МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
**«Южно-Уральский государственный университет
(национальный исследовательский университет)»**
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

РАБОТА ПРОВЕРЕНА
Ведущий программист
ООО «Стратегия роста»
_____ К.Н. Сумороков
“__”_____ 2017 г.

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой, д.ф.-м.н.,
профессор
_____ Л.Б. Соколинский
“__”_____ 2017 г.

РАЗРАБОТКА REST-СЕРВИСА ИНФОРМИРОВАНИЯ КЛИЕНТОВ ОХРАННОЙ ОРГАНИЗАЦИИ

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 02.03.02.2017.13-026-1328.ВКР

Научный руководитель
старший преподаватель кафедры
системного программирования
_____ Н.С. Силкина

Автор работы,
студент группы КЭ-401
_____ Г.И. Федосеев

Ученый секретарь
(нормоконтролер)
_____ О.Н. Иванова

“__”_____ 2017 г.

Челябинск-2017

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность темы

В рамках данной работы рассматривается разработка REST-сервиса информирования клиентов о деятельности охранного предприятия. На сегодняшний день REST является наиболее распространенным подходом к построению API, с которыми взаимодействует множество различных устройств. Актуальность построения REST API, с которым могут взаимодействовать приложения на устройствах клиентов охранной организации, обусловлена потребностями заказчика – одного из охранных предприятий Челябинска.

Цель и задачи исследования

Целью разработки является создание API, доступного для различных клиентских приложений. К основным задачам, служащим достижению этой цели, относятся:

1) изучение и анализ методов построения публичных API в стиле REST;
2) изучение предметной области в объеме, необходимом для проектирования системы;
3) описание требований к разрабатываемому сервису;
4) разработка архитектуры системы в целом и ее отдельных компонентов;
5) реализация сервиса:
   - реализация базы данных;
   - реализация сбора, очистки, интеграции и редукции данных из различных источников;
   - реализация механизмов немедленного оповещения и обратной связи;
   - реализация API;
6) тестирование сервиса;
7) внедрение сервиса.

Объем и структура работы

Общий объем работы составляет 35 стр., основная часть работы содержит 5 глав. Работа включает в себя 3 приложения, объем библиографии составляет 31 источник, объем приложений – 12 страниц.

Краткое содержание работы

В главе «Анализ предметной области» приведены основные понятия и термины, связанные с предметной областью, а также определена роль разрабатываемой системы в деятельности предприятия.

В главе «Проектирование» выявлены ключевые требования к системе, описана архитектура базы данных и всей системы в целом.

В главе «Реализация» подробно рассматриваются используемые средства и приемы, а также необходимое для работы системы окружение.

В главе «Тестирование» описаны различные методы тестирования и отладки, используемые в ходе разработки.

В главе «Внедрение и сопровождение» описан способ разворачивания разрабатываемой системы на рабочей станции заказчика.

В заключении подведены итоги работы, определены достигнутые результаты.

Приложение 1 отражает содержание тестовой базы данных, используемой в процессе разработки и тестирования системы.

В приложении 2 приведена спецификация API разрабатываемого REST-сервиса.

В приложении 3 приведены спецификации вариантов использования разрабатываемой системы.

На рис. 1 представлены три основных бизнес-процесса, протекающих при участии разрабатываемой системы.

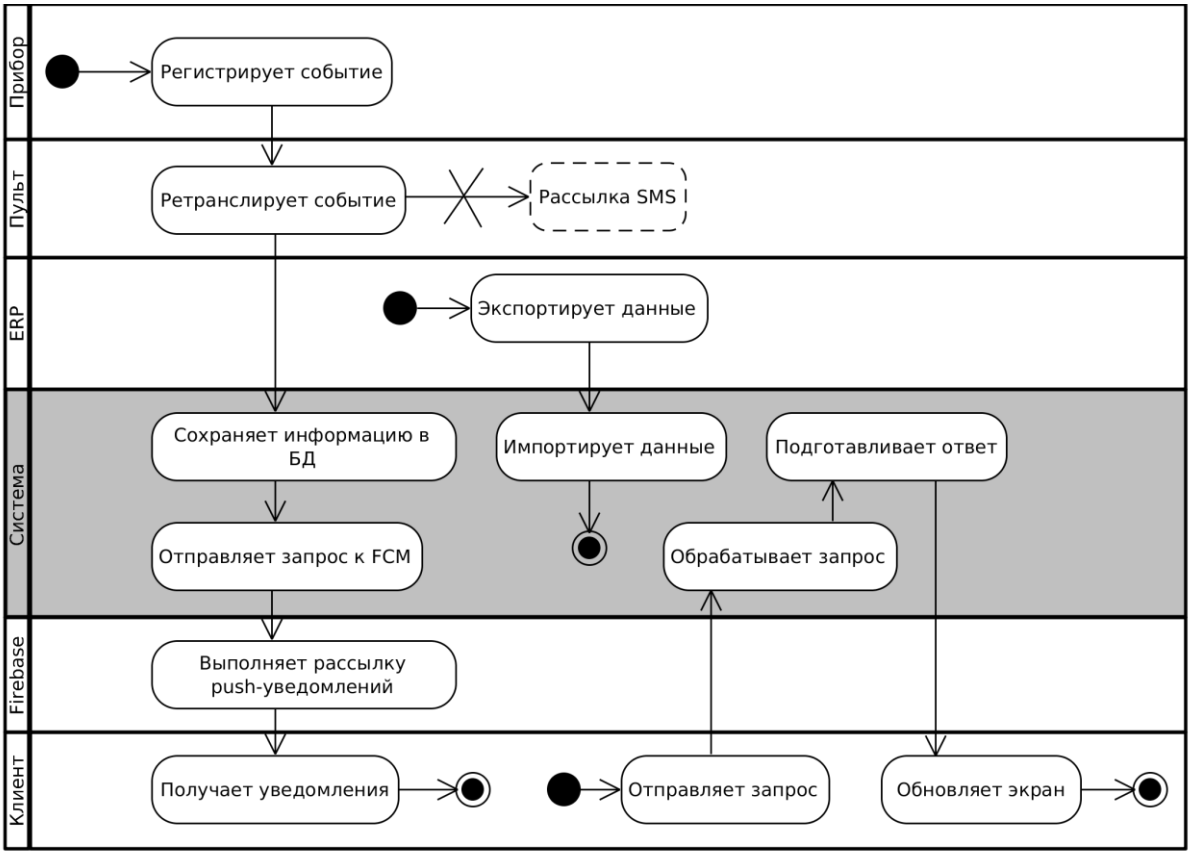

Рис. 1. Бизнес-процессы с участием разрабатываемой системы

Серым цветом на рисунке выделена область ответственности разрабатываемой системы.

2. ПРОЕКТИРОВАНИЕ

2.1. Определение требований к разрабатываемой системе

При анализе требований к разрабатываемому приложению была спроектирована диаграмма вариантов использования, представленная на рис. 2.

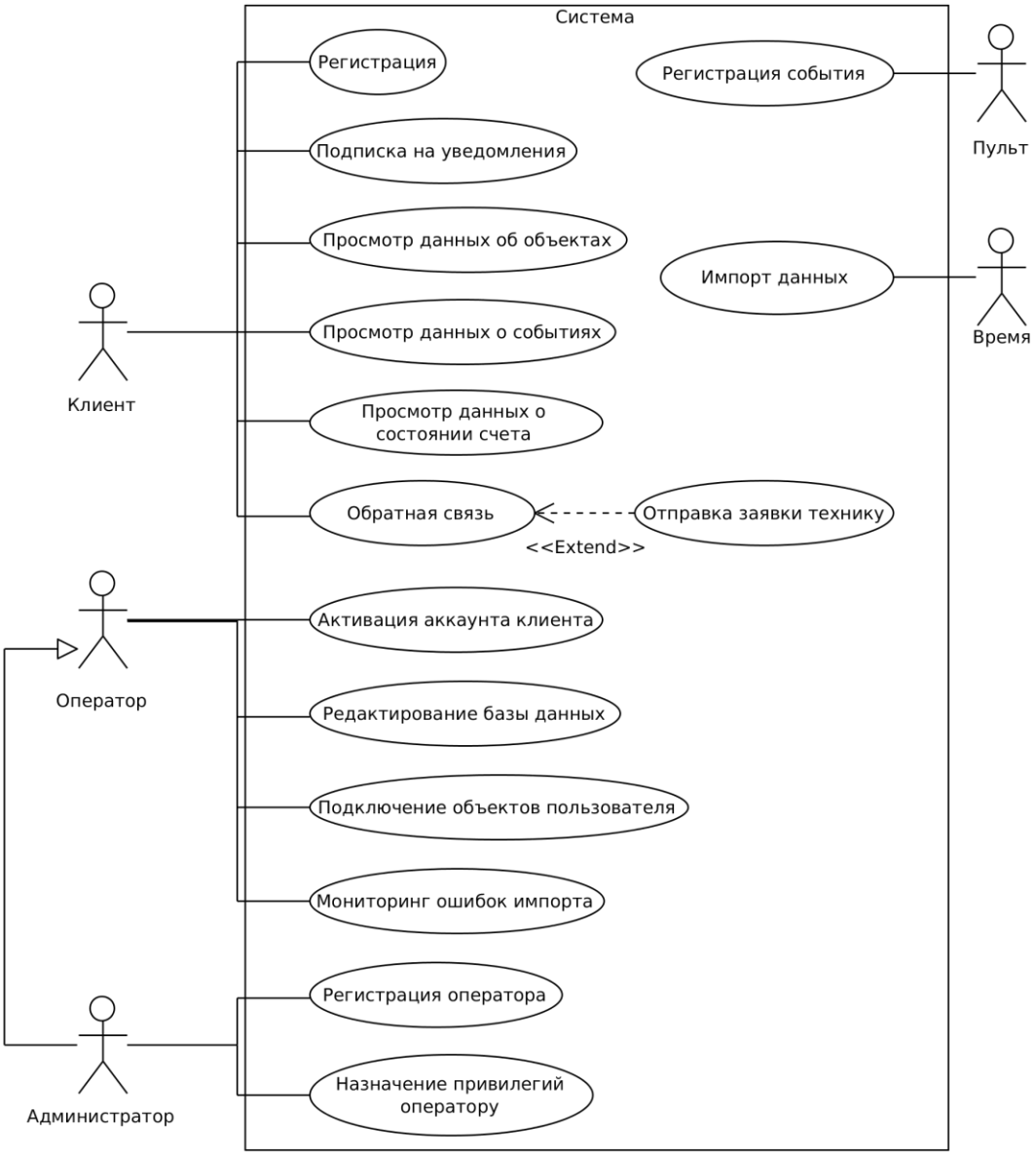

Рис. 2. Диаграмма вариантов использования

Как показано на диаграмме вариантов использования, с системой взаимодействуют пять актеров:

Клиент – программное обеспечение (клиентское приложение), взаимодействующее с API разрабатываемой системы.

Оператор – авторизованный пользователь, осуществляющий взаимодействие с системой при помощи веб-интерфейса и обладающий правами на редактирование/создание/удаление некоторых видов записей базе данных.

Администратор – авторизованный пользователь, осуществляющий взаимодействие с системой при помощи веб-интерфейса и обладающий всеми правами, включая возможность назначать операторов и администраторов.

Пульт – программное обеспечение, настроенное на ретрансляцию событий разрабатываемой системе.

Время – актер, обозначающий сторонний планировщик, запускающий импорт данных по расписанию.

В приложении 3 приведены спецификации ключевых вариантов использования системы.

2.2. Проектирование архитектуры системы

При анализе вариантов использования мною было выявлено, что, во-первых, пульт имеет свои собственные протоколы обмена информацией. Используемое пультовое ПО «Андромеда» [14] может передавать информацию разрабатываемой системе при помощи протокола Sur-Gard [29] и XML-подобного «открытого протокола». Во-вторых, при взаимодействии с клиентским приложением данные могут быть переданы в формате JSON. В-третьих, оператор системы может взаимодействовать с системой посредством веб-интерфейса.

Исходя из этого, логичным решением будет разделить систему на относительно независимые компоненты, обращающиеся к общей базе данных. Схема взаимодействия компонентов изображена на рис. 3.

Рассмотрим подробнее каждый из пяти компонентов, разрабатываемых мною (REST-сервис, административный сайт, подсистема импорта данных, подсистема отправки push-уведомлений и обработчик пакетов, присылаемых пультом), их форматы их входных и выходных данных.

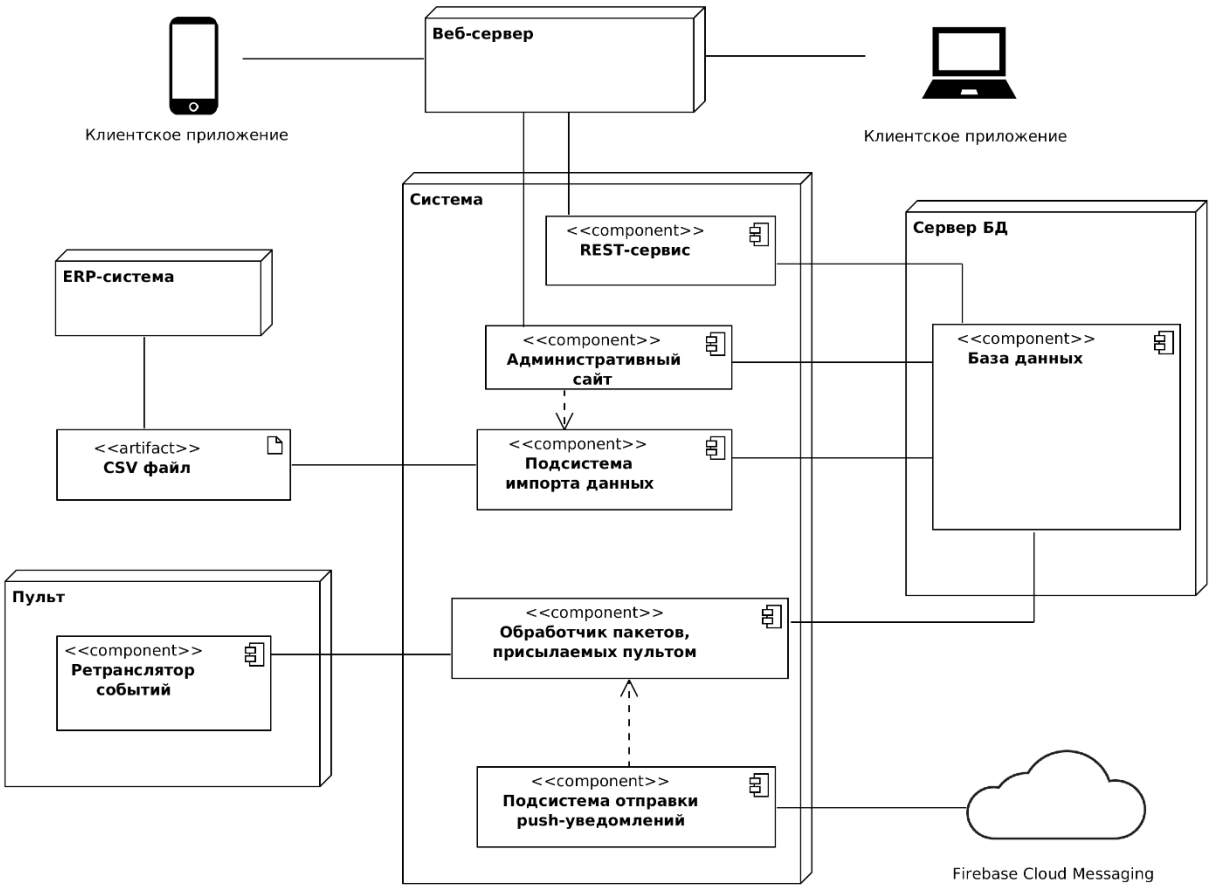

Рис. 3. Компоненты системы

REST-сервис

Данный компонент предназначен для взаимодействия преимущественно с клиентским (например, веб- или мобильным) приложением. В этом компоненте изолирована логика отображения ресурсов [13], отражающих состояние объектов базы данных. Совокупность ресурсов формирует API, который и предоставляет REST-сервис. Данный компонент обеспечивает также разграничения доступа пользователей клиентского приложения к ресурсам.

Формат входных данных: HTTP-запрос на определенный URL.

Формат выходных данных: код ответа HTTP, тело ответа в формате JSON [10].

Административный сайт

Данный компонент предназначен для персонала фирмы. Он позволяет редактировать некоторые записи в базе данных напрямую, осуществлять

импорт данных посредством веб-интерфейса. Содержит ряд страниц с инструкциями по работе.

Формат входных данных: HTTP-запрос, генерируемый браузером пользователя.

Формат выходных данных: веб-страница.

Подсистема импорта данных

Этот компонент содержит ряд функций, преобразующих текстовые файлы в записи базы данных. Имеет консольный интерфейс для обеспечения пакетной автоматизированной обработки файлов, но также может использоваться и через административный сайт.

Формат входных данных: файл в формате CSV [9].

Формат выходных данных: зависит от способа запуска импорта. Может быть как сообщением в консоли о числе успешно импортированных строк, так и веб-страницей.

Обработчик пакетов, присылаемых пультом

Данный компонент преобразует пакеты, формируемые пультом, использующим протоколы передачи данных на базе TCP [31], в записи базы данных. Содержит логику смены состояний охраняемого объекта, происходящей при регистрации нового события.

Форматы входных данных:

- Sur-Gard – строковое представление, состоящее из кодов событий, объектов, спецсимволов и т.п. Подробнее данный формат будет рассмотрен в разделе 3.3.2.
- «Открытый протокол» – разновидность XML, структура которого определена производителем пультового программного обеспечения.

Формат выходных данных: подтверждение получения данных, определенное в спецификации соответствующего протокола.

Подсистема отправки push-уведомлений

После регистрации нового события компонент находит все устройства пользователей, интересы которых это событие затрагивает, после чего

рассылает на них соответствующие push-уведомления при помощи сервиса Firebase Cloud Messaging.

Формат входных данных: отсутствует.

Формат выходных данных: POST-запрос, отправляемый на сервер Firebase.

2.3. Проектирование базы данных

При анализе предметной области мною была спроектирована база данных разрабатываемого приложения, которая приведена на рис. 4. Из соображений безопасности работы охранного предприятия часть таблиц не была приведена. Эти таблицы содержат информацию для авторизации пользователей.

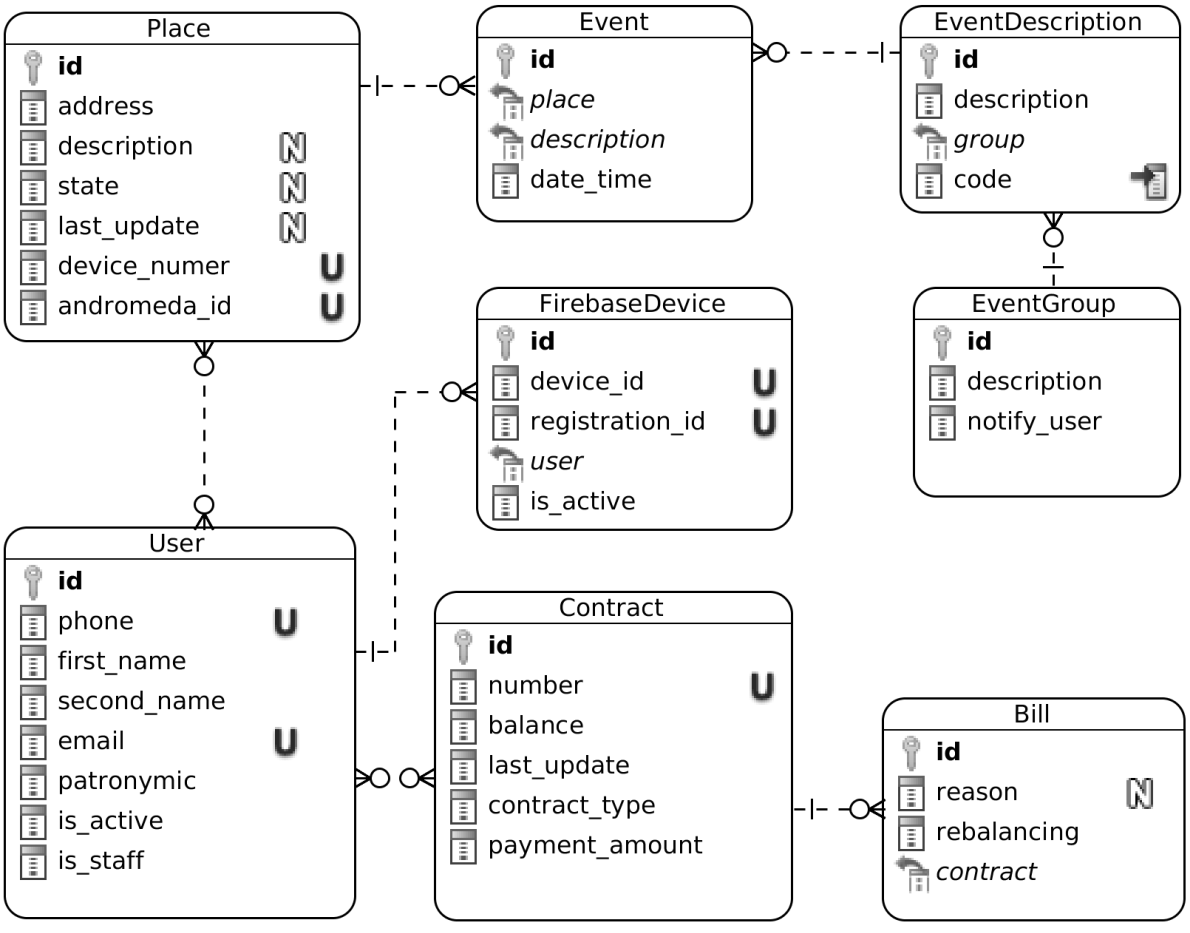

Рис. 4. Схема базы данных

Рассмотрим подробнее каждую из приведенных таблиц. Заметим, поле *id* является первичным ключом в тех таблицах, где оно присутствует.

Пакет *api*

Данный пакет содержит реализацию RESTful API на базе Tastypie (Python, SQL, 1115 строк) и имеет вложенные пакеты.

Далее приведена структура вложенного пакета *api/resources/*, содержащего классы-ресурсы Tastypie (Python, 740 строк):

- contract_resource.py – API, предоставляющий доступ к информации о договорах (Python, 35 строк);
- place_resource.py – API, предоставляющий доступ к информации об объектах (Python, 175 строк);
- feedback_resource.py – API, позволяющий оставлять отзывы и заявки технику (Python, 170 строк);
- user_resource.py – API, реализующий регистрацию, создание и удаление сессий (Python, 320 строк);
- system_resource.py – модуль, содержащий базовый класс для ресурсов, не привязанных к моделям данных (Python, 40 строк);
- custom_model_resource.py – модуль, содержащий базовый класс для ресурсов, основанных на моделях данных Django, использующих юникод (Python, 25 строк);
- В другом вложенном пакете, api/helpers/, находятся вспомогательные классы и функции, необходимые для работы API (Python, SQL, 375 строк):
- auth.py – реализация механизма аутентификации (Python, 30 строк);
- email_helper.py – функции формирования и рассылки электронных писем (Python, 95 строк);
- group_by_day.py – группировка и подсчёт записей таблицы по дням – запрос, который не может быть реализован средствами Django ORM (Python, SQL, 70 строк);
- money_mixin.py – класс, обеспечивающий поддержку полей денежного типа PostgreSQL в ресурасах (Python, 30 строк);

- payment_logic.py – реализация бизнес-логики вычисления даты следующего платежа (Python, 15 строк);
- resource_decorators.py – повторяющиеся проверки вынесены в декораторы, которые применяются к методам классов-ресурсов (Python, 40 строк);
- timezone_serializer.py – правила сериализации времени и даты с указанием часового пояса (Python, 10 строк);
- validators.py – правила проверки вводимых при регистрации значений (Python, 60 строк).

Модули верхнего уровня

Приведем структуру остальных модулей приложения main_app:

- views.py – модуль, который содержит представления, переопределяющие отображение страниц, используемых в процедуре восстановления пароля (Python, 25 строк);
- models/ – модуль, содержащий модели данных Django, с которыми работает API и все остальные классы и приложения проекта (Python, 350 строк):
- constants.py – описание констант и перечислений, используемых в моделях (Python, 20 строк);
- events.py – модели, связанные с обработкой событий (Python, 155 строк);
- places.py – модели, связанные с охраняемыми объектами (Python, 55 строк):
- users.py – модели, связанные с пользователями, договорами и устройствами (Python, 120 строк);
- admin.py – конфигурация административного сайта, относящаяся к моделям данного приложения (Python, 210 строк);
- templates/ – шаблоны веб-страниц (HTML, 360 строк):
- admin/ – шаблоны для административного сайта (HTML, 320 строк);

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы были достигнуты следующие результаты:

1) получены навыки построения публичных API в стиле REST с использованием фреймворка Django-Tastypie.

2) проведен анализ требований к разрабатываемому сервису, учитывающий специфику предметной области;

3) спроектирована архитектура программной системы, продумана ее интеграция с инфраструктурой охранной организации;

4) спроектированы и реализованы компоненты системы:

- база данных;
- REST API;
- административный сайт;
- средства интеграции с пультовым ПО «Андромеда»;
- средства импорта данных из ERP-системы;

5) реализованы механизмы немедленного оповещения пользователей и обратной связи с охранной организацией;

6) работа системы протестирована;

7) получен опыт работы со средством контейнеризации приложений Docker;

8) система внедрена в промышленную эксплуатацию в одной из охранных организаций Челябинска;

9) разработана пользовательская документация для персонала охранной организации.

В рамках данной работы была подготовлена статья [28], принятая к публикации в сборнике «Наука ЮУрГУ».

27. Радченко Г.И. Распределенные вычислительные системы. – Челябинск, ЮУрГУ, 2012. – 182 с.

28. Силкина Н.С., Федосеев Г.И. Опыт разработки REST-сервиса на примере системы информирования клиентов охранного предприятия. // Наука ЮУрГУ: материалы 69-й научной конференции. – Челябинск: Издательский центр ЮУрГУ, 2017. [В печати].

29. Спецификация протокола Sur-Gard. [Электронный ресурс]. URL: http://www.lavina-pult.ru/helpdesk/integraciya_SurGard.pdf (дата обращения: 13.02.2017).

30. Спецификация технологии Push. [Электронный ресурс] URL: https://www.w3.org/TR/push-api/ (дата обращения: 13.02.2017).

31. Таненбаум Э., Уэзеролл Д. Компьютерные сети. 5-е изд. – СПб.: Питер, 2012. – 960 с.

Табл. 4. EventDescription

id	description	group	code
1	Постановка на охрану	1	4534
2	Автоматическая постановка	1	4535
3	Снятие с охраны	2	5111
4	Тревога в зоне периметра	3	1001
5	Пожарная тревога	3	1050
6	Отключение питания	4	8896
7	Тестовый отчет	5	9000

Табл. 5. Bill

id	reason	contract	date	rebalancing
1	Пополнение счета	1	2016-09-11	1000.00
2	Абонентская плата	1	2016-10-01	-300.00
3	Установка оборудования	1	2016-10-02	-3000.00
4	Абонентская плата	2	2016-09-11	-300.00
5	Пополнение	2	2016-10-03	500.00
6	Пополнение	2	2016-10-04	600.00
7	Абонентская плата	3	2016-10-03	-1200.00
8	Перерасчет	3	2016-10-04	400.00
9	Ложный вызов	4	2016-10-02	-350.00

Табл. 6. PlaceUser

id	user	place	user_role
1	1	1	null
2	2	1	1
3	1	3	null
4	4	4	2
5	4	2	null

Табл. 7. Contract

id	number	balance	contract_type	payment_amount
1	1011-01	1000	1	200
2	1011-01П	4000	2	2000
3	1004-08	350	1	1500
4	1800-02	0	2	1500

Табл. 8. ContractUser

id	user	contract
1	1	1
2	2	2
3	1	3
4	4	4
5	2	3

Табл. 9. Event

id	place	description	date_time
1	1	4	2017-04-01 22:25:08
2	2	7	2017-04-02 20:00:01
3	2	6	2017-04-02 22:07:30
4	4	3	2017-04-02 23:05:03
5	2	2	2017-04-03 08:03:20
6	2	3	2017-04-03 13:15:33
7	4	1	2017-04-03 13:17:29
8	3	1	2017-04-03 13:18:11

Приложение 2

Спецификация API

Примеры API ресурсов, предоставляемых сервисом, представлены на рис. [1-7].

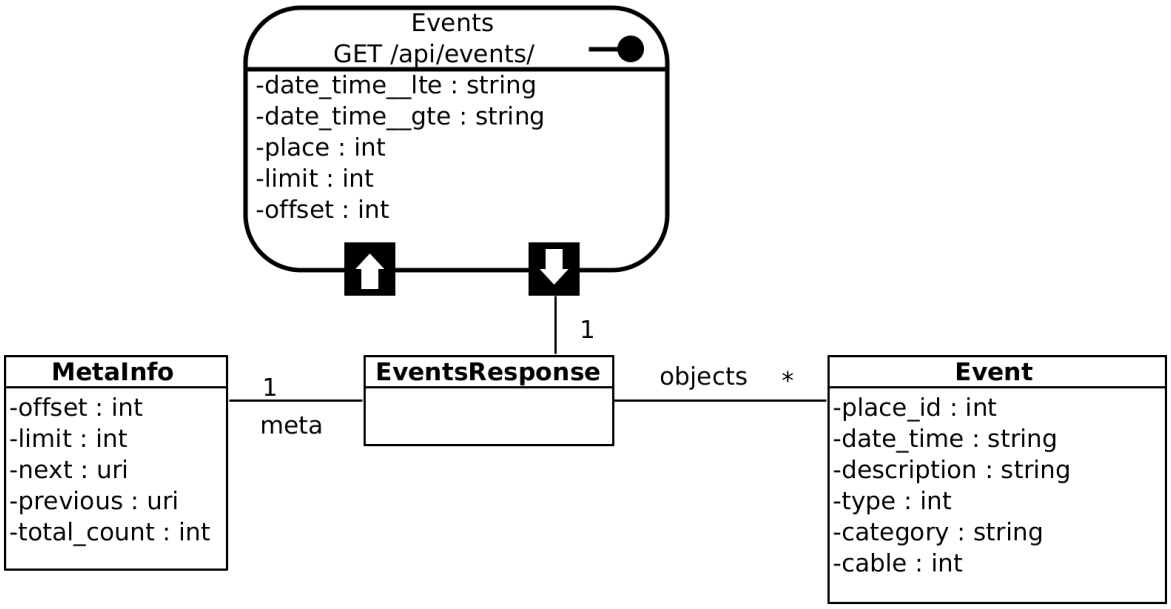

Рис. 1. API списка событий

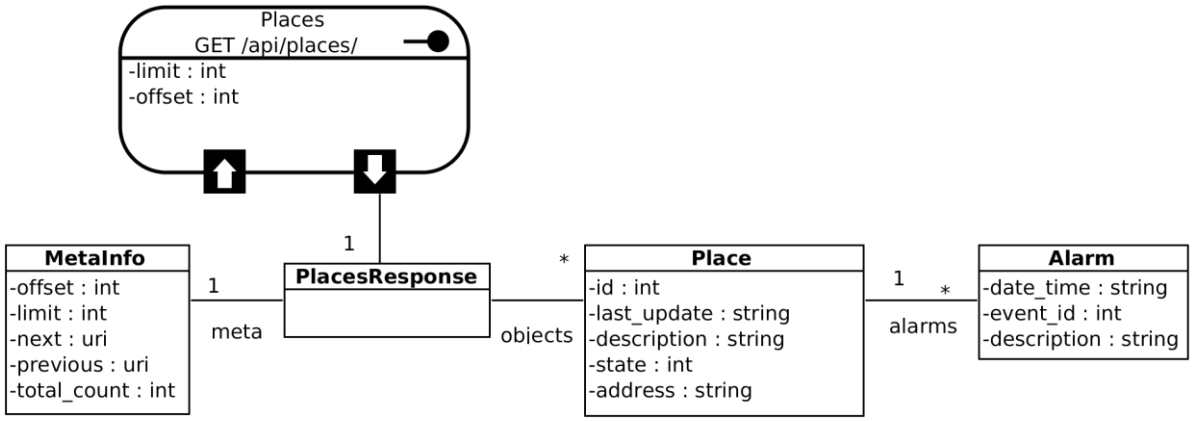

Рис. 2. API списка охраняемых объектов

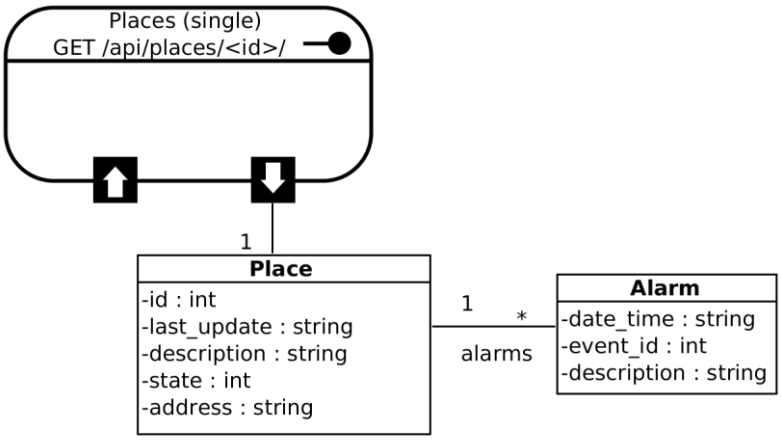

Рис. 3. API объекта

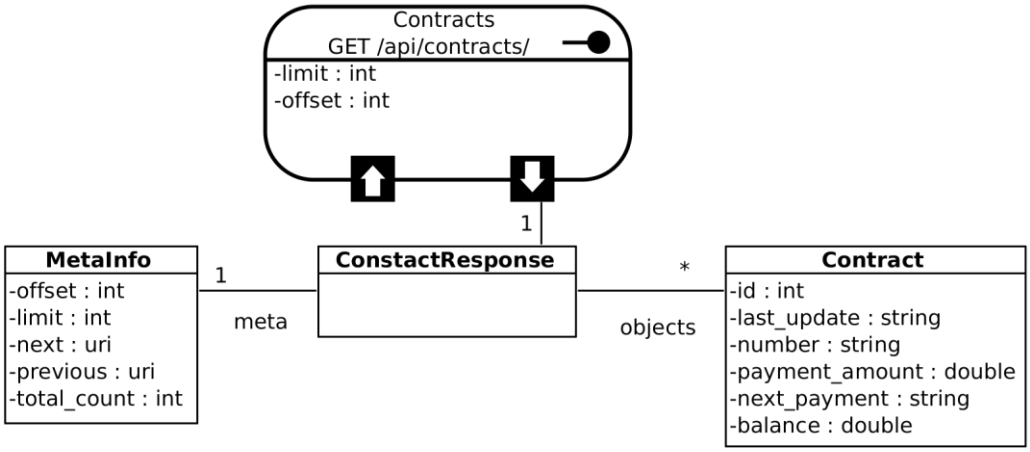

Рис. 4. API списка договоров

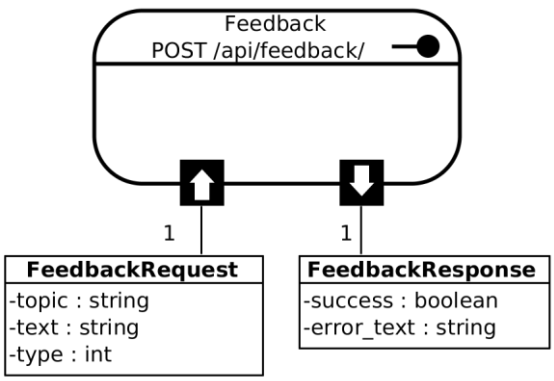

Рис. 5. API отправки отзыва

Табл. 23. Спецификация прецедента «Назначение привилегий оператору»

UseCase: Назначение привилегий оператору
Аннотация: Ограничение доступа оператора к тем или иным разделам административного сайта.
Главные актеры: Администратор.
Предусловия: Создана сессия.
Основной поток: 1. Администратор загружает веб-страницу раздела административного сайта с информацией о пользователях; 2. Администратор заполняет форму с правами оператора; 3. Изменения заносятся в базу данных.
Альтернативные потоки: При несоответствии данных формы ограничениям целостности выводится сообщение об ошибке.