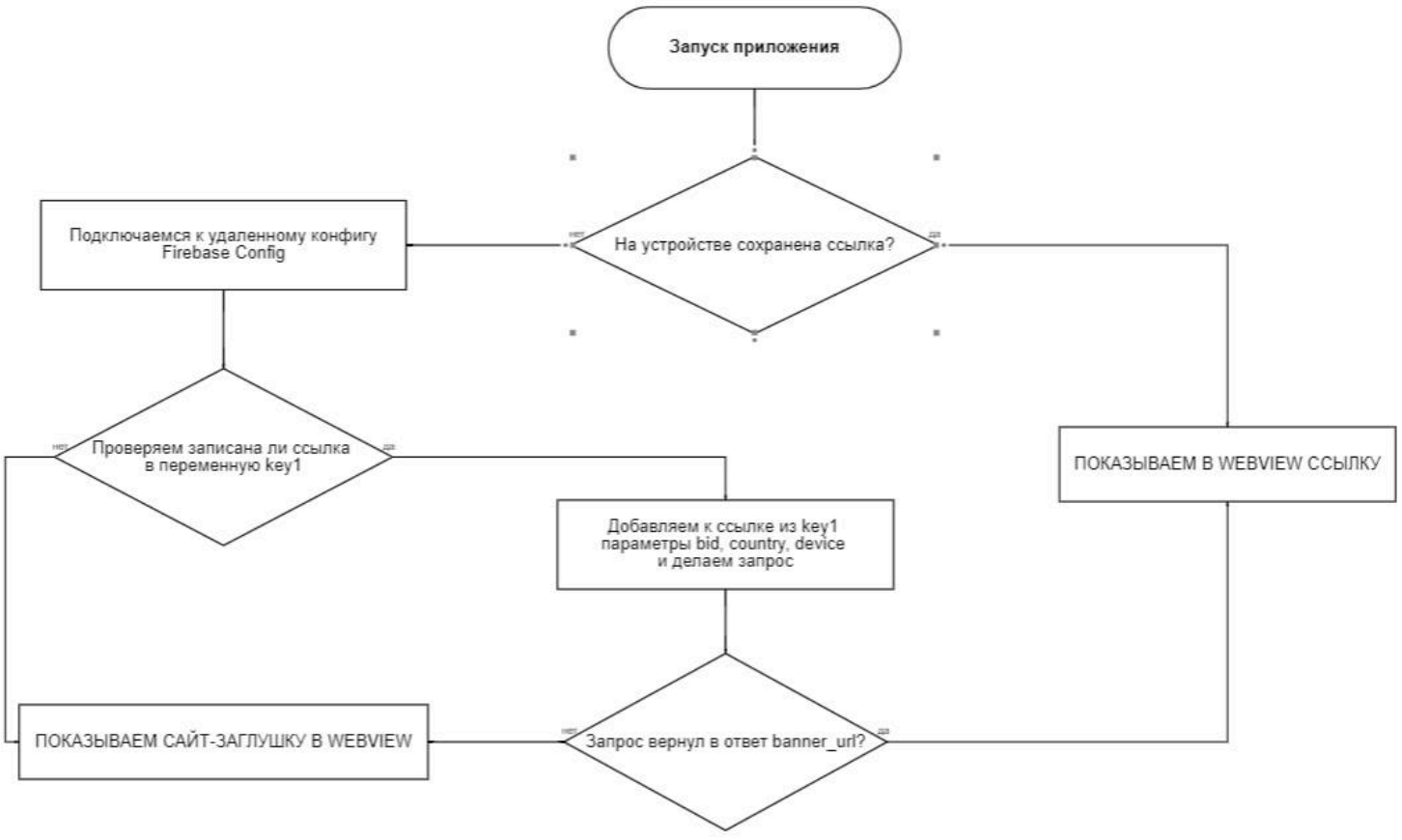
Техническое задание на разработку мобильного приложения “Название приложения”

Краткое описание работы приложения

1. Получаем конфигурацию от **Firebase Config**.
   1.1. Если **Firebase** возвращает ссылку → делаем запрос на **сервер**, получая в ответ **json**.
   1.2. Если в **json** есть ссылка → отображаем ее в **Android System Webview (Chrome tabs)**.
2. В случае отсутствия данных от **Firebase** или **сервера** → отображаем в **Android System Webview (Chrome tabs)** сайт-заглушку (*stub link*).
3. В случае отсутствия подключения к сети → отображаем **сообщение с ошибкой.**

🔔 *ВАЖНО: если мы показываем ссылку, а не заглушку,* ***ее нужно сохранить на устройстве пользователя и каждый раз открывать при старте приложения****. Подробности ниже в п.3*

Рисунок №1. Схема работы приложения



1. Получаем конфигурацию от Firebase Config

Необходимо подключить firebase
https://firebase.google.com/docs/android/setup

Далее подключаем firebase remote config
https://firebase.google.com/docs/remote-config/use-config-android

Ссылка будет в строковой переменной **key1**

🔔 *ВАЖНО: аналитику от Firebase не подключаем к проекту*

2. Создаем запрос к серверу

Если Firebase возвращает ссылку, то делаем к ней GET-запрос, передавая следующие параметры ниже.

Параметры:
- device (модель устройства)**:** `Build.MANUFACTURER +" "+ android.os.Build.MODE`
- country (гео): берется одноуровневый запрос → страна из сим-карты.
- country2 (гео): берется одноуровневый запрос → страна из локали устройства (iso3 country code).
- operator (провайдер сим-карты)**:** `getOperatorName`
- bid (android id): берется двухуровневый запрос: 1 уровень → IMEI устройства и 2 уровень → ANDROID_ID, если уровень 1 не дал результат
`Secure.getString(getContext().getContentResolver(), Secure.ANDROID_ID)`

Пример запроса:
Firebase Config в **key1** вернул `https://test.com/api/v1/launch/test`, мы добавляем параметры:
https://test.com/api/v1/launch/test?device=SamsungE21&country=ru&country2=ru&operator=Megafon&bid=a9542c1f-4bb9

Пример правильного полученного ответа:
{"bid":"a9542c1f-4bb9","country":"RUS","country2":"RUS","operator":"Megafon","device":"SamsungE21","proxied_ip":"2001:470:1f1b:388:cb1a:528d:d6ce:1b2f"}

Ответ в запросе:
Вернется json объект, в котором мы проверяем строку banner_url. В случае, если в ней есть ссылка, отображаем ее в Android System Webview, иначе в WebView отображаем **сообщение с ошибкой**.

```
{“banner_url” = “https://ya.ru”}
```

3. Добавить в gradle обфускацию

Необходимо добавить в gradle режим обфускации:

```
buildTypes {
        release {
```

```
            debuggable false
            minifyEnabled true
            shrinkResources true
            proguardFiles
getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
```

4. Настраиваем Android System WebView (Chrome tabs)

Android System WebView (Chrome tabs) должен быть внедрен, правильно настроен и соответствовать следующим требованиям:

- ✅ Cookies должны сохраняться
- ✅ Должен быть включен JavaScript, Dom и т.д. (см.ниже)
- ✅ Должна быть возможность загружать файлы на сайт
- ✅ Аппаратная кнопка back должна переводить пользователя назад по истории
- ✅ При смене ориентации устройства, сайт не должен перезагружаться
- ✅ В манифесте нельзя указывать только горизонтальную ориентацию
- ✅ Открытая клавиатура: webview должен скроллиться и появляться “statusbar activity” (если стоит фуллскрин → чекать когда клавиатура открывается и отображать “statusbar”)

Необходимые настройки:

```
val cookieManager = CookieManager.getInstance()
cookieManager.setAcceptCookie(true)
val mWebSettings = this.settings
mWebSettings.javaScriptEnabled = true
mWebSettings.loadWithOverviewMode = true
mWebSettings.useWideViewPort = true
mWebSettings.domStorageEnabled = true
mWebSettings.databaseEnabled = true
mWebSettings.setSupportZoom(false)
mWebSettings.allowFileAccess = true
mWebSettings.allowContentAccess = true
mWebSettings.loadWithOverviewMode = true
mWebSettings.useWideViewPort = true
```

🔔 *ОЧЕНЬ ВАЖНО: при первой загрузке ссылки через Android System WebView (Chrome tabs), в методе* ***onPageFinished*** *необходимо* ***сохранить первую ссылку на устройстве пользователя*** *и при последующих запусках приложения → открывать только ее (не делая запрос к Firebase и серверу).*

5. Создаем условие проверки устройства на эмулятор

Следующий блок кода производит проверку того, что за устройство запустило приложение (эмулятор или нет)

Если устройство, запустившее приложение является эмулятором → метод вернет “true”, если обычный пользовательский смартфон → “false”.

В случае, если метод возвращает “true” → сохраняем в “SharedPreferences” ссылку на заглушку, и далее загружаем ее всегда на этом устройстве.
Если метод вернет “false” → продолжаем работу приложения.

🔔 *ВАЖНО: в начале метода есть проверка, на то является ли приложение debug версией. Если версия debug → метод возвращает “false”.*
Сделано это для того, чтобы разработчик мог с эмулятора нормально тестировать приложение, а уже в “release” версии приложения → этот код будет обнаруживать эмулятор.

```
// if this method returns true, save the stub at the shared preferences
    private fun checkIsEmu(): Boolean {
        if (BuildConfig.DEBUG) return false // when developer use this build on emulator

        val phoneModel = Build.MODEL
        val buildProduct = Build.PRODUCT
        val buildHardware = Build.HARDWARE

        var result = (Build.FINGERPRINT.startsWith("generic")
                || phoneModel.contains("google_sdk")
                || phoneModel.lowercase(Locale.getDefault()).contains("droid4x")
                || phoneModel.contains("Emulator")
                || phoneModel.contains("Android SDK built for x86")
                || Build.MANUFACTURER.contains("Genymotion")
                || buildHardware == "goldfish"
                || buildHardware == "vbox86"
                || buildProduct == "sdk"
                || buildProduct == "google_sdk"
                || buildProduct == "sdk_x86"
                || buildProduct == "vbox86p"
                || Build.BOARD.lowercase(Locale.getDefault()).contains("nox")
                || Build.BOOTLOADER.lowercase(Locale.getDefault()).contains("nox")
                || buildHardware.lowercase(Locale.getDefault()).contains("nox")
                || buildProduct.lowercase(Locale.getDefault()).contains("nox"))

        if (result) return true
        result = result or (Build.BRAND.startsWith("generic") && Build.DEVICE.startsWith("generic"))
        if (result) return true
        result = result or ("google_sdk" == buildProduct)
        return result
    }
```

6. Требования к front-end

— При обращении к Firebase, серверу или загрузке ссылки, мы показываем **progressbar** пользователю.
— Приложение работает в **FullScreen** (statusbar скрыт), но когда клавиатура открывается, нужно **statusbar** отображать. Делается через `getWindow().clearFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN;`

— Если не удалось подключиться к Firebase Config или отсутствует интернет соединение при запросе к серверу, отобразить **сообщение с ошибкой**.

🔔 *ВАЖНО: способы реализации Вы выбираете на свое усмотрение.*
*Главный критерий → **нужно чтобы код был написан с нуля, без копипаста**.*
Каждое приложение тестируется на соответствие требованиям.

7. Тестирование

Перед отправкой debug.apk заказчику → просьба проверить за собой результаты работ по следующим параметрам:

✅ Иконка приложения → отображается согласно предоставленному дизайну.

✅ Название приложения → соответствует ТЗ (файл readme).

✅ Загрузочный экран имеет портретную ориентацию + индикатор загрузки.

✅ Фильтр по country/device/bid работает корректно и подтягивают информацию в виде: {"bid":"a9542c1f-4bb9","country":"RUS","country2":"RUS","operator":"Megafon","device":"SamsungE21","proxied_ip":"2001:470:1f1b:388:cb1a:528d:d6ce:1b2f"}

✅ В продукте, открывшемся по по ссылке → работает регистрация/вход пользователя.

✅ При регистрации/оплате по ссылке с открытой клавиатурой → экран скролится + видно поле ввода.

✅ При входе по открывшейся ссылке → поворачивая экран webview страница не перезагружается.

✅ При перезаходе по ссылке → авторизация пользователя не слетает (заходим в профиль и сверяем ранее загруженные данные).

✅ При входе по ссылке → по умолчанию скрылись верхние и нижние бары (fullscreen)

✅ При входе по ссылке → кнопка назад возвращает на предыдущую страницу в рамках webview (chrome tabs).

8. Проверка .aab файла на ошибки перед отправкой

Перед отправкой Android App Bundle заказчику → файл не должен содержать ошибок при загрузке в “Play Market” по следующим параметрам:

🚫 **ОШИБКА:**	✅ **РЕШЕНИЕ:**
Вы загрузили APK или Android App Bundle, который был подписан в режиме отладки.	*Перед сборкой → очистить кэш в студии*

Вы должны подписать их в режиме выпуска	
Целевой уровень API вашего приложения → 30. *Чтобы обеспечить необходимую производительность и безопасность, целевой уровень API должен быть не ниже 31*	*Целевой уровень API приложения должен быть не ниже 31 уровня*

9. FAQ

---

1. ВОПРОС: Какой стек брать для разработки и какие SDK нужно подключить?
   ОТВЕТ: По стеку → ждём от вас предложение. По SDK → Изначально только “Firebase” (appsflyer, oneSignal, appmetrica → нужно подключить только после “обновы” приложения).
2. ВОПРОС: На чьем устройстве производим сборку и создание ключа?
   ОТВЕТ: На устройстве разработчика.

Также, после тестирования приложения и оплаты работ → нужно направить в адрес заказчика архив с исходниками проекта + подписанным ключом + паролем от ключа.

3. ВОПРОС: Оплата после прохождения тестирования или после прохода в маркет?
   ОТВЕТ: Оплатим после завершения тестирования и передачи aab файла.

4. ВОПРОС: Заглушку нужно сделать с функционалом или просто заглушку?
   ОТВЕТ: Да, со всем функционалом.

5. ВОПРОС: Заглушку через webview (chrome tabs) отображаем?
   ОТВЕТ: Да. Ее нужно открыть html-страницей (домен принадлежит нам).

6. ВОПРОС: Какую версию Android брать?
   ОТВЕТ: Нужна поддержка с Android 6.0 (API v.23) до Android 12.0 (API v.32)

7. ВОПРОС: Возможна ли предоплата за проект?
   ОТВЕТ: Нет.

8. ВОПРОС: Оплата за проект или за час работы?
   ОТВЕТ: У нас фиксированная оплата за проект (не за час).

9. ВОПРОС: Как оплатите работу?
   ОТВЕТ: Переводом на карту (кошелек) после тестирования .apk и .aab

10. ВОПРОС: Как оплачиваются последующие обновления?
    ОТВЕТ: Оплачиваются также → фиксированной стоимостью, исходя из объема работ.

11. ВОПРОС: Тулбар у вебвью?
    ОТВЕТ: Только “статус бар” и “нав бар” → иначе скролла контента над клавиатурой не получиться.

12. ВОПРОС: banner_url от firebase получаем, далее запрос к админке → полученный url не пытаемся открыть в webview (chrome tabs), еще один запрос делаем?
    ОТВЕТ: Из firebase получаем key1→ banner_url это ключ в полученном от админки json

13. ВОПРОС: При старте приложения будет подгрузка данных, запуск webview (chrome tabs) и прочее. В это время надо показать пользователю сплэш-скрин с картинкой, лого или тп. Какую картинку установить?
    ОТВЕТ: Просто белый экран с лоадером.

14. ВОПРОС: Как происходит бан? Если тестите сценарий с модератором, то какие флаги должны быть или что-то еще, чтобы прила определяла, что это модератор?
    ОТВЕТ: При бане с админки придет json с пустыми полями (banner_url → пустой).

15. ВОПРОС: Можно увидеть пример, как открывается “statusbar” в приложении?
    ОТВЕТ: см раздел №4 ТЗ.

16. ВОПРОС: Всегда открывается “statusbar” или только когда открыта клавиатура?
    ОТВЕТ: В ТЗ изначально было условие, чтобы приложение работало в “fullscreen”, это когда статус бар скрыт. А показываем его, когда клавиатура открыта, потому что без него система не может скроллить экран.

17. ВОПРОС: Если вдруг я взялся за разработку проекта, но не смог его выполнить или выполнил частично. Мне оплатят мои потраченные часы?
    ОТВЕТ: Такой момент обсуждается индивидуально на старте каждого проекта. Если на старте проекта мы это не обсудили → обсуждаем по факту выполненных работ.

Смагин Антон
Руководитель проектов “Montekrist Media”
Разработка мобильных приложений
Telegram: +7 (912) 609-66-08