресурс] / Ю.В. Новиков. – Электрон.текстовые данные. – М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. – 405 с.

5. Немцова Т. И., Голова С. Ю., Абрамова И. В. Программирование на языке высокого уровня. Программирование на языке Object Pascal (+ CD-ROM); Форум, Инфра-М - Москва, 2009. - 496 с.

6. Мобильное приложение, что такое, определение, новости, статьи, видео [Электронный ресурс]. – https://indicator.ru/tags/ mobilnoeprilozhenie/

7. Миковски, М.С. Разработка одностраничных веб-приложений / М.С. Миковски, Д.К. Пауэлл. - М.: ДМК, 2014. - 512 с.

8. Медникс Зигард, Дорнин Лайрд , Мик Блэйк , Накамура Масуми Программирование под Android; Питер - Москва, 2013. - 560 с.

9. Машнин, Т.С. JavaFX 2.0: разработка RIA-приложений / Т.С. Машнин. - СПб.: BHV, 2012. - 320 с.

10. Машнин, Т.С. Google App Engine Java и Google Web Toolkit: разработка Web-приложений / Т.С. Машнин. - СПб.: BHV, 2014. - 352 с.

11. Машнин, Т.С. Eclipse: разработка RCP-, Web-, Ajax- и Android-приложений / Т.С. Машнин. - СПб.: БХВ-Петербург, 2013. - 384 с.

12. Маркин, А. Разработка приложений баз данных в Delphi Самоучитель / А. Маркин. - М.: Диалог-МИФИ, 2013. - 160 с.

13. Марк, Д. Разработка приложений для iPhone, iPad и iPod touch с использованием iOS SDK / Д. Марк, Д. Наттинг, Д. Ламарш. - М.: Вильямс, 2012. - 624 с.

14. Майер Рето Android 4. Программирование приложений для планшетных компьютеров и смартфонов; Эксмо - Москва, 2013. - 816 с.

15. Майер Рето Android 2. Программирование приложений для планшетных компьютеров и смартфонов; Эксмо - Москва, 2011. - 672 с.

16. Кречмер, Р. Разработка приложений SAP R/3 на языке ABAP/4 / Р. Кречмер, В. Вейс. - М.: Лори, 2014. - 461 с.

17. Компоненты приложения [Электронный ресурс] - https://developer.android.com/guide/components/activities/index.html#Li fecycle

18. Колисниченко, Д.Н. PHP и MySQL. Разработка веб-приложений. Профессиональное программирование / Д.Н. Колисниченко. - СПб.: BHV, 2015. - 592 с.

19. Колисниченко Денис Программирование для Android. Самоучитель; БХВ-Петербург - Москва, 2012. - 272 с.

20. Колисниченко Д. Программирование для Android. Самоучитель /. - СПб.: Санкт-Петербург, 2013. - 736 с.

21. Козловский, П. Разработка веб-приложений с использованием AngularJS / П. Козловский, П. Дарвин. - М.: ДМК, 2014. - 394 с.

22. Книги по Java [Электронный ресурс] - https://lyapidov.ru/category/programming/java/

23. Кирстен, В. Постреляционная СУБД Cache 5. Объектно-ориентированная разработка приложений / В. Кирстен, М. Иринге р. - М.: Бином, 2013. - 416 с.

24. Использование мобильных устройств [Электронный ресурс]. – Режим доступа: http://www.wi-life.ru/stati/wi-fi/marketingovye-stati-2/mobiledevices-use-aruba-research

25. Здзиарски, Дж. iPhone SDK. Разработка приложений / Дж. Здзиарски. - СПб.: BHV, 2013. - 512 с.

26. Жвалевскийм А. Смартфоны Android без напряга. Руководство пользователя /. - СПб.: Санкт-Петербург, 2012. - 224 с.

27. Дэрси, Л. Разработка приложений для Android-устройств. Т. 1: Базовые принципы / Л. Дэрси, Ш. Кондер. - М.: Лори, 2014. - 402 с.

28. Дэрси Л. Разработка приложений для Android-устройств. Базовые принципы /Л. Дэрси, Ш. Кондер – Том 1. – Москва: Эксмо, 2014. – 598 с.

29. Дэйв, М. iOS 5 SDK. Разработка приложений для iPhone, iPad и iPod touch / М. Дэйв, Н. Джек. - М.: Вильямс, 2012. - 672 с.

30. Дронов, В.А. Windows 8: разработка Metro-приложений для мобильных устройств. / В.А. Дронов. - СПб.: BHV, 2013. - 528 с.

31. Дорнин Л. Google Android: программирование для мобильных устройств. — СПб.: БХВ-Петербург, 2012. — 448 с.: ил.+CD-ROM — (Профессиональное программирование)

32. Джордж Шеферд. Программирование на Microsoft Visual C++ .NET. 2011г.

33. Дари, К. AJAX и PHP. Разработка динамических веб-приложений / К. Дари, Б. Бринзаре, Ф. Черчез-Тоза, М. Бусика. - СПб.: Символ-плюс, 2015. - 336 с.

34. Дадян Э, Зеленков Ю. Методы, модели, средства хранения и обработки данных: учебник. – М.: Вузовский учебник, 2017. – 168 с.

35. Гринберг, М. Разработка веб-приложений с использованием Flask на языке Python / М. Гринберг. - М.: ДМК, 2016. - 272 с.

36. Гецманн, П. Разработка приложений для Windows Phone. Архитектура, фреймворки, API / П. Гецманн. - СПб.: BHV, 2014. - 880 с.

37. Гамма Э., Приемы объектно-ориентированного проектирования // Э.Гамма, Р.Хелм, Р.Джонсон. — СпБ: Питер, 2001. — 368 с.

38. Веллинг, Л. Разработка Web-приложений с помощью PHP и MySQL / Л. Веллинг, Л. Томсон. - М.: Вильямс, 2013. - 848 с.

39. Васильев, А.Е. Микроконтроллеры. Разработка встраиваемых приложений / А.Е. Васильев. - СПб.: BHV, 2012. - 304 с.

40. Бутко, В.Р., CASE – технологии моделирования и проектирования АИС – Учебн. Пособие / В. Р. Бутко, В. П. Дерябкин. – Самара: Самарск. Гос. Экон. академ., 2008. – 105 с.

41. Брюс Эккель «Философия Java» (4-е полное издание): СПБ.: Питер,2015 – 1168 с.

42. Блэйк Мик /Программирование под Android. - СПб.: СанктПетербург, 2012. - 496 с.

43. Блог разработчиков Android [Электронный ресурс] - https://androiddevelopers.googleblog.com/2014/07/learn-to-think-like-androiddeveloper.html

44. Аникеев, С.В. Разработка приложений баз данных в Delphi: Самоучитель / С.В. Аникеев, А.В. Маркин. - М.: ДИАЛОГ-МИФИ, 2013. - 160 с.

45. Аллан, А. Программирование для мобильных устройств на iOS: Профессиональная разработка приложений для iPhone, iPad, and iPod Touch / А. Аллан.. - СПб.: Питер, 2013. - 416 с.

46. Аарон Хиллегасс. Objective-C Программирование для Android.2012 г.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Аарон Хиллегасс. Objective-C Программирование для Android.2012 г.

2. Аллан, А. Программирование для мобильных устройств на iOS: Профессиональная разработка приложений для iPhone, iPad, and iPod Touch / А. Аллан.. - СПб.: Питер, 2013. - 416 с.

3. Аникеев, С.В. Разработка приложений баз данных в Delphi: Самоучитель / С.В. Аникеев, А.В. Маркин. - М.: ДИАЛОГ-МИФИ, 2013. - 160 с.

4. Блог разработчиков Android [Электронный ресурс] - https://androiddevelopers.googleblog.com/2014/07/learn-to-think-like-androiddeveloper.html

5. Блэйк Мик /Программирование под Android. - СПб.: СанктПетербург, 2012. - 496 с.

6. Брюс Эккель «Философия Java» (4-е полное издание): СПБ.: Питер,2015 – 1168 с.

7. Бутко, В.Р., CASE – технологии моделирования и проектирования АИС – Учебн. Пособие / В. Р. Бутко, В. П. Дерябкин. – Самара: Самарск. Гос. Экон. академ., 2008. – 105 с.

8. Васильев, А.Е. Микроконтроллеры. Разработка встраиваемых приложений / А.Е. Васильев. - СПб.: BHV, 2012. - 304 с.

9. Веллинг, Л. Разработка Web-приложений с помощью PHP и MySQL / Л. Веллинг, Л. Томсон. - М.: Вильямс, 2013. - 848 с.

10. Гамма Э., Приемы объектно-ориентированного проектирования // Э.Гамма, Р.Хелм, Р.Джонсон. — СпБ: Питер, 2001. — 368 с.

11. Гецманн, П. Разработка приложений для Windows Phone. Архитектура, фреймворки, API / П. Гецманн. - СПб.: BHV, 2014. - 880 с.

12. Гринберг, М. Разработка веб-приложений с использованием Flask на языке Python / М. Гринберг. - М.: ДМК, 2016. - 272 с.

13. Дадян Э, Зеленков Ю. Методы, модели, средства хранения и обработки данных: учебник. – М.: Вузовский учебник, 2017. – 168 с.

14. Дари, К. AJAX и PHP. Разработка динамических веб-приложений / К. Дари, Б. Бринзаре, Ф. Черчез-Тоза, М. Бусика. - СПб.: Символ-плюс, 2015. - 336 с.

15. Джордж Шеферд. Программирование на Microsoft Visual C++ .NET. 2011г.

16. Дорнин Л. Google Android: программирование для мобильных устройств. — СПб.: БХВ-Петербург, 2012. — 448 с.: ил.+CD-ROM — (Профессиональное программирование)

17. Дронов, В.А. Windows 8: разработка Metro-приложений для мобильных устройств. / В.А. Дронов. - СПб.: BHV, 2013. - 528 с.

18. Дэйв, М. iOS 5 SDK. Разработка приложений для iPhone, iPad и iPod touch / М. Дэйв, Н. Джек. - М.: Вильямс, 2012. - 672 с.

19. Дэрси Л. Разработка приложений для Android-устройств. Базовые принципы /Л. Дэрси, Ш. Кондер – Том 1. – Москва: Эксмо, 2014. – 598 с.

20. Дэрси, Л. Разработка приложений для Android-устройств. Т. 1: Базовые принципы / Л. Дэрси, Ш. Кондер. - М.: Лори, 2014. - 402 с.

21. Жвалевскийм А. Смартфоны Android без напряга. Руководство пользователя /. - СПб.: Санкт-Петербург, 2012. - 224 с.

22. Здзиарски, Дж. iPhone SDK. Разработка приложений / Дж. Здзиарски. - СПб.: BHV, 2013. - 512 с.

23. Использование мобильных устройств [Электронный ресурс]. – Режим доступа: http://www.wi-life.ru/stati/wi-fi/marketingovye-stati-2/mobiledevices-use-aruba-research

24. Кирстен, В. Постреляционная СУБД Cache 5. Объектно-ориентированная разработка приложений / В. Кирстен, М. Иринger. - М.: Бином, 2013. - 416 с.

25. Книги по Java [Электронный ресурс] - https://lyapidov.ru/category/programming/java/

26. Козловский, П. Разработка веб-приложений с использованием AngularJS / П. Козловский, П. Дарвин. - М.: ДМК, 2014. - 394 с.

27. Колисниченко Д. Программирование для Android. Самоучитель /. - СПб.: Санкт-Петербург, 2013. - 736 с.

28. Колисниченко Денис Программирование для Android. Самоучитель; БХВ-Петербург - Москва, 2012. - 272 с.

29. Колисниченко, Д.Н. PHP и MySQL. Разработка веб-приложений. Профессиональное программирование / Д.Н. Колисниченко. - СПб.: BHV, 2015. - 592 с.

30. Компоненты приложения [Электронный ресурс] - https://developer.android.com/guide/components/activities/index.html#Li fecycle

31. Кречмер, Р. Разработка приложений SAP R/3 на языке ABAP/4 / Р. Кречмер, В. Вейс. - М.: Лори, 2014. - 461 с.

32. Майер Рето Android 2. Программирование приложений для планшетных компьютеров и смартфонов; Эксмо - Москва, 2011. - 672 с.

33. Майер Рето Android 4. Программирование приложений для планшетных компьютеров и смартфонов; Эксмо - Москва, 2013. - 816 с.

34. Марк, Д. Разработка приложений для iPhone, iPad и iPod touch с использованием iOS SDK / Д. Марк, Д. Наттинг, Д. Ламарш. - М.: Вильямс, 2012. - 624 с.

35. Маркин, А. Разработка приложений баз данных в Delphi Самоучитель / А. Маркин. - М.: Диалог-МИФИ, 2013. - 160 с.

36. Машнин, Т.С. Eclipse: разработка RCP-, Web-, Ajax- и Android-приложений / Т.С. Машнин. - СПб.: БХВ-Петербург, 2013. - 384 с.

37. Машнин, Т.С. Google App Engine Java и Google Web Toolkit: разработка Web-приложений / Т.С. Машнин. - СПб.: BHV, 2014. - 352 с.

38. Машнин, Т.С. JavaFX 2.0: разработка RIA-приложений / Т.С. Машнин. - СПб.: BHV, 2012. - 320 с.

39. Медникс Зигард, Дорнин Лайрд , Мик Блэйк , Накамура Масуми Программирование под Android; Питер - Москва, 2013. - 560 с.

40. Миковски, М.С. Разработка одностраничных веб-приложений / М.С. Миковски, Д.К. Пауэлл. - М.: ДМК, 2014. - 512 с.

41. Мобильное приложение, что такое, определение, новости, статьи, видео [Электронный ресурс]. – https://indicator.ru/tags/ mobilnoeprilozhenie/

42. Немцова Т. И., Голова С. Ю., Абрамова И. В. Программирование на языке высокого уровня. Программирование на языке Object Pascal (+ CD-ROM); Форум, Инфра-М - Москва, 2009. - 496 с.

43. Новиков, Ю.В. Основы локальных сетей [Электронный ресурс] / Ю.В. Новиков. – Электрон.текстовые данные. – М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. – 405 с.

44. Основные этапы разработки мобильных приложений [Электронный ресурс] – Режим доступа: https://spark.ru/startup/componentix/blog/4499/osnovnie-etapirazrabotki-mobilnih-prilozhenij

45. Официальная документация по Android [Электронный ресурс]. - https://developer.android.com/guide/

46. Приложение Google I / O 2017 для Android [Электронный ресурс] - https://github.com/google/iosched

47. Программирование на Java [Электронный ресурс] - http://studyjava.ru/category/uroki-java/

48. . Программирование для android, java [Электронный ресурс] http://davidmd.ru/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-%D0%BF%D0%BE-android/

49. Ретабоуил Сильвен Android NDK. Разработка приложений под Android на C/C++; ДМК Пресс - Москва, 2012. - 496 с.

50. Руководство разработчика [Электронный ресурс] - http://mybiblioteka.su/tom3/7-59852.html

51. Русская документация Android [Электронный ресурс] - http://easyandroid.ru/index.php?p=721

52. Смирнов, Н.В. Проектирование информационных систем по курсовому проектированию / Н.В. Смирнов; Балт. гос. техн. ун-т. – СПб., 2009. – 61с.

53. Тестирование android приложений с помощью реальных устройств [Электронный ресурс] - http://www.fandroid.info/testirovanie-androidprilozhenij-s-pomoshhyu-realnyh-ustrojstv/

54. Уроки по Java [Электронный ресурс] - https://javarush.ru/

55. Уроки программирования [Электронный ресурс] - http://learnandroid.ru/index.html

56. Уроки программирования от Google [Электронный ресурс] - https://developer.android.com/training/index.html

57. Фаулер М., Архитектура корпоративных программных приложений // М.Фаулер. — М.: Издательский дом "Вильяме", 2006. — 544 с.

58. Якоб Нильсен, Ралука Будиу. Как создавать идеально удобные приложения для мобильных устройств. 2013г.

59. Android [Электронный ресурс]. – Режим доступа: http://www.android.com/

60. Android Studio [Электронный ресурс] - https://developer.android.com/studio/index.html

61. Habrahabr.ru [Электронный ресурс]. – Режим доступа: https://habrahabr.ru/posts/programming/

62. Java — Учебник для начинающих программистов [Электронный ресурс] http://proglang.su/java

63. Oracle [Электронный ресурс]. - https://www.oracle.com/index.html

64. Start Android – учебник по Android для начинающих и продвинутых [Электронный ресурс]. – Режим доступа: http://startandroid.ru/ru/