**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**"НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО" (УНИВЕРСИТЕТ ИТМО)**

**ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ (РАБОТУ)**

Студент Лапшин Данила евгеньевич

(Фамилия, И., О.)

Факультет ПИиКТ Группа P4250 Направление (специальность) 09.04.04 Программная инженерия Руководитель Васильев Д. Е., преподаватель

(Фамилия, И., О., должность)

Дисциплина мобильная разработка

Наименование темы: Разработка Backend мобильного приложения «спутник» для лиц с нарушением интеллекта и эмоционально-волевой сферы.

Задание. Разработать бэкенд приложения «спутник». \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_

Краткие методические указания (задачи работы)

Рассмотреть конкурентов приложения «спутник» имеющихся в данной сфере; разработать и обосновать требования к приложению; выбрать и обосновать выбор технологий для разработки приложения; создать бэкенд приложения и описать его; протестировать приложение на соответствие требованиям, описанным в работе.

Содержание пояснительной записки

Оглавление. Введение. Глава 1 теоретическая часть. Анализ конкурентов. Выбор инструментов и технологий для разработки приложения. требования к приложению. Глава 2: Практическая часть. Архитектуры для разработки мобильных приложений. Общая структура приложения. База данных. Модель. модель представлений. Тестирование. Заключение. Список использованных источников.

**"НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО" (УНИВЕРСИТЕТ ИТМО)**

**АННОТАЦИЯ К КУРСОВОМУ ПРОЕКТУ (РАБОТЕ)**

Студент Лапшин Данила Евгеньевич

(Фамилия, И., О.)

Факультет ПИиКТ Группа P4250 Направление (специальность) 09.04.04 Программная инженерия Руководитель Государев И. Б., преподаватель

(Фамилия, И., О., должность)

Дисциплина мобильная разработка

Наименование темы: Разработка Backend мобильного приложения «спутник» для лиц с нарушением интеллекта и эмоционально-волевой сферы

ХАРАКТЕРИСТИКАКУРСОВОГО ПРОЕКТА (РАБОТЫ)

* **Цель и задачи работы** Определены руководителем Предложены студентом

Цель работы — разработка бэкенда мобильного приложения «Спутник»

Задачи работы:

Рассмотреть конкурентов приложения «спутник» имеющихся в данной сфере

разработать и обосновать требования к приложению

выбрать и обосновать выбор технологий для разработки приложения

создать бэкенд приложения и описать его

протестировать приложение на соответствие требованиям, описанным в работе

* **Характер работы** Анализ
* **Содержание работы**

**Рассмотрен и определён смысл разработки мобильного приложения «Спутник», проанализированы его конкуренты, описаны и спроектированы требования к приложению, реализован бэкенд приложения, описана структура приложения, приложение протестировано.**

* **Выводы**

тема Разработка Backend мобильного приложения «спутник» для лиц с нарушением интеллекта и эмоционально-волевой сферы является актуальной. Нарушения данного типа имеются у миллионов людей, а в России нет конкурентов данному приложению. Мировые же конкуренты могут быть не достаточно адаптированы для работы с людьми, которые кроме нарушений в интеллектуальной или эмоционально-волевой сфере имеют сопутствующие нарушения, например зрения, поэтому данная работа является необходимой.

Руководитель

(подпись)

Студент

(подпись)

Д.Е. Васильев Д.Е. Лапшин

**Министерство науки и высшего образования Российской Федерации**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«Национальный исследовательский университет ИТМО»**

**(Университет ИТМО)**

**Факультет программной инженерии и компьютерной техники**

**Направление (специальность) 09.04.04 Программная инженерия**

**Специализация Программирование для незрячих и слабовидящих**

**Дисциплина — мобильная разработка**

**Направление (специальность) 09.04.04 Программная инженерия**

**Специализация Программирование для незрячих и слабовидящих**

**Дисциплина — мобильная разработка**

**Курсовой проект (работа)**

**ТЕМА:** Разработка Backend мобильного приложения «спутник» для лиц с нарушением интеллекта и эмоционально-волевой сферы

ВЫПОЛНИЛ

Студент группы P4250 Лапшин Д.Е.

№ группы ФИО

ПРОВЕРИЛ преподаватель Васильев Д.Е

ученая степень, должность ФИО

САНКТ-ПЕТЕРБУРГ 2024 г.

Оглавление

[Введение 6](#_Toc191992112)

[Глава 1 теоретическая часть 8](#_Toc191992113)

[1.1 Анализ конкурентов 8](#_Toc191992114)

[1.2 Выбор инструментов и технологий для разработки приложения 11](#_Toc191992115)

[1.3 требования к приложению 18](#_Toc191992116)

[Глава 2: Практическая часть 21](#_Toc191992117)

[2.1 Архитектуры для разработки мобильных приложений 21](#_Toc191992118)

[2.2 Общая структура приложения 25](#_Toc191992119)

[2.3 База данных 27](#_Toc191992120)

[2.4 Модель 30](#_Toc191992121)

[2.5 модель представлений 32](#_Toc191992122)

[2.6 Тестирование 34](#_Toc191992123)

[Заключение 35](#_Toc191992124)

[Список использованных источников 36](#_Toc191992125)

# Введение

Люди с нарушениями интеллекта и эмоционально-волевой сферы ежедневно сталкиваются с множеством препятствий в повседневной жизни. Такими припятствиями являются: Ограниченные возможности в общении, ориентации в пространстве, выполнении бытовых задач, ограничения в понимании происходящего часто приводят к социальной изоляции и невозможности жить полноценной жизнью. В современном обществе, где информационные технологии играют одну из ключевых ролей, можно использовать их потенциал для улучшения условий жизни людей с такими нарушениями.

Информационные технологии- это та сфера жизни, которая может улучшить её качество для всех людей, не зависимо от их физических ограничений, при соответствующей реализации инструментов для каждой группы пользователей. Смартфон сегодня есть практически у каждого человека, поэтому возможность реализации помощника в бытовой сфере в виде мобильного приложения, является одним из самых удобных вариантов для пользователя. Такой помощник всегда будет с человеком, если рядом с ним будет телефон. Так же, мобильное приложение можно сделать доступным для большинства пользователей, например, используя программы экранного доступа для людей с нарушениями не только в эмоционально-волевой сфере, но и с нарушением зрения. Существующие мобильные приложения не всегда учитывают специфические особенности данной аудитории, что создаёт необходимость разработки новых, более адаптированных решений, которым и является приложение, представленное в данной работе.

Целью данной курсовой работы является создание Backend для мобильного приложения «Спутник», предназначенного для облегчения взаимодействия пользователей с нарушениями интеллекта и эмоционально-волевой сферы с окружающим миром в бытовой сфере. Для достижения данной цели необходимо решить следующие задачи:

1. Провести анализ существующих мобильных приложений, предназначенных для помощи данной группе пользователей, выявить их преимущества и недостатки.

2. Изучить современные технологии и инструменты, применяемые при разработке мобильных приложений, обосновать выбор оптимальных для реализации проекта.

3. Разработать архитектуру Backend и спроектировать пользовательский интерфейс приложения с учётом особенностей целевой аудитории.

4. Реализовать функционал приложения на основе выбранных технологий, обеспечить его стабильную работу и безопасность.

5. Провести тестирование приложения, оценить его эффективность и соответствие поставленным требованиям.

6. Подвести итоги проделанной работы, сформулировать выводы и наметить перспективы дальнейшего развития проекта.

Объектом исследования является процесс разработки мобильных приложений для пользователей с нарушениями интеллекта и эмоционально-волевой сферы. Предметом исследования выступают технологии и методы разработки Backend для таких приложений.

Методы исследования включают в себя анализ литературных источников и существующих приложений, сравнительный анализ технологий, методологии проектирования и разработки программного обеспечения, а также методы тестирования и оценки эффективности.

Структура работы включает в себя введение, теоретическую и практическую части, заключение, список использованных источников и приложения. В теоретической части будет проведён обзор существующих решений и технологий разработки, обоснован выбор инструментов для реализации проекта. В практической части описывается процесс проектирования и разработки Backend приложения, результаты тестирования и рекомендации по дальнейшему развитию.

Таким образом, разработка Backend мобильного приложения «Спутник» является актуальной задачей, направленной на решение важных социальных проблем и улучшение качества жизни людей с нарушениями интеллекта и эмоционально-волевой сферы.

# Глава 1 теоретическая часть

## 1.1 Анализ конкурентов

Для анализа конкурентов решения, предложенного в данной работе, сначала опишем это решение. Суть приложения «Спутник» заключается в том, что для лиц с нарушениями в эмоционально-волевой и интеллектуальной сферах предоставляются сценарии поведения в различных ситуациях, с которыми человек сталкивается в жизни и которые для обычного человека являются легко преодолимыми и обыденными, а для данной группы пользователей являются сложными для понимания. Например, такими ситуациями могут быть пожар, стирка белья, вызов скорой и так далее. сценарии действий предоставляются в текстовом формате, что позволяет сделать их доступными для большинства групп пользователей, например, не только с нарушениями в интеллектуальной и эмоционально-волевой сферах, но и для людей с сопутствующими другими нарушениями, например зрения или слуха. Теперь можно рассмотреть конкурентов.

Прямых конкурентов данному решению найдено не было, так как в основном приложения для помощи людям с такими нарушениями сосредоточено на психологической помощи, или помощи в социальных взаимодействиях. Приложение «Спутник», сосредоточено на бытовой помощи, хотя и может быть использовано для помощи в социальной сфере. Так как прямых конкурентов нет, можно рассмотреть несколько приложений косвенных конкурентов.

Proloquo2Go

Это приложение, разработанное компанией AssistiveWare на устройства компании Apple для облегчения коммуникации людям, у которых есть с ней проблемы. Оно работает на основе перевода текста, который пишет пользователь в речь. Так же оно потдержывает общение для пользователей с разным уровнем грамотности, например в настройках можно выбрать уровни понимания от отдельных слов до сложных предложений. Ещё это приложение имеет функционал для развития пользователей, например от языка общения состоящего из отдельных слов, оно может развивать пользователя до языка, состоящего из сложных предложений. Так же это приложение поддерживает несколько языков, но среди этих языков нет русского, поэтому для жителей России данное приложение фактически бесполезно. Ещё один недостаток данного приложение, оно платное, а так как приложение будет использоваться инвалидами, то многие из них не смогут его себе позволить.

Следующее приложение выбранное для рассмотрения это приложение Choiceworks.

Choiceworks — это обучающее приложение, разработанное компанией Bee Visual, LLC, предназначенное для помощи детям в выполнении повседневных рутин, управлении эмоциями и улучшении навыков ожидания. Так же это приложение может помоч и людям, с нарушениями в эмоционально-волевой и интеллектуальных сферах.

Основными функциями приложения являются:

Доска расписания (Schedule Board): Помогает детям выполнять последовательность задач, отмечая завершенные шаги и продвигаясь к мотивирующей награде. Можно добавлять таймеры к любому заданию или активности; Доска ожидания (Waiting Board): Обучает навыкам ожидания, таким как поочередность и недопущение прерываний. Ребенок выбирает занятие, которым можно заняться во время ожидания, с установленным таймером;

Доска эмоций (Feelings Board): Помогает пользователям распознавать и выражать свои эмоции, выбирая соответствующую стратегию совладания и последующую деятельность для улучшения настроения; Шкала эмоций (Feelings Scale): Позволяет отслеживать и управлять эмоциями в различных ситуациях на протяжении дня, анализируя данные для выявления тенденций и адаптации окружения для достижения лучших результатов.

Так же приложение поддерживает следующие дополнительные возможности:

Библиотека изображений с более чем 180 предустановленными изображениями и аудиозаписями; Возможность добавления собственных изображений, записи аудио и прикрепления видео к любому элементу доски для неограниченной кастомизации; Создание профилей для персонализации и управления несколькими пользователями; Сохранение неограниченного количества досок для разных детей или рутин; Озвучивание досок детским или взрослым голосом; Возможность печати досок или сохранения их в формате PDF; Настраиваемые темы и звуки таймера.

данное Приложение доступно для устройств на базе iOS (iPhone, iPad) и совместимо с версиями iOS 14.0 и выше.

Недостатки этого приложения в том, что оно не локализовано на русском языке и по большей части сосредоточено на визуальном восприятии информации, что усложняет его использование людьми с нарушениями зрения. Это приложение так же является платным.

Таким образом, можно заметить, что у разрабатываемого приложения «Спутник» не так много конкурентов по всему миру, а тем более в России, да и конкуренты косвенные, так как такого функционала, доступного для людей с нарушениями зрения никто не предлогает.

## 1.2 Выбор инструментов и технологий для разработки приложения

Для того, чтобы разработать мобильное приложение, нужно выбрать 3 базовых технологии:

Операционная система- эта платформа, на которой будет работать приложение.

Язык программирования- это язык, на котором оно будет написано.

Среда разработки- это ide, в которой оно будет разрабатываться.

Начнём с выбора платформы.

Сегодня в мире лидируют две мобильных операционных системы- iOS от компании Apple и Android от компании google. Компании Apple и Google придерживаются разных политик в отношении разработки приложений. Компания Apple чательно следит за качеством приложений, имеет ограниченный набор устройств и закрытые, доступные только на её компьютерах компоненты для разработки, например, IDE xcode. Так же разработчикам нужно оплачивать подписку для возможности выкладывать свои приложения в официальный магазин Apple. Такая политика даёт возможность пользователю получать более качественные и безопасные приложения а разработчику меньше задумываться о тестировании приложений для разных конфигураций устройств. Так же у приложений для устройств apple, как правило, более высокий пользовательский доход, так как устройства Apple сами по себе одни из самых дорогих на рынке. Но это же делает порог входа в разработку для устройств Apple более высоким, чем на устройства Android. Так же, разработчикам не доступны некоторые функции системы и меньше возможностей для интеграции кастомных вещей. Политика же google фактически полностью противоположена политике Apple, так что фактически недостатки Apple являются достоинствами Android и наоборот. для Android можно разрабатывать на любом компьютере, порог входа на официальный магазин более низкий и бесплатный, так же есть и другие магазины независимые от google, например рустор и сами устройства на android могут быть в разы дешевле, чем устройства конкурента. Но огромное количество прошивок, конфигураций, производителей устройств и так далее, делает куда сложнее работу и тестирование приложений на всём этом многообразии, да и качество с безопасностью на более низком уровне.

В перспективе проект «Спутник» желательно будет разработать для обеих платформ, но так как проект социальных и для небогатого слоя населения, то для его разработки была выбрана операционная система Android.

Следующим инструментом для разработки мобильного приложения, который надо выбрать, является язык программирования.

«Языки программирования — это формальные искусственные языки, созданные для написания компьютерных программ. Как и естественные языки, они имеют алфавит, словарный запас, грамматику и синтаксис, а также семантику. Алфавит — разрешенный к использованию набор символов, с помощью которого могут быть образованы слова и величины данного языка. Синтаксис — система правил, определяющих допустимые конструкции языка программирования из букв алфавита. Семантика — система правил однозначного толкования каждой языковой конструкции, позволяющих производить процесс обработки данных».

Языки программирования можно разделить на два больших класса.

1. Низкоуровневые- это языки программирования, на которых инструкции для компьютера пишутся в виде машинного кода, то есть последовательности нулей и единиц. Языки низкого уровня жестко ориентированы на конкретный тип оборудования (система управления процессором, каждый тип процессора имеет свой машинный код).

2. Высокого уровня – «это языки программирования, которые позволяют записывать программы в удобной для человека форме. Эти языки ориентированы не на систему инструкций того или иного процессора, а на систему операторов (инструкций), характерную для написания определенного класса алгоритмов».

У каждого из этих типов языков программирования есть свои преимущества и недостатки. Преимуществом языков низкого уровня является то, что программы, которые на них написаны, написаны машинным кодом. Это значит, что такая программа может работать либо с меньшими затратами аппаратных ресурсов чем программа на языке высокого уровня, либо быстрее на доступных. Недостаток языков низкого уровня в том, что они не удобны для восприятия и работы с ними человека.

Языки высокого уровня наоборот удобны для человека, но программы, которые на них написаны, требуется переводить в машинный код. Для этого у таких языков программирования есть компилятор или интерпретатор. Поэтому они работают либо медленнее программ на языках низкого уровня, либо требуют больше ресурсов. Сейчас большинство программистов используют языки высокого уровня, так как вычислительные ресурсы современных компьютеров позволяют работать программам на высокоуровневых языках с достаточной скоростью.

Для мобильной разработки используется несколько основных языков программирования:

1 Java

Java- язык программирования, разработанный в начале 1990-х годов для программирования бытовой техники. К началу двухтысячных он стал использоваться для написания клиентских приложений, а в дальнейшем стал основным языком для разработки приложений на

Android. Код Java транслируется на компьютер по байтого через виртуальную машину встроенную в сам язык программирования как интерпретатор, что позволяет исполнять Java код на любой платформе. Так же эта виртуальная машина контролирует каждый байт кода, поэтому Java является очень безопасным языком и его используют, например, в банковских приложениях для совершения платёжных операций. Популярность данного языка программирования означает то, что у него есть огромное сообщество, которое решила большинство проблем и множество библиотек и готовых решений.

Однако у Java есть и свои минусы, из-за которых он постепенно теряет позиции, уступая другим более молодым языкам в сфере мобильной разработки. Этот язык Может быть сложным для изучения новичками из-за особенностей синтаксиса и концепций. Так же из-за виртуальной машины в большинстве случаях производительность Java-приложений может уступать приложениям, написанным на других языках.

2 Kotlin

Kotlin- это язык программирования постепенно заменяющий Java в мобильной разработки. Этот язык был создан в 2010 году, а в 2019 году Google объявила его основным языком разработки на Androyd. Kotlin основан на Java, но по сравнению с ним его синтаксис более прост. Так же в нём интегрирована система, не дающая совершать распространённые ошибки, такие как NullPointerException. Это делает программирования на данном языке проще, чем на Java.

Недостатки у данного языка во многом такие же как у Java, так как он основан на его базе, хотя как было написано выше они и были частично исправлены.

3 C++

C++- это компилируемый язык программирования со статической типизацией. Это значит, что типы переменных не могут изменяться после их объявления и для работы этого языка кроме среды разработки нужен компилятор. C++ впервые был выпущен Бьёрном Страуструпом в 1983 как дальнейшее развитие языка C. Этот язык имеет несколько преимуществ:

Одно из них это Кроссплатформенность. Так же программы написанные на C++ работают быстрее чем на других языках программирования. Недостатками C++ является сложный синтаксис и необходимость наличия кроме среды разработки ещё и компилятора. Так же разработчик должен сам следить за памятью, что усложняет разработку. Из-за своих плюсов и минусов используется в основном для разработки высокопроизводительных приложений или компьютерных игр.

4 JavaScript (с использованием React Native):

JavaScript- это язык программирования, который был изначально создан для web разработки, но его плюсы позволили ему проникнуть во многие сферы программного обеспечения, от работы с большими базами данных до мобильных приложений.

Разработка мобильных приложений на JavaScript реализуется библиотекой React Native. Данная библиотека позволяет создавать приложения для всех популярных мобильных операционных систем.

Простота и рациональность применения. Простую задачку можно решить за 5 минут, не надо делать лишнюю работу. Для сложных заданий есть варианты решения, можно подобрать лучший, адаптировать.

Недостатком данного решения является производительность. Приложения на React Native могут уступать в производительности нативным приложениям, хотя это не всегда важно.

5 Dart с использованием Flutter

библиотека Flutter — это инструмент с открытым исходным кодом от Google, предназначенный для создания красивых, быстрых и нативных приложений для мобильных устройств, веба и десктопа из единой кодовой базы.

Язык программирования Dart: Flutter использует язык программирования Dart, также разработанный Google. Dart компилируется непосредственно в машинный код, что обеспечивает высокую производительность приложений.

Flutter предоставляет богатый набор готовых блоков для создания интерфейсов пользователя в мобильных приложениях. такими блоками могут быть кнопки, текстовые поля, изображения и т. д.

у Flutter есть функция так называемой Горячей перезагрузки, которая позволяет видеть изменения кода в режиме реального времени, без необходимости перезапускать приложение.

Flutter позволяет получить доступ к функциям устройства, таким как камера, геолокация, датчики и т.д., через стандартные плагины, что очень удобно при разработке приложений.

Недостатком Flutter может быть размер приложений, которые могут быть немного больше по размеру, чем приложения на других языках. Так же на данном фреймворке делается небольшое количество приложений в сравнении с тем же Kotlin.

Для приложения «Спутник» был выбран язык программирования Kotlin, так как это рекомендованный язык программирования от Google для разработки под Android и на нём делается большинство приложений под данную опирационную систему, что даёт огромные возможности по поиску решений, компонентов, ответов на вопросы и так далее. Так же под этот язык программирования настроена и основная среда разработки, созданная под Android, а именно Android studio.

Среды разработки.

Среда разработки (или ide)- это окружение для разработки программного обеспечения, представляющее собой набор инструментов для разработки, таких как компиляторы, редакторы кода с автодополнением, отладчики, шаблоны проектов, инструменты для управления зависимостями, сборщики проектов и другое.

фактически для разработки и сборки приложений под операционную систему Android есть только одна среда разработки, это Android studio. Остальные ide не имеют функционала для сборки приложений, а значит всегда придётся использовать Android studio.

## 1.3 требования к приложению

Приложение «спутник» разрабатывается для того, чтобы обеспечить помощь людям с нарушениями эмоционально-волевой и интеллектуальной сферы. Функционал приложения описанный ниже, позволит обеспечить её не только им, но и людям с сопутствующими заболеваниями, например слепотой, чего в мире почти никто не делает. Целевой аудиторией данного приложения являются люди с нарушениями в эмоционально-волевой и интеллектуальной сферах, их родственники, преподаватели и социальные работники, взаимодействующие с ними.

Требования к приложению.

Основные функции приложения:

Добавление, удаление и редактирование памяток и инструкций.

Добавление в раздел «Избранное» памяток и инструкций.

Удаление из раздела «Избранное» памяток и инструкций.

Чтение памяток и инструкций.

Поиск памяток и инструкций.

Разделение инструкций на тематические разделы

Приложение будет доступно только для смартфонов на базе опирационной системы Android версии 8.0 и выше. – инструкции должны храниться локально на устройстве для постоянного доступа пользователя.

Пользовательские требования:

Для обычного пользователя Должно быть реализовано 2 экрана с навигацией между ними: главный экран, избранное и экран администратора.

Инструкции должны быть максимально однозначными и понятными.

Интерфейс приложения должен быть максимально простым и понятным, не содержать лишние элементы и не быть перегруженным. Экран администратора должен для доступа к своему функционалу запрашивать авторизацию пользователя по логину и паролю. Для администратора должен быть доступен весь выше описанный функционал, в то время как обычный пользователь не должен иметь возможности изменять данные инструкций и администраторов, кроме добавления или удаления инструкций в или из раздела избранное. Так же администратор должен иметь возможность поиска администраторов на административном экране. Функционал администратора отделён от функционала обычного пользователя для того, чтобы люди с нарушениями в эмоционально-волевой или интеллектуальной сфере не могли сломать приложение.

Всё приложение должно быть доступно для программ экранного доступа.

Производительность:

Приложение должно работать на как можно большем числе устройств, от флагманских телефонов, до бюджетного сегмента.

Приложение должно переходить от элемента к элементу интерфейса, например между экранами, менее чем за 0.5 с.

Приложение должно быть доступно круглосуточно и обеспечивать надёжную работу.

Требования к устройству:

Операционная система не меньше Android версии 8.0

Системные требования не меньше 2 gb оперативной памяти и не меньше 100 mb памяти на накопителе.

Тестирование:

Функциональное тестирование- протестировать все функции на предмет их работоспособности.

# Глава 2: Практическая часть

## 2.1 Архитектуры для разработки мобильных приложений

Каждое мобильное приложение имеет свою архитектуру, которая определяет его структуру и взаимодействие между компонентами. Архитектура мобильного приложения — это структурный каркас, определяющий способы организации кода и взаимодействия между различными компонентами приложения. Выбор архитектурного подхода играет ключевую роль в разработке устойчивого, масштабируемого и поддерживаемого приложения.

Не исключение и приложение, разрабатываемое в данной работе, поэтому для выбора архитектуры Рассмотрим несколько из них.

1. MVC (Model-View-Controller)

MVC — один из базовых архитектурных паттернов, который разделяет приложение на три основных компонента:

Model (Модель), компонент, который Отвечает за управление данными, бизнес-логику и правила приложения.

View (Представление): Отвечает за отображение информации пользователю. Это пользовательский интерфейс.

Controller (Контроллер): Обрабатывает пользовательский ввод, взаимодействие между Model и View.

Эта архитектура работает так

• Пользователь взаимодействует с View.

• View передаёт событие Controller'у.

• Controller обрабатывает событие, обновляет Model.

• Model уведомляет View об изменениях.

• View обновляется с учётом новых данных.

Преимуществами данной архитектуры являются Простота и понятность: Легко начать использовать и понимать; Разделение ответственности: Логика разделяется между компонентами, что позволяет их описывать и обновлять по отдельности.

Недостатками данной архитектуры является тесная связь между компонентами, что затрудняет их разделение для разработчика и плохая масштабируемость, так как контроллер может быть перегружен.

2. MVP (Model-View-Presenter)

MVP — улучшение MVC, разделяющее логику на:

•

Model (Модель): Данные и бизнес-логика.

•

View (Представление): Интерфейс пользователя, пассивный, не содержит логики.

•

Presenter (Презентер): Посредник между View и Model, содержит всю логику представления.

Работает так:

• View делегирует все действия Presenter'у.

• Presenter обрабатывает действия, запрашивает данные у Model.

• Presenter обновляет View.

Преимущества

• Легкость тестирования: Presenter можно легко тестировать отдельно от View.

• Чистое разделение обязанностей: View не содержит логики, Presenter не зависит от платформы.

Недостатки

• Много шаблонного кода: Наличие интерфейсов для View.

• Увеличение количества классов Может усложнять проект.

3. Clean Architecture или чистая архитектура

Clean Architecture — концепция, предложенная Робертом Мартином, направленная на создание архитектур с независимыми слоями, разделяющими ответственности:

•

Entities (Сущности): Бизнес-модели приложения.

•

Use Cases (Сценарии использования): Реализуют бизнес-логику.

•

Interface Adapters (Адаптеры интерфейса): Конвертируют данные между слоями.

•

Frameworks and Drivers (Фреймворки и драйверы): Внешние системы, UI, базы данных.

Преимуществами данной архитектуры являются возможность лёгкой смены технологий, так как она не зависит от фреймворков; лёгкость тестирования, так как компоненты изолированы друг от друга и из-за этого же возможность легко обновлять и долго поддерживать приложения. Недостатки у этой архитектуры в том, что она сложная, поэтому требует много времени на разработку и разработчик, который ей пользуется, должен хорошо понимать архитектурные принципы в целом.

MVM (model view viewmodel)

Архитектура MVM состаит из следующих компонентов:

Model- сущность, обеспечивающая бизнес-логику приложения (работу с данными, вычисление и прочие функции внутренней части приложения).

Viewmodel- сущность, обеспечивающая логику представления данных для пользовательского интерфейса. Именно с ней связывается ui для обновления данных.

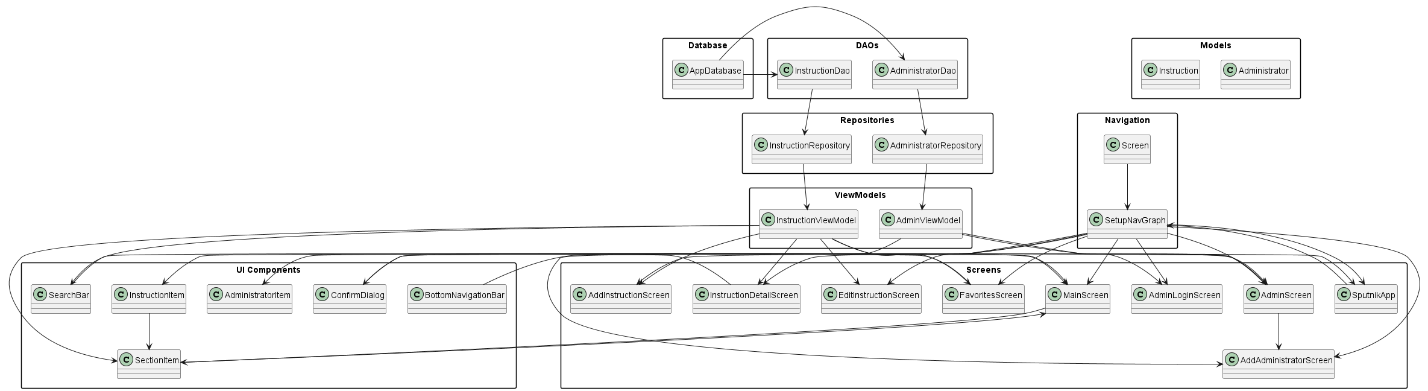
View- пользовательский интерфейс.

Преимуществами данной архитектуры является слабая связь между компонентами, например нету контроллера, который следит за состояниями каждого компонента и обновляет их при необходимости. Данная архитектура работает с реактивными потоками данных, тоесть view подписывается на veiwmodel и при её обновлении получает данные в режиме потока. Такой способ организации взаимодействия между компонентами позволяет использовать асинхронные операции, а так же менять компоненты приложения не зависимо друг от друга, например, если изменить один экран в view остальное приложение не понадобиться менять, что возможно пришлось бы делать в случае, если бы был контроллер, связывающий все компоненты.

Таким образом, для разработки приложения «Спутник» после рассмотрения выше описанных архитектур было принято решение использовать архитектуру MVM. Так же эта архитектура встроена во фреймворк GetPack compose, который используется для создания интерфейса данного приложения, что делает разработку более удобной.

## 2.2 Общая структура приложения

Теперь рассмотрим структуру получившегося приложения. В данной работе подробно будет рассматриваться логика работы приложения (бэкенд), но для понимания как устроено приложение в целом, приведена схема, описывающая общую структуру проекта.



## 2.3 База данных

Теперь, имея общее представление о структуре проекта, рассмотрим логику его работы. В приложении спутник реализован следующий функционал:

управление инструкциями (удаление, добавление, редактирование, отображение, добавление и удаление из избранного, поиск по инструкциям и разделам), идентификация администраторов, управление администраторами (отображение, добавление, удаление и поиск по администраторам).

Для реализации описанного функционала было решено спроектировать базу данных. получившаяся База данных имеет следующий вид:

1 таблица instructions, отвечающая за хранение статей. В данной таблице есть такие поля: id- генерируется с помощью автоинкремента для каждой новой инструкции. Является её уникальным идентификатором. Второе поле это название инструкции (title)- поле, отвечающее за заголовок инструкции, который будет отображаться пользователю. Третье поле- content. В этом поле храниться содержание инструкции, которое и должен читать пользователь. Следующее поле- sectionname- является названием раздела, к которому принадлежит инструкция. Ещё одно поле- url позволяет добавлять ссылку на изображение, с которым инструкция может показываться пользователю данное поле является не обязательным и может быть пустым. Последним полем таблицы instructions, является поле isfavorit, определяющее находится ли инструкция в избранном или нет.

2 administrator- таблица для хранения данных администраторов, которые могутизменять контент в приложении. В данной таблице есть 2 поля- login и password. Этих поля позволяют надёжно идентифицировать администратора.

Для реализации базы данных описанной выше структуры была использована система управления базами данных sqlite, являющаяся отличной небольшой системой, пригодной для баз данных, встроенных локально в приложения. Эта система управления базами данных, или сокращённо СУБД, работает с реляционными, то есть табличными базами данных и использует такой популярный язык описания баз данных как SQL. Для разработки на языке Kotlin с использованием системы sqlite существует библиотека Room, которая и была использована в данном проекте. Так же, все операции с базой данных были реализованы в асинхронном виде для того, чтобы обращение к базе данных не блокировало основной поток приложения, а происходило в фоновых потоках. Для асинхронного выполнения операций используется библиотека coroutin.

## 2.4 Модель

Теперь, описав базу данных, с которой работает приложение, опишем бизнес-логику, использующуюся в нём. В архитектуре MVM за бизнес-логику отвечает сущность, которую называют моделью. В данной сущности реализуется весь функционал, необходимый для приложения. В случае приложения Спутник для реализации его функционала была создана модель следующей структуры:

1 работа с базой данных. Работа с базой данных сосредоточена в четырёх файлах: Instruction, Administrator, InstructionDao и AdministratorDao. В файлах бес окончания Dao происходит инициализация таблиц для базы данных, а в файлах с окончанием Dao реализован интерфейс работы с ними с помощью sql запросов для всего функционала приложения по изменению данных.

2 репозитории

Дальше реализованный функционал из файлов dao используется в папке repositories, в которой реализованы два файла для работы с администраторами и инструкциями соответственно. В файлах репозиториях функции из предыдущих интерфейсов оборачиваются в классы с соответствующими названиями, что позволяет инкапсулировать работу с данными и тестировать или изменять эти классы отдельно от view или модели представлений.

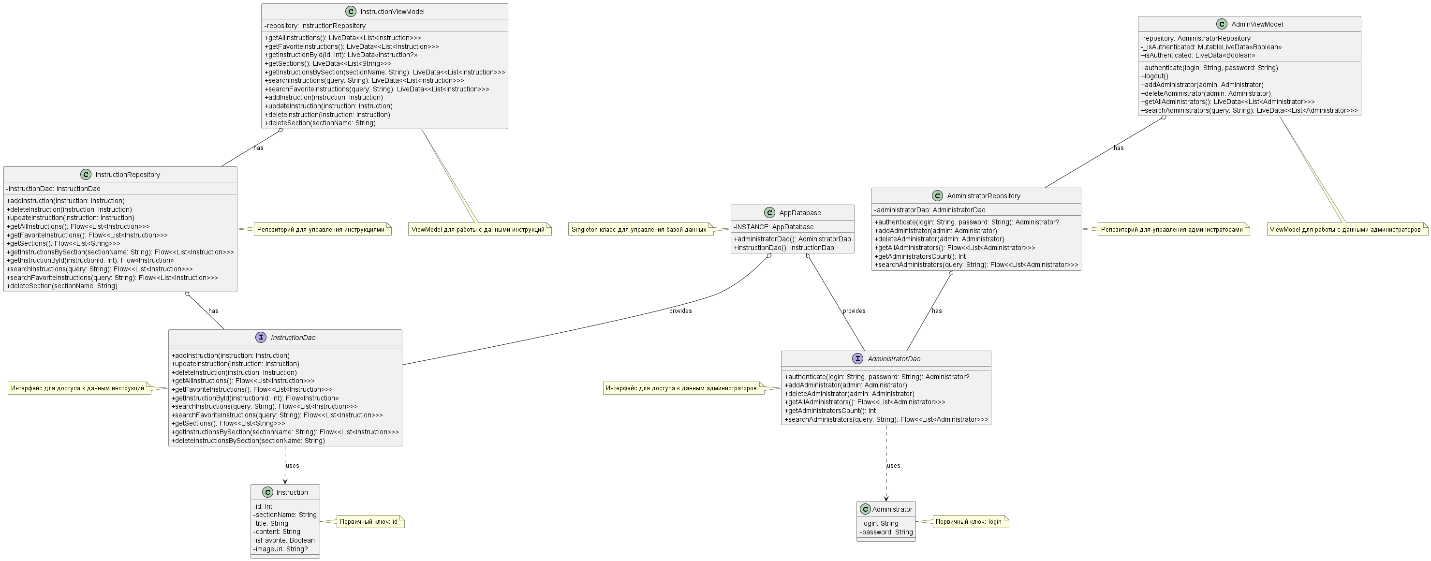
инициализация базы данных.

Последней частью модели является файл AppDataBase, который создаёт базу данных при сборке приложения.

Ниже представлена схема описанной модели.

## 2.5 модель представлений

Следующей сущностью, которая была реализована в проекте является так называемая модель представлений или viewmodel. В данную сущность выносятся все функции описанные в моделе и уже с ней работает пользовательский интерфейс. Такая структура нужна для изоляции пользовательского интерфейса от бизнес-логики и возможности в определённых приделах изменять интерфейс или логику отдельно друг от друга, что упрощает разработку приложений. В данном проекте модель реализована в папке viewmodel, где для инструкторов и администраторов есть файл с соответствующим названием. Дальше в архитектуре MVM идёт пользовательский интерфейс, который может быть любым, в рамках того, что позволяет реализовать логика, описанная выше. Ниже дана схема логики работы приложения, для наглядного представления.



## 2.6 Тестирование

Приложение было протестировано несколькими пользователями на устройствах с разными версиями операционной системы Android. Тестирование показало, что приложение полностью соответствует описанным требованиям и выполняет весь необходимый функционал.

# Заключение

Таким образом в данной работе было выполнено следующее:

определены цели и задачи работы.

2. Разработаны требования к приложению Спутник и обосновано почему эти требования такие.

Проведён обзор решений конкурентов и сравнение с приложением Спутник.

Описана целевая аудитория продукта.

Описаны технологии, использованные для его создания и обоснован выбор конкретных решений.

Создан и описан бэкенд приложения.

Приложение было протестировано и показало свою работоспособность и выполнение поставленных задач

Этогом курсовой работы явилось готовое приложение, которое можно использовать в сфере, в которой у него практически нет конкурентов: помощь людям с нарушениями в интеллектуальной и эмоционально-волевой сфере.

# Список использованных источников

1. О приложении Choiceworks

<https://www.beevisual.com/?utm_source>

2 о приложении Proloquo2Go

<https://www.assistiveware.com/blog/new-to-proloquo2go-training?utm_source>

3 создание приложений для людей с ограниченными возможностями

<https://appbooster.com/blog/kak-sozdat-prilozhenie-dlya-lyudey-s-ogranichennymi-vozmozhnostyami/?utm_source>

4 об архитектурах мобильных приложений

https://vc.ru/dev/1485809-obzor-osnovnyh-arhitekturnyh-patternov-proektirovaniya-mobilnyh-prilozhenii

5 гайд по созданию мобильных приложений

<https://www.simbirsoft.com/blog/kak-sozdat-mobilnoe-prilozhenie-gayd-ot-idei-do-pervogo-reliza/>

6 несколько статей о GetPack compose

<https://habr.com/ru/articles/798755/?utm_source=chatgpt.com>

<https://habr.com/ru/articles/757572/?utm_source=chatgpt.com>

<https://appmaster.io/ru/blog/jetpack-compose-rukovodstvo-dlia-nachinaiushchikh?utm_source=chatgpt.com>

7 документация к GetPack compose

<https://developer.android.com/develop/ui/compose/documentation>

8 документация по Android

https://developer.android.com/docs

9 документация по Kotlin

https://kotlinlang.org/docs/home.html

10 начала использования Android Studio

https://skillbox.ru/media/code/chto-takoe-android-studio-i-kak-ey-polzovatsya/