# Общее описание

Приложение – сервис знакомств. Состоит из 3 частей: экран входа, экран регистрация (визард с полями анкеты пользователя), основной раздел (WebView с мобильной версией сайта знакомств) и сервиса по приему и отображению push-сообщений.

## Требования к приложению

* Поддержка Android 5.0+
* Все тексты должны быть на английском. Логин должен отображаться как Login и так далее. Все сообщения также должны отображаться на английском (goolgle translate в помощь).

Рисунок 1. Экраны приложения

Первые 2 экрана – фиктивные и необходимы лишь для обхода модераторов google. Реальных запросов с этих активити поступать не будет. Реальным пользователям необходимо отображать сразу 3 раздел с WebView. Дизайн-макеты первых двух экранов отсутствуют. Дизайн на усмотрение разработчика. Достаточно разместить стандартные инпуты.

# Экраны

## Экран входа

Активити с полями логин, пароль и 2 кнопками: войти и регистрация. При нажатии на кнопку “регистрация” открывается экран регистрации. При нажатии на кнопку “войти” происходит отображение сообщения “Неверный логин/пароль. Обратитесь в службу поддержки”. Данный экран является фиктивным, реальных запросов никуда не отправляется.

## Экран регистрации

Визард с полями анкеты пользователя. Состоит из стандартных полей: логин, пароль, пароль еще раз, email, дата рождения, пол, имя. Обязательно должна присутствовать возможность добавить фото профиля. Анкету можно дополнить такими полями, как: вес, рост, “о себе”, место работы, образование и так далее. Тут разработчик должен проявить немного фантазии и создать подобие реальной анкеты для сервиса знакомств. Группировка полей и дизайн – на усмотрение разработчика. Данные с данного визарда никуда не отправляются. При нажатии на кнопку “Готово ” пользователь возвращается на экран авторизации и ему отображается сообщение “регистрация завершена”.

**Обязательные требования для раздела**

* Минимум 4 экрана в визарде
* Возможность прикреплять фото

Рисунок 2. Визард регистрации

## Экран с WebView

Основной раздел, который отображает WebView с мобильной версией сайта.

**Требования к работе WebView**

* Обязательно в манифесте использовать параметр android:usesCleartextTraffic="true"
* должен поддерживать всю работу с JS
* работу с localstorage
* загрузку фото и снимков с камеры (например, поиск по фото в яндекс картинках)
* javaScriptEnabled={true}
* allowUniversalAccessFromFileURLs={true}
* allowFileAccess={true}
* domStorageEnabled={true}
* cacheEnabled={true}
* Должна поддерживаться кнопка назад в браузере
* Должен запоминаться последний загруженный URL
* <https://stackoverflow.com/questions/27490192/android-webview-display-the-last-seen-page>
* Должна быть поддержка редиректов
* <https://stackoverflow.com/questions/4066438/android-webview-how-to-handle-redirects-in-app-instead-of-opening-a-browser>
* WebView не должна терять куки
* WebView не должна перезагружать страницу при повороте экрана (поворот можно вообще залочить в вертикальной ориентации)

# Логика работы приложения

## Проверка пользователя и отображение экранов

Приложение при помощи специального запроса на сервер определяет реальный это пользователь или же модератор google. В зависимости от этого отображается либо первоначальный экран с формой авторизации (для модераторов), либо экран с webview (для реальных пользователей).

**При запуске приложения происходят следующие проверки:**

* Проверяется локальная дата устройства. Если она меньше даты X (Дата зашивается в приложение. Дата будет указана в сопроводительном письме к данному ТЗ. Параметр **DateX**), то отображается экран авторизации.
* Если же она больше даты X, то выполняется запрос на сервер, который определяет какой из экранов отобразить пользователю.

**Запрос проверки:**

URL запроса будет указан в сопроводительном письме к данному ТЗ. Параметр **RestURL.**

POST: **RestURL**

**Header:** Content-Type: application/json

Данные в формате json

{

    packageName: bundleId,

    emulator: isEmulator,

    debug: false,

    androidId: ANDROID\_ID,

    source: referrer.installReferrer,

    fromPush: fromPush,

    pushCount: pushCount,

}

bundleId - имя пакета

isEmulator - эмулятор или нет (true/false)

debug - всегда false для релизной версии. True для того, чтобы протестировать работу webview.

ANDROID\_ID - уникальный ID устройства

referrer.installReferrer - параметр из библиотеки https://developer.android.com/google/play/installreferrer/

fromPush - true/false - идет ли запрос после открытия пуш сообщения

pushCount – 0, если fromPush == false. Если fromPush == true, то какой этот пуш по счету для данного устройства (приходит в пуше в параметре pushCount)

**Формат ответа**

{

    result: true/false,

    url: "https://ya.ru",

    lastUrl: true/false

}

Если result == false, то это модератор и ему необходимо отобразить экран авторизации. Если жеresult == true, то это реальный пользователь. Ему необходимо отобразить экран с webview. Проверяем другой параметр:lastUrl. Если lastUrl == false, то отображаем сайт из параметра url. Если lastUrl == true, то проверяем есть ли в памяти webview - последний url. Если да, открываем webview с ним. Если нет, то открывает webview с пришедшим url.

**Пример кода:**

if (res.data.result) {

DefaultPreference.get('LAST\_URL').then(value => {

  let url = res.data.url;

  if (value != null && res.data.lastUrl) {

    url = value;

  }

  props.navigation.replace('WebView', {url: url})

});

* Если отсутствует доступ в интернет или же произошла ошибка выполнения запроса из предыдущего пункта, то отображается экран авторизации.

## Работ Push-сервиса

Приложение должно уметь регистрировать pushkey и обрабатывать входящие сообщения: Заголовок + текст.

### 3.2.1 Отправка Push Key

Необходимо реализовать запрос на сервер. Запрос выполняется в рамках метода onNewToken firebase сервиса.

GET: **FirebaseURL**?key=PUSH\_KEY&androidId=ANDROID\_ID

PUSH\_KEY - ключ для отправки push уведомлений (параметр метода onNewToken)

ANDROID\_ID - уникальный ID устройства

В ответ ничего не придет. Лишь вернется 200. Так исторически сложилось ☺

### 3.2.2 Обработка пушей

**Формат пуш сообщения:**

data: {"apps":"server","type":"push","title":"To you (1) a new message","message":"How are you!","pushCount":"1"}

При клике по пушу, открывается приложение и выполняется стандартная проверка из пункта 3.1.

## 3.3 Поддержка Яндекс AppMetric

Добавить поддержку метрики. Вставить в класс Application следующий код:

YandexMetricaConfig config = YandexMetricaConfig.newConfigBuilder("64815e57-65d7-4267-945f-b35d1e49f8e7").build();

 // Initializing the AppMetrica SDK.

 YandexMetrica.activate(getApplicationContext(), config);

 // Automatic tracking of user activity.

 YandexMetrica.enableActivityAutoTracking(this);

 YandexMetrica.setUserProfileID(Utils.getAndroidId(getContentResolver()));

Ключ, который выделен красным заменить на ключ из сопроводительного письма.

# Входные данные

В сопроводительном письме будут переданы следующие данные:

**FirebaseURL**- для пушей

**RestURL** - для rest запросов

**DateX** - Дата X

Google-services.json

Иконки приложения и пушей

Название приложение

Название пакета

# Тестирование и процесс выполнения задания

## 5.1 Процесс взаимодействия

Взаимодействие заказчика и исполнителя происходит на основании договора, заключенного на фриланс платформе через безопасную сделку.

Для тестирования заказчиком необходимо передать ему 2 APK:

1. Обычное приложение со всеми проверками, чтобы его выкинуло на экран заглушки (авторизация + регистрация)
2. APK с отключённой проверкой даты и параметром запроса к серверу debug:true, чтобы открылся экран с webview

Если все хорошо, то заказчик закрывает договор на платформе, исполнителю переводятся деньги. После исполнитель отправляет файлы из раздела 6.

## 5.2 Кейсы тестирования

1. Проверить что работает кнопка назад в браузере
2. Проверить, что можно прикрепить картинку в браузере (например, поиск по картинке в яндексе)
3. Поверить, что при закрытии приложения и повторном открытии запоминается последний url
4. Проверить корректность пушей
5. Проверить, что все параметры в запросе передаются корректные, нигде не осталось заглушек, мок-объектов итд
6. Пройтись по приложению и половить падения

**Убедительная просьба: тестировать приложения перед отправкой (хотя бы по этому списку)**

# Что необходимо от разработчика

* Исходный код
* Подписанный aab или apk файл