📝 План оставшихся задач (TODO)

Проект: Video Marketplace App
Дата создания: 23 декабря 2025
Статус: Готов к дальнейшей разработке
Приоритет: От высокого к низкому

🎯 Краткий обзор

Все критические технические проблемы решены. Проект полностью настроен и готов к разработке новых функций. Ниже представлен структурированный план задач на ближайшие 4 месяца разработки.

🚀 Фаза 0: Немедленные задачи (0-1 неделя)

⚡ Критичность: ВЫСОКАЯ

Задача 0.1: Тестирование собранного APK

Приоритет: 🔴 Критичный
Время: 2-3 дня
Ответственный: QA Engineer / Developer

Подзадачи:
- [ ] Установить APK на 3-5 различных Android устройств
- [ ] Android 5.0 (минимальная версия)
- [ ] Android 10+ (популярная версия)
- [ ] Android 14 (target версия)
- [ ] Протестировать регистрацию и вход (Email + Google)
- [ ] Проверить навигацию по всем экранам
- [ ] Протестировать видео плеер
- [ ] Проверить работу Firebase Authentication
- [ ] Проверить синхронизацию данных с Firestore
- [ ] Найти и задокументировать все баги
- [ ] Создать список исправлений

Результат: Список багов и готовность к исправлениям

Документация: См. `TESTING_INSTRUCTIONS.md`

Задача 0.2: Исправление найденных багов

Приоритет: 🔴 Критичный
Время: 2-4 дня
Зависит от: Задача 0.1

Ожидаемые проблемы:
- [ ] UI глюки на разных размерах экранов
- [ ] Проблемы с подключением к Firebase
- [ ] Ошибки в навигации
- [ ] Memory leaks в видео плеере
- [ ] Проблемы с кэшированием изображений

Результат: Стабильная версия без критичных багов

📱 Фаза 1: MVP функционал (1-4 недели)

⚡ Критичность: ВЫСОКАЯ

Задача 1.1: Интеграция платежной системы 💰

Приоритет: 🔴 Критичный
Время: 1-2 недели
ROI: Высокий (основной источник дохода)

Варианты реализации:

Вариант A: Stripe (рекомендуется для международного рынка)

Преимущества:
- Поддержка 135+ валют
- Apple Pay / Google Pay
- Отличная документация
- Flutter SDK готов

Подзадачи:
- [ ] Зарегистрировать Stripe аккаунт
- [ ] Добавить зависимость `flutter_stripe: ^10.1.1`
- [ ] Настроить API ключи (publishable & secret)
- [ ] Создать UI для payment flow
- [ ] Реализовать экран ввода карты
- [ ] Настроить webhook для подтверждения платежей
- [ ] Добавить обработку ошибок
- [ ] Протестировать в test mode
- [ ] Перейти на production keys

Стоимость: 2.9% + $0.30 за транзакцию

Документация: https://stripe.com/docs/payments/accept-a-payment?platform=flutter

---

Вариант B: Click/Payme (для рынка Узбекистана)

Преимущества:
- Локальные платежные системы
- Низкие комиссии (1.5-2%)
- Поддержка местных карт

Подзадачи:
- [ ] Регистрация в Click/Payme
- [ ] Изучение API документации
- [ ] Создание интеграции (нет готового Flutter SDK)
- [ ] Реализация прямых HTTP запросов
- [ ] Тестирование в sandbox
- [ ] Production deployment

Сложность: Выше, чем Stripe (нужна custom интеграция)

Рекомендация: Начать со Stripe для быстрого MVP, потом добавить Click/Payme для локального рынка.

Структура Firestore для платежей:

```
payments/
  ├── paymentId
  │   ├── userId: string
  │   ├── videoId: string
  │   ├── amount: number
  │   ├── currency: string (USD/UZS)
  │   ├── status: string (pending/completed/failed)
  │   ├── paymentMethod: string (stripe/click/payme)
  │   ├── stripePaymentIntentId: string
  │   ├── createdAt: timestamp
  │   └── completedAt: timestamp
```

Задача 1.2: Система рейтингов и отзывов ⭐

Приоритет: 🟡 Средний
Время: 3-5 дней
ROI: Средний (улучшает trust и engagement)

Подзадачи:
- [ ] Создать Firestore коллекцию `reviews`
- [ ] Добавить модель `Review` в код
- [ ] Создать UI для оставления отзыва
- [ ] Rating stars (1-5)
- [ ] Текстовый комментарий
- [ ] Опционально: фото/видео
- [ ] Реализовать отображение отзывов на странице видео
- [ ] Добавить сортировку (по дате, по рейтингу)
- [ ] Реализовать “Helpful” кнопку
- [ ] Добавить модерацию (флаг inappropriate)
- [ ] Вычислять средний рейтинг видео
- [ ] Показывать рейтинг в списке видео

Структура данных:

```
reviews/
  ├── reviewId
  │   ├── videoId: string
  │   ├── userId: string
  │   ├── userName: string
  │   ├── userPhotoUrl: string
  │   ├── rating: number (1-5)
  │   ├── comment: string
  │   ├── helpful: number
  │   ├── inappropriate: boolean
  │   └── createdAt: timestamp

videos/ (добавить поля)
  ├── averageRating: number
  └── reviewCount: number
```

Security Rules:

```
match /reviews/{reviewId} {
  allow read: if true;
  allow create: if request.auth != null &&
                request.auth.uid == request.resource.data.userId;
  allow update: if request.auth != null &&
                request.auth.uid == resource.data.userId;
  allow delete: if request.auth != null &&
                request.auth.uid == resource.data.userId;
}
```

Задача 1.3: Push-уведомления 🔔

Приоритет: 🟡 Средний
Время: 3-4 дня
ROI: Средний (retention и engagement)

Подзадачи:
- [ ] Добавить `firebase_messaging: ^14.7.10`
- [ ] Настроить Firebase Cloud Messaging в консоли
- [ ] Получить server key
- [ ] Реализовать получение FCM token
- [ ] Сохранять токены в Firestore (коллекция `users`)
- [ ] Создать Cloud Function для отправки уведомлений
- [ ] Реализовать типы уведомлений:
- [ ] Новый заказ (для продавца)
- [ ] Успешная покупка (для покупателя)
- [ ] Новый отзыв на ваше видео
- [ ] Ответ на комментарий
- [ ] Новые видео от избранных авторов
- [ ] Добавить настройки уведомлений в профиле
- [ ] Обработка нажатий на уведомления (deep linking)
- [ ] Локальные уведомления для offline событий

Cloud Function пример (Node.js):

```
exports.sendNewOrderNotification = functions.firestore
  .document('orders/{orderId}')
  .onCreate(async (snap, context) => {
    const order = snap.data();
    const sellerToken = await getSellerFCMToken(order.sellerId);

    await admin.messaging().send({
      token: sellerToken,
      notification: {
        title: 'Новый заказ!',
        body: `Покупка видео "${order.videoTitle}" на $${order.price}`,
      },
      data: {
        orderId: context.params.orderId,
        type: 'new_order',
      },
    });
  });
```

Примечание: Требует настройки Firebase Cloud Functions (бесплатно до 2M вызовов/месяц)

Задача 1.4: Создание release keystore и подписание APK

Приоритет: 🟢 Низкий (но обязательный для публикации)
Время: 1 час

Подзадачи:
- [ ] Создать release keystore файл
`bash`
`  keytool -genkey -v -keystore video-marketplace-release.jks \`
`    -keyalg RSA -keysize 2048 -validity 10000 \`
`    -alias video-marketplace`
- [ ] Сохранить пароли в безопасном месте (password manager)
- [ ] Создать `android/key.properties` с release данными
- [ ] Добавить в `.gitignore`:
`key.properties`
`  *.jks`
`  *.keystore`
- [ ] Обновить `android/app/build.gradle` для release signing
- [ ] Собрать signed release APK
- [ ] Протестировать signed APK

⚠️ **КРИТИЧНО:** НИКОГДА не коммитить keystore файл или пароли в Git!

🎨 Фаза 2: Улучшение UX (4-8 недель)

⚡ Критичность: СРЕДНЯЯ

Задача 2.1: Система чатов 💬

Приоритет: 🟡 Средний
Время: 1-2 недели
ROI: Средний (улучшает коммуникацию покупатель-продавец)

Подзадачи:
- [ ] Добавить зависимости:
`yaml`
`  dependencies:`
`    dash_chat_2: ^0.0.21`
- [ ] Создать Firestore структуру для чатов
- [ ] Реализовать список чатов
- [ ] Создать UI экрана чата
- [ ] Добавить отправку текстовых сообщений
- [ ] Реализовать real-time обновление (Stream)
- [ ] Добавить отправку изображений
- [ ] Показывать “typing...” индикатор
- [ ] Подсчет непрочитанных сообщений
- [ ] Push-уведомления для новых сообщений
- [ ] Возможность блокировки пользователя

Структура данных:

```
chats/
  ├─ chatId
  │   ├─ participants: [userId1, userId2]
  │   ├─ participantNames: {userId: name}
  │   ├─ participantPhotos: {userId: photoUrl}
  │   ├─ lastMessage: string
  │   ├─ lastMessageTime: timestamp
  │   ├─ lastMessageSenderId: string
  │   └─ unreadCount: {userId: number}

messages/
  ├─ chatId (subcollection)
  │   └─ messageId
  │       ├─ senderId: string
  │       ├─ text: string
  │       ├─ imageUrl: string (optional)
  │       ├─ timestamp: timestamp
  │       ├─ read: boolean
  │       └─ type: string (text/image/system)
```

Security Rules:

```
match /chats/{chatId} {
  allow read, write: if request.auth != null &&
    request.auth.uid in resource.data.participants;

  match /messages/{messageId} {
    allow read: if request.auth != null &&
      request.auth.uid in get(/databases/$(database)/documents/chats/$(chat-
Id)).data.participants;
    allow create: if request.auth != null &&
      request.auth.uid == request.resource.data.senderId;
  }
}
```

Задача 2.2: Аналитика для продавцов 📊

Приоритет: 🟡 Средний
Время: 1 неделя
ROI: Средний (помогает продавцам оптимизировать продажи)

Подзадачи:
- [ ] Добавить `firebase_analytics: ^10.8.0`
- [ ] Создать экран “Seller Dashboard”
- [ ] Реализовать метрики:
- [ ] Общий доход
- [ ] Доход за период (день/неделя/месяц)
- [ ] Количество продаж
- [ ] Просмотры vs. покупки (conversion rate)
- [ ] Топ-5 самых продаваемых видео
- [ ] График продаж (line chart)
- [ ] Распределение по категориям (pie chart)
- [ ] География покупателей (если доступно)
- [ ] Добавить экспорт в CSV/PDF
- [ ] Визуализация данных (charts):

`yaml`
`  dependencies:`
`    fl_chart: ^0.68.0`

Дополнительные события аналитики:

```
// Трекинг событий
FirebaseAnalytics.instance.logEvent(
  name: 'video_view',
  parameters: {'video_id': videoId, 'category': category},
);

FirebaseAnalytics.instance.logEvent(
  name: 'video_purchase',
  parameters: {'video_id': videoId, 'price': price, 'seller_id': sellerId},
);
```

Задача 2.3: Рекомендательная система 🎯

Приоритет: 🟡 Средний
Время: 2-3 недели
ROI: Высокий (увеличивает продажи)

Подходы:

Уровень 1 (Простой): Популярность и категории

Время: 3-5 дней
- [ ] Рекомендации на основе категории текущего видео
- [ ] Топ продаваемых видео
- [ ] Недавно добавленные видео
- [ ] Видео от того же автора

Уровень 2 (Средний): История пользователя

Время: 1 неделя
- [ ] Трекинг просмотров пользователя
- [ ] Создание коллекции `user_interactions`
- [ ] Рекомендации на основе истории
- [ ] “Похожие видео” на основе просмотренных категорий
- [ ] “Вам может понравиться” на главной

Уровень 3 (Продвинутый): Machine Learning

Время: 2-3 недели
- [ ] Использование Abacus.AI для ML рекомендаций
- [ ] Collaborative filtering
- [ ] Content-based filtering
- [ ] Персонализированная лента

Пример использования Abacus.AI:

```dart
import 'package:abacusai/abacusai.dart';

final client = AbacusAIClient(apiKey: 'YOUR_API_KEY');

// Получить рекомендации для пользователя
final recommendations = await client.getRecommendations(
  userId: currentUser.id,
  limit: 10,
  context: {
    'current_category': currentCategory,
    'price_range': userPriceRange,
  },
);
```

Рекомендация: Начать с Уровня 1, потом постепенно добавлять Уровень 2 и 3.

Задача 2.4: Улучшение поиска 🔍

Приоритет: 🟢 Низкий
Время: 3-5 дней

Подзадачи:
- [ ] Добавить фильтры:
- [ ] По цене (min-max slider)
- [ ] По категории (checkbox list)
- [ ] По рейтингу (⭐⭐⭐⭐⭐)
- [ ] По дате добавления
- [ ] По популярности (views/sales)
- [ ] Реализовать сортировку:
- [ ] По релевантности
- [ ] По цене (↑↓)
- [ ] По дате (новые/старые)
- [ ] По рейтингу
- [ ] По популярности
- [ ] Добавить автодополнение в поисковой строке
- [ ] Сохранение истории поиска
- [ ] “Недавние поиски”
- [ ] Suggestions на основе популярных запросов

Опциональные улучшения:
- [ ] Algolia для full-text search (платно, но мощно)
- [ ] Elasticsearch (требует backend)

🌟 Фаза 3: Расширенный функционал (8-12 недель)

⚡ Критичность: НИЗКАЯ

Задача 3.1: Админ панель 🛠️

Приоритет: 🟢 Низкий
Время: 2-3 недели
ROI: Низкий (но важен для управления платформой)

Технологии:
- Flutter Web для админки
- Firebase Admin SDK
- Cloud Functions для backend логики

Функционал:
- [ ] Dashboard с общей статистикой
- [ ] Управление пользователями:
- [ ] Список всех пользователей
- [ ] Просмотр профилей
- [ ] Блокировка/разблокировка
- [ ] Удаление аккаунтов
- [ ] Модерация контента:
- [ ] Список всех видео
- [ ] Одобрение/отклонение новых видео
- [ ] Удаление нарушающего контент
- [ ] Просмотр отчетов (reports)
- [ ] Управление категориями:
- [ ] Добавление новых категорий

- [ ] Редактирование существующих
- [ ] Удаление категорий
- [ ] Финансовая панель:
- [ ] Общий доход платформы
- [ ] Комиссии
- [ ] Выплаты продавцам
- [ ] Рефанды
- [ ] Обработка жалоб и споров
- [ ] Аналитика платформы
- [ ] Управление промо-кодами

Структура Firestore:

```
admins/
  ├── adminId
  │   ├── email: string
  │   ├── role: string (super_admin/moderator/support)
  │   └── permissions: array

reports/
  ├── reportId
  │   ├── reporterId: string
  │   ├── targetType: string (video/user/review)
  │   ├── targetId: string
  │   ├── reason: string
  │   ├── description: string
  │   ├── status: string (pending/reviewed/resolved)
  │   └── createdAt: timestamp
```

Задача 3.2: Многоязычность 🌍

Приоритет: 🟢 Низкий
Время: 1 неделя

Поддерживаемые языки:
- 🇺🇸 Английский (en)
- 🇷🇺 Русский (ru)
- 🇺🇿 Узбекский (uz)

Подзадачи:
- [ ] Добавить зависимости:

```yaml
  dependencies:
    flutter_localizations:
      sdk: flutter
    easy_localization: ^3.0.5
```

- [ ] Создать структуру локализации:

```
assets/
    └── translations/
        ├── en.json
        ├── ru.json
        └── uz.json
```

- [ ] Перевести все строки UI
- [ ] Добавить выбор языка в настройках
- [ ] Протестировать на всех экранах
- [ ] RTL support (если планируется арабский)

Примеры файлов перевода:

```
// en.json
{
  "home": "Home",
  "search": "Search",
  "profile": "Profile",
  "sign_in": "Sign In",
  "sign_up": "Sign Up"
}

// ru.json
{
  "home": "Главная",
  "search": "Поиск",
  "profile": "Профиль",
  "sign_in": "Войти",
  "sign_up": "Регистрация"
}
```

Задача 3.3: Социальные функции 👥

Приоритет: 🟢 Низкий
Время: 1-2 недели

Подзадачи:
- [ ] Подписки на авторов:
- [ ] Кнопка “Follow/Unfollow”
- [ ] Список подписчиков
- [ ] Список подписок
- [ ] Уведомления о новых видео от подписок
- [ ] Лента активности:
- [ ] Новые видео от подписок
- [ ] Популярные видео дня/недели
- [ ] Активность друзей (если есть)
- [ ] Избранное:
- [ ] Добавление в избранное (❤️)
- [ ] Список избранных видео
- [ ] Синхронизация между устройствами
- [ ] Плейлисты:
- [ ] Создание плейлистов
- [ ] Добавление видео в плейлисты
- [ ] Публичные/приватные плейлисты
- [ ] Поделиться плейлистом
- [ ] Поделиться в соцсетях:
- [ ] Facebook
- [ ] Twitter

- [ ] Telegram
- [ ] WhatsApp
- [ ] Instagram (через deep link)

Зависимости:

```yaml
dependencies:
  share_plus: ^10.1.3 # уже установлен
```

Структура данных:

```
follows/
  ├── followId
  │   ├── followerId: string (кто подписался)
  │   ├── followingId: string (на кого подписались)
  │   └── createdAt: timestamp

favorites/
  ├── favoriteId
  │   ├── userId: string
  │   ├── videoId: string
  │   └── createdAt: timestamp

playlists/
  ├── playlistId
  │   ├── userId: string (владелец)
  │   ├── title: string
  │   ├── description: string
  │   ├── public: boolean
  │   ├── videoIds: array
  │   └── createdAt: timestamp
```

Задача 3.4: Веб-версия приложения 🌐

Приоритет: 🟢 Низкий
Время: 2-3 недели
ROI: Средний (расширяет аудиторию)

Преимущества:
- Доступность на всех платформах
- SEO оптимизация
- Нет необходимости скачивать приложение
- Быстрый onboarding

Подзадачи:
- [ ] Настроить Flutter Web build:
`bash`
`  flutter build web --release`
- [ ] Адаптировать UI для веба:
- [ ] Responsive design
- [ ] Mouse hover states
- [ ] Keyboard navigation
- [ ] Desktop-friendly layout

- [ ] Настроить Firebase Hosting:
`bash`
`firebase init hosting`
`firebase deploy --only hosting`
- [ ] Оптимизация производительности:
- [ ] Lazy loading
- [ ] Code splitting
- [ ] Image optimization
- [ ] Caching strategies
- [ ] SEO оптимизация:
- [ ] Meta tags
- [ ] Open Graph tags
- [ ] sitemap.xml
- [ ] robots.txt
- [ ] Покупка домена и настройка SSL
- [ ] Google Analytics для веб-трафика

Оценка времени: 2-3 недели для полноценной веб-версии

Примечание: Flutter Web имеет ограничения с некоторыми плагинами. Может потребоваться использование web-specific пакетов.

🚀 Фаза 4: Масштабирование и оптимизация (12-16 недель)

⚡ Критичность: ОЧЕНЬ НИЗКАЯ

Задача 4.1: iOS версия 🍎

Приоритет: 🟢 Низкий
Время: 2-4 недели

Требования:
- Mac с Xcode
- Apple Developer account ($99/год)
- Тестовые iOS устройства

Подзадачи:
- [ ] Настроить Xcode проект
- [ ] Конфигурировать Firebase для iOS
- [ ] Адаптировать специфичные для iOS элементы
- [ ] Протестировать на iPhone/iPad
- [ ] App Store submission
- [ ] Review и публикация

Задача 4.2: Оптимизация производительности ⚡

Приоритет: 🟢 Низкий
Время: Ongoing (непрерывно)

Подзадачи:
- [ ] Profiling и нахождение bottleneck’ов
- [ ] Оптимизация загрузки изображений
- [ ] Lazy loading для списков видео
- [ ] Кэширование данных
- [ ] Минимизация Firestore reads
- [ ] Оптимизация видео доставки (CDN, adaptive streaming)
- [ ] Уменьшение размера APK
- [ ] Memory leak detection и исправление

Инструменты:
- Flutter DevTools
- Firebase Performance Monitoring
- Android Profiler

Задача 4.3: Маркетинг и рост 📈

Приоритет: 🟡 Средний (зависит от бизнес-целей)
Время: Ongoing

Активности:
- [ ] ASO (App Store Optimization)
- [ ] Социальные сети (Instagram, TikTok, YouTube)
- [ ] Content marketing (блог, гайды)
- [ ] Influencer partnerships
- [ ] Referral program
- [ ] Email marketing
- [ ] Paid ads (Google Ads, Facebook Ads)
- [ ] PR и media coverage

⏰ Временная шкала (Timeline)

Неделя 1-2: Стабилизация

- ✅ Тестирование APK
- ✅ Исправление багов
- 📋 Release keystore
- 📋 Первые реальные пользователи (beta)

Неделя 3-4: Monetization

- 💰 Интеграция Stripe/Click
- ⭐ Рейтинги и отзывы
- 🔔 Push-уведомления

Неделя 5-8: Улучшение UX

- 💬 Чат система
- 📊 Аналитика для продавцов
- 🎯 Рекомендательная система (простая)

- 🔍 Улучшенный поиск

Неделя 9-12: Расширенный функционал

- 🛠️ Админ панель
- 🌍 Многоязычность
- 👥 Социальные функции

Неделя 13-16+: Масштабирование

- 🌐 Веб-версия
- 🍎 iOS версия (если есть Mac)
- ⚡ Оптимизация
- 📈 Маркетинг

🎯 Приоритизация задач (Сортировка по важности)

🔴 MUST HAVE (Обязательно):

1. Тестирование APK и исправление багов
2. Платежная система (Stripe или Click/Payme)
3. Рейтинги и отзывы
4. Release keystore для production

🟡 SHOULD HAVE (Желательно):

1. Push-уведомления
2. Чат система
3. Аналитика для продавцов
4. Рекомендательная система (базовая)

🟢 NICE TO HAVE (Дополнительно):

1. Админ панель
2. Многоязычность
3. Социальные функции
4. Улучшенный поиск
5. Веб-версия
6. iOS версия

💰 Оценка ресурсов

Человеко-часы:

Фаза	Время	Человеко-часы
Фаза 0	1 неделя	40h
Фаза 1	3 недели	120h
Фаза 2	4 недели	160h
Фаза 3	4 недели	160h
Фаза 4	4+ недели	160h+
ИТОГО	**16+ недель**	**640h+**

Бюджет (приблизительно):

- Firebase (Blaze план): $0-50/месяц (в рамках free tier)
- Stripe/Click комиссии: 2-3% от транзакций
- Домен: $10-15/год
- Apple Developer: $99/год (если iOS)
- Google Play: $25 one-time
- Облачные сервисы (если нужны): $0-100/месяц

Total первый год: ~$200-500 + комиссии от транзакций

❓ Вопросы для обсуждения

Бизнес-модель:

1. Какую комиссию платформа будет брать с продаж? (рекомендуется 10-20%)
2. Будет ли subscription модель для продавцов?
3. Нужны ли featured listings (платное продвижение)?

Целевая аудитория:

1. Фокус на местный рынок (Узбекистан) или международный?
2. Какие категории видео в приоритете?
3. B2C или B2B модель?

Технические:

1. Нужна ли iOS версия в первый год?
2. Приоритет веб-версии или мобильных приложений?
3. Требуется ли видео транскодинг и адаптивный стриминг?

Маркетинг:

1. Бюджет на маркетинг?

2. Стратегия привлечения первых пользователей?
3. Partnerships с контент-криэйторами?

📚 Рекомендации по дальнейшей разработке

1. Agile подход

- Работать спринтами по 2 недели
- Регулярные демо и ретроспективы
- Приоритизация фич на основе user feedback

2. MVP первый

- Сначала выпустить минимальный функционал
- Получить feedback от реальных пользователей
- Итеративно улучшать

3. Качество > Количество

- Лучше 5 отлично работающих функций, чем 15 багованных
- Тщательное тестирование перед релизом
- Code review и best practices

4. Метрики успеха

Отслеживать KPI:
- DAU/MAU (Daily/Monthly Active Users)
- Retention rate (возвращаемость)
- Conversion rate (просмотры → покупки)
- Average order value
- Churn rate
- Customer Lifetime Value (CLV)

5. Безопасность

- Регулярное обновление зависимостей
- Security audit Firebase rules
- HTTPS везде
- Шифрование sensitive данных
- GDPR/CCPA compliance (если планируете работать в EU/US)

6. Масштабируемость

- Оптимизация Firestore queries
- Индексы для быстрых запросов
- Caching strategies
- CDN для медиа-контента
- Мониторинг performance

🔗 Полезные ресурсы

Документация:

- Flutter Docs (https://docs.flutter.dev/)
- Firebase Docs (https://firebase.google.com/docs)
- Stripe Flutter SDK (https://stripe.com/docs/payments/accept-a-payment?platform=flutter)

Обучение:

- Flutter & Firebase Course (https://fireship.io/)
- Udemy Flutter Courses (https://www.udemy.com/topic/flutter/)
- Flutter YouTube Channel (https://www.youtube.com/c/flutterdev)

Сообщество:

- Flutter Discord (https://discord.gg/flutter)
- r/FlutterDev (https://www.reddit.com/r/FlutterDev/)
- StackOverflow Flutter (https://stackoverflow.com/questions/tagged/flutter)

✅ Чек-лист перед началом новой задачи

Перед тем как начать любую задачу:

- [ ] Прочитать документацию
- [ ] Проверить существующие примеры/tutorials
- [ ] Оценить время реализации
- [ ] Проверить совместимость с текущим кодом
- [ ] Создать отдельную ветку в Git
- [ ] Написать unit/widget тесты
- [ ] Code review перед мержем
- [ ] Обновить документацию

🎯 Итоговые рекомендации

Для разработки в новом чате:

1. **Начните с Фазы 0** - протестируйте существующий APK
2. **Фокус на Фазе 1** - это критичные функции для MVP
3. **Не спешите с Фазами 3-4** - сначала получите user feedback
4. **Приоритет = Платежи** - без них нет бизнеса
5. **Документируйте все** - будущее “вы” скажет спасибо

Quick Wins (быстрые улучшения):

- Добавить splash screen с брендингом
- Улучшить UI/UX существующих экранов
- Добавить loading states и error handling
- Оптимизировать изображения

- Добавить onboarding tutorial для новых пользователей

Создано: 23 декабря 2025
Версия: 1.0 FINAL
Следующий review: После завершения Фазы 1

Удачи в разработке! 🚀