СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 4](#_Toc169694045)

[1. ОБЩАЯ ЧАСТЬ 5](#_Toc169694046)

[1.1. Цель разработки 5](#_Toc169694047)

[1.2. Средства разработки 5](#_Toc169694048)

[2. СПЕЦИАЛЬНАЯ ЧАСТЬ 7](#_Toc169694049)

[2.1. Постановка задачи 7](#_Toc169694050)

[2.1.1. Подробные требования к проекту 7](#_Toc169694051)

[2.2. Внешняя спецификация 7](#_Toc169694052)

[2.2.1. Описание задачи 7](#_Toc169694053)

[2.2.2. Входные и выходные данные 13](#_Toc169694054)

[2.2.3. Методы 17](#_Toc169694055)

[2.2.4. Тесты 20](#_Toc169694056)

[2.3. Проектирование 22](#_Toc169694057)

[2.3.1. Схема архитектуры приложения 22](#_Toc169694058)

[2.3.2. Логическая схема данных 22](#_Toc169694059)

[2.3.3. Физическая схема данных 22](#_Toc169694060)

[2.3.4. Структурная схема 22](#_Toc169694061)

[2.3.5. Функциональная схема 22](#_Toc169694062)

[2.3.6. Диаграмма классов 22](#_Toc169694063)

[2.3.7. Схема тестирования 25](#_Toc169694064)

[2.3.8. Схема пользовательского интерфейса 26](#_Toc169694065)

[2.4. Результат работы программы 26](#_Toc169694066)

[3. ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ 29](#_Toc169694067)

[3.1. Инструментальные средства 29](#_Toc169694068)

[3.2. Отладка программы 29](#_Toc169694069)

[3.3. Защитное программирование 30](#_Toc169694070)

[ЗАКЛЮЧЕНИЕ 32](#_Toc169694071)

[СПИСОК ИСПОЛЬЗУЕМЫХ МАТЕРИАЛОВ 34](#_Toc169694072)

ПРИЛОЖЕНИЕ А. Техническое задание

ПРИЛОЖЕНИЕ Б. Текст программы

ПРИЛОЖЕНИЕ В. Сценарий и результаты тестовых испытаний

ПРИЛОЖЕНИЕ Г. Руководство пользователя

# ВВЕДЕНИЕ

В последние годы медитация стала одной из наиболее популярных практик для улучшения психического здоровья и общего благополучия. В условиях современного мира, где уровень стресса и тревожности постоянно растет, в связи с сидячим образом жизни, неправильным питанием, отсутствием физических нагрузок, перенасыщенностью информацией, медитация предлагает эффективные методы для достижения внутреннего спокойствия и гармонии.

Медитация – это практика, направленная на концентрацию внимания и достижение состояния осознанности и спокойствия.

Регулярная практика медитации способствует снижению уровня стресса, тревожности и депрессии. Она улучшает когнитивные функции, способствует более качественному сну и повышает общую удовлетворенность жизнью. В условиях повседневного напряжения и быстрых темпов жизни, медитация становится не только способом расслабления, но и важным инструментом для поддержания психического здоровья.

Мобильные приложения для медитаций предоставляют пользователям возможность практиковать медитацию в любое удобное для них время и в любом месте. Доступность и удобство использования таких приложений делают медитацию доступной для широкой аудитории, независимо от их опыта и уровня подготовки.

В рамках выполнения курсовой работы необходимо разработать мобильное приложение «MAU». Оно позволит вести дневник медитаций, записывать свои впечатления от выполнения практики, упростит доступ к качественным материалам для проведения медитаций, таким как статьи и музыкальные композиции.

Разработка мобильного приложения «MAU» направлена на создание эффективного и удобного инструмента, который поможет пользователям справляться с повседневными стрессами, улучшать свое психическое здоровье и достигать внутренней гармонии.

# ОБЩАЯ ЧАСТЬ

## Цель разработки

Целью разработки является создание удобного инструмента для проведения практик медитации, а также предоставление материалов, способствующих повышению качества и глубины медитационного опыта.

## Средства разработки

Для проектирования, разработки и тестирования мобильного  приложения, реализующего свой функционал на базе платформы Android, были использованы программные средства, представленные в Таблице 1.

Таблица 1 - программные средства

|  |  |  |  |
| --- | --- | --- | --- |
| № | Тип средства | Название средства | Назначение |
| 1 | 2 | 3 | 4 |
| 1 | Операционная система | Microsoft Windows 11 Pro | Организация  взаимодействия  программ и  пользователя |
| 2 | Операционная система | Android 14 | Организация  взаимодействия  программ и  пользователя |
| 3 | Инструментальные  средства разработки  программного решения | Android Studio Jellyfish 2023.3.1 | Разработка  мобильного  приложения |
| 4 | Текстовый редактор | Microsoft Office LTSC  стандартный 2021 | Написание  документации |

В качестве средств вычислительной техники при разработке ПО  использовался ноутбук Asus Vivobook pro 15, для использования  приложения – мобильное устройство на базе Android Google Pixel 7, технические характеристики устройств представлены в таблице 2.

Таблица 2 - технические характеристики средств вычислительной техники

|  |  |  |
| --- | --- | --- |
| № | Тип оборудования | Наименование оборудования |
| 1 | 2 | 3 |
| Для разработки | | |
| Ноутбук Asus Vivobook pro 15 | | |
| 1 | Размер экрана: | 15,6 |
| 2 | Разрешение экрана: | 1920x1080 |
| 3 | Линейка процессора: | AMD Ryzen 7 5800H |
| 4 | Количество ядер процессора: | 8 |
| 5 | Оперативная память: | 16 ГБ |
| 6 | Тип видеокарты: | Дискретная |
| 7 | Операционная система: | Windows 11 Pro |
| Google Pixel 7 | | |
| 1 | Размер экрана: | 6.2 дюйма, AMOLED |
| 2 | Разрешение экрана: | 2400 x 1080 пикселей, частота обновления 120 Гц |
| 3 | Линейка процессора: | Qualcomm Snapdragon 8 Gen 2 |
| 4 | Количество ядер процессора: | 8 |
| 5 | Оперативная память: | 8 ГБ |
| 6 | Тип видеокарты: | Adreno 730 |
| 7 | Операционная система: | Android 14 |
| Для использования | | |
| Google Pixel 7 | | |
| 1 | Размер экрана: | 6.2 дюйма, AMOLED |
| 2 | Разрешение экрана: | 2400 x 1080 пикселей, частота обновления 120 Гц |
| 3 | Линейка процессора: | Qualcomm Snapdragon 8 Gen 2 |
| 4 | Количество ядер процессора: | 8 |
| 5 | Оперативная память: | 8 ГБ |
| 6 | Тип видеокарты: | Adreno 730 |
| 7 | Операционная система: | Android 14 |

Таблица 3 содержит информацию о минимальных и рекомендуемых технических характеристиках, необходимых для комфортного использования создаваемого программного обеспечения.

Таблица 3 - минимальные и рекомендуемые требования

|  |  |  |
| --- | --- | --- |
| № | Тип оборудования | Наименование оборудования |
| 1 | 2 | 3 |
| Максимальные или рекомендуемые | | |
| 1 | Разрешение экрана | Full HD (1080x1920) и выше |
| 2 | Линейка процессора | Qualcomm Snapdragon 6xx и выше |
| 3 | Количество ядер процессора | 4 ядра |
| 4 | Оперативная память | 4 ГБ и больше |
| 5 | Общий объем всех накопителей | 64 ГБ и больше |
| 6 | Операционная система | Android 10 и выше |
| 7 | Клавиатура | Виртуальная |
| Минимальные | | |
| 1 | Разрешение экрана | HD (720x1280) |
| 2 | Линейка процессора | Qualcomm Snapdragon 4xx |
| 3 | Количество ядер процессора | 4 ядра |
| 4 | Оперативная память | 2 ГБ |
| 5 | Общий объем всех накопителей | 32 ГБ |
| 6 | Операционная система | Android 7.0 |
| 7 | Клавиатура | Виртуальная |

# СПЕЦИАЛЬНАЯ ЧАСТЬ

## Постановка задачи

Разработать мобильное приложение «MAU», направленное на поддержку психического здоровья пользователей через практику медитации и осознанности.

Необходимо разработать функциональный интерфейс для ведения дневника медитаций и записи результатов практик, обеспечить доступ к качественным материалам для медитаций, таким как статьи и аудиофайлы, реализовать систему настроек аккаунта пользователя, а также плеер для прослушивания аудиоматериалов.

### Подробные требования к проекту

Подробные требования к разработке приложения «MAU» представлены в Приложении А. Техническое задание.

## Внешняя спецификация

### Описание задачи

Необходимо разработать мобильное приложение для медитаций «MAU» на языке Java. Приложение разрабатывается для упрощения проведения практик медитаций, а также доступа к качественным ресурсам для улучшения своего ментального здоровья.

Для хранения информации о пользователях, статьях, заметках и музыкальных композициях должны быть использованы компоненты Firebase, такие как Firebase Firestore, Firebase Storage, Firebase Authentication.

В приложении должно быть предусмотрено 2 роли: администратор и пользователь.

Роль администратора должна включать в себя следующие функции:

1. Авторизация;
2. Настройка профиля:

* Настройка подписки на push-уведомления;
* Изменение пароля от учетной записи;

1. Выход из системы;
2. Управление «дневником медитаций»:

* Просмотр списка своих заметок в дневнике, а также каждой заметки отдельно;
* Добавление новых заметок;
* Редактирование и удаление существующих заметок;

1. Управление статьями:

* Просмотр списка статей, а также каждой статьи отдельно;
* Добавление новых статей;
* Редактирование и удаление существующих статей;

1. Управление аудиотекой:

* Просмотр списка медитативной музыки;
* Добавление новой музыки;
* Удаление существующей музыки;

1. Прослушивание медитативной музыки:

* Возможность паузы, воспроизведения и завершения музыки;

Роль пользователя должна включать в себя следующие функции:

1. Регистрация:

* Регистрация новых пользователей в системе;

1. Авторизация:

* Авторизация существующих пользователей с использованием уникальных учетных данных;

1. Настройка профиля:

* Настройка подписки на push-уведомления;
* Изменение имени пользователя;
* Изменение пароля от учетной записи;

1. Выход из системы;
2. Удаление учетной записи из системы:

* Удаление всех данных, связанных с аккаунтом пользователя;

1. Управление «дневником медитаций»:

* Просмотр списка своих заметок в дневнике, а также каждой заметки отдельно;
* Добавление новых заметок;
* Редактирование и удаление существующих заметок;

1. Управление статьями:

* Просмотр списка статей, а также каждой статьи отдельно;

1. Управление аудиотекой:

* Просмотр списка медитативной музыки;

1. Прослушивание медитативной музыки:

* Возможность паузы, воспроизведения и завершения музыки;

Администратор может войти в систему, введя следующие учетные данные:

* Адрес электронной почты – [admin@gmail.com](mailto:admin@gmail.com);
* Пароль – admin123!!!.

Администратор заведомо знает свои учетные данные, пользователи должны пройти процесс регистрации, для входа в систему.

Приложение должно иметь адаптивный под различные устройства интерфейс.

На рисунке 1 представлен алгоритм регистрации пользователя в системе:

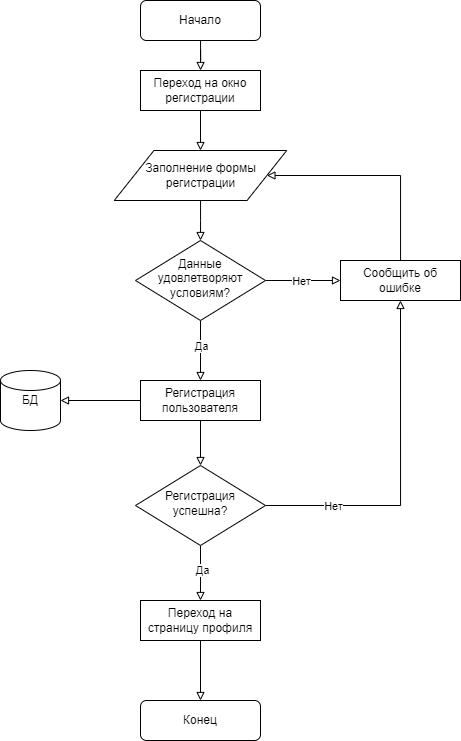


Рисунок 1 – алгоритм функции регистрации пользователя в системе

При запуске приложения пользователь попадает на окно авторизации, ему необходимо нажать на ссылку для перехода на окно регистрации, после чего необходимо заполнить форму регистрации, которая включает в себя поля «Логин», «Email» и «Пароль», и нажать на кнопку «Регистрация». Данные подвергаются проверке, если они не соответствуют условиям выводится соответствующая ошибка и пользователю необходимо ввести данные, удовлетворяющие условиям. При успешно пройденной проверке происходит регистрация пользователя, она подразумевает под собой создание записи о пользователе в базе данных. При успешном завершении процесса регистрации пользователь перенаправляется на страницу профиля. При неудачной регистрации выводится соответствующая ошибка.

На рисунке 2 представлен алгоритм авторизации пользователя в системе:

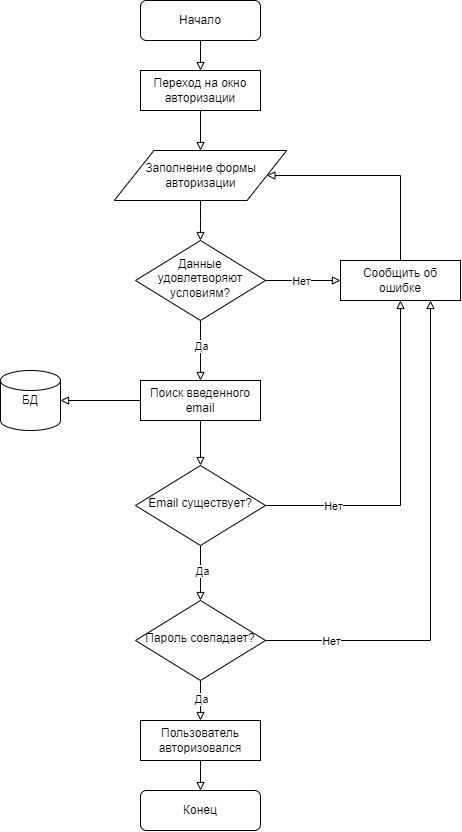


Рисунок 2 - алгоритм функции авторизации пользователя

При запуске приложения пользователь попадает на страницу авторизации, где ему необходимо заполнить форму, включающую в себя 2 поля: «Email» и «Пароль», и нажать на кнопку «Войти». Если данные не удовлетворяют условиям валидации – выводится соответствующая ошибка и пользователю необходимо ввести учетные данные повторно. Если данные удовлетворяют условиям валидации введенный адрес электронной почты проверяется на наличие в БД, если такой адрес существует пароль проверяется на соответствие. Если email или пароль были введены неверно – выводится соответствующая ошибка, иначе – пользователь успешно авторизуется в системе и попадает на страницу профиля.

На рисунке 3 представлен общий алгоритм добавления данных в базу данных:

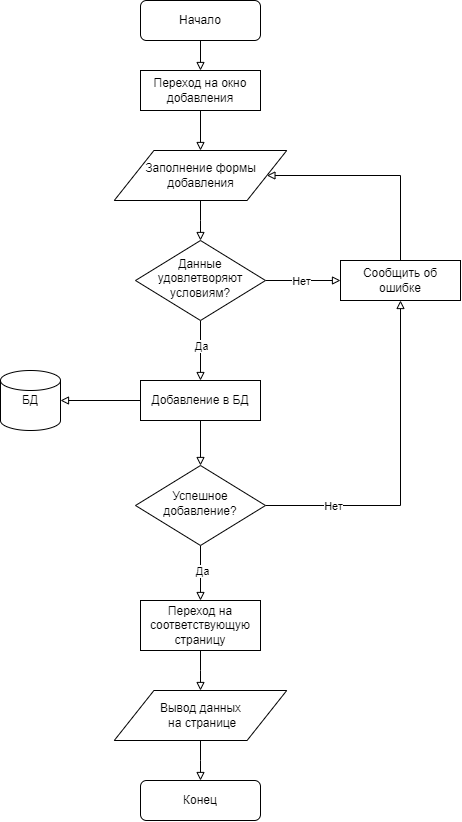


Рисунок 3 - алгоритм функции добавления данных в БД

При запуске процесса добавления данных пользователь попадает на окно добавления, где ему необходимо заполнить форму, включающую в себя поля, зависимые от того или иного вида добавляемой записи (например заметка имеет поля «Дата» и «Описание»). После заполнения формы, если введенные данные не удовлетворяют условиям валидации, выводится соответствующая ошибка, и пользователю необходимо ввести данные повторно. Если данные удовлетворяют условиям валидации, происходит добавление данных в базу данных.

После попытки добавления данных в базу проверяется успешность операции. Если данные не были успешно добавлены, выводится соответствующая ошибка. В случае успешного добавления данных осуществляется переход на соответствующую страницу, где выводятся добавленные данные.

### Входные и выходные данные

В таблице 4 представлены входные данные мобильного приложения «MAU».

Таблица 4 - входные данные

| № | Поле | Тип данных | Ограничения | Формат ввода | Описание |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 |
| Окно авторизации | | | | | |
| 1 | Адрес электронной почты | Строка | [a-zA-Z0-9\\+\\.\\\_\\%\\-\\+]{1,256}" +"\\@" +"[a-zA-Z0-9][a-zA-Z0-9\\-]{0,64}" +"(" +  "\\." +"[a-zA-Z0-9][a-zA-Z0-9\\-]{0,25}" | Поле ввода | Ввод адреса электронной почты для авторизации |
| 2 | Пароль | Строка | ^(?=.\*[0-9])(?=.\*[!@#$%^&\*])[A-Za-z0-9!@#$%^&\*]{8,}$ | Поле ввода | Ввод пароля для авторизации |
| Окно регистрации | | | | | |
| 1 | Логин | Строка | ^(?=.\*[A-Z])[A-Za-z]{6,18}$ | Поле ввода | Ввод имени пользователя (логина) для регистрации пользователя |
| 2 | Пароль | Строка | ^(?=.\*[0-9])(?=.\*[!@#$%^&\*])[A-Za-z0-9!@#$%^&\*]{8,}$ | Поле ввода | Ввод пароля для регистрации |
| 3 | Адрес электронной почты | Строка | [a-zA-Z0-9\\+\\.\\\_\\%\\-\\+]{1,256}" +"\\@" +"[a-zA-Z0-9][a-zA-Z0-9\\-]{0,64}" +"(" +  "\\." +"[a-zA-Z0-9][a-zA-Z0-9\\-]{0,25}" | Поле ввода | Ввод адреса электронной почты для регистрации |
| Окно добавления заметки | | | | | |
| 1 | Дата | Дата | Нет ограничений | Выбор даты | Выбор даты из календаря |
| 2 | Описание заметки | Строка | ^[1, ]$ | Поле ввода | Ввод описания заметки |
| Окно изменения заметки | | | | | |
| 1 | Дата | Дата | Нет ограничений | Выбор даты | Выбор даты из календаря |
| 2 | Описание заметки | Строка | ^[1, ]$ | Поле ввода | Ввод описания заметки |
| Окно добавления статьи | | | | | |
| 1 | Заголовок | Строка | ^[1, ]$ | Поле ввода | Ввод заголовка статьи |
| 2 | Описание | Строка | ^[1, ]$ | Поле ввода | Ввод описания статьи |
| 3 | Текст статьи | Строка | ^[1, ]$ | Поле ввода | Ввод текста статьи |
| 4 | URL фото статьи | Строка | ^[1, ]$ | Поле ввода | Ввод url ссылки на картинку статьи |
| Окно изменения статьи | | | | | |
| 1 | Заголовок | Строка | ^[1, ]$ | Поле ввода | Ввод заголовка статьи |
| 2 | Описание | Строка | ^[1, ]$ | Поле ввода | Ввод описания статьи |
| 3 | Текст статьи | Строка | ^[1, ]$ | Поле ввода | Ввод текста статьи |
| 4 | URL фото статьи | Строка | ^[1, ]$ | Поле ввода | Ввод url на картинку статьи |
| Окно добавления мелодии | | | | | |
| 1 | Аудио | Файл | Расширения .mp3, .aac, .wav, .ogg | Выбор файла | Выбор аудиофайла |
| 2 | Название мелодии | Строка | ^[1, 30]$ | Поле ввода | Ввод названия мелодии |
| 3 | Описание | Строка | ^[1, 70]$ | Поле ввода | Ввод описания мелодии |
| 4 | URL фото мелодии | Строка | ^[1, }$ | Поле ввода | Ввод url на картинку мелодии |
| Модальное окно изменения имени пользователя | | | | | |
| 1 | Логин | Строка | ^(?=.\*[A-Z])[A-Za-z]{6,18}$ | Поле ввода | Ввод нового имени (логина) |
| Модальное окно изменения пароля | | | | | |
| 1 | Пароль | Строка | ^(?=.\*[0-9])(?=.\*[!@#$%^&\*])[A-Za-z0-9!@#$%^&\*]{8,}$ | Поле ввода | Ввод нового пароля |

В таблице 5 представлены выходные данные мобильного приложения «MAU».

Таблица 5 - выходные данные

| № | Поле | Тип данных | Ограничения | Описание |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| Страница профиля | | | | |
| 1 | Логин | Строка | ^(?=.\*[A-Z])[A-Za-z]{6,18}$ | Вывод логина пользователя |
| 2 | Адрес электронной почты | Строка | [a-zA-Z0-9\\+\\.\\\_\\%\\-\\+]{1,256}" +"\\@" +"[a-zA-Z0-9][a-zA-Z0-9\\-]{0,64}" +"(" +  "\\." +"[a-zA-Z0-9][a-zA-Z0-9\\-]{0,25}" | Вывод адреса электронной почты пользователя |
| 3 | Дата | Дата | В формате dd.MM.yyyy | Вывод даты добавления заметки |
| 4 | Описание | Строка | ^[1, ]$ | Вывод описания заметки |
| Страница статей | | | | |
| 1 | Заголовок | Строка | ^[1, ]$ | Вывод заголовка статьи |
| 2 | Описание | Строка | ^[1, ]$ | Вывод описания статьи |
| 3 | Фото статьи | Изображение | Нет ограничений | Вывод картинки статьи |
| Страница просмотра заметки | | | | |
| 1 | Дата | Дата | В формате dd.MM.yyyy | Вывод даты добавления заметки |
| 2 | Описание | Строка | ^[1, ]$ | Вывод описания заметки |
| Страница просмотра статьи | | | | |
| 1 | Заголовок | Строка | ^[1, ]$ | Вывод заголовка статьи |
| 2 | Описание | Строка | ^[1, ]$ | Вывод описания статьи |
| 3 | Текст статьи | Строка | ^[1, ]$ | Вывод текста статьи |
| Главная страница | | | | |
| 1 | Фото мелодии | Изображение | Нет ограничений | Вывод картинки мелодии |
| 2 | Название мелодии | Строка | ^[1, 30]$ | Вывод названия мелодии |
| 3 | Описание | Строка | ^[1, 70]$ | Вывод описания мелодии |
| Страница настроек | | | | |
| 1 | Логин | Строка | ^(?=.\*[A-Z])[A-Za-z]{6,18}$ | Вывод логина пользователя |
| 2 | Адрес электронной почты | Строка | [a-zA-Z0-9\\+\\.\\\_\\%\\-\\+]{1,256}" +"\\@" +"[a-zA-Z0-9][a-zA-Z0-9\\-]{0,64}" +"(" +  "\\." +"[a-zA-Z0-9][a-zA-Z0-9\\-]{0,25}" | Вывод адреса электронной почты пользователя |
| Окно плеера | | | | |
| 1 | Название мелодии | Строка | ^[1, 30]$ | Вывод названия мелодии |
| 2 | Описание | Строка | ^[1, 70]$ | Вывод описания мелодии |
| 3 | Фото мелодии | Изображение | Нет ограничений | Вывод картинки мелодии |

### Методы

В процессе разработки программного кода необходимо производить отладку кода (debugging), в свою очередь отладка включает в себя:

Использование отладчика (debugger) – это инструмент, встроенный в среду разработки, который позволяет контролировать выполнение программы и анализировать её поведение.

Процесс начинается с установки точек останова в тех местах кода, где необходимо приостановить выполнение программы для анализа. После установки точек останова можно запустить приложение в режиме отладки, нажав кнопку «Debug» (значок жука) на панели инструментов. Программа будет запущена, и выполнение остановится на первой достигнутой точке останова.

Также можно начать пошаговое выполнение кода, для подробного анализа каждой операции. Во время остановки программы можно просматривать и изменять значения переменных и многое другое.

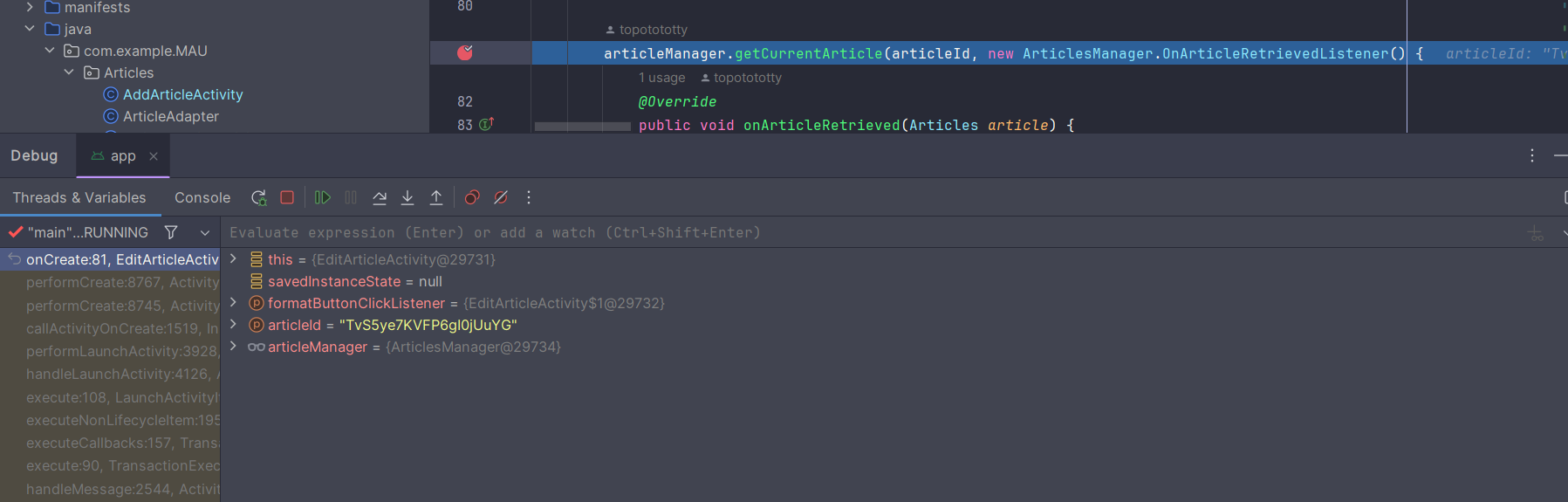


Рисунок 4 – использование отладчика

Логирование (logging) – это процесс записи сообщений о работе программы в лог-файл или консоль для последующего анализа. Логирование помогает отслеживать поведение приложения и диагностировать проблемы.

Логирование производилось при помощи класса Log из пакета android.util, Класс Log предоставляет несколько методов для записи логов на различных уровнях, таких как d (debug), i (info), w (warn), e (error), и v (verbose), они позволяют эффективно фильтровать и анализировать логи, фокусируясь на наиболее важных и релевантных сообщениях.

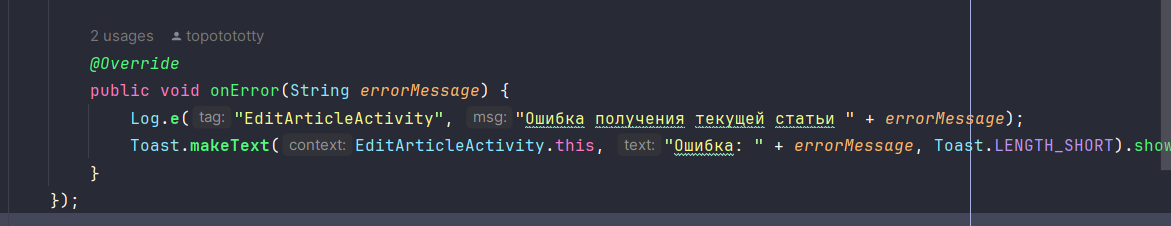


Рисунок 5 - логирование ошибок

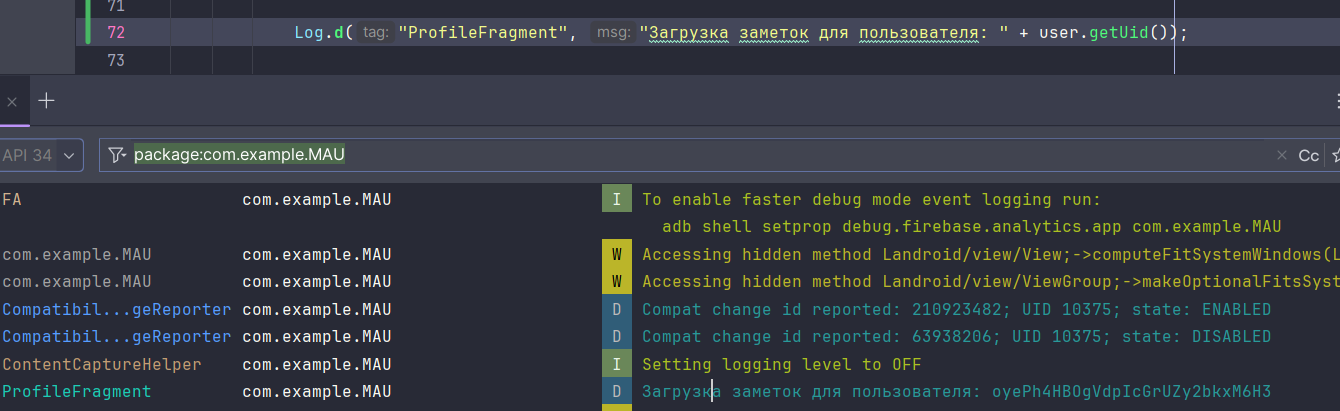


Рисунок 6 - логирование загрузки заметок для пользователя

Рефакторинг – это процесс, направленный на улучшение структуры написанного кода без изменения его внешнего поведения. Его основная цель – сделать код более чистым, понятным и поддерживаемым. Рефакторинг позволяет снизить число ошибок, устранить повторяющийся код, повысить производительность и т. д.



Рисунок 7 - код до рефакторинга (1)



Рисунок 8 - код до рефакторинга (2)



Рисунок 9 - код до рефакторинга (3)

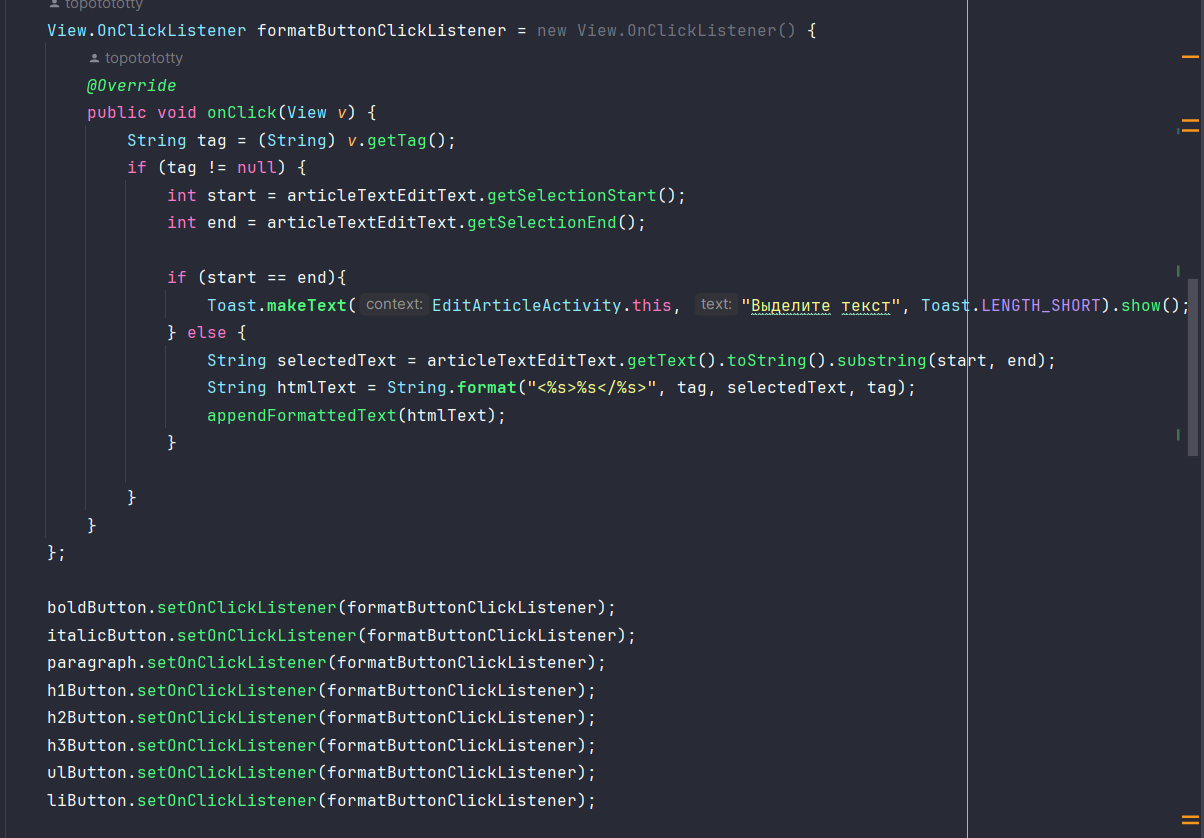


Рисунок 10 - код после рефакторинга

### Тесты

Тестирование по тестам – тестирование по предварительно написанным тест-кейсам, предназначено для проверки работы функций мобильного приложения и выявления дефектов программы.

1. По исполнению кода:

Динамическое тестирование – метод тестирования программного обеспечения, при котором выполняется работающая программа с целью проверки ее поведения и функциональности в реальном времени.

1. По уровню тестирования:

Системное – проверка работы всей системы на соответствие заявленным требованиям к программному продукту. Этот вид тестирования охватывает все аспекты работы системы, включая функциональность, производительность, безопасность и удобство использования

1. По целям:

Функциональное тестирование – тип тестирования ПО, направленный на проверку функциональности системы в соответствии с требованиями и спецификациями. Его основная цель – убедиться в том, что все функции и особенности системы работают правильно.

1. По степени автоматизации:

Ручное – без использования дополнительных программных средств.

1. По позитивности сценария:

Позитивное тестирование направлено на проверку системы при использовании корректных и ожидаемых входных данных. Цель этого вида тестирования — убедиться, что система выполняет свои функции правильно, когда все условия соблюдены.

Негативное тестирование проверяет систему на устойчивость и правильное поведение при использовании некорректных, неожиданных или ошибочных входных данных. Цель этого вида тестирования — убедиться, что система правильно обрабатывает ошибки и исключительные ситуации, не приводя к сбоям или некорректной работе.

1. По знанию системы:

Тестирование «белого ящика» – тестирование программного продукта с доступом к коду.

Тестирование «черного ящика» – тестирование без доступа к коду продукта.

1. По разработке тестовых испытаний:

На основе требований – требование было определено до начала тестирования.

## Проектирование

## Схема архитектуры приложения

На рисунке … представлена схема архитектуры мобильного приложения «MAU».

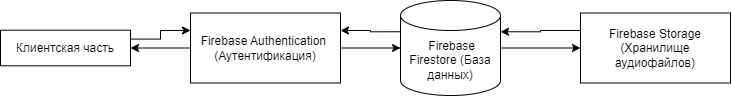


Рисунок 11 - схема архитектуры приложения

## Логическая схема данных

Хранение данных в Firebase Firestore осуществляется в виде коллекций и документов. Каждая коллекция состоит из множества документов, и каждый документ может содержать вложенные данные, такие как поля или вложенные коллекции.

Firebase Firestore является примером нереляционной базы данных, следовательно логическая модель данных не предоставляется.

## Физическая схема данных

Firebase Firestore является примером нереляционной базы данных, следовательно физическая модель данных не предоставляется.

## Диаграмма классов

На рисунках 12 - 16 представлена диаграмма классов приложения «MAU».

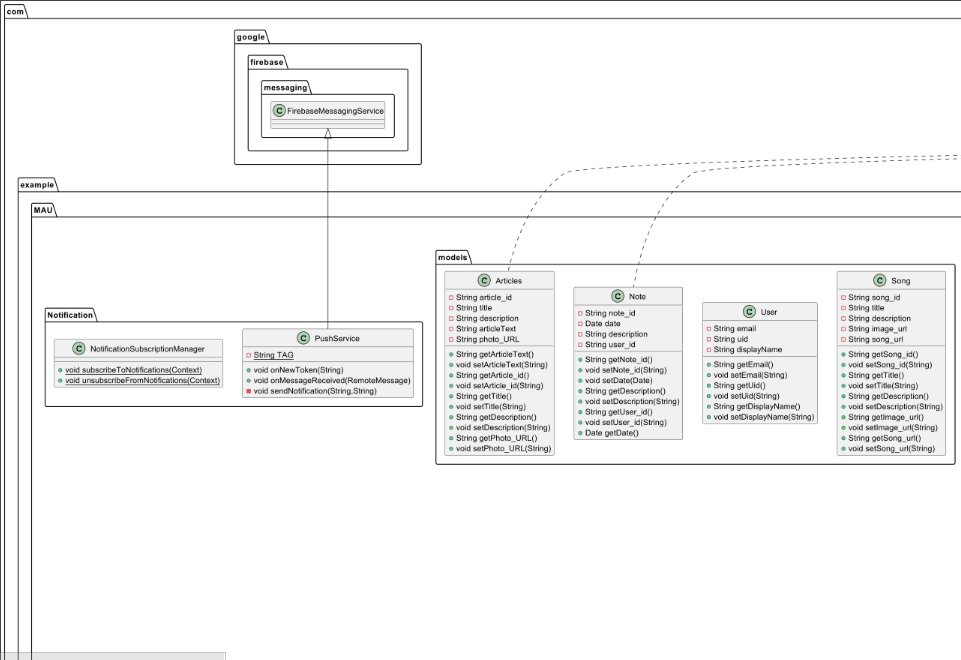


Рисунок 12 - диаграмма классов (1)

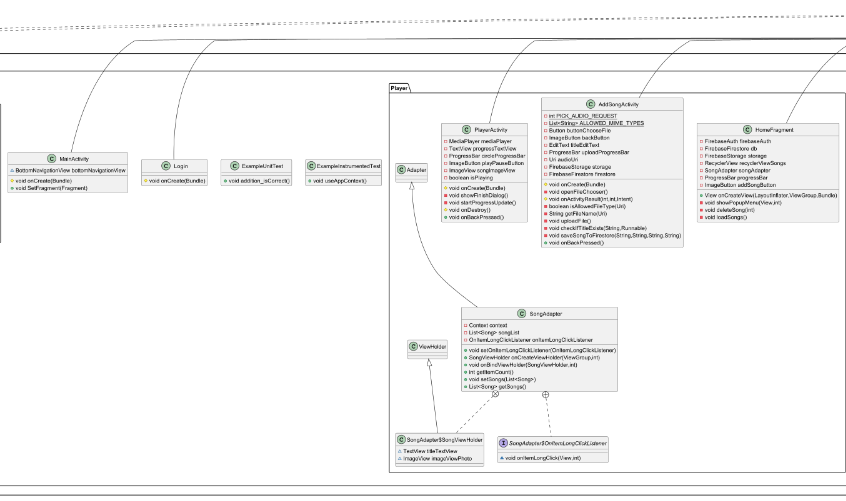


Рисунок 13 - диаграмма классов (2)

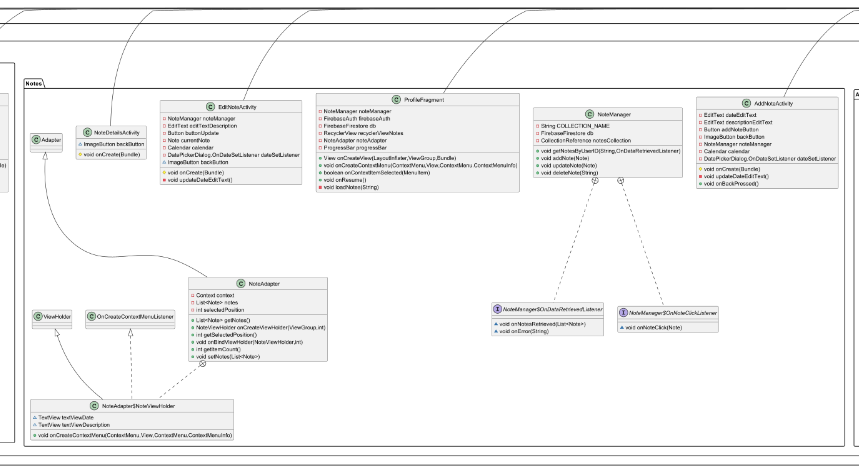


Рисунок 14 - диаграмма классов (3)

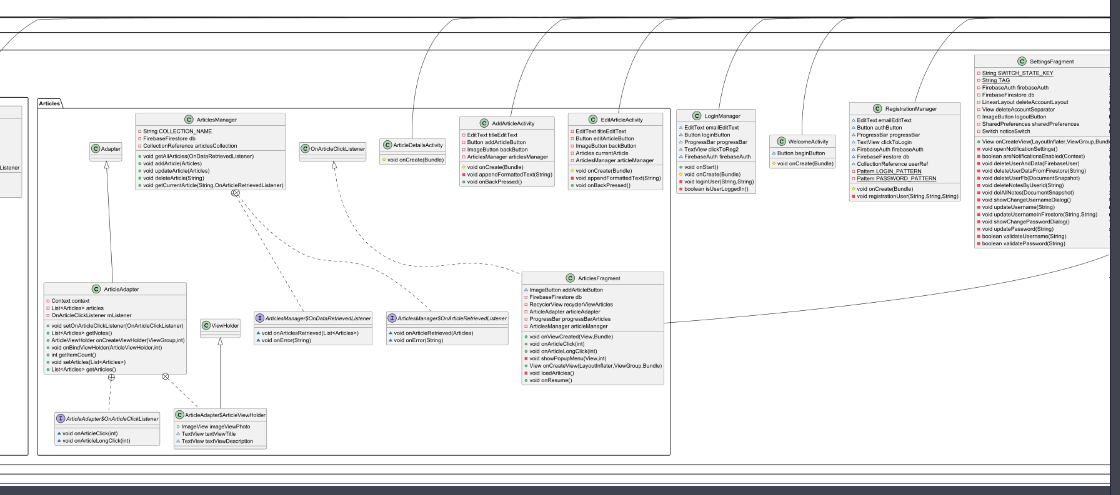


Рисунок 15 - диаграмма классов (4)

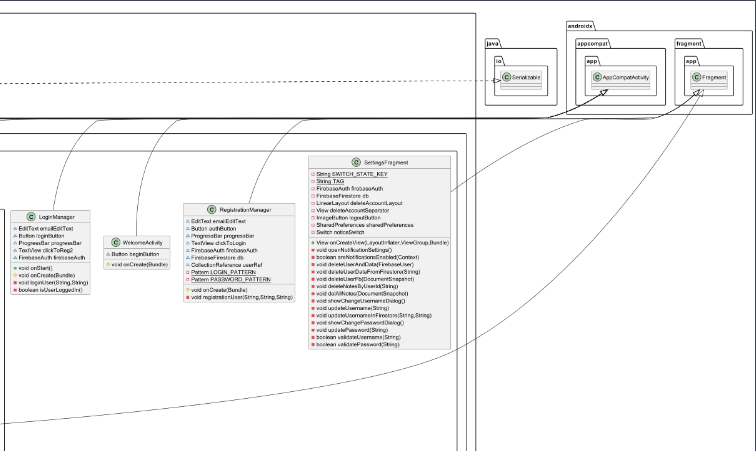


Рисунок 16 - диаграмма классов (5)

## Схема тестирования

Схема тестирования представляет собой структурированный план или модель, описывающую процесс тестирования программного обеспечения или системы. Она используется для определения стратегии тестирования, этапов, методов и ресурсов, необходимых для проведения тестирования.

На рисунке … представлена схема тестирования мобильного приложения «MAU»

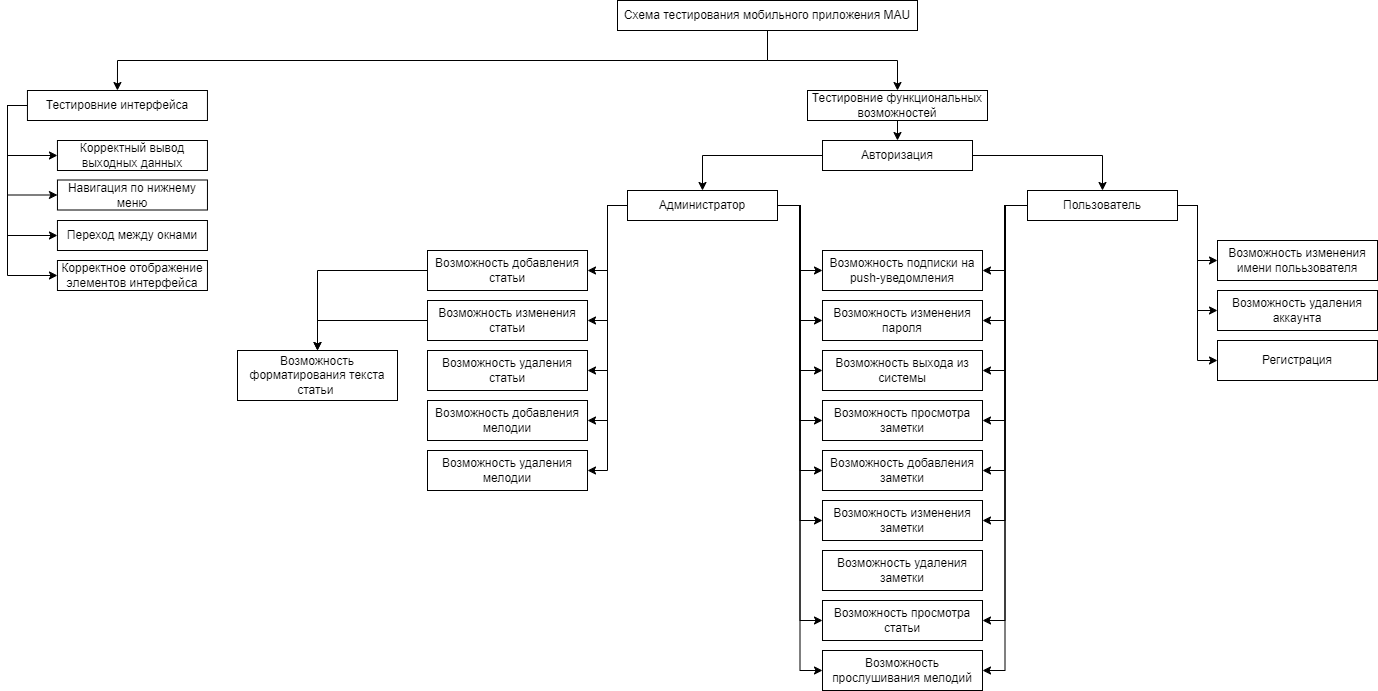


Рисунок 17 - схема тестирования

## Схема пользовательского интерфейса

Схема пользовательского интерфейса представляет собой визуальное отображение структуры и элементов интерфейса пользователя.

На рисунке 18 представлена схема пользовательского интерфейса мобильного приложения «MAU», голубым цветом обозначены страницы или окна, доступные только администратору.

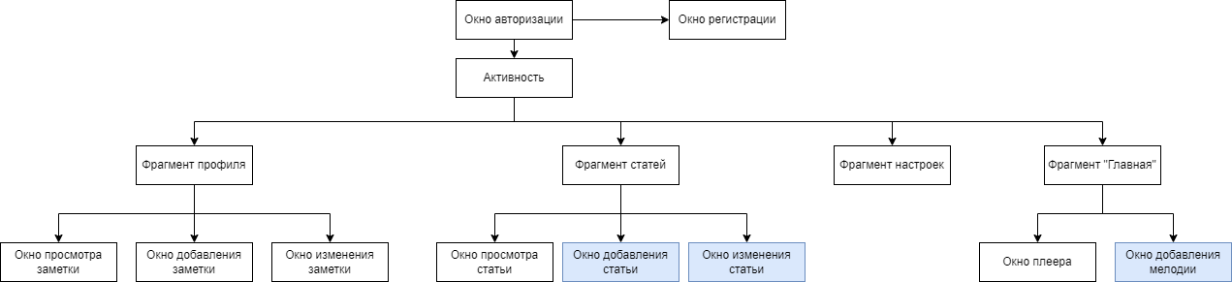


Рисунок 18 - схема пользовательского интерфейса

## Результат работы программы

На рисунке 19 представлено окно авторизации, главное окно с навигацией, а также страница профиля и статей.

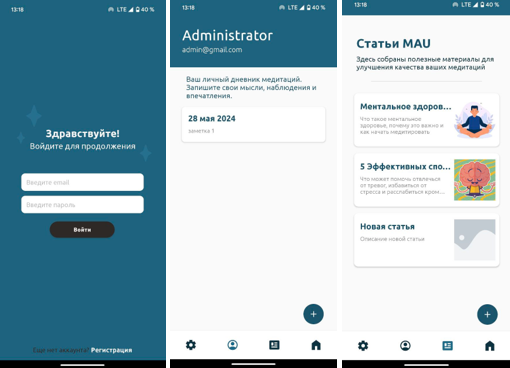


Рисунок 19 - работа программы (1)

На рисунке 20 представлено окно просмотра статьи, добавления мелодии и редактирования статьи.

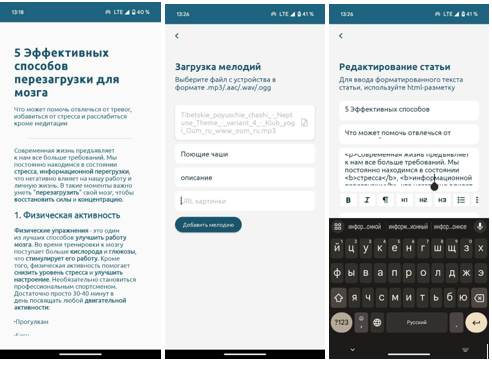


Рисунок 20 - работа программы (2)

На рисунке 21 представлена страница настроек администратора, а также добавления заметки.

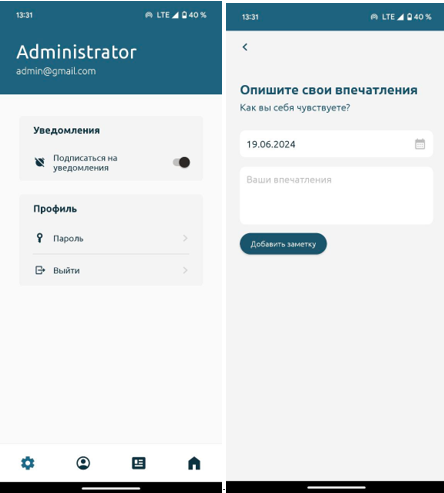


Рисунок 21 - работа программы (3)

# ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ

## Инструментальные средства

Для разработки мобильного приложения для медитаций «MAU» на языке Java была использована интегрированная среда разработки Android Studio. Она обладает широкими возможностями для создания и отладки приложений на платформе Android.

В сравнении с другими интегрированными средами разработки для Android, такими как Eclipse и IntelliJ IDEA, Android Studio обладает более простым и удобным интерфейсом, что позволяет быстро создавать приложения.

Для хранения информации о пользователях, статьях, заметках и музыкальных композициях были использованы компоненты Firebase, такие как Firebase Firestore, Firebase Storage и Firebase Authentication. Эти инструменты обеспечивают надежное и масштабируемое хранение данных, а также безопасность и аутентификацию пользователей.

## Отладка программы

Отладка мобильного приложения производилась при помощи встроенных инструментов Android Studio. Эти инструменты включают в себя:

Эмуляторы и физические устройства:

Возможность тестирования приложения на различных виртуальных устройствах и реальных устройствах для проверки совместимости и производительности.

Логирование (Logcat):

Средство просмотра логов работы приложения, что позволяет отслеживать ошибки, предупреждения и информационные сообщения.

Отладчик (Debugger):

Инструмент для пошагового выполнения кода, проверки значений переменных и состояния приложения в реальном времени.

## Защитное программирование

В программном коде предусмотрена обработка всех возможных ошибок при эксплуатации программы, неожиданных вылетов приложения в различных частях кода при помощи блока if-else, try-catch, а также в некоторых частях с помощью регулярных выражений для того, чтобы введенные данные соответствовали требуемому формату.



Рисунок 23 - блок if-else (1)

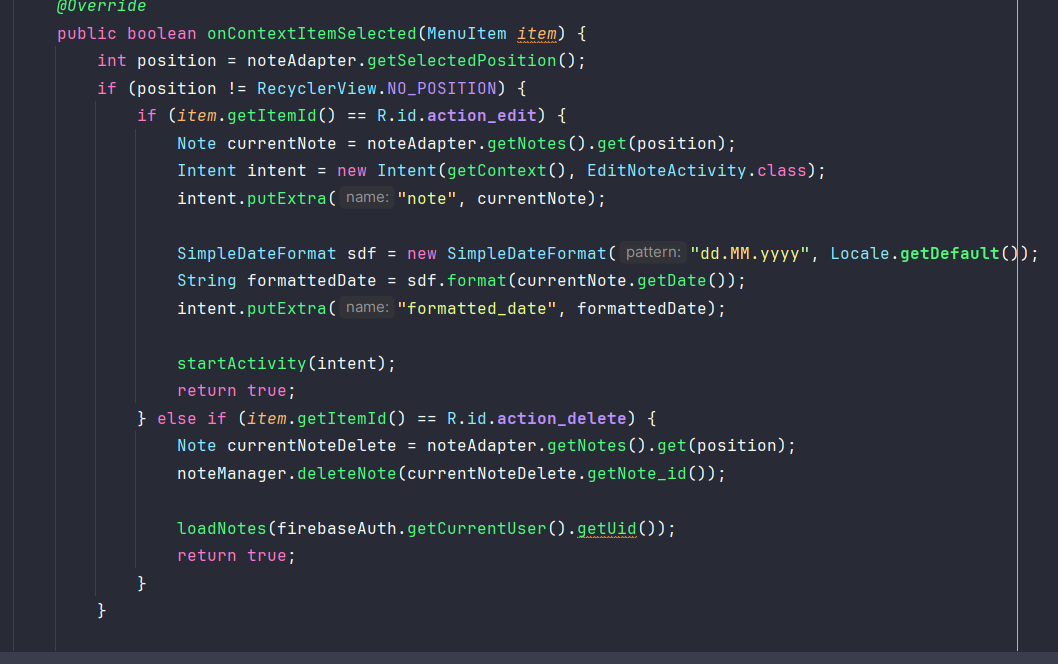


Рисунок 24 - блок if-else (2)

* 1. Характеристики программы

Разработанное мобильное приложение может запускаться на любых телефонах Android, которые имеют версию операционной системы не ниже 7.0

# ЗАКЛЮЧЕНИЕ

Разработка приложения «MAU» включала исследование потребностей пользователей, анализ существующих решений и внедрение лучших практик в области мобильных технологий и дизайна пользовательского интерфейса.

В ходе выполнения курсового проекта было разработано мобильное приложение «MAU», предназначенное для практики медитации.

В ходе выполнения курсового проекта на тему «Разработка мобильного приложения для медитаций «MAU»» были выполнены все поставленные требования.

В процессе анализа предметной области, была выполнена работа по постановке и описанию задачи.

В рамках курсового проекта были выполнены все необходимые этапы проектирования, включая создание различных схем и моделей данных, а также описание входных и выходных данных. Это обеспечило эффективное начало разработки мобильного приложения, которое было создано в соответствии с предъявленными требованиями и успешно выполняет поставленные задачи.

Выполнение курсового проекта позволило закрепить и расширить практические навыки в разработке и проектировании мобильных приложений. Особое внимание было уделено использованию Firebase для управления базой данных и аутентификации, что упростило реализацию серверной части и улучшило производительность приложения. Разработка мобильного приложения велась на языке программирования Java, что позволило использовать преимущества его объектно-ориентированного подхода и обеспечить надежную работу приложения.

На этапе тестирования было проведено тщательное исследование всех функциональных возможностей приложения, что позволило выявить и исправить недостатки, связанные с функциональностью, безопасностью и производительностью. В результате тестирования стало очевидно, что приложение работает стабильно, без сбоев и вылетов, благодаря применению различных методов защитного программирования.

Результат выполнения курсового проекта полностью соответствовал поставленным задачам и позволил получить новые знания и ценный опыт в области мобильной разработки

Проект показал, что в условиях современного мира, где уровень стресса и тревожности постоянно растет, такие приложения могут стать важным инструментом для улучшения психического здоровья и общего благополучия.

# СПИСОК ИСПОЛЬЗУЕМЫХ МАТЕРИАЛОВ

1. Разработка пользовательского интерфейса. [Электронный ресурс] URL: <https://developer.android.com/guide/topics/ui> (Дата обращения: 17.02.2024).
2. Документация Android Studio. . [Электронный ресурс] URL: <https://developer.android.com/studio> (Дата обращения: 13.06.2024).
3. Обработка ошибок и исключений в Android. . [Электронный ресурс] URL: <https://developer.android.com/studio/debug> (Дата обращения: 20.04.2024).
4. Создание макета для приложения. . [Электронный ресурс] URL: <https://www.figma.com/> (Дата обращения: 11.02.2024).
5. Исправление ошибок в проекте Android. . [Электронный ресурс] URL: <https://stackoverflow.com/questions/73735042/android-studio-resource-linking-failed-for-resource-xml-not-found> (Дата обращения: 05.05.2024).
6. Обзор архитектурных компонентов Android. . [Электронный ресурс] URL: <https://developer.android.com/jetpack/guide> (Дата обращения: 18.06.2024).
7. Firestore documentation . [Электронный ресурс] URL: <https://cloud.google.com/firestore/docs> (Дата обращения: 03.04.2024)