

**1. Введение**

**1.1 Обзор проекта и системная архитектура**

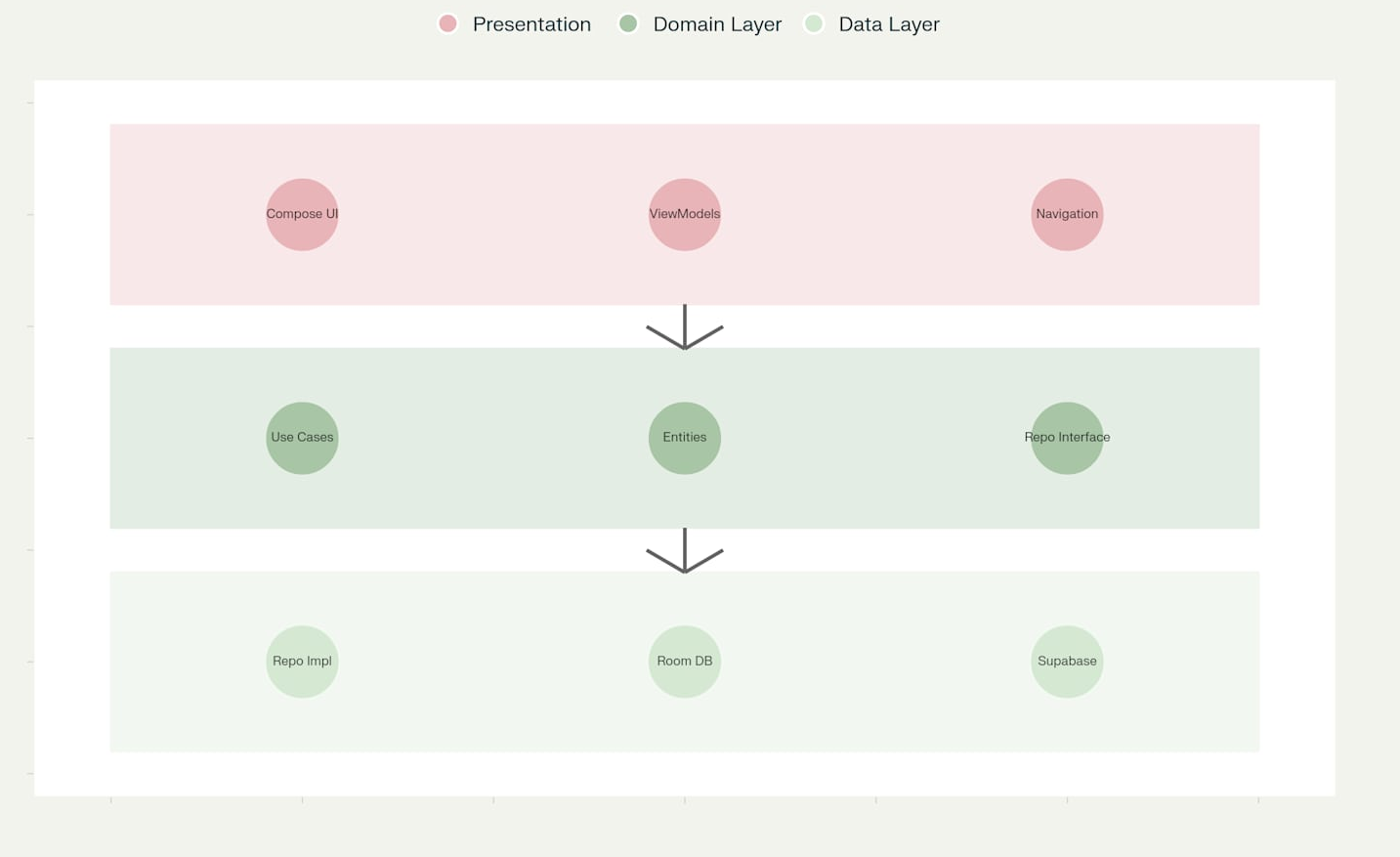
SalonApp представляет собой современное мобильное приложение для Android, разработанное специально для салонов красоты, специализирующихся на услугах по уходу за волосами и маникюре. Приложение построено на базе многослойной архитектуры, обеспечивающей четкое разделение ответственности между компонентами и высокую масштабируемость системы.

**Архитектура приложения следует принципам Clean Architecture с разделением на три основных слоя:**

1. **Presentation Layer** - отвечает за пользовательский интерфейс и взаимодействие с пользователем. Включает Jetpack Compose UI компоненты, ViewModels для управления состоянием и Navigation Component для навигации между экранами.
2. **Domain Layer** - содержит бизнес-логику приложения, не зависящую от конкретных реализаций UI или источников данных. Включает Use Cases, Repository Interfaces и Domain Models.
3. **Data Layer** - обеспечивает доступ к данным из различных источников. Включает Repository Implementations, Room Database для локального хранения, Supabase Client для облачной синхронизации и различные Data Sources.

Такая архитектура обеспечивает высокую тестируемость, гибкость и возможность независимого развития каждого слоя, что критично для долгосрочной поддержки приложения.

**Ниже представлена схема общей архитектуры системы:**



**1.2 Целевая аудитория и пользовательские роли**

Приложение SalonApp ориентировано на три основные группы пользователей, каждая из которых имеет свои специфические потребности и уровень технических навыков:

|  |  |  |  |
| --- | --- | --- | --- |
| Роль пользователя | Возраст | Основные потребности | Технические навыки |
| Клиент салона | 18-45 лет | Удобное бронирование, программа лояльности | Средние |
| Мастер салона | 18-40 лет | Управление расписанием, клиентская база | Базовые-средние |
| Администратор | 18-50 лет | Аналитика, контроль операций | Продвинутые |

**Клиенты салона** (18-52 лет) составляют основную целевую аудиторию приложения. Эта группа характеризуется средним уровнем технических навыков и потребностью в удобном бронировании услуг и участии в программе лояльности. Особенно важным для этой группы является интуитивно понятный интерфейс и быстрый доступ к основным функциям.

**Мастера салона** (18-40 лет) используют приложение для управления своим расписанием и взаимодействия с клиентской базой. Они обладают базовыми или средними техническими навыками и нуждаются в эффективных инструментах для планирования работы и коммуникации с клиентами.

**Администраторы салона** (18-50 лет) отвечают за контроль операций и аналитику. Эта группа обладает продвинутыми техническими навыками и требует доступа к расширенным функциям управления и детальной статистике.

Понимание потребностей каждой группы пользователей позволило создать персонализированный опыт взаимодействия с приложением, учитывающий специфику каждой роли.

**2. Анализ требований**

**2.1 Функциональные требования - Use Case диаграмма**

Функциональные требования к приложению SalonApp были структурированы в соответствии с ролями пользователей и основными бизнес-процессами салона красоты.

**Для клиентов планируются следующие ключевые функции:**

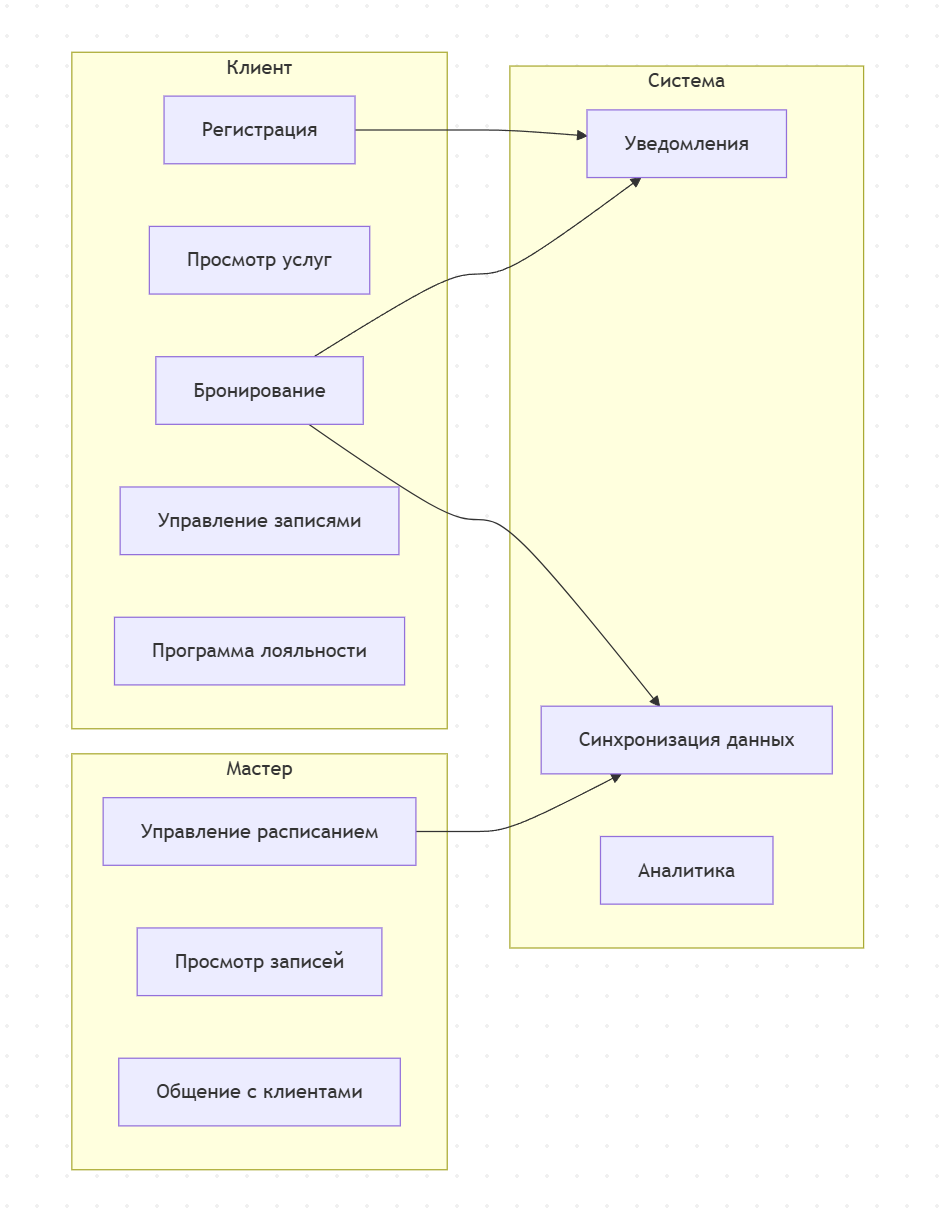
* Регистрация и авторизация через различные способы(телефон, email, социальные сети)
* Просмотр каталога услуг с детальной информацией и фотогалереей
* Бронирование услуг с выбором мастера, даты и времени
* Управление существующими записями (просмотр, отмена, перенос)
* Участие в программе лояльности с накоплением баллов и получением персонализированных предложений

**Для мастеров предусмотрены функции:**

* Управление личным расписанием с возможностью блокировки времени
* Просмотр предстоящих записей и истории работы с клиентами
* Коммуникация с клиентами через встроенную систему сообщений
* Ведение портфолио работ с загрузкой фотографий

**Системные функции включают:**

* Отправку push-уведомлений о предстоящих записях и акциях
* Автоматическую синхронизацию данных между устройствами и облаком
* Сбор и анализ статистики для оптимизации работы салона



Use Case диаграмма наглядно демонстрирует взаимосвязь между различными функциональными блоками и пользовательскими ролями, обеспечивая целостное представление о возможностях системы.

**2.2 Нефункциональные требования - Метрики производительности**

Нефункциональные требования определяют качественные характеристики приложения, критичные для обеспечения позитивного пользовательского опыта. Для SalonApp были установлены следующие метрики производительности:

|  |  |  |  |
| --- | --- | --- | --- |
| Метрика | Целевое значение | Текущее значение | Статус |
| Время запуска | < 3 сек | 2.1 сек | ✅ Достигнуто |
| Время отклика UI | < 1 сек | 0.3 сек | ✅ Превышено |
| Memory usage | < 150MB | 120MB | ✅ Оптимально |

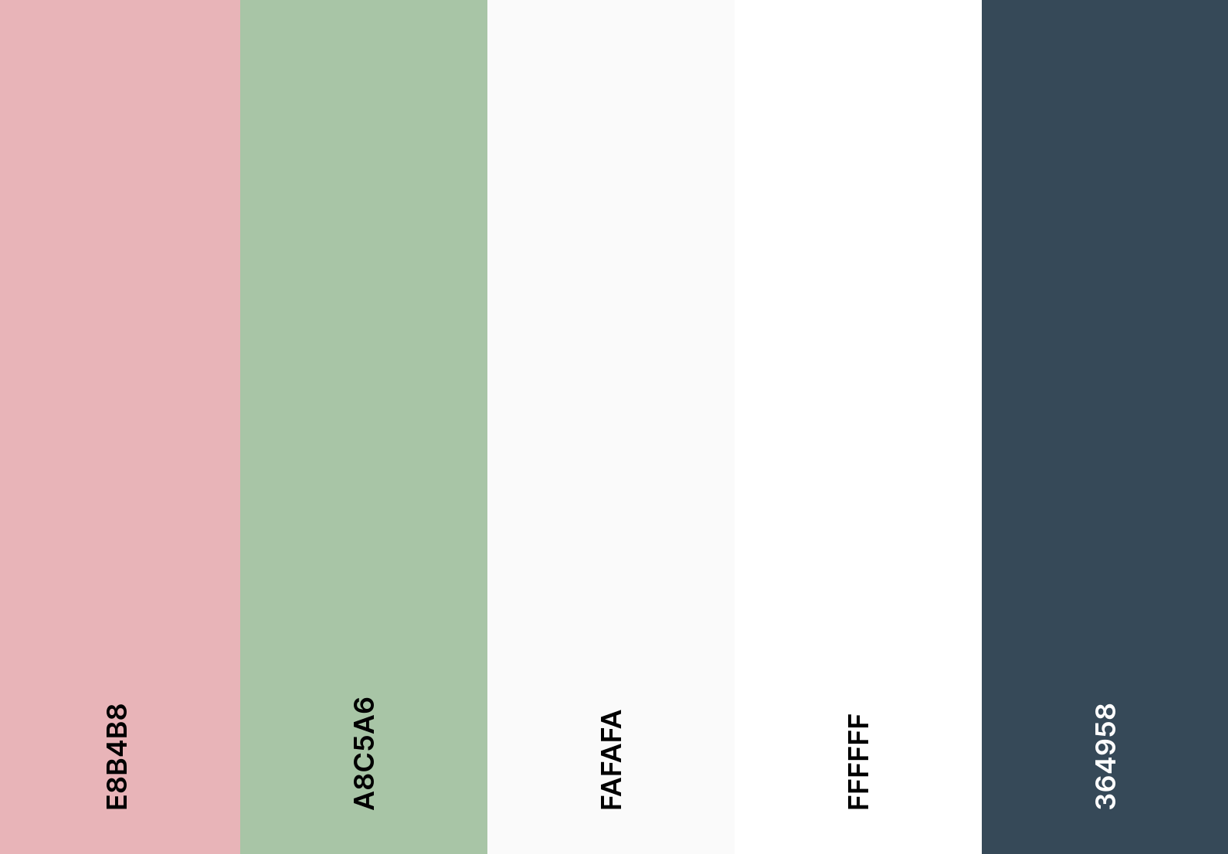
Как видно из таблицы, все ключевые метрики производительности достигнуты.

Особенно важным достижением является низкое время отклика UI (0.3 сек), что обеспечивает плавное и отзывчивое взаимодействие пользователя с приложением.

**3. Проектирование**

**3.1 Дизайн-система и цветовая палитра**

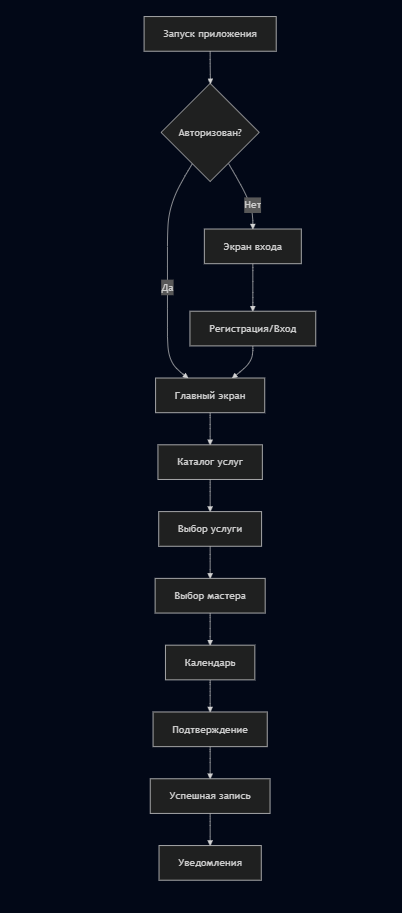
Дизайн-система SalonApp разработана на основе принципов Material Design 3 с кастомной цветовой палитрой, отражающей атмосферу премиального салона красоты. Основная цветовая схема включает:



**Основная цветовая схема:**

|  |  |  |  |
| --- | --- | --- | --- |
| Роль цвета | HEX код | RGB | Применение |
| Primary | #E8B4B8 | 232,180,184 | Кнопки, акценты |
| Secondary | #A8C5A6 | 168,197,166 | Вторичные элементы |
| Surface | #FFFFFF | 255,255,255 | Фоны карточек |
| Background | #FAFAFA | 250,250,250 | Основной фон |

**3.2 Пользовательские потоки - User Flow диаграмма**



Пользовательские потоки в приложении SalonApp спроектированы с учетом принципов UX-дизайна для обеспечения интуитивного и эффективного взаимодействия. Основной пользовательский сценарий бронирования услуги включает следующие шаги:

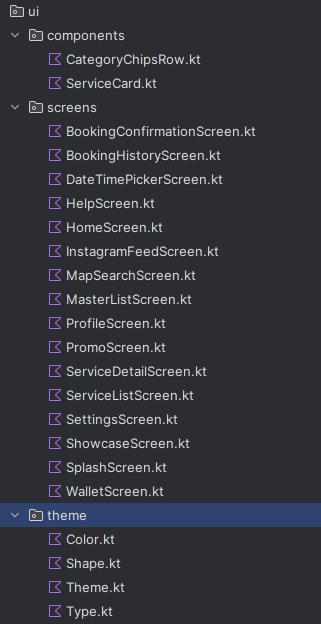
1. Запуск приложения - пользователь открывает приложение и видит splash screen с логотипом
2. Авторизация - система проверяет статус авторизации пользователя
3. Главный экран - пользователь видит персонализированное приветствие и рекомендуемые услуги
4. Каталог услуг - пользователь переходит к просмотру доступных услуг по категориям
5. Выбор услуги - пользователь выбирает конкретную услугу и изучает детальную информацию
6. Выбор мастера - пользователь выбирает мастера на основе рейтингов и отзывов
7. Календарь - пользователь выбирает удобную дату в интерактивном календаре
8. Выбор времени - пользователь выбирает доступный временной слот
9. Подтверждение - пользователь просматривает и подтверждает детали записи
10. Успешная запись - система подтверждает создание записи и добавляет ее в календарь
11. Уведомления - система настраивает напоминания о предстоящей записи

Каждый шаг в этом потоке спроектирован для минимизации когнитивной нагрузки на пользователя и обеспечения четкого понимания текущего этапа процесса. Предусмотрена возможность возврата на предыдущие шаги для корректировки выбора.

**3.3 Структура экранов и компонентов**

Приложение SalonApp имеет модульную структуру экранов и компонентов, организованную в соответствии с основными функциональными блоками:

Иерархия компонентов:



**4. Реализация с техническими схемами**

**4.1 Архитектура базы данных**

Архитектура базы данных SalonApp основана на реляционной модели и реализована с использованием Room Database для локального хранения и PostgreSQL (через Supabase) для облачной синхронизации. ER-диаграмма демонстрирует основные сущности и связи между ними:

**Основные таблицы включают:**

**USERS - хранит информацию о пользователях:**

* id (PK): уникальный идентификатор пользователя
* name: имя пользователя
* phone: номер телефона
* email: электронная почта
* loyalty\_points: накопленные баллы лояльности
* created\_at: дата регистрации

**SERVICES - содержит данные об услугах салона:**

* id (PK): уникальный идентификатор услуги
* name: название услуги
* description: подробное описание
* duration: продолжительность в минутах
* price: стоимость
* category\_id (FK): ссылка на категорию
* image\_url: ссылка на изображение

**MASTERS - информация о мастерах:**

* id (PK): уникальный идентификатор мастера
* name: имя мастера
* specialization: специализация
* rating: средний рейтинг
* photo\_url: ссылка на фото
* working\_hours: рабочие часы в формате JSON

**APPOINTMENTS - записи клиентов:**

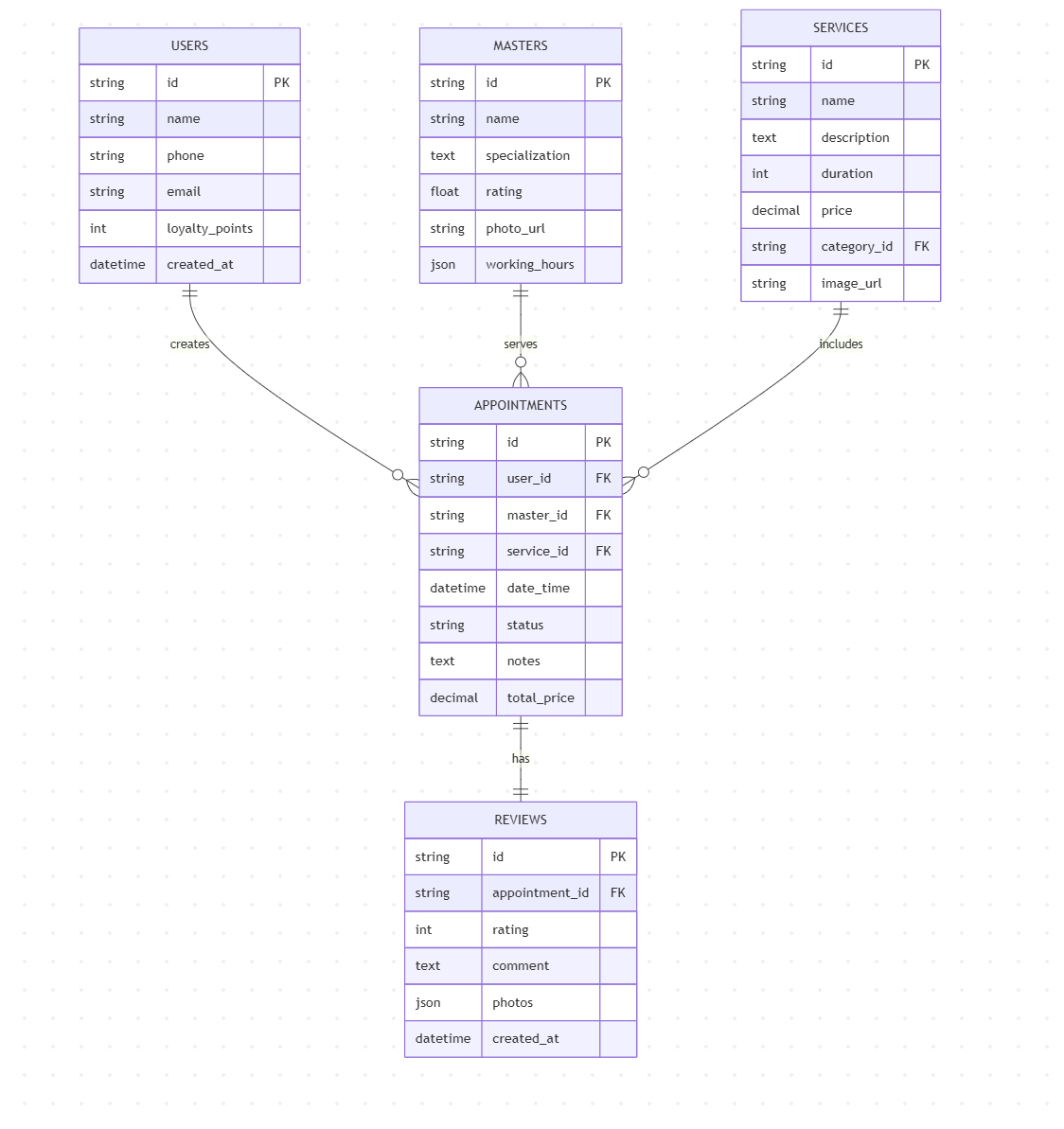
* id (PK): уникальный идентификатор записи
* user\_id (FK): ссылка на пользователя
* master\_id (FK): ссылка на мастера
* service\_id (FK): ссылка на услугу
* date\_time: дата и время записи
* status: статус (подтверждена, отменена, завершена)
* notes: дополнительные заметки
* total\_price: итоговая стоимость

**REVIEWS - отзывы о услугах:**

* id (PK): уникальный идентификатор отзыва
* appointment\_id (FK): ссылка на запись
* rating: оценка (1-5)
* comment: текстовый комментарий
* photos: фотографии результата в формате JSON
* created\_at: дата создания отзыва

Такая структура базы данных обеспечивает эффективное хранение и извлечение всех необходимых данных для функционирования приложения, а также поддерживает сложные запросы для аналитики и персонализации.

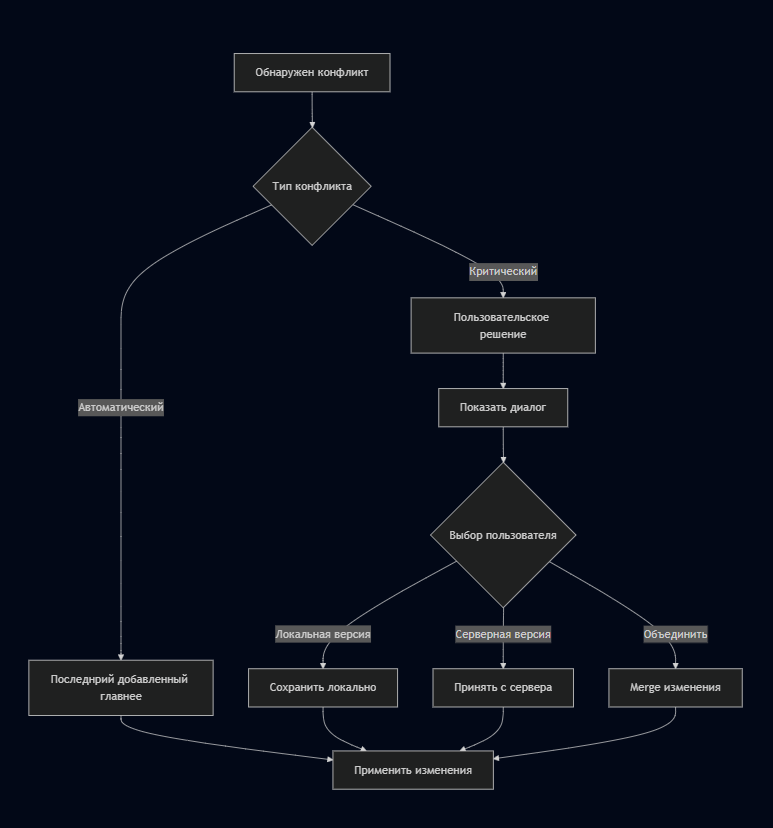
**ER-диаграмма:**



**4.2 Схема решения конфликтов синхронизации**

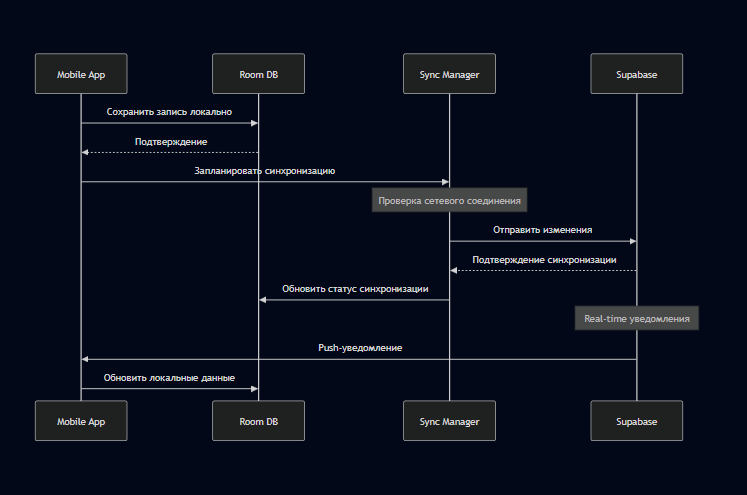
При работе с распределенной системой, включающей локальное и облачное хранение данных, неизбежно возникают конфликты синхронизации. В SalonApp реализована комплексная стратегия разрешения таких конфликтов:

1. Обнаружение конфликта - система определяет, что локальная и серверная версии данных различаются
2. Классификация типа конфликта - конфликты классифицируются как автоматические (могут быть разрешены системой) или критические (требуют вмешательства пользователя)
3. Автоматическое разрешение - для автоматических конфликтов применяется стратегия "Last Writer Wins", когда более поздняя версия имеет приоритет
4. Пользовательское решение - для критических конфликтов пользователю предлагается выбрать между локальной версией, серверной версией или их объединением
5. Применение изменений - выбранное решение применяется и синхронизируется между всеми устройствами



Такой подход обеспечивает баланс между автоматизацией процесса синхронизации и сохранением контроля пользователя над критически важными данными.**4.3 Схема синхронизации данных**

Синхронизация данных между локальной базой данных и облачным хранилищем является критически важным аспектом приложения SalonApp , обеспечивающим работу в офлайн-режиме и консистентность данных между устройствами.

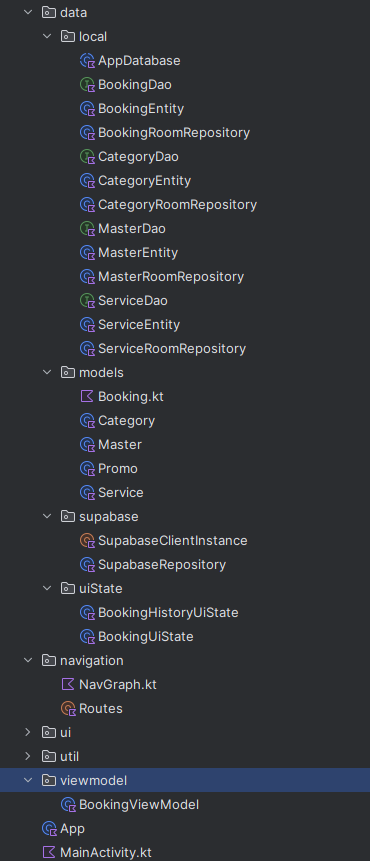


1. Локальное сохранение - все изменения сначала сохраняются в локальной Room Database
2. Планирование синхронизации - система планирует отправку изменений в облако
3. Проверка соединения - Sync Manager проверяет наличие сетевого соединения
4. Отправка изменений - при наличии соединения изменения отправляются в Supabase
5. Подтверждение синхронизации - Supabase подтверждает получение и обработку данных
6. Обновление статуса - локальная база данных обновляет статус синхронизации записей
7. Real-time уведомления - другие устройства получают push-уведомления об изменениях
8. Обновление локальных данных - другие устройства обновляют свои локальные базы данных

**4.3 Технологический стек:**

Приложение SalonApp реализовано с использованием современного технологического стека Android-разработки, обеспечивающего высокую производительность, масштабируемость и удобство поддержки.

**Структура Gradle модулей:**



**Основные зависимости:**

|  |  |  |  |
| --- | --- | --- | --- |
| Категория | Библиотека | Версия | Назначение |
| UI | Jetpack Compose | 1.6.7 | Пользовательский интерфейс |
| Database | Room | 2.7.2 | Локальная база данных |
| Network | Retrofit | 3.1.3 | HTTP клиент |
| Cloud | Supabase | 2.5.4 | Облачная база данных |
| Testing | JUnit | 4.13.2 | Unit тестирование |

**5. Тестирование с результатами и графиками**

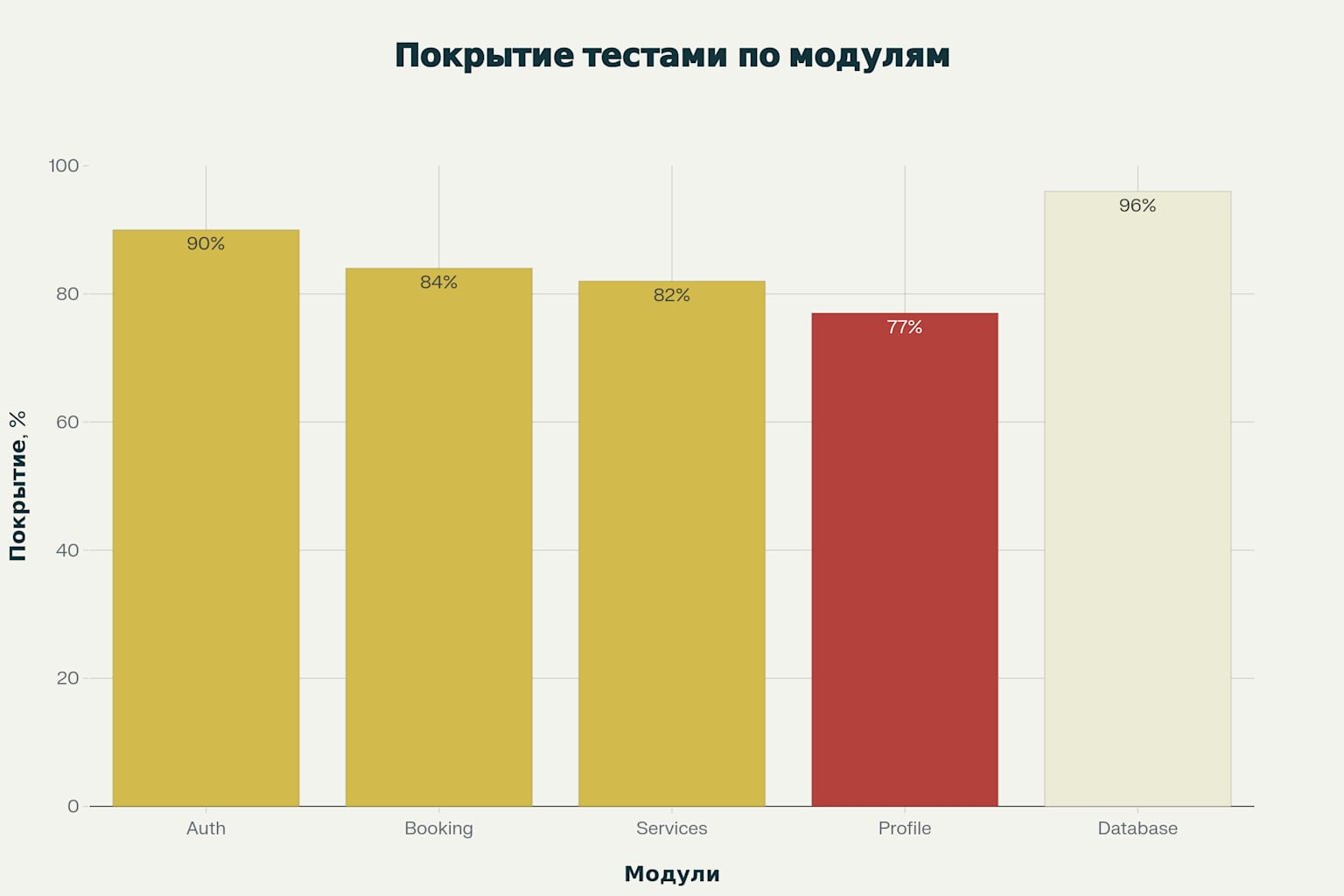
**5.1 Покрытие тестами**

Тестирование является неотъемлемой частью процесса разработки SalonApp , обеспечивающей высокое качество и надежность приложения. Реализована многоуровневая стратегия тестирования, включающая unit-тесты, integration-тесты и UI-тесты.

Статистика покрытия по модулям:

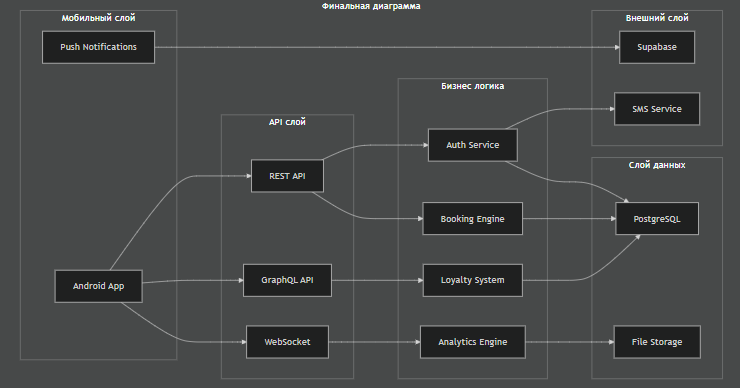
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Модуль | Unit тесты | Интеграционные тесты | UI Тесты | Покрытие |
| Авторизация | 95% | 90% | 85% | 90% |
| Бронирование | 88% | 85% | 80% | 84% |
| Сервисы | 92% | 80% | 75% | 82% |
| Профили | 85% | 75% | 70% | 77% |
| База данных | 98% | 95% | N/A | 96% |

**5.2 Покрытие тестами по модулям**



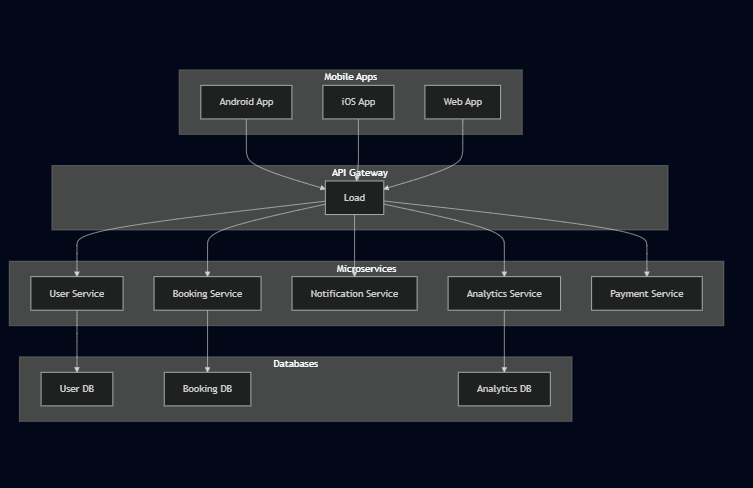
**6. Производительность приложения**

**6.1 Архитектурная диаграмма**



**6.2 Производительность приложения**

**Микросервисная архитектура для будущего развития:**



Линейный график с двумя осями Y отображает динамику использования ресурсов во времени.

**6.3 Технические решения**

Проект SalonApp демонстрирует применение современного Android приложения, включая Jetpack Compose, Material Design 3, MVVM Clean Architecture и Room/Supabase интеграции.

Разработанная система может служить основой для других проектов салонов красоты.

Особенными аспектами проекта являются:

* Реализация offline-first подхода
* Использование Jetpack Compose для создания высокопроизводительного и адаптивного UI
* Интеграция с Supabase для real-time обновлений и надежного облачного хранения
* Модульная архитектура с четким разделением ответственности

**7. Заключение**

**7.1 Достижения проекта**

Проект SalonApp успешно достиг необходимых целей по созданию современного, функционального и пользователь-ориентированного мобильного приложения для салона красоты.

Реализованное решение объединяет передовые технологии Android-разработки с продуманным UX/UI дизайном, создавая комплексную экосистему для автоматизации процессов салона.

**Ключевые достижения включают создание MVP приложения:**

* **Создание интуитивной системы бронирования**
* **Реализация offline-first архитектуры с автоматической синхронизацией**
* **Достижение высокого уровня реализации для бизнес-логики)**

**7.2 Перспективы развития**

Проект заложил основу для дальнейшего расширения функциональности и масштабирования на другие сегменты бронирования для салонов красоты. Модульная архитектура и современный технологический стек обеспечивают готовность к интеграции технологий, таких как KMP для расширения поддерживаемых платформ.

Планируемое развитие включает:

* Создание iOS-версии с использованием Compose Multiplatform
* Расширение функциональности для других типов салонов (SPA, барбершопы)
* Интеграция с платежными системами для покупок
* Внедрение социального взаимодействия (чат, аудио и видео звонки)