

Санкт-Петербургский государственный университет

Программная инженерия

Соколова Полина Александровна

**Социальная сеть вещей как элемент
универсальной цифровой платформы
«умного дома»**

Выпускная квалификационная работа

Научный руководитель:
ст. преп. Баклановский М. В.

Рецензент:
ст. преп. Зеленчук И. В.

Санкт-Петербург
2018

SAINT-PETERSBURG STATE UNIVERSITY

Software Engineering

Polina Sokolova

Social network of things as element of universal digital smart home platform

Graduation Thesis

Scientific supervisor:
assistant prof. Maxim Baklanovsky

Reviewer:
assistant prof. Ilya Zelenchuk

Saint-Petersburg
2018

Оглавление

Введение	4
1. Постановка задачи	7
2. Обзор	8
2.1. Существующие платформы «умного дома»	8
2.2. Используемые технологии	11
3. Архитектура системы	12
3.1. Социальная сеть вещей	13
3.2. База данных	14
4. Android-клиент	16
4.1. Регистрация и вход	16
4.2. Считывание NFC-меток	17
4.3. Поиск по названию	18
4.4. Добавление профиля новой вещи	19
4.5. Профиль добавленной вещи	20
4.6. Профиль модели	21
5. Веб-сервер	24
6. Апробация	27
Заключение	29
Список литературы	31

Введение

Интернет вещей – это сеть физических устройств, транспортных средств, бытовой техники и других предметов, оснащенных электроникой, программным обеспечением, датчиками, приводами и сетевым подключением, которые позволяют этим объектам подключаться друг к другу и обмениваться данными [1]. Каждая вещь уникально идентифицируется через встроенную вычислительную систему, но способна взаимодействовать в рамках существующей инфраструктуры Интернета. Интернет вещей позволяет объектам восприниматься или быть контролируемыми удаленно в рамках существующей сетевой инфраструктуры, создавая возможности для более прямой интеграции физического мира в компьютерные системы, в результате повышая эффективность, точность и экономическую выгоду в дополнение к сокращению вмешательства человека.

Весь Интернет вещей можно разделить на промышленный и потребительский. Примером промышленного Интернета вещей является «умный город» – концепция объединения информационных и коммуникационных технологий и различных физических устройств, подключенных к сети Интернет для повышения эффективности обслуживания и удовлетворения нужд граждан [2].

Потребительский Интернет вещей представляют системы «умного дома». «Умный дом» – это система автоматизации и компьютеризации дома, включающая в себя контроль и автоматизацию освещения, отопления, вентиляцию, кондиционирование воздуха и безопасность, а также бытовую технику. Современные системы обычно состоят из переключателей и датчиков, подключенных к центральному концентратору, из которого система управляется пользовательским интерфейсом, который взаимодействует либо с настенным терминалом, программным обеспечением для мобильных телефонов, планшетным компьютером или веб-интерфейсом [3].

Рынок «умных домов» в России находится на стадии формирования. В 2013 году, до экономического кризиса, оборот таких систем показы-

вал развитие и составлял около 4 млрд. рублей. Основные пользователи этой технологии в мире – жители США и Западной Европы, где сосредоточено около 90% от общемирового количества «умных домов», и с каждым годом их число растет. В России рынку мешают развиваться высокая стоимость этих разработок и неготовность населения вкладываться в них. Тем не менее, перспективы для роста есть [4]. Эксперты прогнозируют большой рост спроса на комплексные системы уже к 2022 году, и если сейчас рынок находится на начальной стадии, то его ждёт неизбежный, лавинообразный рост, в среднем на 21% в год [5].

«Умная вещь» – это электронное устройство, обладающее центральным процессором и имеющее выход в Интернет для коммуникации с другими устройствами. На сегодняшний день в области «умных домов» и Интернета вещей нет стандартизации, так что все такие устройства работают на разных протоколах (проводных и беспроводных). Кроме того, каждый производитель, как правило, предлагает свою систему контроля для выпускаемых «умных вещей». В результате, у пользователя, желающего автоматизировать свое жилище, нет единого интерфейса взаимодействия со всеми «умными вещами» в рамках системы «умного дома».

Помимо этого, на данный момент существует огромное количество вещей, которые не могут быть включены в систему «умного дома» и быть управляемыми дистанционно, поскольку не обладают выходом в Интернет. Таким образом, на сегодняшний день не существует универсальной цифровой платформы «умного дома» с единым интерфейсом взаимодействия пользователя и его вещей и полной поддержкой концепции Интернета вещей как глобальной сети.

В настоящее время большой популярностью пользуются мобильные сервисы. Это связано с тем, что около 67% жителей планеты используют такие мобильные устройства, как смартфоны и планшеты. Популярность мобильных устройств обусловлена их доступностью, возможностью быть всегда под рукой и выполнять задачи, аналогичные тем, что осуществляются с помощью компьютера.

Сегодня самой продаваемой мобильной операционной системой яв-

ляется Android. Доля глобальных продаж устройств на базе этой платформы составляет 87.7% по данным на первое полугодие 2017 года [6]. Популярность Android-устройств связана в первую очередь с ценовой доступностью.

Единым интерфейсом взаимодействия пользователя и его вещей может выступить социальная сеть вещей. Сервис социальной сети вещей – это онлайн-платформа, которая используется для создания социальных связей, но не между отдельными пользователями, как в социальной сети людей, а между их вещами.

Мы покупаем новую вещь, например, роутер, настраиваем её по инструкции, но почему-то мобильный телефон не может подключиться к сети. Тогда мы обращаемся к Интернету для решения этого вопроса: ищем ответы на форумах в надежде, что кто-то уже сталкивался с данной проблемой.

Социальная сеть вещей может решить проблему объединения людей для обсуждения их покупок, обеспечивая обмен опытом использования вещей между различными пользователями.

1. Постановка задачи

Целью данной дипломной работы является разработка сервиса социальной сети вещей для последующего включения в систему «умного дома». Для достижения этой цели были сформулированы следующие задачи.

1. Проанализировать существующие платформы «умного дома».
2. Выполнить проектирование архитектуры социальной сети вещей.
3. Разработать модуль социальной сети вещей в Android-приложении «Дом вещей».
4. Разработать веб-сервер социальной сети вещей.
5. Выполнить тестирование созданного приложения.

2. Обзор

Представьте, что у вас есть много разных вещей, как "умных", так и не обладающих выходом в Интернет. Вы хотите хранить информацию о своих вещах, но не в бумажном виде дома, а в оцифрованном, с возможностью иметь к ней доступ в любое время. Кроме того, вам хочется, чтобы "умные вещи" могли выполнять какие-то примитивные действия без прямого участия хозяина. Например, включение кондиционера при превышении определенной температуры. Помимо этого, вы также хотите иметь систему, которая бы диагностировала проблемы с вашими вещами и предлагала пути их решения.

Проанализируем существующие на рынке платформы "умного дома" с целью определения, подходят ли они для решения вышеперечисленных проблем. В ходе обзора будут рассмотрены следующие платформы: HomeKit, MajorDoMo, openHAB, Thinger.

2.1. Существующие платформы «умного дома»

С помощью платформы HomeKit можно подключить, объединить и контролировать всю «умную технику» в доме, имея под рукой iOS-устройство. Уже более 50 брендов выпускают гаджеты с поддержкой HomeKit [7].

HomeKit обеспечивает бесшовную интеграцию между устройствами, которые поддерживают Home Automation Protocol, и iOS-устройствами от Apple, открывая новые возможности в домашней автоматизации. Продвигая общий протокол для устройств домашней автоматизации и сделав публичное API, доступное для настройки и взаимодействия с этими устройствами, HomeKit делает возможным рынок, где приложения для управления своим домом не зависят от поставщика, который делает устройства для домашней автоматизации и где эти устройства от различных производителей могут быть интегрированы в единое целое без продавцов, которым необходимо сотрудничать непосредственно друг с другом.

MajorDoMo – это программное обеспечение, которое позволяет сде-

лать из компьютера (Windows/Linux) контроллер управления домашней автоматикой. Одним из основных преимуществ MajorDoMo является возможность одновременной работы с оборудованием на различных стандартах и от разных производителей.

Для работы с оборудованием в системе существуют так называемые модули, которые обеспечивают обмен данными по соответствующим протоколам между оборудованием, внутренними объектами системы и пользовательским интерфейсом. В системе реализована поддержка множества аппаратных протоколов (ZWave, ModBus, KNX, 1-Wire и т.д.), и их количество продолжает расти. Система является расширяемой, и любой пользователь может написать собственный модуль.

Взаимодействие с системой происходит через веб-интерфейс, но есть и мобильная версия под операционную систему Android, которая позволяет управлять сервером умного дома MajorDoMo.

OpenHAB – это проект домашней автоматизации с открытым исходным кодом, нацеленный на создание универсальной платформы для объединения всей домашней «умной техники» в единую систему управления [8]. OpenHAB реализует единую шину, то есть позволяет объединить все устройства с разными протоколами в единую сеть, абстрагируя пользователя от каждого конкретного протокола. Таким образом, можно пользоваться единым средством управления (скажем, приложением на смартфоне) и реализовать сколь угодно сложную логику взаимосвязи между устройствами.

В результате открытости уже сейчас openHAB поддерживает более 100 различных протоколов «умных устройств». Это, в свою очередь, дает возможность пользователю выбирать устройства для своего дома по самым разнообразным параметрам, практически не ограничиваясь возможностями всей платформы.

Thinger – это централизованный ресурс, через который каждый пользователь может просматривать, управлять и организовывать взаимодействие между различными устройствами [9]. У каждого пользователя в рамках этой платформы есть профиль. Также профиль заводится на каждую вещь отдельно. И пользователь может к своему профилю

привязать профили своих вещей, тем самым иметь единый интерфейс для управления ими всеми.

Платформа включает в себя две подсистемы: сайт (визуальная составляющая, которую видят все пользователи) и веб-сервис, представляющий единый программный интерфейс (RESTful API), который используется непосредственно вещами для передачи данных в систему и также конечными приложениями или различными виджетами, которые получают или передают данные этим вещам.

В настоящее время большая часть функций системы закрыта от публичного использования сторонними разработчиками и используется только для работы с корпоративными клиентами.

Представьте, что у вас дома установлена одна из вышеперечисленных систем домашней автоматизации. В один момент у вас ломается стиральная машина: она выключилась и больше не включается. Что делать?

Если такая проблема возникает впервые, вы начинаете обзванивать знакомых или спрашивать в Интернете в надежде, что кто-то может помочь вам получить ответ на свой вопрос. Хорошо, когда есть знакомый специалист, готовый проконсультировать вас по телефону. Но так бывает не у всех людей и не всегда, особенно когда дело касается решения сложных технических вопросов.

Платформа "умного дома" не может вам помочь. Во-первых, стиральная машина не включена в систему, поэтому вы не можете получить детальной информации о том, что случилось. Во-вторых, платформа домашней автоматизации нацелена в первую очередь на управление вещами, а не на обмен информацией между ними, например, для самодиагностики.

Для решения этой проблемы необходимо создание универсальной цифровой платформы "умного дома", поддерживающей включение в систему как "умных вещей", так и не обладающих выходом в Интернет. Кроме того, важно обеспечить социальную функцию платформы, заключающуюся в возможности обмениваться опытом использования вещей между различными пользователями.

2.2. Используемые технологии

В данной дипломной работе используются следующие технологии.

1. Near-field communication (NFC) – технология беспроводной передачи данных малого радиуса действия между двумя устройствами [10]. NFC-устройства используются в бесконтактных платежных системах, аналогичных тем, которые применяются в кредитных картах и смарт-картах электронных билетов, и позволяют мобильным платежам заменять или дополнять эти системы.
2. Android – это мобильная операционная система, разработанная Google на основе модифицированной версии ядра Linux и другого программного обеспечения с открытым исходным кодом, и предназначенная в основном для мобильных устройств с сенсорным экраном, таких как смартфоны и планшеты [11]. Система Android предоставляет разностороннюю платформу приложений, на основе которой можно создавать инновационные приложения и игры для мобильных устройств в среде языка Java [12].
3. MySQL – реляционная система управления базами данных с открытым исходным кодом [13]. Обычно MySQL используется в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы.

3. Архитектура системы

Для того, чтобы сделать возможным включение в систему «умного дома» любой вещи, необходимо создание единого интерфейса взаимодействия пользователя и его вещей, как «умных», так и не обладающих выходом в Интернет. Таким интерфейсом выступает Android-приложение «Дом вещей».

«Умные вещи» по проводным и беспроводным протоколам связи подключены к центральному контроллеру, который, в свою очередь, соединен с домашним веб-сервером по Wi-Fi (Рис. 1). Роль домашнего веб-сервера может выполнять, например, компьютер или мобильное устройство, но обязательно находящееся с центральным контроллером в одной сети Wi-Fi.

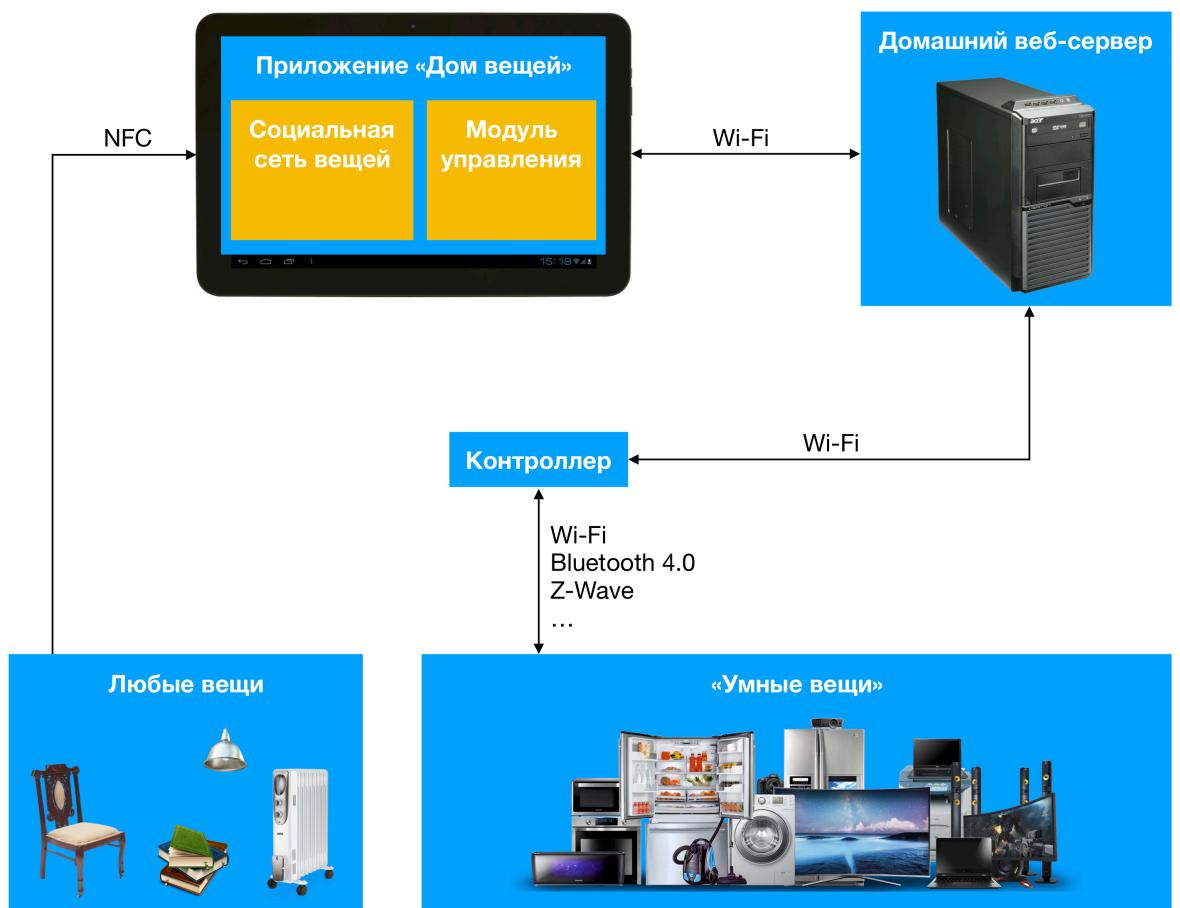


Рис. 1: Архитектура платформы «умного дома»

Модуль управления в приложении «Дом вещей» взаимодействует с домашним веб-сервером и необходим для осуществления удаленного управления и мониторинга «умных вещей». При подключении новой «умной вещи» к центральному контроллеру домашний веб-сервер автоматически добавляет объект в систему. Кроме того, информация о новой вещи отправляется в модуль управления Android-приложения «Дом вещей», и в социальную сеть вещей объект добавляется без участия пользователя.

Вещи, которые не могут быть включены в систему вышеописанным образом, снабжаются NFC-метками для их идентификации в социальной сети вещей. Для добавления таких вещей в социальную сеть необходимо провести сканирование NFC-метки встроенным или внешним NFC-считывателем.

3.1. Социальная сеть вещей

Модуль социальной сети вещей в приложении «Дом вещей» необходим для хранения информации о собственных вещах в системе «умного дома» и обмена пользовательским опытом использования вещей, включенных в разные системы. Таким образом, социальная сеть вещей позволяет создание глобальной сети вещей, не ограничиваясь рамками одного «умного дома».

Социальная сеть вещей представляет собой Интернет-сервис, состоящий из Android-клиента, Java-сервера и базы данных MySQL (Рис. 2).

Android-клиент отправляет соответствующие HTTP-запросы на сервер с целью добавления, редактирования, удаления или получения данных. Java-сервер, в свою очередь, принимает HTTP-запрос от клиента, посыпает запрос к базе данных MySQL и получает выборку данных. После этого сервер формирует и отправляет ответ клиенту в XML-формате.

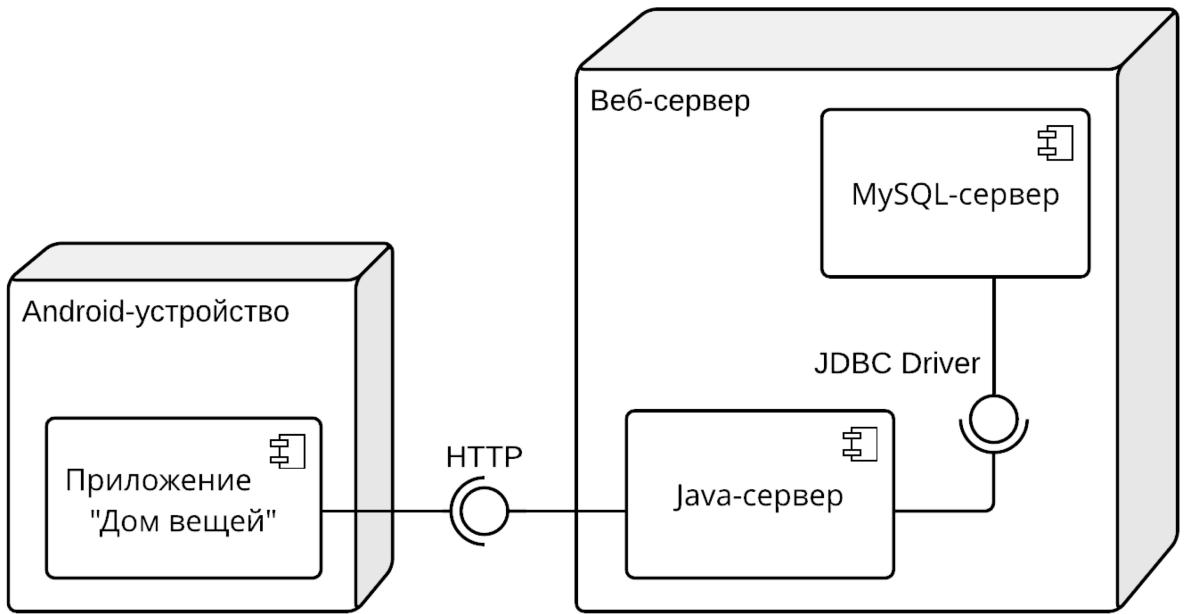


Рис. 2: Диаграмма компонентов социальной сети вещей

3.2. База данных

Глобальная база данных MySQL состоит из 12 таблиц (Рис. 3). В таблице purchase хранятся записи с информацией о добавленных пользователями вещах. Таблица device содержит записи с параметрами всех существующих товаров.

Все таблицы, кроме таблицы purchase, изначально заполнены данными, но при необходимости могут быть обновлены добавлением необходимых записей. Например, если пользователь сообщает о том, что его вещи нет в списке предложенных, то после проведения определенных проверок вещь может быть добавлена в таблицу device.

В таблице user хранятся логины и хеш-значения паролей.

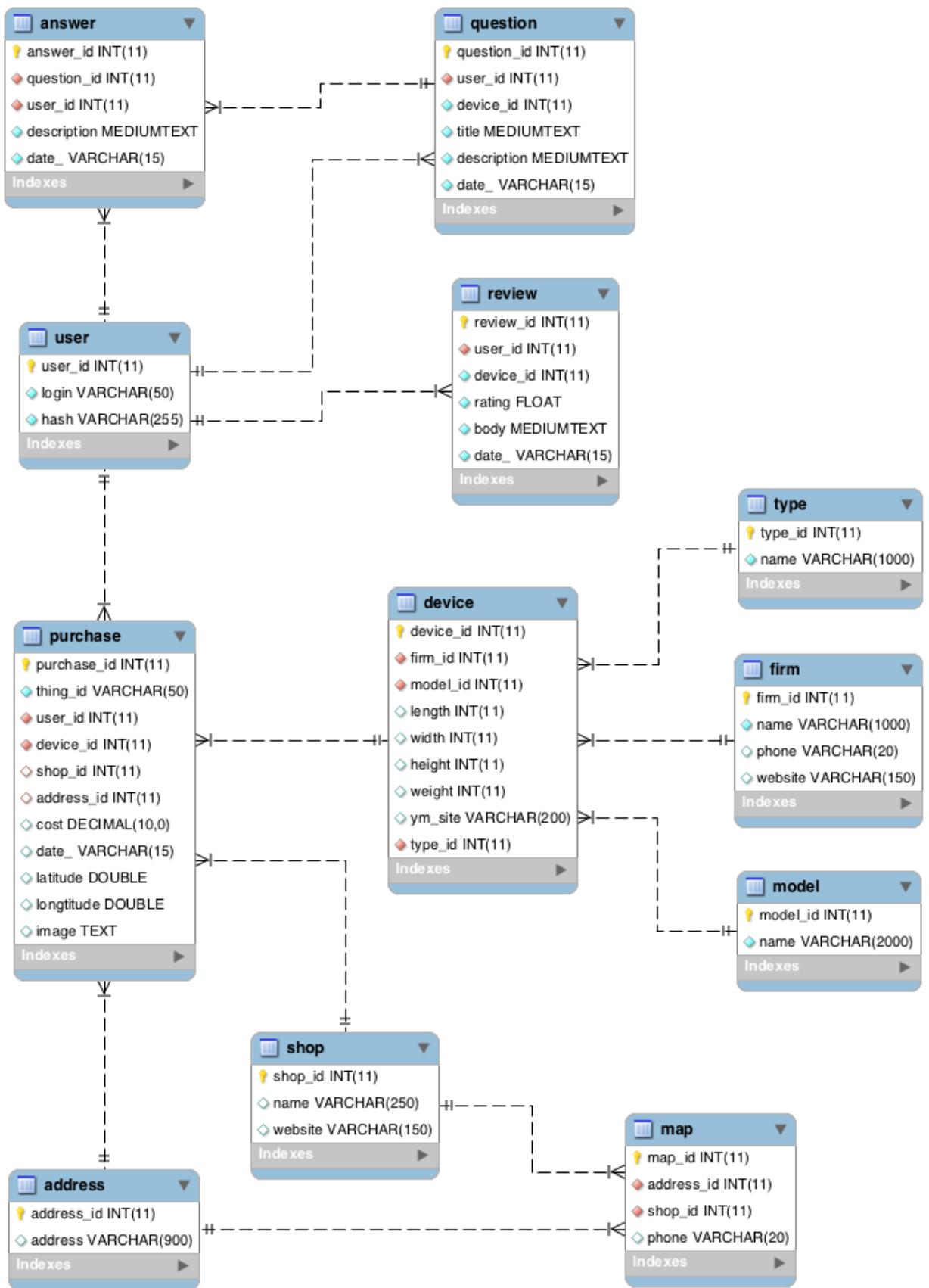


Рис. 3: Схема базы данных MySQL

4. Android-клиент

4.1. Регистрация и вход

Использование Android-приложения «Дом вещей» начинается с экранов регистрации и входа (Рис. 4). Пользователю предлагается зарегистрировать новую учетную запись, состоящую из логина и пароля, или войти под уже существующей.

После ввода логина и пароля пользователем на экране регистрации Android-клиент отправляет POST-запрос на сервер для создания новой учетной записи. Если такой логин уже существует в базе данных, будет получен код "409 Conflict". На экране приложения появится сообщение с соответствующей подписью.

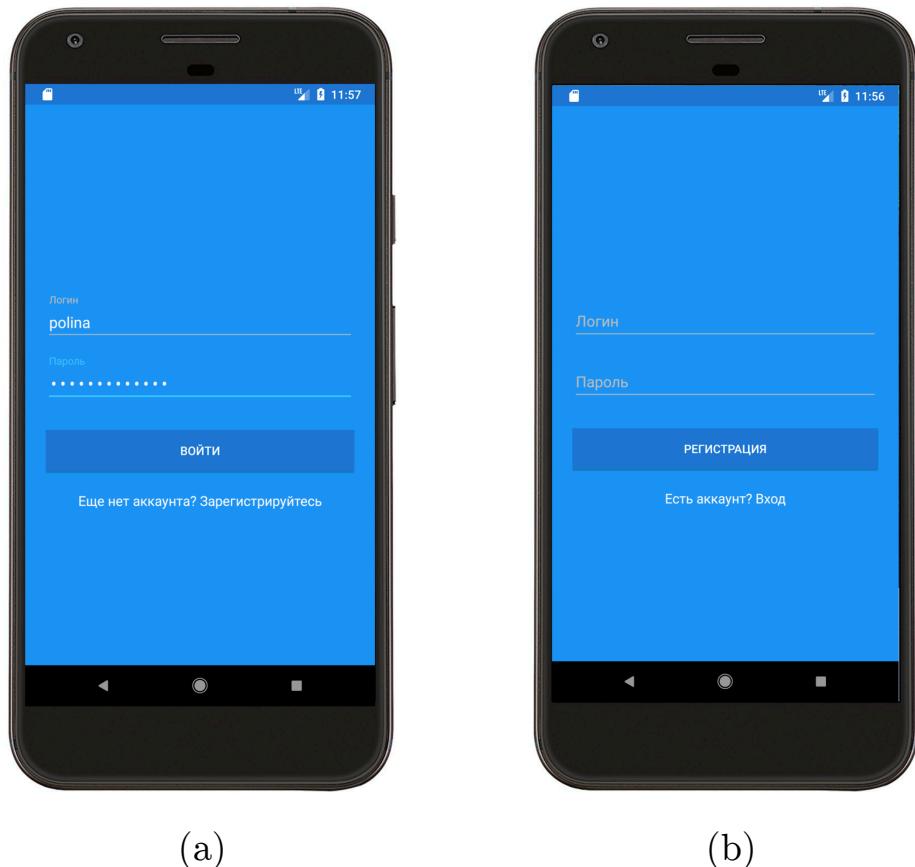


Рис. 4: Пользовательские интерфейсы Android-приложения «Дом вещей»: (a) – экран входа, (b) – экран регистрации

После ввода логина и пароля пользователем на экране входа Android-клиент отправляет GET-запрос на сервер для проверки связки логин и пароль. В случае неправильного ввода параметров с сервера будет получен код ответа "401 Unauthorized".

В случае успешного создания нового аккаунта или ввода корректных логина и пароля уже существующей учетной записи будет получен код "200 OK", и пользователю будет доступен главный экран приложения.

4.2. Считывание NFC-меток

После входа в систему пользователю доступен главный экран Android-приложения со списком всех добавленных вещей (Рис. 5а). Для добавления новой вещи пользователю необходимо поднести NFC-считыватель к NFC-метке на соответствующей вещи.

В результате курсовой работы 2016-2017 учебного года мной была реализована поддержка сканирования NFC-меток внешним RFID-считывателем [14]. Сейчас в Android-приложении существует также возможность считывать NFC-метки встроенным в мобильное устройство NFC-адаптером.

Идентификационный номер NFC-метки NTag213, используемой в данной системе, занимает 7 байт памяти. Встроенный NFC-адаптер считывает и воспринимает его в шестнадцатеричном виде с порядком от старшего к младшему, так что идентификационный номер состоит из 14 символов. Внешний RFID-считыватель считывает только первые 4 байта в десятичной системе счисления.

Для представления идентификационного номера в едином формате при сканировании любым NFC- или RFID-устройством осуществляется преобразование из десятичного представления номера в шестнадцатеричное. Кроме того, в полученной записи изменяется порядок следования символов на порядок от старшего к младшему.

При сканировании NFC-метки встроенным NFC-считывателем происходит сначала поиск равного идентификационного номера в списке полученных с веб-сервера вещей. Если вещь с таким номером существует

ет, то осуществляется переход на экран профиля вещи. В противном случае, если номер NFC-метки занимает 7 байт, то берутся первые 4 байта и им ищется соответствие в списке вещей.

Если вещь с восьмисимвольным идентификационным номером найдена, то происходит переход на экран профиля вещи. Такая ситуация может возникнуть, когда при добавлении вещи используется внешний RFID-считыватель, распознающий первые 4 байта, а в дальнейшем при сканировании задействуется встроенный NFC-адаптер,читывающий все 7 байт идентификационного номера NFC-метки. Если вещь не найдена, то осуществляется переход на экран добавления профиля вещи.

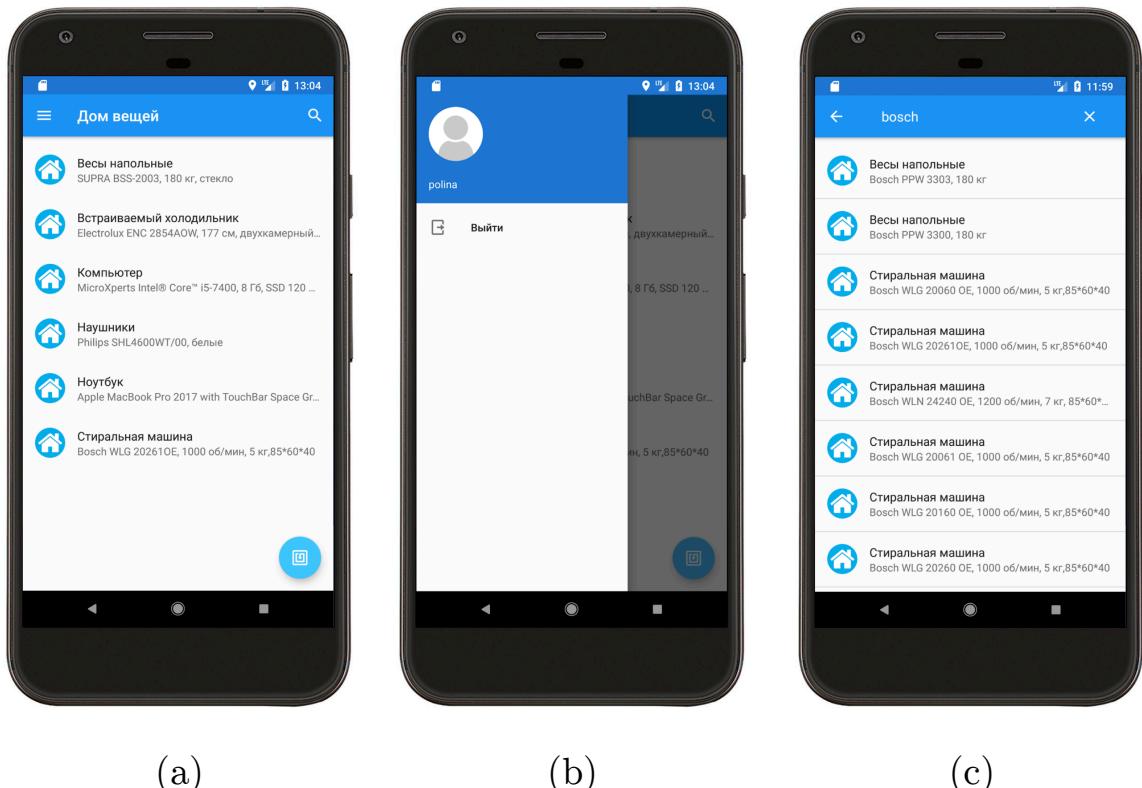


Рис. 5: Пользовательские интерфейсы Android-приложения «Дом вещей»: (а) – главный экран, (б) – панель навигации, (с) – экран поиска

4.3. Поиск по названию

После сканирования NFC-метки пользователю предлагается выбрать свою вещь из списка предложенных (Рис. 5с). Для получения актуаль-

ной информации Android-клиент посыпает GET-запрос на веб-сервер и получает ответ в XML-формате.

Полученные данные сохраняются в XML-файле на мобильном устройстве. В GET-запросе также передается идентификационный номер последней вещи в сохраненном локально XML-файле, так что при каждом таком запросе отправляются не все данные, а только новые, еще не добавленные в XML-файл на мобильном устройстве.

Пользователь вводит модель своей вещи в строку поиска, список всех вещей фильтруется в соответствии с введенными символами, и пользователю показывается обновленный список вещей для выбора.

Кроме того, функции поиска по названию вещи и переход к профилю модели могут быть осуществлены при нажатии на кнопку "Поиск" на главном экране, то есть без создания профиля вещи.

4.4. Добавление профиля новой вещи

После выбора вещи из предлагаемого списка, пользователь заполняет информацию о покупке.

В предыдущей версии, реализованной мной в курсовой работе 2016-2017 учебного года, было 3 последовательных экрана добавления нового профиля: параметры объекта, покупки и сервисного обслуживания [14]. Такой вариант оказался неудобным для пользователя, поскольку требовал тратить значительное количество времени для самостоятельного поиска и заполнения данных.

Сейчас на экране добавления нового профиля пользователь может заполнить только самые необходимые данные о покупке (Рис. 6а).

После добавления информации и нажатия на кнопку сохранения пользователем Android-клиент отправляет POST-запрос с введенными параметрами Java-серверу на добавление записи в глобальную базу данных. Если поле соответствующего параметра не заполнено, то веб-серверу передается значение "NULL".

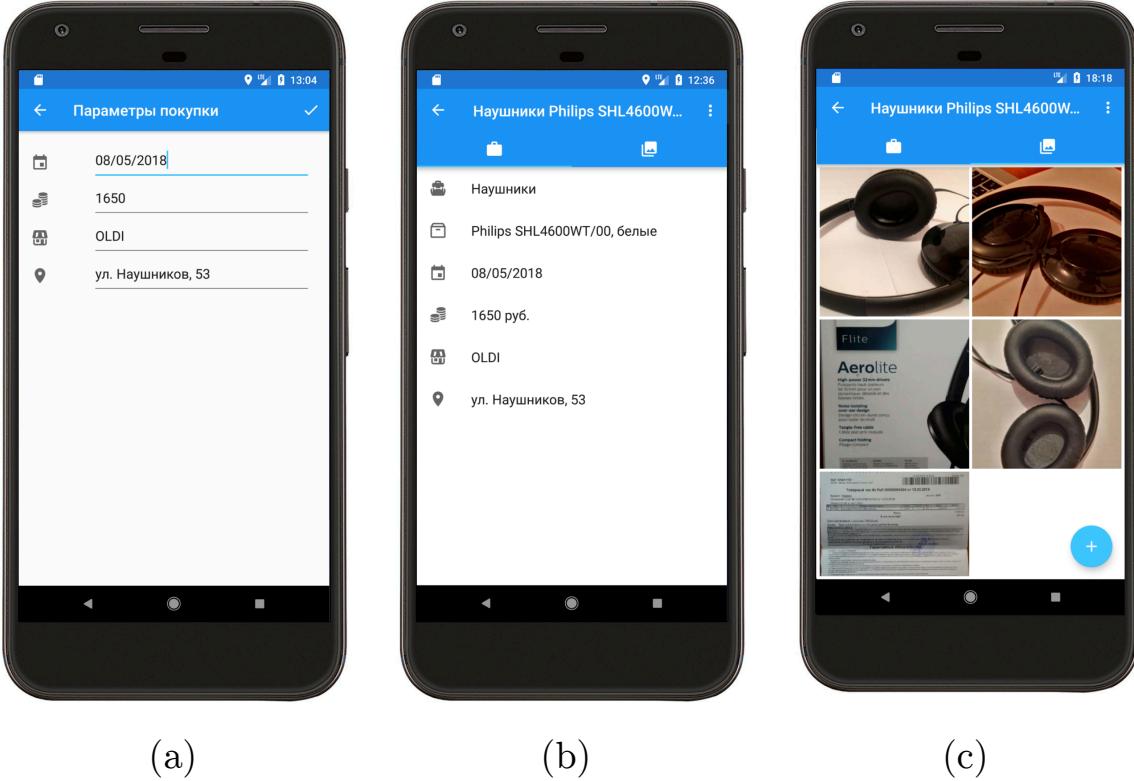


Рис. 6: Пользовательские интерфейсы Android-приложения «Дом вещей»: (а) – экран добавления профиля новой вещи, (б) – экран информации о покупке, (с) – экран фотографий

4.5. Профиль добавленной вещи

В предыдущей версии, реализованной мной в курсовой работе 2016–2017 учебного года, профиль вещи состоял из экрана с фотографией вещи и списком параметров, как введенных пользователем вручную, так и подгруженных с сервера [14]. Сейчас экран профиля вещи представлен интерфейсом с вкладками: информация о покупке и фотографии. (Рис. 6б, 6с). Таким образом, стало возможным добавление фотографий из галереи мобильного устройства.

Пользователю доступны функции редактирования и удаления профиля. После нажатия на кнопку редактирования осуществляется переход к экрану, аналогичному экрану добавления нового профиля, только с заполненными соответствующими полями. При сохранении введенных данных происходит отправка POST-запроса на веб-сервер. По-

сле нажатия на кнопку удаления профиля Android-клиент посыпает DELETE-запрос на удаление профиля вещи.

4.6. Профиль модели

Помимо добавления информации о собственной вещи, пользователь также может обмениваться впечатлениями о модели в целом. Экран профиля модели представлен интерфейсом с вкладками: отзывы и вопросы (Рис. 7).

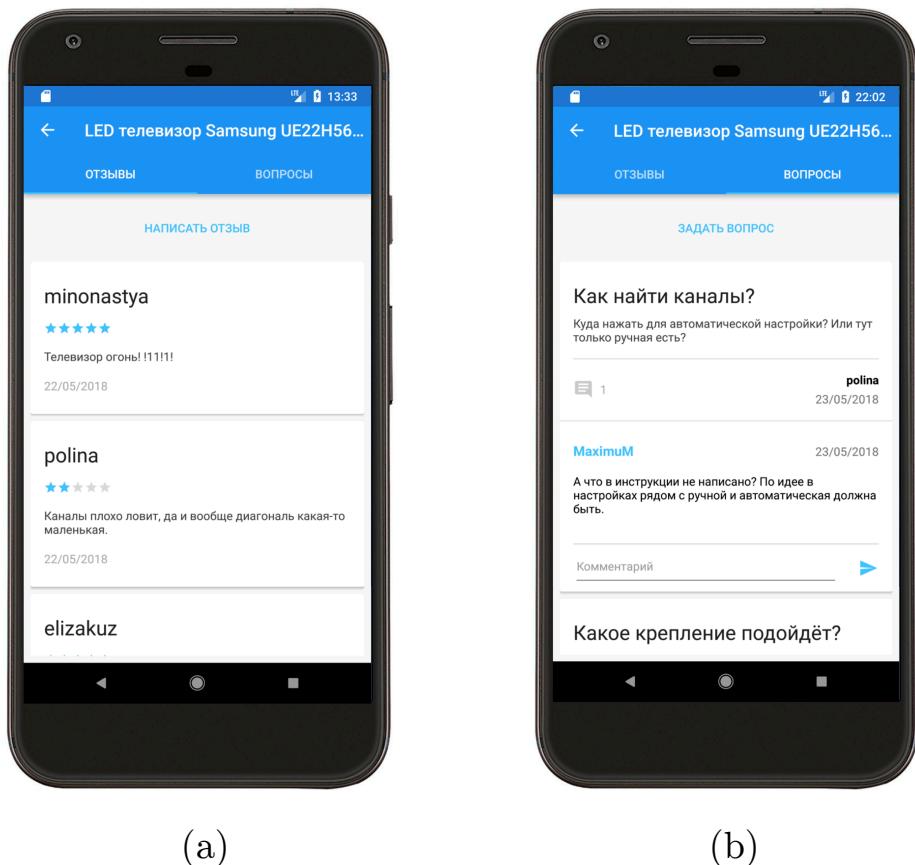


Рис. 7: Пользовательские интерфейсы Android-приложения «Дом вещей»: (а) – экран отзывов, (б) – экран вопросов

На вкладке "Отзывы" пользователь может как узнавать мнения других людей о товаре, так и добавлять свои отзывы по использованию вещи. Кроме того, у покупателей часто возникают вопросы по настройке и эксплуатации товара, ответы на которые не могут быть найдены, например, в прилагаемой к вещи инструкции. На вкладке "Вопросы"

пользователь может задавать интересующие его вопросы, получать на них ответы, а также помогать другим людям решать возникающие с вещью проблемы.

При переходе пользователя в профиль модели Android-клиент отправляет GET-запросы на получение информации об отзывах и вопросах о вещи. Для добавления отзыва или вопроса пользователь нажимает на кнопку "Добавить отзыв" или "Задать вопрос" соответственно, и ему демонстрируется экран добавления отзыва или вопроса (Рис. 8). После ввода всей необходимой информации пользователем Android-клиент отправляет POST-запрос на сервер.

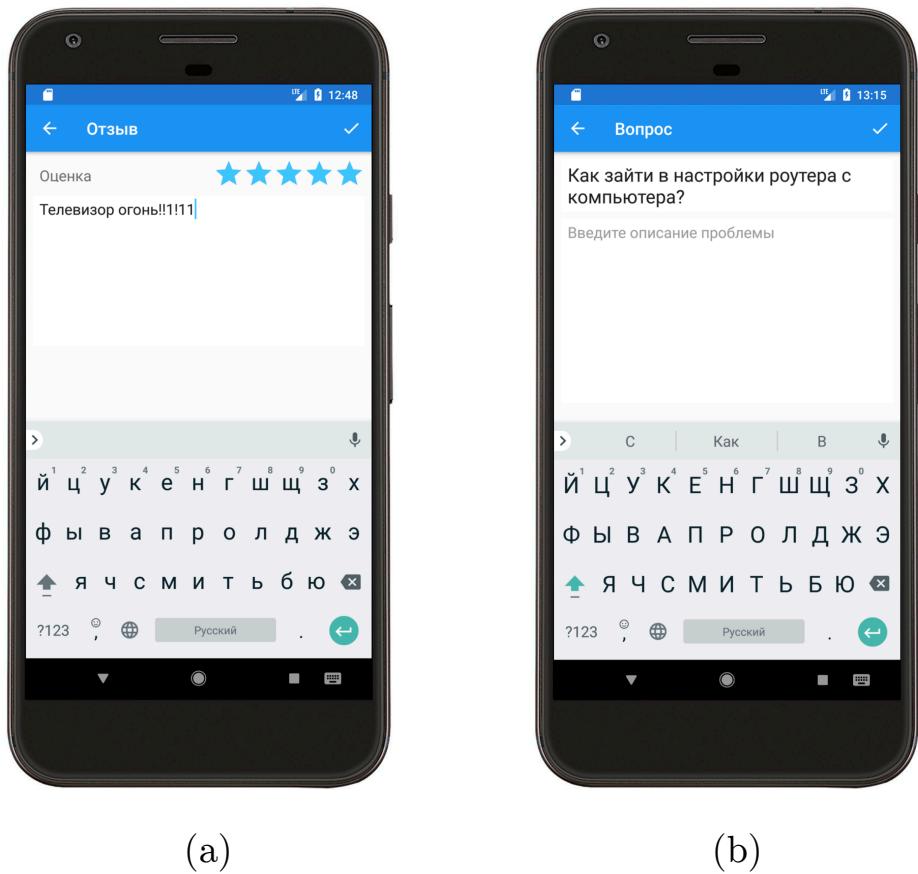


Рис. 8: Пользовательские интерфейсы Android-приложения «Дом вещей»: (а) – экран добавления отзыва, (б) – экран добавления вопроса

Переход в профиль модели осуществляется как из профиля добавленной вещи, так и из экрана поиска вещи. Таким образом, пользова-

тель может читать чужие отзывы и ответы на возникающие вопросы еще до покупки товара и его добавления в список вещей.

5. Веб-сервер

Веб-сервер написан на Java. Сервер создает сокет и ждет HTTP-запросы на порт 8080. При подключении клиента создается новый поток, и в нем начинается обработка запроса, которая заключается в определении метода запроса, *request-uri* и разборе параметров [15]. Сервер обрабатывает запросы: GET, POST, DELETE.

Веб-сервер взаимодействует с MySQL-сервером через JDBC Driver и отправляет SQL-запросы: SELECT, INSERT, UPDATE, DELETE.

GET-запрос отправляется Android-клиентом, когда ему необходимо получить данные.

1. Корректность логина и пароля. Параметры: логин и пароль. Запрос: SELECT к таблице user.
2. Доступные вещи. Параметр: идентификационный номер последней вещи в списке. Запрос: SELECT к таблице device. Тело ответа: список вещей с большими идентификационными номерами. Данные были получены в результате парсинга сайта Юлмарт. В результате получилось скачать информацию о 18413 вещах.
3. Информация о покупке. Параметры: логин пользователя и идентификационный номер вещи, полученный при считывании NFC-метки. Запрос: SELECT к таблицам purchase, device, type, firm, model, shop, address. Тело ответа: название, модель, дата покупки, стоимость, магазин, адрес.
4. Изображения покупки. Параметры: логин пользователя и идентификационный номер вещи. Запрос: SELECT к таблице purchase. Тело ответа: список URI изображений.
5. Отзывы. Параметры: логин пользователя и идентификационный номер добавленной вещи или модели. Запрос: SELECT к таблицам user, review. Тело ответа: список отзывов с логинами, оценками, комментарием и датой публикации.

6. Вопросы. Параметры: логин пользователя и идентификационный номер добавленной вещи или модели. Запрос: SELECT к таблицам user, question, answer. Тело ответа: список вопросов с идентификационным номером вопроса, логином, заголовком, комментарием, датой публикации и количеством ответов.
7. Ответы. Параметры: логин пользователя и идентификационный номер вопроса. Запрос: SELECT к таблицам user, question, answer. Тело ответа: список ответов с идентификационным номером ответа, логином, комментарием, датой публикации.

При успешной обработке запроса в теле HTTP-ответа клиенту отправляются данные в XML-формате.

POST-запрос может быть отправлен Android-клиентом при создании или редактировании информации.

1. Создание учетной записи. Параметры: логин и пароль. Запрос: INSERT к таблице user. Ключ формируется по стандарту PBKDF2.
2. Создание и редактирование профиля вещи. Параметры: идентификационный номер вещи, полученный при считывании NFC-метки, логин пользователя, идентификационный номер модели, дата, стоимость, магазин, адрес, широта и долгота. Запрос: INSERT или UPDATE к таблице purchase.
3. Добавление фотографий в профиль вещи. Параметры: логин пользователя, идентификационный номер вещи, URI изображения. Запрос: INSERT или UPDATE к таблице purchase.
4. Добавление отзыва или вопроса. Параметры: логин пользователя, идентификационный номер модели, заголовок (для вопроса), описание, рейтинг (для отзыва), дата. Запрос: INSERT к таблице review или question.
5. Добавление ответа. Параметры: логин пользователя, идентификационный номер вопроса, описание, дата. Запрос: INSERT к таблице answer.

При успешной обработке запроса в теле HTTP-ответа клиенту отправляется "Object inserted" или "Object updated" в зависимости от совершенных действий.

Android-клиент посыпает DELETE-запрос при удалении профиля добавленной вещи или фотографий из профиля. В качестве параметров передаются идентификационный номер удаляемой вещи и логин пользователя, а также URI удаляемого изображения. Сервер посыпает DELETE-запрос к таблице purchase.

При успешной обработке запроса в теле HTTP-ответа клиенту отправляется "Object deleted".

В HTTP-ответе клиенту может быть передан один из кодов состояния: 200, 401, 404, 409, 500.

При успешной обработке HTTP-запроса Java-сервер посыпает код ответа об успешном статусе "200 OK". Если хеш-значение пароля, соответствующего переданному логину не совпадает с хеш-значением переданного, то клиент получает код "401 Unauthorized". В случае, когда модель не может быть найдена в базе данных, сервер посыпает клиенту код "404 Not Found". При попытке добавления в базу данных user уже существующего логина клиент получает код "409 Conflict". Если в ходе обработки возникли ошибки на сервере, например, некорректный запрос к базе данных, то клиенту отправляется код ответа о внутренней ошибке сервера "500 Internal Server Error".

Для передачи тела ответа используется собственный протокол, основанный на XML.

6. Апробация

Апробация социальной сети вещей состояла из тестирования двух основных частей системы: мобильного приложения и веб-сервера.

Тестирование мобильного приложения заключалось в проверке эргономичности (usability testing) пользовательского интерфейса. Пользователям предлагалось выполнить несколько стандартных сценариев: регистрация, вход, создание нового профиля вещи, добавление и удаление фотографий, поиск и написание отзывов о не добавленной в список вещи, добавление вопросов и ответов о вещи.

Как правило, у пользователей возникали проблемы со сканированием NFC-меток. Это связано, в первую очередь, с недостатком опыта использования NFC-считывателя в связке с NFC-метками.

Кроме того, некоторыми пользователями были высказаны замечания относительно недостаточного количества доступных для выбора вещей: отсутствие уже не выпускаемых на данный момент товаров. Например, в базе данных нет iPad 2, выпущенного в 2011 году.

В целом, мобильное приложение было описано респондентами как юзабельное, с интуитивно понятным интерфейсом.

Для тестирования производительности веб-сервера использовалась утилита ApacheBench. В ходе проведения тестирования оказалось, что веб-сервер поддерживает обслуживание не более 52 пользователей одновременно для всех запросов. Кроме того, при 52 одновременных пользователях, посылающих по 50 последовательных запросов (кроме первого запроса на получение списка всех доступных вещей), среднее время обработки запроса не превышает 0.8 с.

Обработка запроса на получение списка всех доступных вещей, посыпаного пользователем в первый раз (когда списка вещей у пользователя еще нет), не превышает 7 с. Это не критично, поскольку запрос отправляется не в ответ на действия пользователя в мобильном приложении, а с запуском главного экрана. Время обработки последующих запросов, когда список вещей у пользователя заполнен полностью или частично, в среднем не превышает 2,6 с.

По результатам тестирования можно сделать вывод о том, что на первом этапе разработке системы прототип является жизнеспособным: мобильный интерфейс показался пользователям удобным, а веб-сервер справляется с небольшой нагрузкой.

Заключение

В ходе данной работы были получены следующие результаты.

1. Проведен анализ существующих платформ "умного дома": Apple HomeKit, MajorDoMo, openHAB, Thingster.
2. Разработана архитектура социальной сети вещей: Android-клиент, Java-сервер, база данных MySQL.
3. Реализовано Android-приложение социальной сети вещей со следующим функционалом: считывание NFC-меток встроенным NFC-адаптером, регистрация и вход в систему под уникальным логином и паролем, пользовательский интерфейс для просмотра профиля модели, взаимодействие с веб-сервером по протоколу HTTP.
4. Реализован веб-сервер на Java со следующим функционалом: получение HTTP-запросов клиента, отправка соответствующих запросов к базе данных MySQL и получение выборки, формирование и отправка ответа клиенту.
5. Проведено тестирование сервиса: нагружочное тестирование веб-сервера и юзабилити-тестирование мобильного приложения.

В качестве возможного применения Интернет-сервиса можно выделить помочь пользователю с выбором качественного сервисного центра рядом для ремонта его вещи. Коммерциализация продукта может обеспечиваться за счет привлечения мелкого бизнеса: отдельных сервисных центров и индивидуальных мастеров. Сервис будет предлагать пользователю оптимальный вариант, а мастерские, поскольку заинтересованы в продвижении и рекламе своих услуг, будут платить за то, чтобы попасть в список возможных рекомендаций.

Следующим возможным применением Интернет-сервиса является хранение истории эксплуатации вещи для последующего обмена ею с другими пользователями, экспертами и мастерами. Это позволяет экономить время и деньги пользователя, ограждая его от необходимости везти свою вещь в сервисный центр или вызывать мастера на дом.

Кроме того, помимо обсуждения вопросов с другими пользователями, сервис может предоставлять услугу «задать вопрос эксперту». Это бывает полезно, когда вопрос технически сложен и для его решения необходимы специальные знания, которыми не обладает рядовой пользователь. Эксперты привлекаются со стороны, а для пользователя эта услуга предполагается платной.

Список литературы

- [1] Wikipedia. Internet of Things. Wikipedia, The Free Encyclopedia. URL: https://en.wikipedia.org/wiki/Internet_of_things.
- [2] Wikipedia. Smart city. Wikipedia, The Free Encyclopedia. URL: https://en.wikipedia.org/wiki/Smart_city.
- [3] Wikipedia. Home automation. Wikipedia, The Free Encyclopedia. URL: https://en.wikipedia.org/wiki/Home_automation.
- [4] Черкасов Денис. Кому нужен «умный дом»: перспективы развития технологии в России. РБК. 2017. URL: <https://realty.rbc.ru/news/59a6b61b9a79471042a5627a>.
- [5] Беркан Алиса. Стоит ли выходить на российский рынок умных домов? Rusbase. 2017. URL: <https://rb.ru/longread/umnyj-dom-v-rossii/>.
- [6] Global mobile OS market share in sales to end users from 1st quarter 2009 to 2nd quarter 2017. Statista, The Statistic Portal. 2017. URL: <https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/>.
- [7] Умный дом, послушный дом. Apple Inc. URL: <https://www.apple.com/ru/ios/home/>.
- [8] openHAB. openHAB, empowering the smart home. URL: <https://www.openhab.org>.
- [9] Thinger – социальная сеть умных вещей. YouTube. URL: <https://www.youtube.com/watch?v=990PVAmW0oM>.
- [10] Near-field communication. Wikipedia, The Free Encyclopedia. URL: https://en.wikipedia.org/wiki/Near-field_communication.
- [11] Android (operating system). Wikipedia, The Free Encyclopedia. URL: [https://en.wikipedia.org/wiki/Android_\(operating_system\)](https://en.wikipedia.org/wiki/Android_(operating_system)).

- [12] Общие сведения о платформе Android. Android Developers. URL:
<https://developer.android.com/guide/index.html>.
- [13] MySQL. Wikipedia, The Free Encyclopedia. URL:
<https://en.wikipedia.org/wiki/MySQL>.
- [14] Соколова Полина. Разработка интерфейса для базы данных вещей. Курсовая работа, Кафедра Системного Программирования, Математико-механический факультет СПбГУ. 2017.
- [15] Hypertext Transfer Protocol – HTTP/1.1. RFC 2616. 1999. URL:
<https://tools.ietf.org/html/rfc2616>.