Индивидуальное задание на курсовую работу

Тема курсовой работы: Разработка Backend мобильного приложения «спутник» для лиц с нарушением интеллекта и эмоционально-волевой сферы.

Студент: Лапшин Данила Евгеньевич

Группа: P4250

Руководитель: Васильев Дмитрий Евгеньевич

Цель работы: Создание мобильного приложения, облегчающего пользователям с нарушением интеллекта и эмоционально-волевой сферы взаимодействие с окружающим мире в бытовой сфере.

Задачи работы:

Провести анализ существующих мобильных приложений для помощи данной группе пользователей.

Изучить современные технологии и инструменты, применяемые при разработке мобильных приложений.

Разработать архитектуру и дизайн пользовательского интерфейса приложения.

Реализовать функционал приложения с использованием выбранных технологий.

Провести тестирование приложения и оценить его эффективность.

Завершение работы, подведение итогов, описание литературы.

Содержание работы:

Введение

Актуальность темы.

Цель и задачи работы.

Объект и предмет исследования.

Методы исследования.

Теоретическая часть

Обзор существующих решений – анализ существующих мобильных приложений для помощи людям с нарушением интеллекта и эмоционально-волевой сферы, оценка их функциональных возможностей, преимуществ и недостатков.

Технологии и инструменты разработки – обзор современных технологий и инструментов, применяемых при разработке мобильных приложений, обоснование выбора конкретных технологий для реализации проекта.

Обоснование разработки – определение целевой аудитории приложения, постановка требований к функционалу и дизайну приложения, ожидаемые результаты и их значимость.

Практическая часть

Проектирование приложения – разработка архитектуры приложения, создание схемы навигации и взаимодействия пользователя с приложением, проектирование базы данных.

Реализация приложения – описание процесса разработки с использованием выбранных технологий, реализация основных модулей и компонентов приложения, интеграция с внешними сервисами или устройствами (если потребуется).

Сбор статистики для оценки эффективности приложения и выдачи рекомендаций пользователям (в перспективе)

Тестирование приложения – разработка плана тестирования, проведение функционального и пользовательского тестирования, анализ результатов тестирования и выявление возможных проблем.

Документация – подготовка пользовательской инструкции, описание технической документации для разработчиков.

Заключение

Выводы – подведение итогов проделанной работы.

Достижение цели и задач – оценка достижения поставленных целей и задач.

Перспективы развития – рекомендации по дальнейшему развитию и улучшению приложения.

Список использованных источников

Перечень литературы, статей и других источников, использованных при выполнении работы.

Приложения

Дополнительные материалы: код, большие схемы (если потребуется).

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

## 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.

1. 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.

## Выбор инструментов и технологий для разработки приложения

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

1. Операционная система- эта платформа, на которой будет работать приложение.
2. Язык программирования- это язык, на котором оно будет написано.
3. 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 в основном используется несколько языков программирования, которые рассмотрены в данной научно-исследовательской работе.

4.1 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.

4.2 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.

4.3 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.

4.4 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.

4.5 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.

## Обоснование разработки и 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 памяток и инструкций.

- Добавление в раздел "Избранное" памяток и инструкций.

- Удаление из раздела "Избранное" памяток и инструкций.

- Чтение памяток и инструкций.

- 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:

Должно быть реализовано 2 экрана с навигацией между ними: главный экран и избранное.

Инструкции должны быть максимально однозначными и понятными.

Интерфейс приложения должен быть максимально простым и понятным, не содержать лишние элементы и не быть перегруженным.

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

## Архитектура приложения

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.

Так как приложение «спутник» не является сложным и высоконагруженным, для его создания было принято решение использовать архитектуру MVC.

## Реализация приложения