МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**
Кафедра математической кибернетики и компьютерных наук

РАЗРАБОТКА ANDROID ПРИЛОЖЕНИЯ ДЛЯ ТАЙМ МЕНЕДЖМЕНТА

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 411 группы
направления 02.03.02 — Фундаментальная информатика и информационные технологии
факультета КНиИТ
Павленко Егора Романовича

Научный руководитель
доцент, к. ф.-м. н. _____ Ю. Н. Кондратова

Заведующий кафедрой
к. ф.-м. н., доцент _____ С. В. Миронов

Саратов 2023

Введение.

В современном мире ритм жизни постоянно ускоряется, и организация времени становится ключевым фактором успешности в работе и личной жизни. Эффективное использование времени и планирование задач позволяет достигать поставленных целей, снижает уровень стресса и повышает общую продуктивность. Тайм-менеджеры играют важную роль в этом процессе, предоставляя инструменты для планирования, контроля и анализа использования времени. В последнее время наблюдается рост популярности мобильных и веб-приложений для тайм-менеджмента, однако, многие из них не полностью удовлетворяют потребности пользователей и имеют ряд недостатков.

Существующие приложения часто обладают ограниченными функциональными возможностями, не всегда удобны в использовании, могут быть перегружены ненужными функциями или наоборот, не предлагать необходимого функционала. Кроме того, многие приложения являются платными или предлагают ограниченный функционал в бесплатной версии, что также является препятствием для их широкого использования.

Актуальность данного исследования обусловлена необходимостью создания новых тайм-менеджеров, которые могут решить существующие проблемы и ограничения текущих приложений. Проблематика исследования связана с разработкой эффективных алгоритмов и методов для обработки и анализа данных, а также с выбором подходящих технологий и языков программирования для создания backend приложения, обеспечивающего стабильную, надежную и безопасную работу тайм-менеджера.

Целью данной работы является разработка Тайм-менеджера в виде Android-приложения. Для этого необходимо решить следующие задачи:

- Изучение существующих тайм-менеджеров и выявление их недостатков и ограничений.
- Определение функциональных требований к разрабатываемому тайм-менеджеру.
- Выбор инструментов для разработки приложения.
- Разработка и реализация мобильного приложения под операционную систему Android на языке Koltin.

Выпускная квалификационная работа состоит из введения, 2 глав, заключения, списка использованных источников, включающего 22 наименования, работа изложена на 56 листах текста, с приведенным кодом программного про-

дукта. Далее приведены наименования глав:

1. Обзор технологий и инструментов;
2. Реализация приложения для тайм-менеджмента.

Основное содержание работы.

В первой главе рассматриваются технологии и инструменты, которые использовались в процессе разработки приложения.

1. Принципы тайм-менеджмента - эта секция исследует основы тайм-менеджмента, его понятие, цели и методы. Рассмотрены основные подходы к планированию времени и организации работы, а также важность эффективного использования времени в современном обществе. Основные подходы к тайм-менеджменту включают методы, такие как постановка СМАРТ целей, "техника Помодоро и многие другие. Важность эффективного использования времени в современном обществе не может быть недооценена, особенно в эпоху информационного перегруза.
2. Особенности и преимущества Android-системы в контексте разработки мобильных приложений. Здесь рассматриваются особенности Android как платформы для разработки мобильных приложений, включая её широкую распространенность, открытость, гибкость и мощные инструменты разработки.Открытость кода позволяет разработчикам легко адаптировать систему под свои потребности, а широкая распространенность делает Android идеальной платформой для достижения большой аудитории. Кроме того, Android предоставляет богатый набор API и библиотек, которые облегчают разработку мобильных приложений.
3. Существующие тайм-менеджеры на Android и их функциональность. Этот раздел представляет собой обзор наиболее популярных существующих приложений для управления временем на платформе Android. Основной акцент делается на их функциональность, простоту использования и оценку пользователей. Среди рассматриваемых приложений описаны такие, как Todoist, Trello, Wunderlist, Evernote, Google Keep, TickTick, Any.do. На основе их функционала был выделен ряд наиболее значимых для пользователей функций приложений для управления временем.
4. Архитектура Android-приложения. Рассматриваются основные компоненты и принципы архитектуры Android-приложения, а также стек технологий и библиотек для создания гибкого и поддерживаемого приложения под операционную систему Android. В разделе описывается используемая в приложении архитектура Model-View-ViewModel (MVVM), позволяющая разделять логику и интерфейс, что улучшает читаемость кода, упрощает

тестирование и поддержку приложения, а также позволяет автоматически обновлять View при изменении данных в ViewModel при помощи Data Binding.

5. Обзор средств разработки для Android-приложений. В этом разделе дается обзор основных инструментов, используемых для разработки Android приложений, среди которых IntelliJ IDEA, Android Studio, Eclipse, Xamarin, React Native, Flutter. Также происходит выявление основных преимуществ Android Studio по сравнению с другими средами разработки.
6. Сравнительный анализ языков программирования для Android Studio. Здесь проводится сравнение языков программирования Java, C++ и Kotlin для разработки приложений на Android. Анализируются их основные особенности, преимущества и недостатки. Выявляется наиболее подходящий под поставленные цели язык программирования.
7. Обзор средств разработки для Backend REST приложений. В этом разделе рассматриваются основные инструменты и технологии для создания серверной части приложения, включая такие языки программирования, как Python, Kotlin, JavaScript и PHP. Так же объясняется почему для разработки был выбран язык Kotlin с фреймворком Ktor.
8. Обзор возможных баз данных для Backend REST приложений. Этот раздел обсуждает различные варианты баз данных, которые могут быть использованы в серверной части приложения, среди них SQL и NoSQL базы данных, их особенности, преимущества и недостатки. Также детально описывается документо-ориентированная NoSQL база данных MongoDB, выбранная в качестве базы данных для разрабатываемого приложения.
9. Размещение приложения на облачный хостинг: плюсы и минусы. В данном разделе рассматриваются варианты размещения серверной части приложения в облаке. Описываются основные преимущества использования облачного хостинга, а именно масштабируемость, быстрое размещение и управление, а также имеющиеся недостатки, в числе которых стоимость, зависимость от поставщика и сложность в использованни.
10. Jetpack Compose в мобильной разработке Android приложений. В этом разделе анализируется новый декларативный UI-фреймворк Jetpack Compose, его особенности, преимущества и применение в разработке приложений.

Вторая часть автореферата посвящена реализации приложения для тайм-

менеджмента, которая должна удовлетворять следующим требованиям:

- Регистрация и аутентификация пользователей. Пользователи смогут создавать учетные записи и входить в систему с помощью своих учетных данных.
- Создание локальных и глобальных задач. Пользователи смогут создавать как локальные задачи, которые видны только им самим, так и глобальные задачи, которые могут быть видны и доступны для отметки другими пользователями. Это позволит пользователям совместно работать над проектами и делиться задачами.
- Отметка задач другими пользователями. Пользователи смогут отмечать задачи других пользователей, чтобы указать на их выполнение или прогресс. Это поможет улучшить коммуникацию и сотрудничество между участниками.
- Редактирование всех видов задач. Пользователи смогут редактировать и обновлять информацию о своих задачах, включая описание, приоритет, дату выполнения и другие атрибуты. Это позволит им вносить изменения в свой план работы и адаптировать его под изменяющиеся условия.
- Поиск друзей в приложении: Пользователи смогут искать и добавлять друзей внутри приложения. Это создаст возможность для совместной работы над задачами, обмена идеями и поддержки друг друга в достижении целей.
- Настройка push уведомлений: Пользователи смогут настраивать уведомления в приложении для получения push-уведомлений о предстоящих задачах, напоминаниях и других событиях. Это поможет им быть в курсе и не пропускать важные сроки и события.

В качестве инструментов для разработки Android приложения были выбраны:

- Язык программирования Kotlin, обладающий чистым и синтаксисом, совместимостью с Java, поддерживающий лямбда-выражения и функциональное программирование, предлагающий корутины, а также являющийся официальным поддерживаемым языком для разработки Android приложений.
- Среда разработки Android Studio, являющаяся официальной IDE для Android от компании Google.
- XML и Compose для разработки пользовательского интерфейса.

— Для работы с сетевыми запросами: OkHttp и Retrofit.
— Локальная база данных: Room.

Инструменты для разработки REST API приложения на Ktor:

— Язык программирования: Kotlin.
— Фреймворк Ktor, поддерживающий асинхронность, корутины, дающий инструменты для создания модульной архитектуры, а также являющийся простым в использовании.
— Инструмент сборки Gradle.
— RESTful архитектура.
— База данных MongoDB, имеющая гибкую схему данных, расширенное индексирование, высокую доступность и масштабируемость.

Ниже представлены основные моменты по реализации приложения, которые будут подробно рассмотрены:

1. Разработка RestApi приложения на базе Ktor. В разделе происходит описание фреймворка Ktor, в основе которого лежат модули, представляющие собой набор функций и классов, настроенных для выполнения определенных задач. Модули Ktor могут быть настроены для обработки различных типов запросов, включая HTTP, WebSocket и другие, а также для взаимодействия с различными видами баз данных, такими как SQL, NoSQL. В разработанном приложении модуль Ktor настроен для работы с MongoDB и обработки различных типов запросов.
2. Размещение приложения на облачный хостинг. В данном разделе описываются необходимые действия для успешного развертывания приложения на платформе Render.com. Для этого было выполнено создание JAR-файла проекта с помощью инструмента сборки Gradle. Также был написан Dockerfile, который содержит все команды, необходимые для сборки образа Docker, проект был выложен на GitHub, а также было развернуто и запущено приложение на Render.com для его связи с репозиторием GitHub.
3. Retrofit и OkHttp для работы с сетью. В разделе описывается использование различных инструментов, а именно Retrofit, являющегося типобезопасным HTTP-клиентом для Android и Java, который упрощает процесс работы с API и позволяет разработчикам объявлять вызовы интерфейса с помощью аннотаций. В свою очередь OkHttp является HTTP-клиентом, который эффективно обрабатывает запросы и обеспечивает управление соединением,

используя HTTP/2 и постоянные соединения. С его помощью были также созданы пользовательские интерсепторы, которые используются для добавления заголовков, логирования и других задач.

4. Миграция на ViewBinding. В разделе происходит описание использования ViewBinding, являющегося функцией, которая генерирует простой класс привязки для каждого XML макета в модуле проекта. Эти классы содержат прямые ссылки на все представления, имеющие идентификатор, в соответствующих макетах, упрощая работу с представлениями в коде приложения. Для этого было произведено включение ViewBinding в файле build.gradle проекта, проведена замена использования findViewById на ViewBinding в коде Fragment.
5. Использование Room в качестве базы данных в Android. В данном разделе происходит описание процесса создания базы данных с использованием Room. Room представляет собой библиотеку, которая является абстракцией над SQLite и позволяет более удобно и безопасно работать с базой данных, а также включает в себя три основных компонента, которые были реализованы в приложении:
   — Database: класс, аннотированный @Database, являющийся контейнером с методами доступа к данным.
   — Entity: класс, аннотированный @Entity, представляющий собой таблицу в базе данных
   — Dao: интерфейс или абстрактный класс, аннотированный @Dao, использующийся для определения методов доступа к данным.
6. Использование внедрения зависимостей для улучшения кода. В разделе описывается внедрение зависимостей или DI, являющееся методом проектирования программного обеспечения, который облегчает разработку масштабируемого и поддерживаемого кода. В разрабатываемом приложении для этого используется библиотека Hilt, для которой была проведена настройка в проекте, а именно были добавлены необходимые зависимости в файл конфигурации Gradle и проведено аннотирование основного класса приложения с помощью @HiltAndroidApp. Также были определены модули, которые предоставляют зависимости для различных частей приложения, внедрены зависимости в нужные компоненты приложения с помощью аннотации @AndroidEntryPoint, определены зависимости, кото-

рые должны быть внедрены с помощью аннотации @Inject. В результате использования Hilt для внедрения зависимостей, структура кода приложения стала более чистой и понятной.

7. Способы оптимизации android приложения. В разделе рассказывается об оптимизации приложений на Android и устранении утечек памяти. Далее приведены некоторые из ключевых используемых инструментов для оптимизации приложения и устранения утечек памяти. Среди них использование встроенных инструментов профилирования в Android Studio, управление жизненным циклом объектов в приложении, избавление от неэффективных макетов (компоновок), определяющих отображение виджетов и элементы пользовательского интерфейса на экране, избавление от внутренних и анонимных классов, неявно содержащих ссылку на внешний класс. Кроме того сборщики мусора также освобождают память, удаляя неиспользуемые объекты.
8. Разработка основных экранов в приложении. В данном разделе было проведено описание разработки экранов регистрации и авторизации, основного экрана, экрана просмотра заметки или задачи, экрана с настройками профиля.

   На экране регистрации и входа в систему при создании аккаунта пользователю необходимо указать такие данные, как имя учетной записи, логин и пароль, причем на всех полях ввода установлены подсказки. После успешной регистрации пользователю показывается всплывающая подсказка об успешной регистрации и происходит перенаправление на экран входа в свою учетную запись. На данном экране пользователю требуется ввести логин (адрес электронной почты или номер телефона) и пароль.

   Основной экран разделен на две основные части: "Глобальные задачи"и "Локальные задачи чтобы пользователи могли легко отличать глобальные задачи, которые синхронизируются с сервером и доступны на всех устройствах пользователя, от локальных задач, которые сохраняются и доступны только на конкретном устройстве.

   На экране просмотра заметки или задачи в зависимости от типа задачи (локальной или глобальной), который передается в качестве параметра навигации, настройка интерфейса экрана просмотра заметок меняется. В случае редактирования задачи, пользователь может менять ее содержа-

ние, статус выполнения, приоритет и любые другие параметры, которые были установлены при создании задачи. Это помогает поддерживать актуальность информации о задачах и обеспечивает эффективное управление задачами.

На экране "Настройки"пользователю предоставляются обширные возможности для взаимодействия с приложением, а именно возможность редактирования персональных данных, установления связи с другими пользователями приложения при помощи функций поиска и добавления друзей в "круг общения а также выхода из учетной записи.

Таким образом, была произведена разработка основных компонентов архитектуры приложения, базы данных и интерфейса.

Заключение.

Целью данной работы было создание функционального и удобного приложения, позволяющего пользователям эффективно управлять своим временем и задачами. В процессе реализации работы был успешно разработан тайм-менеджер для платформы Android, с использованием backend-части на основе фреймворка Ktor, полный код которого представлен в приложении.

В процессе работы были проведены исследования и анализ существующих тайм-менеджеров, что помогло определить основные требования и функциональности, которые должны быть реализованы в разрабатываемом приложении. Также были изучены архитектурные подходы и паттерны разработки мобильных приложений, что позволило выбрать Clean Architecture для проектирования приложения.

В результате проведенной работы был получен тайм-менеджер, который соответствует поставленным требованиям. Разработанное приложение предоставляет удобные возможности планирования и управления временем, помогая пользователям быть более организованными и эффективными. В отличие от многих платных и импортных аналогов, это приложение является бесплатным и открытым для пользователей, предоставляя им доступ ко всем функциям без необходимости вложений.

Однако, в дальнейшем развитии проекта возможно внедрение дополнительных функций и улучшений. Например, можно добавить функциональность синхронизации с календарными приложениями, расширить интеграцию с другими сервисами и много другое.

В целом, данная дипломная работа позволила не только разработать полноценное приложение для управления временем на платформе Android, но и получить глубокие познания в разработке backend-части с использованием фреймворка Ktor. Результатом работы является функциональный и привлекательный тайм-менеджер, который может быть полезен для широкого круга пользователей.