ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук
Департамент программной инженерии

СОГЛАСОВАНО Профессор департамента программной инженерии факультета компьютерных наук, доктор технических наук. _____ Д.В. Александров «___» _____ 2016 г.	УТВЕРЖДАЮ Академический руководитель образовательной программы «Программная инженерия» _____ В.В. Шилов «_ _» _____2016 г.

Мобильный социальный сервис с геопозиционированием на платформе Android

Техническое задание

ЛИСТ УТВЕРЖДЕНИЯ
RU.17701729.508890-01 ТЗ 01-1-ЛУ

Исполнитель
студент группы БПИ 142
_____/Берюхов А.С./
«___»_____2016 г.

2016

Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата
RU.1770172 9.50889 0-01 81 01-1-ЛУ				

УТВЕРЖДЕНО

RU.17701729.508890-01 81 01-1-ЛУ

Мобильный социальный сервис с геопозиционированием на платформе Android

Пояснительная записка
RU.17701729.508890-01 81 01-1

Листов 24

Инв. № подл.	*Подп. и дата*	*Взам. Инв. №*	*Инв. № дубл.*	*Подп. и дата*
RU.17701729.508890-01 81 01-1				

2016

СОДЕРЖАНИЕ

Номер изменения	Подпись ответственного за внесение изменения	Дата внесения изменения

Номер изменения	Подпись ответственного за внесение изменения	Дата внесения изменения

1. ВВЕДЕНИЕ

1.1. Наименование программы

Полное наименование программы – «Мобильный социальный сервис с геопозиционированием на платформе Android».

Краткое наименование программы – «Quid Pro Quo».

1.2. Документы, на основании которых ведется разработка

1) Приказ Национального исследовательского университета «Высшая школа экономики» № 6.18.1-02/1112-19 от 11.12.2015.

Номер изменения	Подпись ответственного за внесение изменения	Дата внесения изменения

2. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ

2.1. Функциональное назначение

Программа предназначена для добавления записей (постов) с указанием определенного местоположения пользователя (текущего – определеннного устройствой пользователя, или заданного по адресу), отображать для него такие записи других пользователей в пределах заданного радиуса поиска (т.е. записей, привязанных к местоположениям, находящимся на расстоянии, не большем заданному от определенного пользователем местоположения) с возможностью добавления и просмотра комментариев.

2.2. Эксплуатационное назначение

Программа может быть использована для размещения и поиска информации, полезной в данном местоположении, в том числе, относящейся к обмену товарами или услугами.

2.3. Краткая характеристика области применения

Решение выше обозначенной задачи имеет потенциальное применение в разных областях:

1. Размещение объявлений, предупреждений и другой информации, актуальной для людей, находящихся в конкретном месте.
2. Распространение срочной информации о происшествиях (например, пробках или авариях) в конкретный промежуток времени.
3. Общение внутри территориальной группы (например, жильцы многоквартирного дома, сотрудники одного предприятия или учащиеся одного учебного заведение), в том числе с целью обмена товарами и услугами.

3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

3.1. Постановка задачи на разработку программы

Разрабатываемое приложение должно обеспечивать выполнение перечисленных ниже функций:

1) Регистрация нового пользователя.

2) Авторизация пользователя.

3) Восстановление пароля по email.

4) Редактирование части информации о пользователе.

5) Определение местоположения, используя сервисы Google Play.

6) Определение местоположения по адресу.

7) Добавление новой записи.

8) Отображение списка запрашиваемых записей.

9) Просмотр подробной информации о конкретной записи.

10) Отображение места привязки записи на карте (используя сторонние картографические приложения).

11) Добавление и отображение комментариев к записям.

3.2. Описание алгоритма функционирования программы

Приложение состоит из клиентской части, которая взаимодействует с сервисом бэкэнда Parse посредством Parse SDK и API Яндекс Геокодер.

Parse SDK обеспечивает интерфейс для настройки, создания, изменения и удаления таблиц и записей в базе данных.

В основе сервиса Parse лежит MongoDB база данных.

Конфигурация базы данных сервиса Parse производится непосредственно из клиентского обращения при первом обращении к сервису.

3.2.1. Алгоритм представления времени добавления записи

Для отображения времени и даты добавления записи или комментария в более коротком виде используется особый алгоритм (исходный текст в классе DataTransform).

На вход принимаются две даты в формате GregorianCalendar (из пакета java.util): первая – дата добавления записи, втора – дата, относительно которой вычисляется упрощенный вид отображения первой.

Если обе даты совпадают до дня (это используется, если запись сделана сегодня), отображается только время в формате "HH:mm".

Если даты различаются на один день (это используется, если запись сделана вчера), отображается слово «yesterday» или «вчера» в зависимости от локализации.

Если даты совпадает с точностью до года, отображаются только день и месяц в формате "d MMM".

В противном случае (это используется, если запись сделана даже не в этом году), отображается дата в формате "d MMM Y".

С помощью SimpleDateFormat (из пакета java.text) формируется строка, которая возвращается алгоритмом.

3.2.2. Алгоритм отображения части квадратного изображения в виде круглого

Алгоритм используется для отображения аватаров пользователей в виде круга (исходный код во вложенном классе ImageHelper класса MainActivity).

На вход поступает изображение в формате Bitmap. После преобразований с помощью методов классов Canvas и Paint круглое изображение возвращается в этом же формате.

3.3. Описание и обоснование выбора метода организации входных и выходных данных

Входные данные вводятся пользователем с помощью клавиатуры, либо с фотокамеры устройства, либо с устройства определения местоположения.

Выходные данные отображаются на экране устройства.

Выбор метода организации входных и выходных данных обоснован требованиями технического задания.

3.4. Описание и обоснование выбора состава технических и программных средств

3.4.1. Описание выбора состава технических средств

Для запуска и работы приложения требуется мобильное устройство, со следующими свойствами:

1) Устройство с ОС Android 4.0 и выше, совместимое с Google Play[1].
2) Не менее 10 МБ свободного места на накопителе.
3) Доступ в интернет.
4) Клавиатура или поддержка сенсорного ввода.
5) Устройство определения геопозиции (опционально).

3.4.2. Обоснование выбора состава технических средств

Выбор состава технических средств обусловлен техническим заданием.

Минимальная версия ОС Android 4.0 обусловлена тем, что 94% устройств на платформе Android используют версию 4.0 и старше[3].

3.4.3. Описание выбора состава программных средств

Для запуска и работы программы требуется компьютер, со следующим предустановленным программным обеспечением:

1) операционная система Android 4.0 и новее;

2) Google Play Services 8.7.02.

3.4.4. Обоснование выбора состава программных средств

Выбор версии Android объяснен выше.

Выбор версии Google Play Services обусловлен официальными рекомендациями Google [4].

4. ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

4.1. Предполагаемая востребованность

Приложение может применяться для личного использования пользователями (см. п.2). При небольшом усовершенствовании и после набора сервисом аудитории возможно внедрение механизмов монетизации и сотрудничество с юридическими лицами, желающими выделить свои записи.

4.2. Экономические преимущества разработки по сравнению с аналогами

Выявлены следующие аналоги приложения:

- SwopShop
- FarPost
- Krack
- Зигзаг

Представленное приложение имеет ряд преимуществ:

1. Изменяемый радиус поиска.
2. Легкая регистрация.
3. Отсутствие комиссии при продаже/обмене товаров или услуг.
4. Разграничение записей по типу: дать/взять.

4.3. Бизнес модель

Для оценки потенциального применения приложения в бизнесе проведено построение бизнес модели по шаблону А. Остервальдера (рис. 1).

Ценностные предложения сервиса: новизна, новые функции, удобство, доступность (бесплатное скачивание) (см. п.4.2).

Ключевые виды деятельности: разработка приложения, поддержка (решение проблем, возникающих у пользователей, поддержка взаимодействия приложения с серверной частью, исправление обнаруженных ошибок), модерация – удаление

записей, содержащих неприемлемое содержание, применение санкций к пользователям, создающим такие записи.

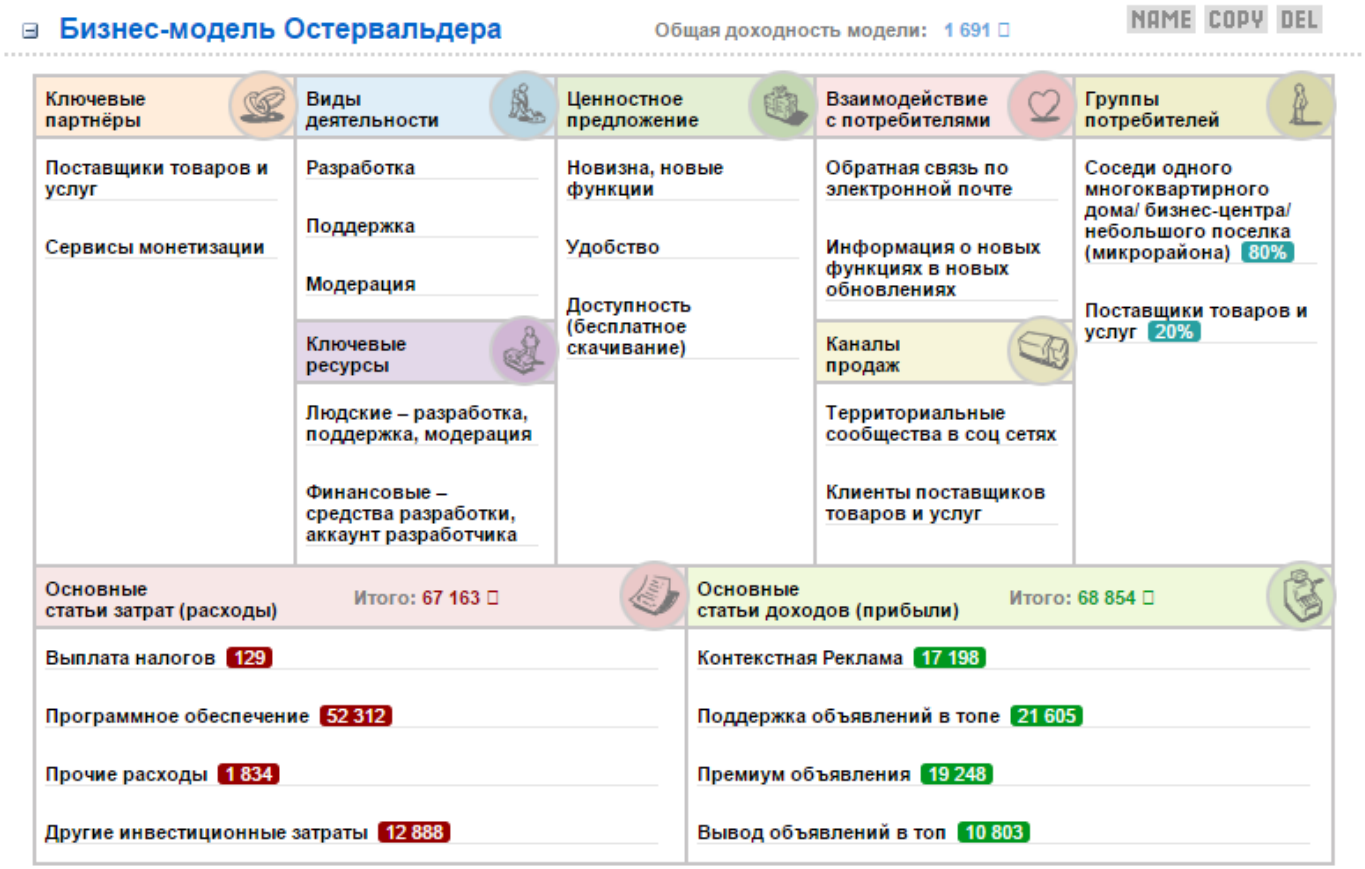

Рисунок 1 Бизнес модель Остервальдера.

Ключевые ресурсы: людские – для разработки, поддержки и модерации; финансовые – для закупки лицензий средств разработки, взнос за аккаунт разработчика в сервисах распространения приложений, оплата услуг серверных мощностей.

Ключевыми партнерами проекта могут стать поставщики товаров и услуг, размещающие рекламные записи в приложении на особых условиях; сервисы монетизации (сторонние решения для извлечения прибыли с рекламы).

Основные статьи затрат (расходов): разработка программного обеспечения, оплата серверных мощностей, аккаунт разработчика (для публикации), налоги.

Номер изменения	Подпись ответственного за внесение изменения	Дата внесения изменения

Взаимоотношения с клиентами: обратная связь по электронной почте, рассылка с информацией об обновлениях.

Каналы продаж: распространение через студенческие или территориальные сообщества в социальных сетях (через которые проходит лишний траффик предложений по купле/обмену товаров и услуг сейчас); клиенты поставщиков товаров и услуг.

Группы потребителей: соседи одного многоквартирного дома/ бизнес-центра/небольшого поселка (микрорайона), поставщики товаров и услуг.

Потоки поступления доходов: контекстная реклама, премиум объявления.

Номер изменения	Подпись ответственного за внесение изменения	Дата внесения изменения

5. ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ

1. Справка Google Play. Системные требования. [Электронный ресурс] / URL: https://support.google.com/googleplay/answer/2844198 (Дата обращения 23.03.2016, режим доступа: свободный).

2. Единая система программной документации – М.: ИПК Издательство стандартов, 2000.

3. История версий Android – Википедия. [Электронный ресурс] / URL: https://ru.wikipedia.org/wiki/История_версий_Android (Дата обращения 23.11.2015, режим доступа: свободный).

4. Making Your App Location-Aware. Android Developers. [Электронный ресурс] / URL: http://developer.android.com/intl/ru/training/location/index.html (Дата обращения 2.02.2016, режим доступа: свободный).

5. Toasts. Android Developers. [Электронный ресурс] / URL: http://developer.android.com/intl/ru/guide/topics/ui/notifiers/toasts.html (Дата обращения 20.04.2016, режим доступа: свободный).

6. Parse Tutorials [Электронный ресурс] / URL: https://parse.com/tutorials (Дата обращения 23.11.2015, режим доступа: свободный).

7. Parse Android API Reference. [Электронный ресурс] / URL: https://parse.com/docs/android/api/ (Дата обращения 23.11.2015, режим доступа: свободный).

8. Stack Overflow community. [Электронный ресурс] / URL: http://stackoverflow.com/ (Дата обращения 23.11.2015, режим доступа: свободный).

9. Moodle. Software design 2015-2016. Readings. [Электронный ресурс] / URL: http://moodle.cs.hse.ru/mod/folder/view.php?id=164 (Дата обращения 23.11.2015, режим доступа: свободный).

Номер изменения	Подпись ответственного за внесение изменения	Дата внесения изменения

10. Material design – Google design guidelines [Электронный ресурс] / URL: https://www.google.com/design/spec/material-design (Дата обращения: 20.01.2016, режим доступа: свободный, на англ. языке).

11. FloatingActionButton – Library on GitHub [Электронный ресурс] / URL: https://github.com/makovkastar/FloatingActionButton (Дата обращения: 12.02.2016, режим доступа: свободный (The MIT License (MIT)), на англ. языке).

12. Иконография. Проектирование Android приложений. [Электронный ресурс] / URL: http://developer-android.unlimited-translate.org/design/style/iconography.html (Дата обращения 2.12.2015, режим доступа: свободный).

13. Material Palette - Material Design Color Palette Generator. [Электронный ресурс] / URL: http://www.materialpalette.com/ (Дата обращения 22.01.2016, режим доступа: свободный).

14. Создание бизнес-плана с нуля. [Электронный ресурс] / URL: https://bpe24.ru/ (Дата обращения 23.10.2015, режим доступа: свободный).

15. error 100 i/o failure · Issue #325 · ParsePlatform/Parse-SDK-Android · GitHub. [Электронный ресурс] / URL: https://github.com/ParsePlatform/Parse-SDK-Android/issues/325 (Дата обращения 23.01.2016, режим доступа: свободный).

Номер изменения	Подпись ответственного за внесение изменения	Дата внесения изменения

6. ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ, ПОЛЕЙ И МЕТОДОВ

6.1. Application

Класс, содержащий глобальные константы, метод onCreate, в котором инициализируется SDK Parse и регистрируются подклассы QuidPost и QuidComment класса ParseObject для создания соответствующих таблиц в сервисе Parse.

Также внутри содержится класс ParseLogInterceptor, который необходим для функционирования SDK Parse в Android API 14-19 версий (bugfix от разработчиков Parse) [15].

6.2. QuidPost

Класс, наследующий класс ParseObject, определяющий структуру для хранения записей (постов) на сервере Parse.

Методы: геттеры и сеттеры – инкапсулируют обращение к полям записей:

- `public String getName()`
- `public void setName(String value)`
- `public ParseUser getAuthor()`
- `public void setAuthor(ParseUser value)`
- `public String getText()`
- `public void setText(String value)`
- `public ParseGeoPoint getLocation()`
- `public void setLocation(ParseGeoPoint value)`
- `public PostState getState()`
- `public void setState(PostState value)`
- `public static ParseQuery<QuidPost> getQuery()`

Перечисление PostState определяет возможные виды постов.

6.3. QuidComment

Класс, наследующий класс ParseObject, определяющий структуру для хранения комментариев к записям (постам) на сервере Parse.

Методы: геттеры и сеттеры – инкапсулируют обращение к полям записей:

- `public String getText()`
- `public void setText(String value)`
- `public ParseUser getAuthor()`
- `public void setAuthor(ParseUser value)`
- `public String getPostID()`
- `public void setPostID(String value)`
- `public static ParseQuery<QuidComment> getQuery()`

6.4. DataTransform

Содержит два метода:

- `private static String GetTimePassed(Date date, Date now , Context context)`
- `public static String GetTimePassedTillNow(Date date , Context context)`

Второй метод использует первый и отображает время, прошедшее со значения date до текущего момента (до значения now) в упрощенном виде, в соответствии с локализацией, используемой в context.

6.5. activities

Далее будут описаны классы, помещенные в соответствующий каталог, и представляющие возможное поведение для отдельных состояний (экранов) приложения. В соответствие каждому классу (кроме класса DispatchActivity) соответствует XML-файл разметки экрана, содержащийся в директории res/layout.

6.5.1. DispatchActivity

Основная активность, запускаемая при запуске приложения. Не имеет соответствующего XML-файла с разметкой. В зависимости от того, содержатся ли данные об активном пользователе, отображает MainActivity (если содержится), или WelcomeActivity с предложением зарегистрироваться или войти в зарегистрированную учетную запись.

Наследует класс android.app.Activity. Переопределяет метод `protected void onCreate(Bundle savedInstanceState)`.

6.5.2. WelcomeActivity

Активность, запускаемая после DispatchActivity, если приложение не находит информацию об активном пользователе. Имеет графический интерфейс, описанный в файле activity_welcome.xml. Отображает название и краткую информацию о приложении (это первый экран, который видит пользователь), а также две кнопки: «Зарегистрироваться» и «Войти» (здесь и далее названия кнопок и других элементов интерфейса могут отличаться в зависимости от локализации), по нажатию которых осуществляется переход к соответствующим активностям.

Наследует класс android.app.Activity. Переопределяет метод `protected void` **`onCreate`**`(Bundle savedInstanceState).`

6.5.3. SignUpActivity

Активность, запускаемая по нажатию кнопки «Зарегистрироваться» в WelcomeActivity. Имеет графический интерфейс, описанный в файле activity_signup.xml. Отображает три текстовых поля для ввода имени пользователя, email и пароля, и кнопку «Зарегистрироваться», по нажатию которой происходит проверка корректности введенных данных и регистрация пользователя в системе.

Наследует класс android.app.Activity. Переопределяет метод `protected void` **`onCreate`**`(Bundle savedInstanceState).`

Содержит два собственных метода. Первый метод `private void` **`signup`**`()` осуществляет проверку данных и процесс регистрации. Второй – `private String check(String name, String passw, String email)` – проверяет введенные данные на корректность и возвращает строку “OK” в случае успеха, либо текст ошибки.

Номер изменения	Подпись ответственного за внесение изменения	Дата внесения изменения

6.5.4. LogInActivity

Активность, запускаемая по нажатию кнопки «Войти» в WelcomeActivity. Имеет графический интерфейс, описанный в файле activity_login.xml. По умолчанию отображает два текстовых поля для ввода имени пользователя и пароля, кнопку «Войти», по нажатию которой происходит проверка корректности введенных данных и вход пользователя в приложение. Ниже также расположена кнопка «Забыли пароль», по нажатию которой отображается дополнительное текстовое поле для ввода email и кнопка «Восстановить пароль», по нажатию которой, в случае существующего аккаунта с данным email, пользователю высылается письмо с указаниями по восстановлению пароля.

Наследует класс android.app.Activity. Переопределяет метод `protected void` **`onCreate`**`(Bundle savedInstanceState).`

Содержит два собственных метода. Первый метод `private void` **`login`**`()` осуществляет проверку данных и процесс входа в приложение. Второй – `private String check(String name, String passw)` – проверяет введенные данные на корректность и возвращает строку “OK” в случае успеха, либо текст ошибки.

6.5.5. MainActivity

Активность, запускаемая после DispatchActivity, если приложение находит информацию об активном пользователе. Имеет графический интерфейс, описанный в файле activity_main.xml. Отображает меню с кнопками «Настройки аккаунта» и «Определить местоположение», по нажатию которых происходит переход в соответствующие активности; строкой, отображающей имя пользователя.

Ниже меню располагается список записей, найденных в заданном местоположении (или строку, сообщающую об отсутствии записей) и «плавающая»

Номер изменения	Подпись ответственного за внесение изменения	Дата внесения изменения

кнопка[2], для добавления новых записей, по нажатию которой вызывается NewPostActivity.

Внешний вид пунктов списка определяется файлом quid_post_item.xml.

Наследует класс android.app.Activity. Переопределяет методы: `protected void` **`onCreate`**`(Bundle savedInstanceState), public void onBackPressed(),protected void onResume().`

Содержит собственные методы: private void doListQuery() – делает запрос с полученными параметрами к серверу Parse и отображает результаты в виде списка; `private void` **`startLocationActivity`**`()` – запускает GetLocationActivity; `private ParseGeoPoint` **`geoPointFromLocation`**`(Location loc)` – преобразует координату – объект типа Location в объект типа ParseGeoPoint.

6.5.6. GetLocationActivity

Активность, запускаемая по нажатию кнопки «Определить местоположение» в MainActivity. Имеет графический интерфейс, описанный в файле activity_get_location.xml. Отображает меню с кнопками «Назад» и «Сохранить местоположение», по нажатию которых происходит переход в MainActivity без сохранения изменений либо с изменением местоположения; строкой, отображающей подсказку для пользователя.

Ниже расположены: кнопка «Определить местоположение через геолокацию», по нажатию которой программа пытается определить местоположение с помощью встроенных функций устройства; текстовое поле для ввода адреса и кнопка «Найти по адресу», по нажатию которой программа пытается определить местоположение по введенному адресу; элемент с выбором пункта из списка предложенных для определения радиуса поиска записей.

Наследует класс android.app.Activity, реализует интерфейсы GoogleApiClient.ConnectionCallbacks, GoogleApiClient.OnConnectionFailedListener.

Переопределяет методы `protected void` **`onCreate`**`(Bundle savedInstanceState),` `protected void` **`onStart`**`(),` `protected void` **`onStop`**`(),` `public void` **`onConnected`**`(Bundle bundle),` `public void` **`onConnectionSuspended`**`(int i),` `public void` **`onConnectionFailed`**`(@NonNull ConnectionResult connectionResult).`

Содержит собственный метод `public String` **`getJSON`**`(String urle)` – возвращает строковое представление объекта JSON, получаемого в качестве GET-запроса по URL, передаваемому в виде строки urle.

6.5.7. UserPageActivity

Активность, запускаемая по нажатию кнопки «Настройки аккаунта» в MainActivity. Имеет графический интерфейс, описанный в файле activity_user_page.xml. Отображает меню с кнопками «Назад» и «Сохранить местоположение», по нажатию которых происходит переход в MainActivity без сохранения изменений либо с изменением данных пользователя; строкой, отображающей подсказку для пользователя; кнопкой «Выйти» для выхода из текущего аккаунта и возвращения в DispatchActivity.

Ниже расположены: строка-подсказка «Основная информация», текстовые поля с именем пользователя, email и именем пользователя в telegram (при наличии), изображение аватара.

Наследует класс android.app.Activity. Переопределяет метод `protected void` **`onCreate`**`(Bundle savedInstanceState).`

6.5.8. NewPostActivity

Активность, запускаемая по нажатию кнопки «Новая запись» в MainActivity. Имеет графический интерфейс, описанный в файле activity_new_post.xml.

Отображает меню с кнопками «Назад» и «Добавить запись», по нажатию которых происходит переход в MainActivity без сохранения изменений либо с добавлением записи; строкой, отображающей подсказку для пользователя.

Ниже расположены: текстовые поля для ввода заголовка и описания записи; элемент с выбором пункта из списка предложенных для определения типа записи; кнопка «Добавить новую запись», которая дублирует действие кнопки «Добавить запись» из меню.

Наследует класс android.app.Activity. Переопределяет метод `protected void` **`onCreate`**`(Bundle savedInstanceState).`

Содержит метод `private void` **`post`**`()`, который осуществляет процесс добавления записи в систему.

6.5.9. PostInfoActivity

Активность, запускаемая по нажатию на выбранную запись в MainActivity. Имеет графический интерфейс, описанный в файле activity_post_info.xml. Отображает меню с кнопками: «Назад», по нажатию которой происходит переход в MainActivity; «Поделиться», по нажатию которой отображается окно с выбором подходящего приложения, в которое передается текстовое представление текущей записи; строкой, отображающей подсказку для пользователя.

Ниже расположены: аватар пользователя, опубликовавшего запись, его имя, название и описание записи, кнопка «Показать пост на карте» – отображает окно с выбором подходящего картографического приложения, в котором будет отображена точка, к которой прикреплена запись.

Еще ниже расположен список комментариев, оставленных к этой записи (формат которых определяется файлом quid_comment_item.xml), текстовое поле для ввода нового и кнопка «Добавить» для добавления нового комментария.

Наследует класс android.app.Activity. Переопределяет метод `protected void` **`onCreate`**`(Bundle savedInstanceState).`

Содержит методы: private String getLocHref() – для получения URL для отображения геометки записи в стороннем приложении; private void addcomment() – для добавления комментария в систему; private void doCommentsQuery() – для получения и отображения в виде списка комментариев с сервера; private void findPostByID(String thisPostId) – для загрузки записи по её ID.

6.5.10. MyPostsActivity

Активность, запускаемая после нажатия соответствующей кнопки меню. Имеет графический интерфейс, описанный в файле activity_my_posts.xml. Отображает меню с кнопками «Настройки аккаунта» и «Определить местоположение», по нажатию которых происходит переход в соответствующие активности; строкой, отображающей имя пользователя.

Ниже меню располагается список записей, созданных текущим пользователем.

Внешний вид пунктов списка определяется файлом quid_post_item.xml.

Наследует класс android.app.Activity. Переопределяет методы: `protected void` **`onCreate`**`(Bundle savedInstanceState), public void onBackPressed(),protected void onResume().`

Содержит собственные методы: private void doListQuery() – делает запрос с полученными параметрами к серверу Parse и отображает результаты в виде списка; `private void` **`startLocationActivity`**`()` – запускает GetLocationActivity; `private ParseGeoPoint` **`geoPointFromLocation`**`(Location loc)` – преобразует координату – объект типа Location в объект типа ParseGeoPoint.

Номер изменения	Подпись ответственного за внесение изменения	Дата внесения изменения

6.5.11. MyCommentsActivity

Активность, запускаемая после после нажатия соответствующей кнопки меню. Имеет графический интерфейс, описанный в файле activity_my_comments.xml. Отображает меню с кнопками «Настройки аккаунта» и «Определить местоположение», по нажатию которых происходит переход в соответствующие активности; строкой, отображающей имя пользователя.

Ниже меню располагается список комментариев, оставленных текущим пользователем.

Внешний вид пунктов списка определяется файлом quid_comment_item.xml.

Наследует класс android.app.Activity. Переопределяет методы: `protected void` **`onCreate`**`(Bundle savedInstanceState), public void onBackPressed(),protected void onResume().`

Содержит собственные методы: private void doListQuery() – делает запрос с полученными параметрами к серверу Parse и отображает результаты в виде списка; `private void` **`startLocationActivity`**`()` – запускает GetLocationActivity; `private ParseGeoPoint` **`geoPointFromLocation`**`(Location loc)` – преобразует координату – объект типа Location в объект типа ParseGeoPoint.

Номер изменения	Подпись ответственного за внесение изменения	Дата внесения изменения

Лист регистрации изменений									
Изм.	Номера листов (страниц)				Всего листов (страниц) в документе	№ документа	Входящий № сопроводительного документа и дата	Подпись	Дата
	измененных	замененных	новых	аннулированных					

Номер изменения	Подпись ответственного за внесение изменения	Дата внесения изменения