**Введение**

BookReader — это мобильное приложение для чтения книг в формате .txt, разработанное с акцентом на простоту, производительность и удобство пользователя. Проект реализован студентами второго курса и служит отличным примером современного подхода к созданию Android-приложений.

**Цели и задачи проекта**

Основная цель — создать лёгкое, быстрое и надёжное приложение для чтения электронных книг, которое:

* Позволяет пользователю организовать собственную библиотеку,
* Сохраняет позицию последнего прочтённого фрагмента,
* Предлагает базовый, но комфортный пользовательский опыт,
* Поддерживает светлую и тёмную тему интерфейса.

**Архитектура и структура проекта**

В проекте реализована архитектура **MVVM (Model-View-ViewModel)**:

* **Model** — отвечает за работу с данными и бизнес-логику (например, хранение информации о книгах, настройках).
* **View** — реализует пользовательский интерфейс (экраны, компоненты, темы).
* **ViewModel** — управляет состоянием приложения и логикой отображения, связывает Model и View.

Такой подход обеспечивает модульность, удобство поддержки и расширяемость кода.

**Технологический стек**

* **Kotlin** — основной язык программирования.
* **Jetpack Compose** — современный декларативный UI-фреймворк, позволяющий быстро создавать адаптивные и красивые интерфейсы.
* **DataStore** — современное решение для хранения пользовательских настроек и состояния.
* **Navigation Compose** — библиотека для организации навигации между экранами.
* **Gradle Kotlin DSL** — система сборки и управления зависимостями.
* **JUnit, Espresso, Compose UI Test** — инструменты для модульного и UI-тестирования.

**Ключевые технические решения**

1. **MVVM-архитектура** — чёткое разделение ответственности между слоями.
2. **Jetpack Compose** — декларативный подход к созданию UI, поддержка темизации.
3. **DataStore** — надёжное хранение пользовательских данных (например, позиции чтения, выбранной темы).
4. **Navigation Compose** — простая и надёжная навигация между экранами.
5. **Репозиторий** — инкапсуляция работы с данными, что облегчает тестирование и развитие приложения.
6. **Тестирование** — обеспечение стабильности и надёжности работы приложения.

**Особенности проекта**

* **Минимализм** — только самые необходимые функции, ничего лишнего.
* **Высокая производительность** — быстрое открытие и чтение книг даже на слабых устройствах.
* **Сохранение позиции** — пользователь всегда продолжает чтение с того места, где остановился.
* **Поддержка светлой и тёмной темы** — комфортное чтение в любых условиях.
* **Открытый исходный код** — проект доступен для изучения и доработки.

**Перспективы развития**

Проект обладает гибкой архитектурой и современным стеком, что открывает широкие возможности для развития:

* Добавление поддержки новых форматов книг (EPUB, FB2, PDF),
* Синхронизация библиотеки и позиции чтения через облако,
* Расширенные настройки персонализации,
* Закладки, заметки, поиск по тексту,
* Интеграция с онлайн-библиотеками,
* Поддержка аудиокниг и TTS.

**Заключение**

BookReader — это пример современного, лёгкого и надёжного мобильного приложения, созданного с учётом актуальных стандартов разработки. Проект демонстрирует грамотное применение архитектурных паттернов, современных технологий и ориентирован на удобство конечного пользователя. Благодаря своей структуре и выбранным решениям, приложение легко поддерживать и развивать, что делает его отличной основой для дальнейших улучшений и новых функций.

### 

### 1. MainActivity.kt

**Где:** app/src/main/java/com/reina/ebookreader/MainActivity.kt

**Для чего:**

* Точка входа в приложение.
* Запускает Compose UI и инициализирует навигацию между экранами.

**2. BookViewModel.kt**

**Где:** app/src/main/java/com/reina/ebookreader/viewmodel/BookViewModel.kt

**Для чего:**

* Управляет состоянием списка книг, текущей книги, позиции чтения.

### Связывает данные (модели, репозиторий) и UI (экраны). 3. BookRepository.kt

**Где:** app/src/main/java/com/reina/ebookreader/repository/BookRepository.kt

**Для чего:**

* Инкапсулирует логику работы с книгами: загрузка, сохранение, обновление.
* Служит единым источником данных для ViewModel.

**4. Book.kt**

**Где:** app/src/main/java/com/reina/ebookreader/model/Book.kt

**Для чего:**

* Описывает модель данных книги (название, путь к файлу, позиция чтения и т.д.).

### 5. HomeScreen.kt, BookDetailScreen.kt, SettingsScreen.kt

**Где:** app/src/main/java/com/reina/ebookreader/screens/

**Для чего:**

* Реализуют основные экраны приложения:
* HomeScreen — библиотека книг.
* BookDetailScreen — чтение выбранной книги.
* SettingsScreen — настройки приложения.

### 6. ThemeViewModel.kt, ThemePreferences.kt, ui/theme/

**Где:**

* viewmodel/ThemeViewModel.kt
* data/ThemePreferences.kt
* ui/theme/

**Для чего:**

* Управляют темой приложения (светлая/тёмная).
* Сохраняют выбор пользователя и применяют оформление.

### 7. Navigation

**Где:** navigation/

**Для чего:**

* Описывает маршруты и переходы между экранами приложения.

### 8. DataStore

**Где:** Используется в ThemePreferences.kt и других классах для хранения настроек.

**Для чего:**

* Сохраняет пользовательские данные (например, тему, позицию чтения) между сессиями.

### 9. build.gradle.kts, libs.versions.toml

**Где:**

* app/build.gradle.kts
* gradle/libs.versions.toml

**Для чего:**

* Описывают зависимости, плагины и настройки сборки проекта.