МИНОБРНАУКИ РОССИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”

Факультет компьютерных наук

Кафедра информационных систем и сетевых технологий

Проведение концертов на различных площадках и продажа электронных билетов на эти концерты.

Курсовая работа

по дисциплине

Технологии программирования

09.03.02. Информационные системы и технологии

Информационные системы и сетевые технологии

Преподаватель \_\_\_\_\_\_\_ В. С. Тарасов, ассистент

Обучающиеся \_\_\_\_\_\_\_ Д. Н. Трофимов, Н. А. Гуляев, 3 курс, д/о

Руководитель \_\_\_\_\_\_\_\_ И. Ю. Иванов, ассистент

Воронеж 2019

Оглавление

Оглавление 2

Введение 3

Постановка задачи 4

Анализ предметной области 7

Проектирование 9

Use-case диаграмма 9

Диаграмма классов 10

ER-диаграмма 11

Диаграммы последовательности 12

Диаграммы взаимодействия 13

Диаграмма объектов 15

Диаграмма деятельности 16

Тестирование 17

Swagger 18

Siege нагрузочное тестирование 19

Реализация 22

Главная страница 22

Просмотр информации о событии 22

Покупка билета 23

Купленный билет и его печать 23

Возврат билета 24

Создание контракта 24

Добавление билетов 25

Работающее приложение можно посмотреть по ссылке 25

Заключение 26

Отчет по команде: 26

Введение

Поход на концерт — праздник для любого человека. Хочется получать положительные эмоции с самого начала. То есть с выбора события и с покупки билета на него. Поэтому мы разработали веб-приложение именно для этих целей. Но мы пошли дальше и решили облегчить жизнь менеджерам, составляющим концерты. Им будет предоставлена возможность самостоятельно создавать и изменять события. Во время создания расчет стоимости билетов происходит на основе данных о затратах и прибыли, которую планируется получить. Также составителем указывается ожидаемый процент купленных билетов, для того чтобы уменьшить риск не окупить проект.

В нашем городе действует всего 2 сайта по продаже билетов: afisha.yandex и vrn.kassir. Эти площадки не предоставляют никакого функционала для организаторов концертов, поэтому менеджерам будет удобно пользоваться нашим веб-приложением. Также мы постараемся сделать удобный и интересный дизайн, чтобы привлечь покупателей. Комиссия при покупке билетов на существующих сайтах составляет 10%. Так как у нас не будет физических касс и штат сотрудников не такой большой, то есть не нужны лишние затраты на аренду и зарплату, комиссия может упасть на 2-4%. Впоследствии планируется создание системы «постоянный покупатель» с личными акционными купонами.

Постановка задачи

1. Должно быть разработано клиент-серверное приложение, предназначенное для людей, составляющих концерты и для тех, кто покупает билеты на них. Первым оно поможет не забыть ничего важного при планировании концерта и составит договор. Для вторых оно будет удобной площадкой для просмотра предстоящих событий и покупки билетов на них.

2. Начало работы — февраль 2019, окончание — июнь 2019

3. Организация концертов — трудная и многогранная задача: договориться с артистом, выбрать подходящее место для проведения, найти площадки для распространения, рассчитать стоимость билетов и так далее. Наше приложение предназначено для автоматизации части работы менеджера — волокиты с площадкой для продажи и реализации билетов.

4. Потенциальными пользователями нашего приложения являются менеджеры, которые вместо личной встречи с представителями сайтов по продаже билетов и других организаций, предпочитают в любое время создавать и редактировать предстоящие события и простые покупатели.

5. Целью программы является автоматизация процесса создания музыкального события и процесса размещение его на площадке для продажи билетов.

6. Функции продукта

В приложении может быть два типа пользователей:

Менеджер может создавать событие, редактировать его и удалять (до публикации). При создании события указываются:

* Место (город, адрес), время и дата проведения;
* Тип площадки (крытая, открытая, стадион);
* В зависимости от типа площадки указываются: типы мест (танцпартер, сидячие места), их количество и ожидаемый процент купленных билетов;
* Общий расход на проведение концерта и ожидаемая прибыль;

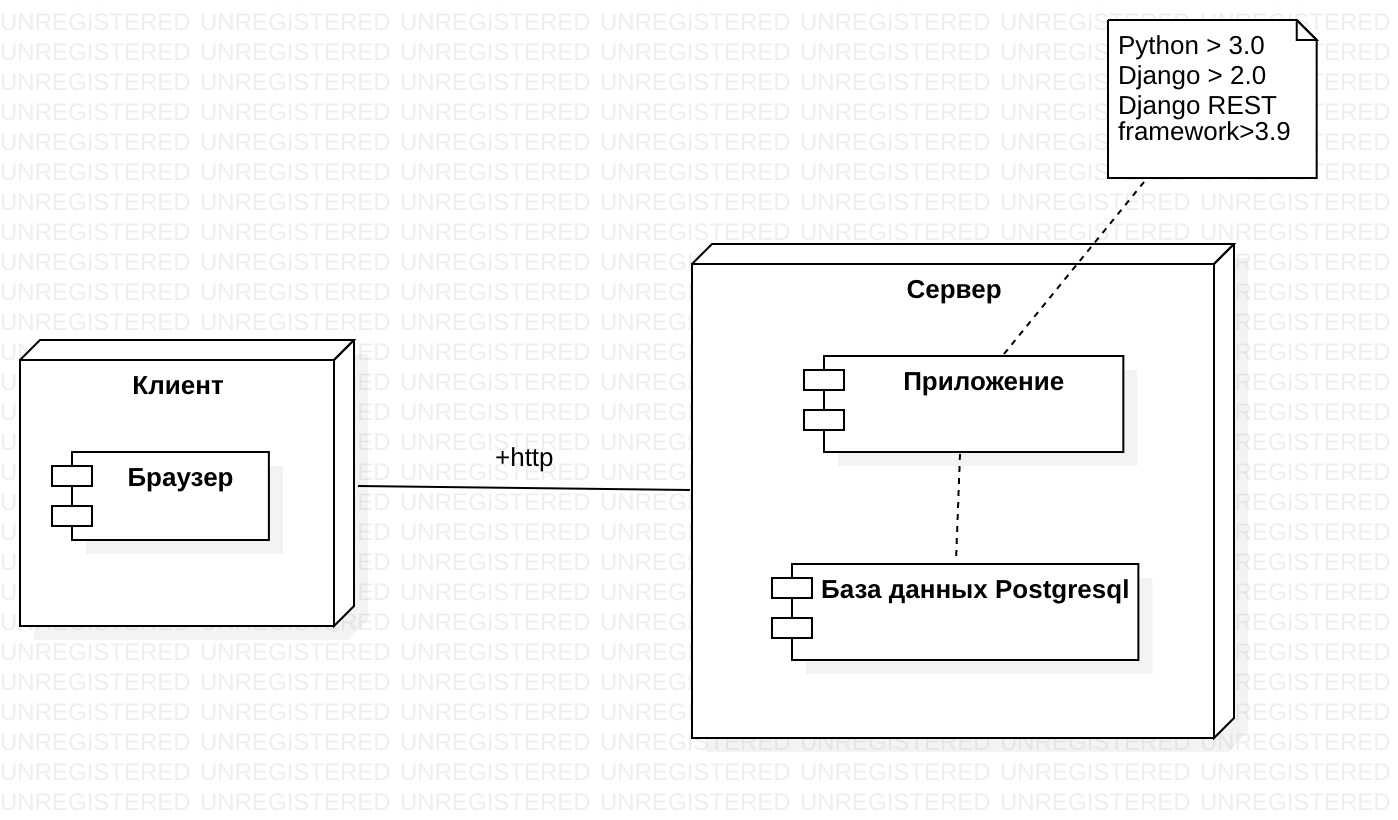
Приложение рассчитывает стоимость билетов на основе введенных данных и создает контракт.

Клиент может:

* Получить информацию о: месте и времени проведения события, стоимости билетов, условиях возврата билетов;
* Купить билеты;
* Вернуть билеты.

Для покупки клиенту надо будет ввести номер телефона и адрес электронной почты. Приложение предложит ему распечатать электронную версию билета.

7. Для бэкенда приложения требуется сервер на операционной системе Linux с предустановленным интерпретатором python версии 3.6.0 или старше с пакетным менеджером pip 18.0 или старше, а также база данных postgresql версии 10 или старше; для интерпретатора python требуется установка фреймворков Django версии 2.0 или старше, Django REST фреймворк (версия зависит от версии Django).



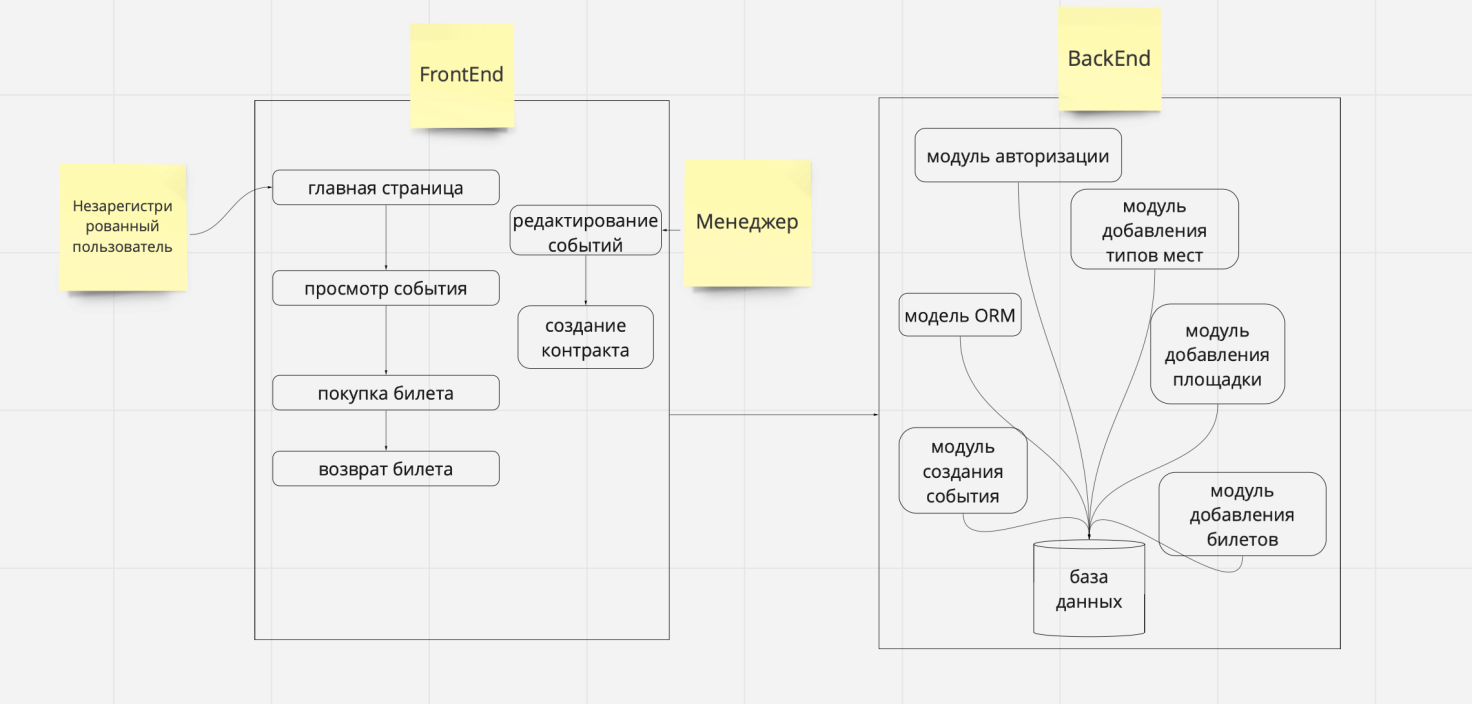
База данных postgresql выбрана, потому что это свободное программное обеспечение для коммерческих проектов, способное обрабатывать огромное количество данных, стремящееся соответствовать стандарту ANSI-SQL:2008, отвечающее требованиям ACID (атомарность, согласованность, изолированность и надежность) и известное своей ссылочной и транзакционной целостностью.  
 Фреймворк Django был выбран, потому что он относительно быстро разворачивается, легко масштабируется, является "живым" фреймворком (регулярно обновляется и поддерживается разработчиками и комьюнити), поддерживается ORM с Postgresql "из коробки" (сразу, после установления фреймворка).

8. Для использования системы требуется персональный компьютер или мобильное устройство с браузером и доступом в интернет.

Приложение будет корректно работать на персональных компьютерах в браузерах: Mozilla Firefox, Safari, Google Chrome, Microsoft Edge.

Приложение будет корректно работать в мобильных браузерах: Safari и Google Chrome.

Максимальное число запросов в секунду - 100.

9. Модульная схема

Анализ предметной области

Планируется разработка автоматизированной системы для описанного проекта.

1. Процесс начинается с регистрации менеджера в системе в качестве которого могут выступать физические и юридические лица. Далее он создает событие и заполняет данные о нем. После подтверждения билеты поступают в продажу. Первым этапом, после регистрации менеджера, является создание события. Пользователь должен ввести информацию о концерте. В нее входят:

* Название концерта;
* дата и время проведения;
* затраченные средства;
* ожидаемая прибыль;
* площадка для проведения.

Менеджером создается новая площадка или выбирается одна из уже существующих. Ему требуется указать типы мест на ней, их количество и коэффициент. После этого менеджер может выбрать ожидаемый процент купленных билетов для каждого типа мест. На основе введенных данных происходит расчет стоимости билетов. После этого менеджер сохраняет введенные данные. Создается событие на основной странице. Чтобы билеты появились в продаже, их надо добавить к определенному контракту во вкладке «Билеты» или выбрать его во вкладке «Контракты» и добавить билеты там. Также менеджер может добавить площадки и типы мест для площадок для использования их в других контрактах.

2. Незарегистрированный пользователь может посмотреть информацию о событиях, купить и вернуть билеты. Для покупки билета клиент на главной странице должен выбрать интересующее его событие, тип места и нажать «купить билет». После этого ему предстоит заполнить форму с электронным адресом и мобильным телефоном. После заполнения появится билет с уникальным номером и QR-кодом, который можно распечатать. Для возврата билета пользователь должен ввести код билета и электронный адрес, который использовался при покупке. После этого он снова станет доступен для других клиентов.

Выделим базовые сущности данной предметной области, которые образуют структуру проектируемой ИС:

1. Аккаунт пользователя. Атрибуты — идентификационный номер пользователя, ФИО, имя аккаунта, пароль, тип пользователя. Так как в будущем планируется развитие системы, добавим поле «тип пользователя» в сущность «Аккаунт пользователя», чтобы легко можно было добавить регистрацию пользователей.
2. Контракт. Атрибуты — идентификационный номер контракта, дата создания, дата последнего изменения, ожидаемая прибыль, затраты, название события.
3. Площадка для проведения. Атрибуты — идентификационный номер площадки, название, адрес, дата.
4. Тип мета. Атрибуты — идентификационный номер, вместимость, название.
5. Билет. Атрибуты — идентификационный номер, стоимость, электронный адрес и мобильный телефон покупателя.

Функциональные возможности:

* ведение БД (запись, чтение, модификация, удаление в архив);
* обеспечение логической непротиворечивости БД;
* обеспечение защиты данных от несанкционированного или случайного доступа (определение прав доступа).

Проектирование

Use-case диаграмма

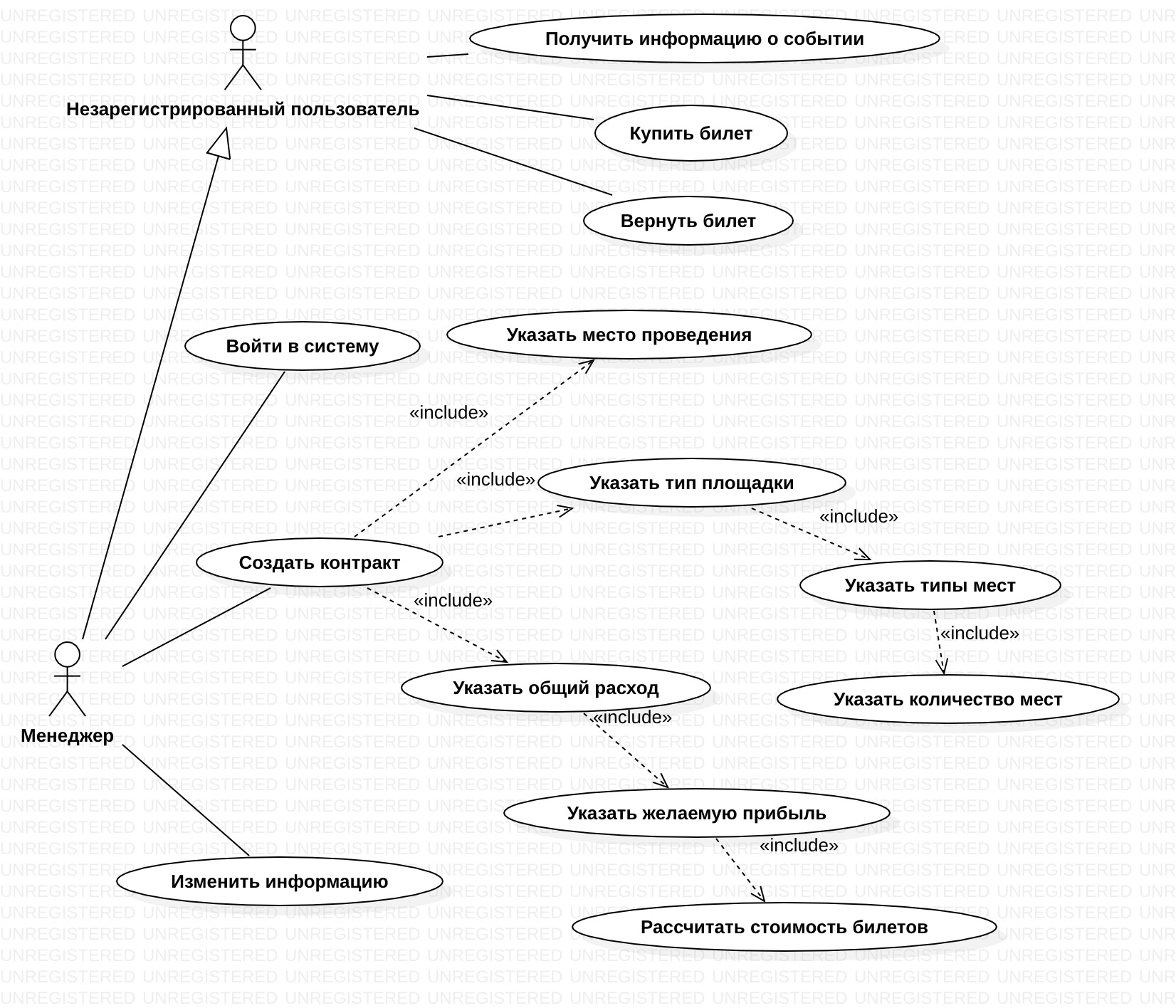
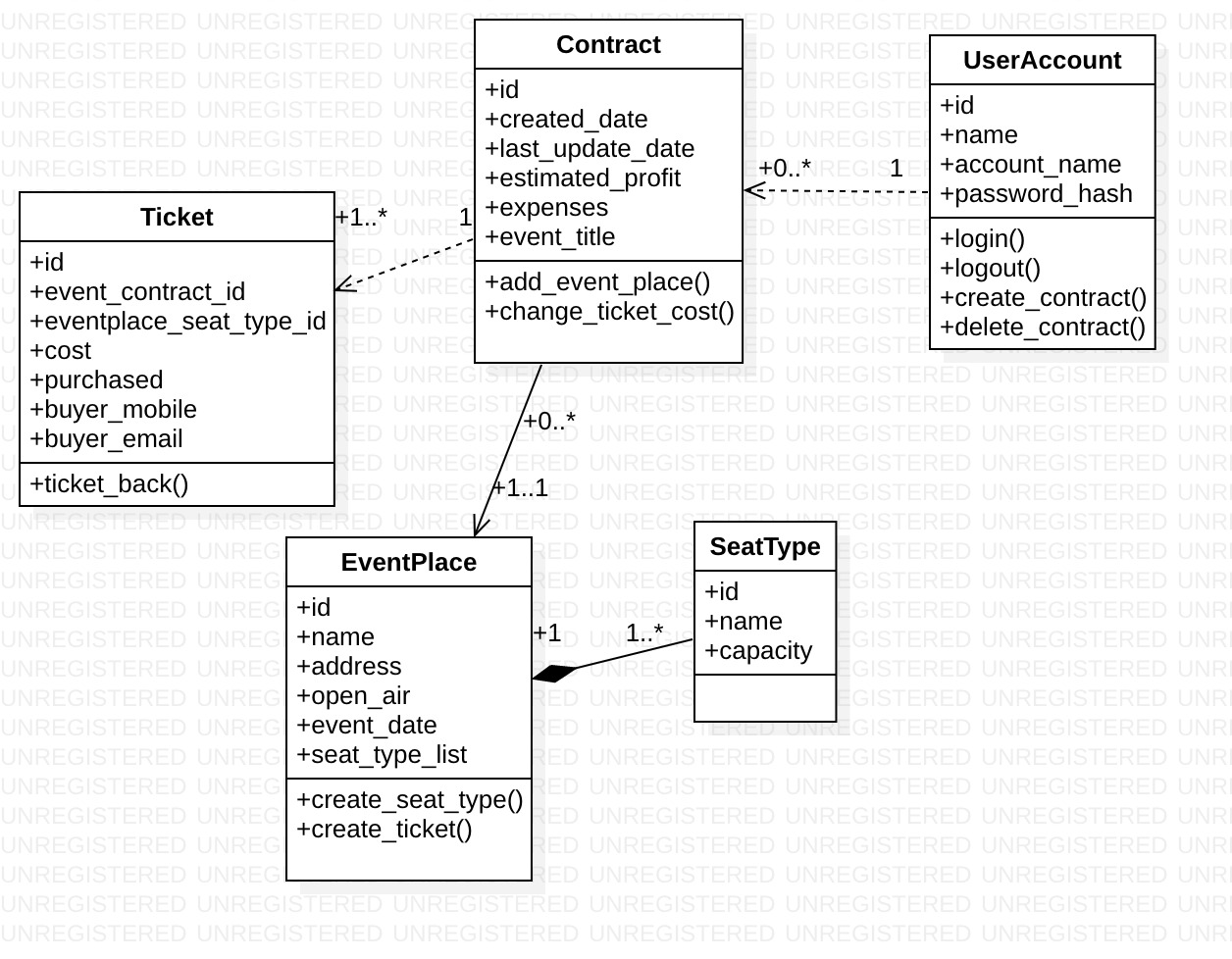
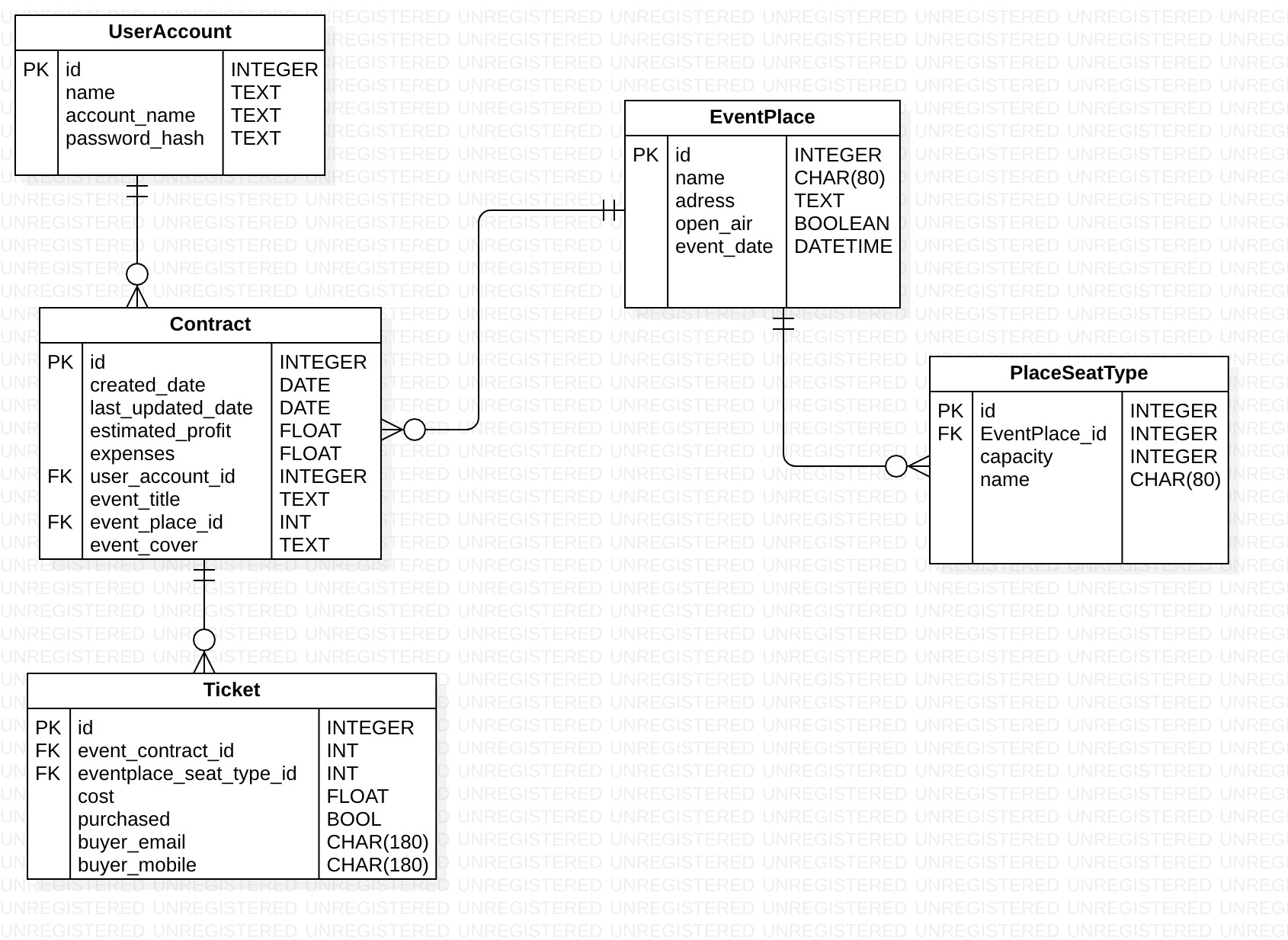


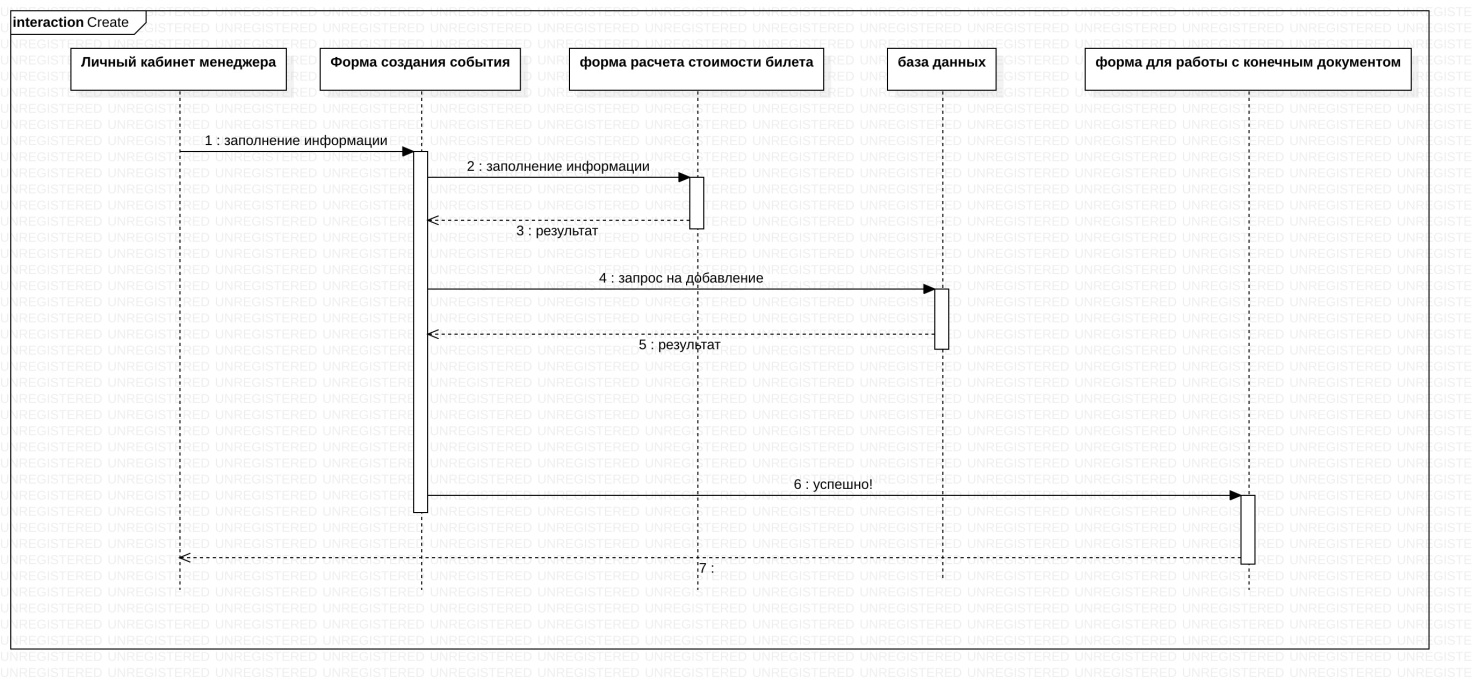
Диаграмма классов

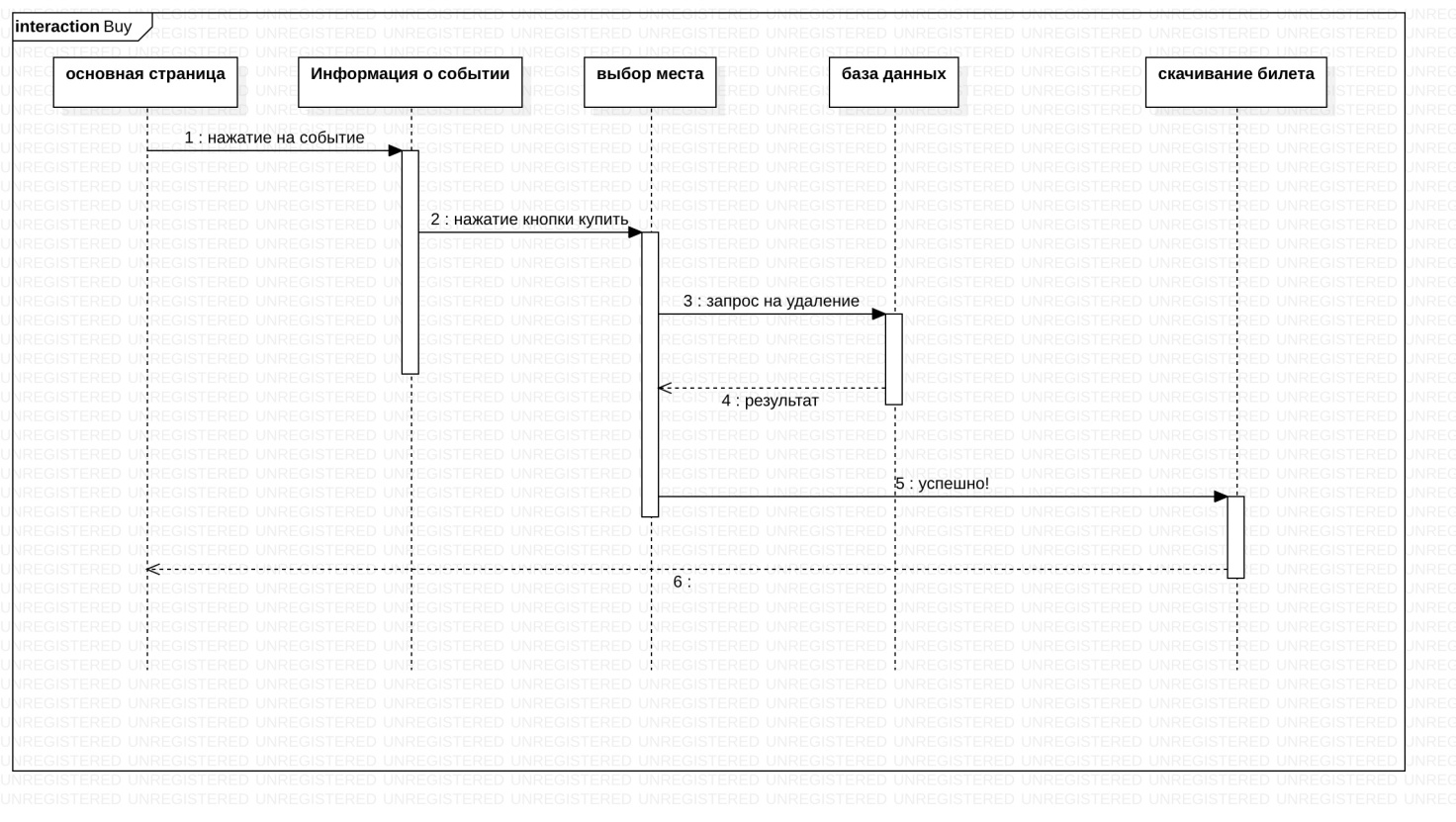


ER-диаграмма

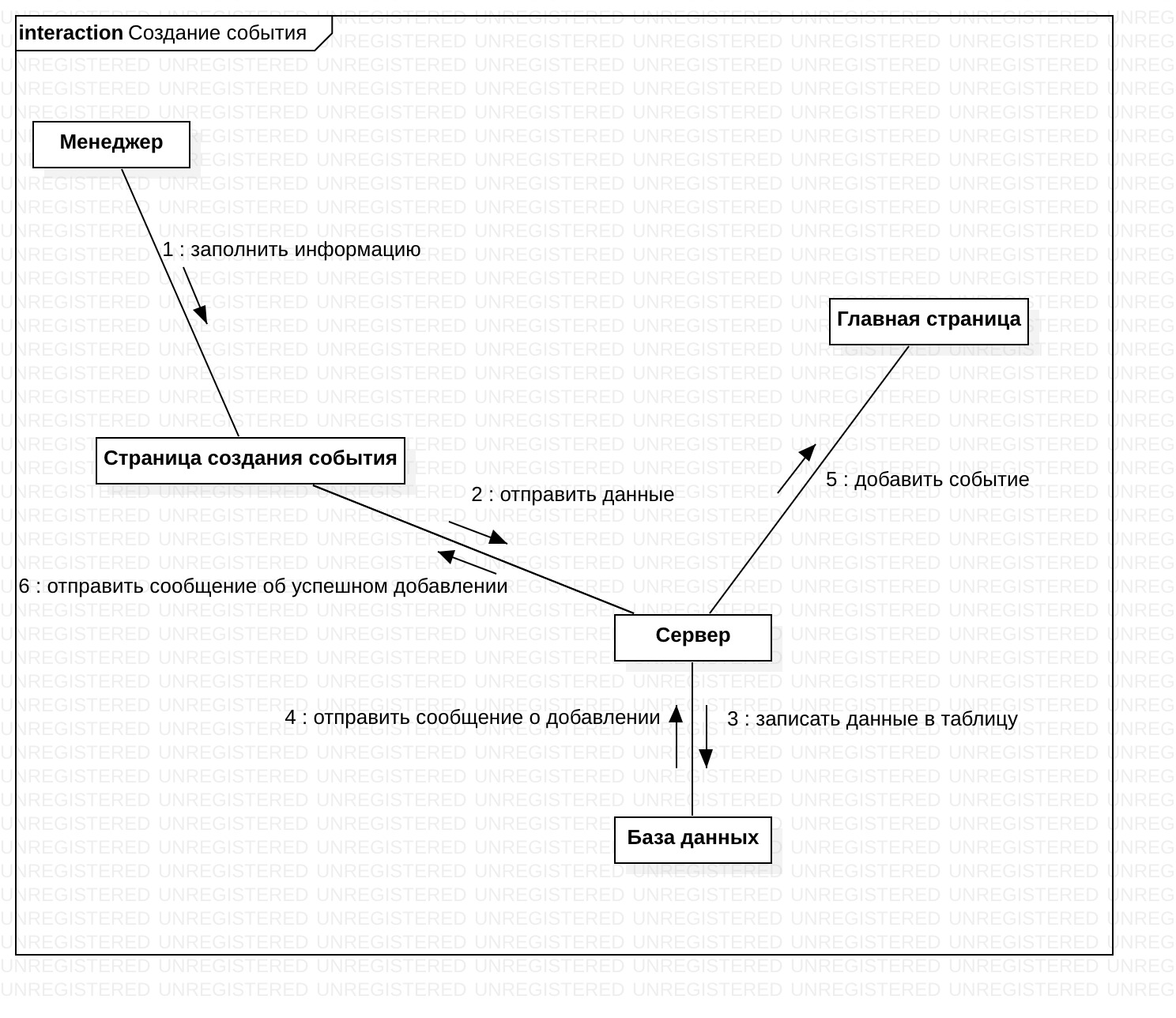


Диаграммы последовательности





Диаграммы взаимодействия



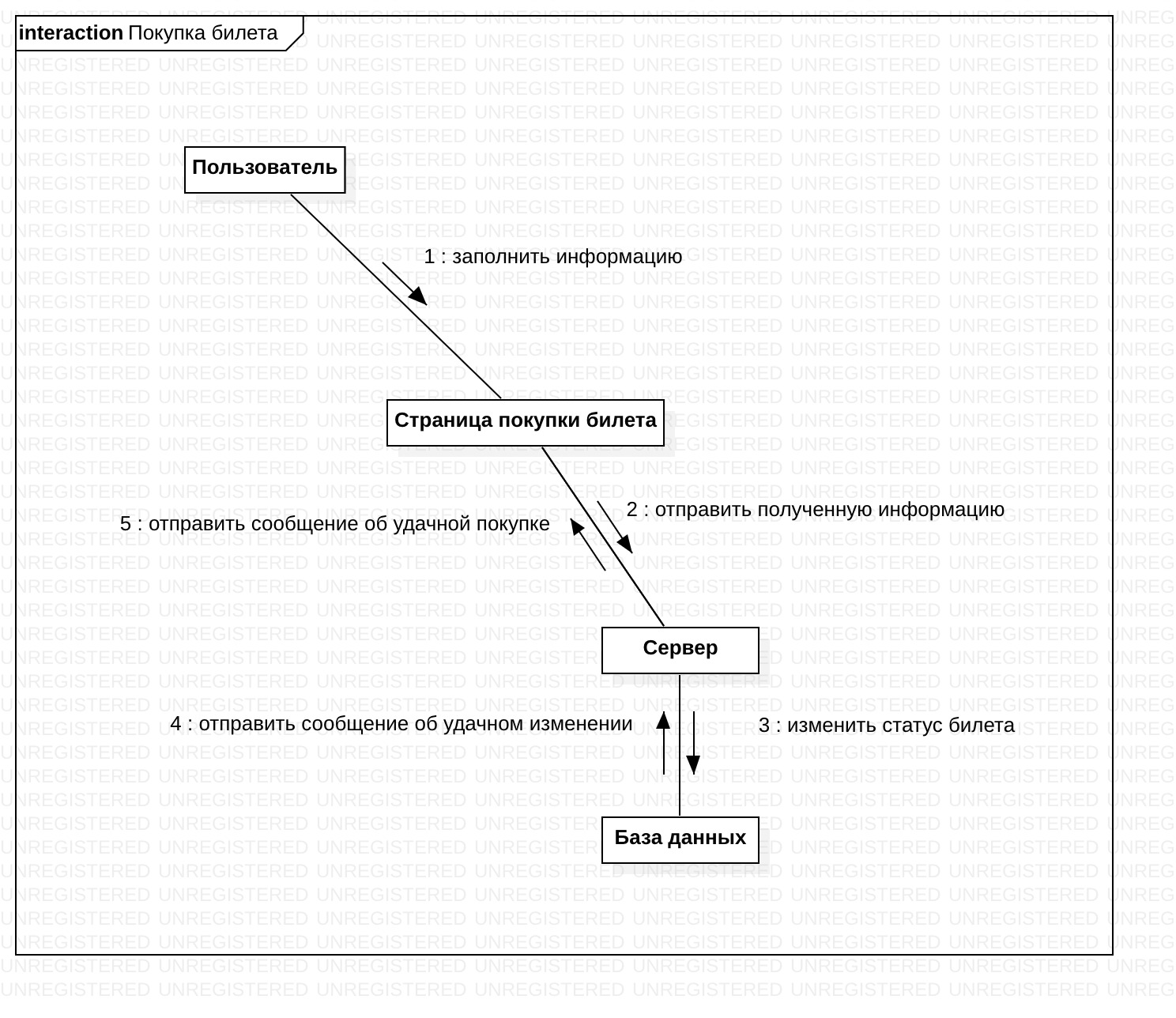


Диаграмма объектов

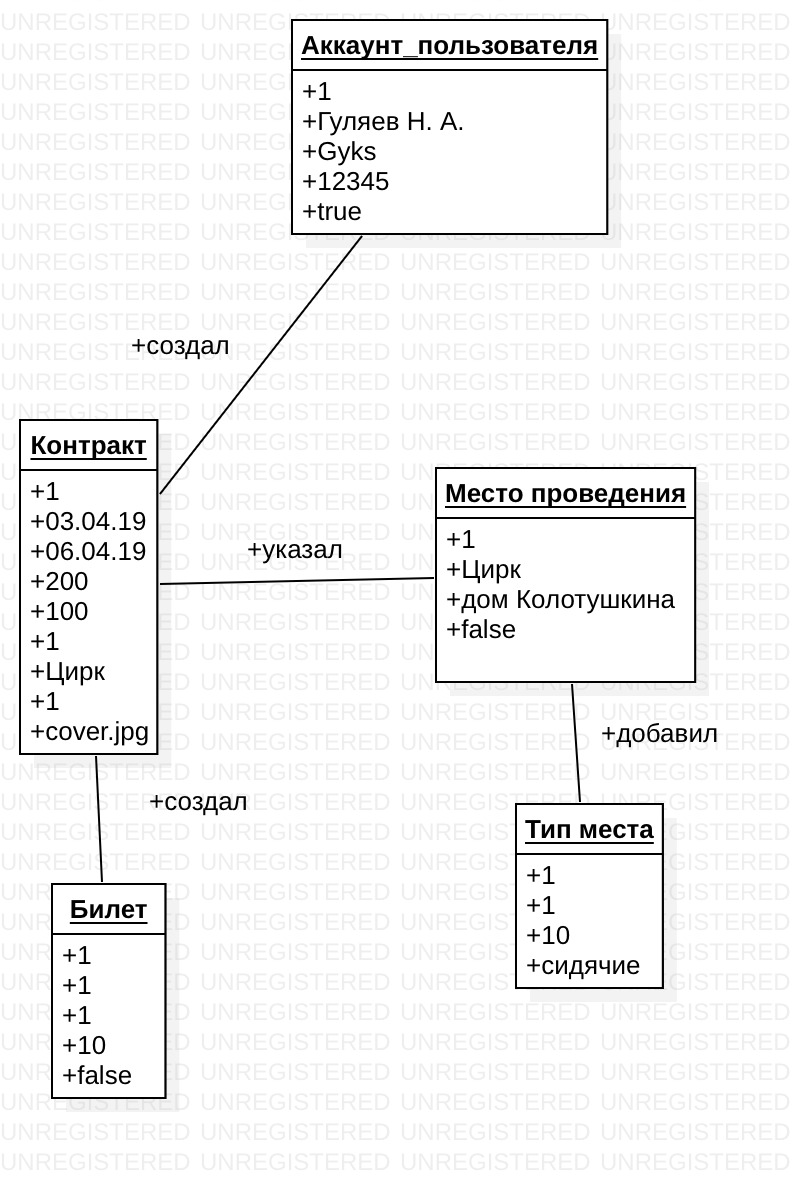
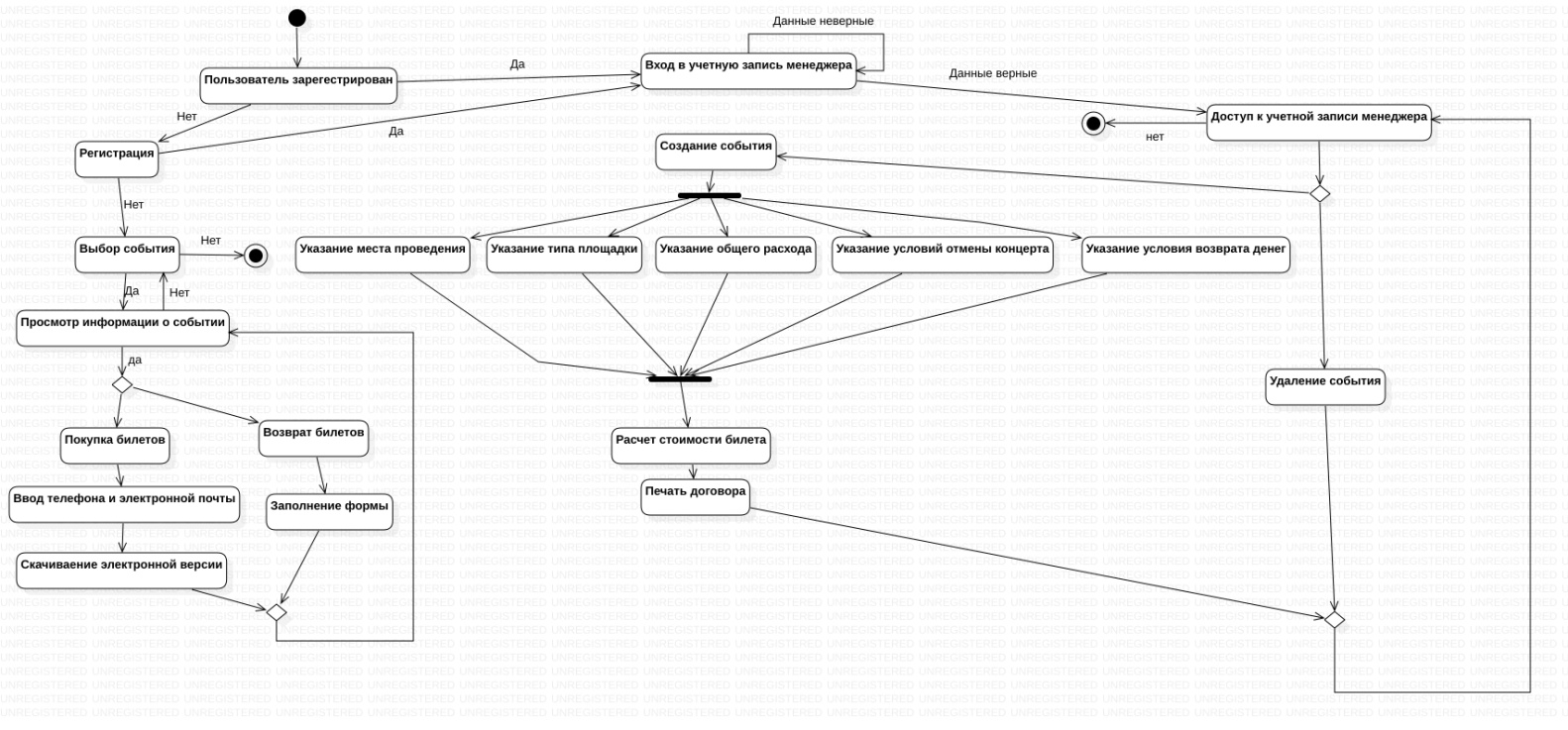


Диаграмма деятельности



Тестирование

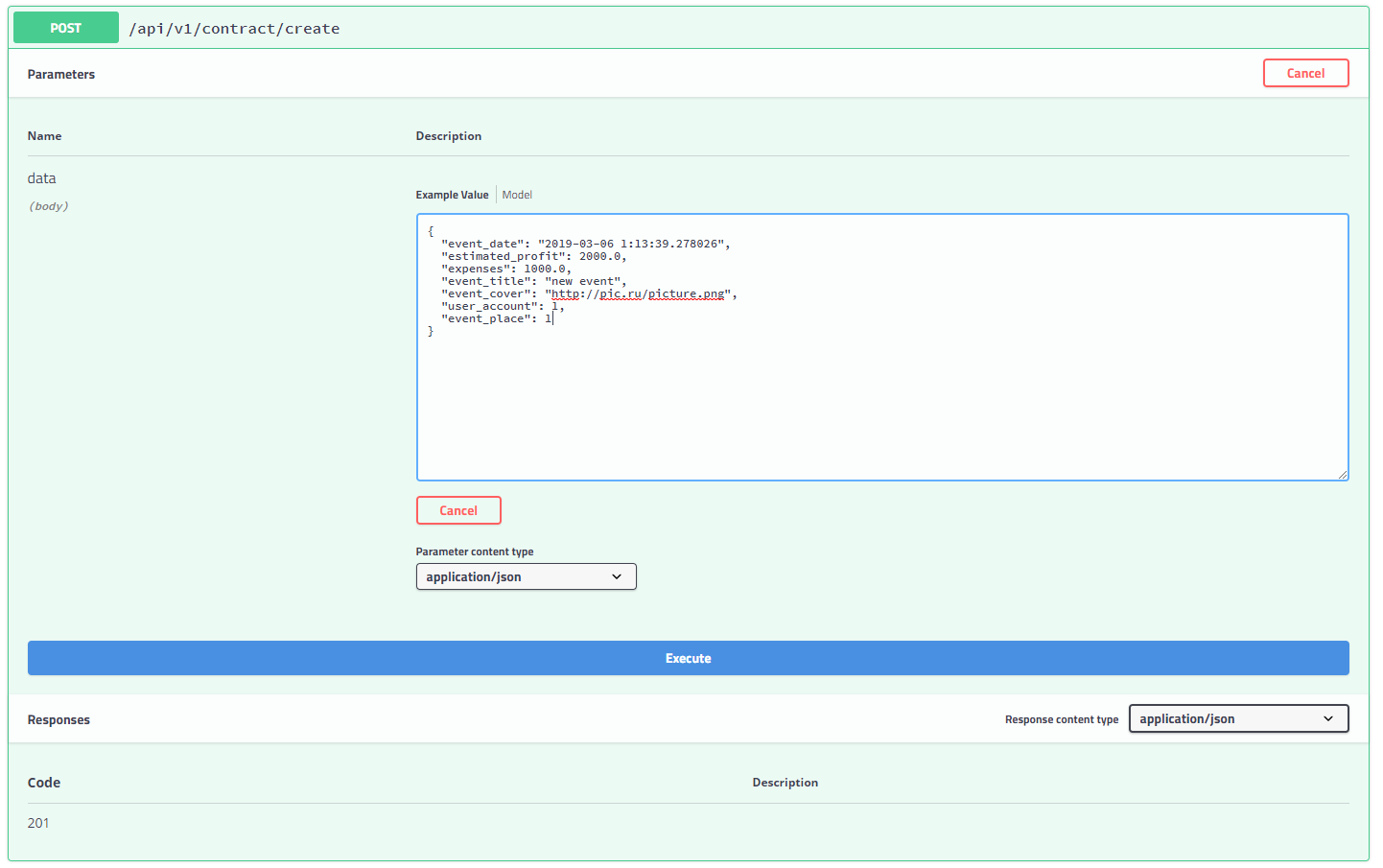
Целью тестирования данного приложения является проверка корректной работы его функций и удобства для пользователя. Тестирование будет проводиться вручную, методом «неформального» тестирования с позиции конечного пользователя приложения.

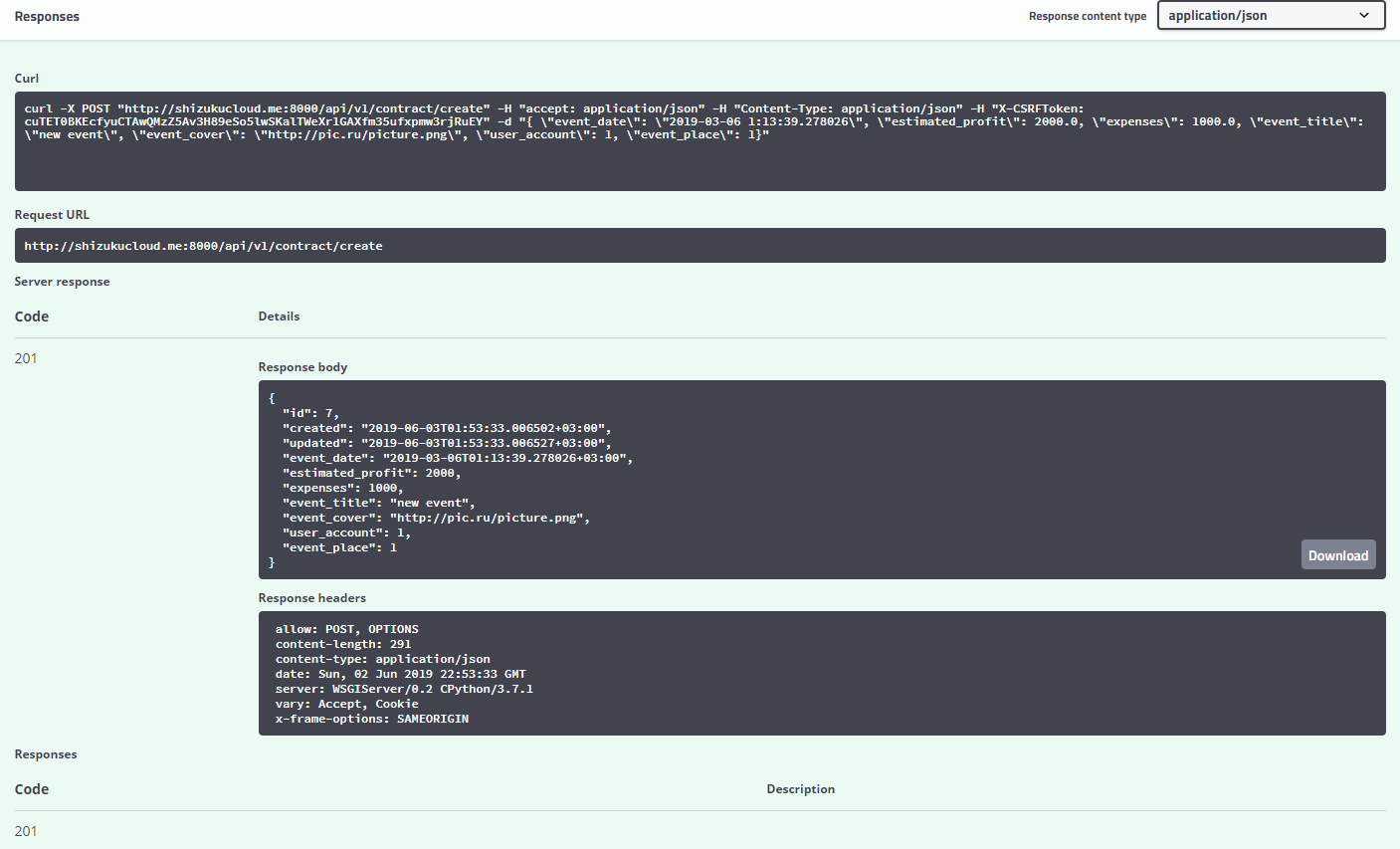
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Название | Use case | Шаг | Ожидаемый результат | Отметка о прохождении |
| 1 | Стартовая страница | Получить информацию о событии | Просмотр списка событий | Отображение страницы со списком событий, возможность открытия отдельного события, списка площадок, кнопка «вернуть». |  |
| 2 | Отдельное событие | Получить информацию о событии | Просмотр информации о событии | Открытие страницы с информацией о событии, на которой есть кнопки для покупки билетов. |  |
| 3 | Покупка | Купить билет | Покупка билета | Открытие страницы с полями для ввода электронной почты и мобильного телефона. После ввода доступна электронная версия билета |  |
| 4 | Возврат | Вернуть билет | Возврат билета | Открытие страницы на которой выводится стоимость возврата билета. |  |
| 5 | Авторизация | Войти в систему | Авторизация | Отображение страницы с возможностью ввода логина/пароля с кнопкой войти. |  |
| 6 | Просмотр событий | Войти в систему | Просмотр созданных ранее событий | Отображение страницы с созданными ранее событиями, площадками, типами мест, билетами и контрактами. |  |
| 7 | Изменение | Изменить информацию | Изменение информации | Переход в один из разделов (билеты, контракты, площадки, типы мест), отображение введенной ранее информации и возможность ее изменения. |  |
| 8 | Создание события | Создать контракт | Создание события | Открытие формы для создания события, на которой нужно указать информацию о событии и отрегулировать стоимость билетов. |  |

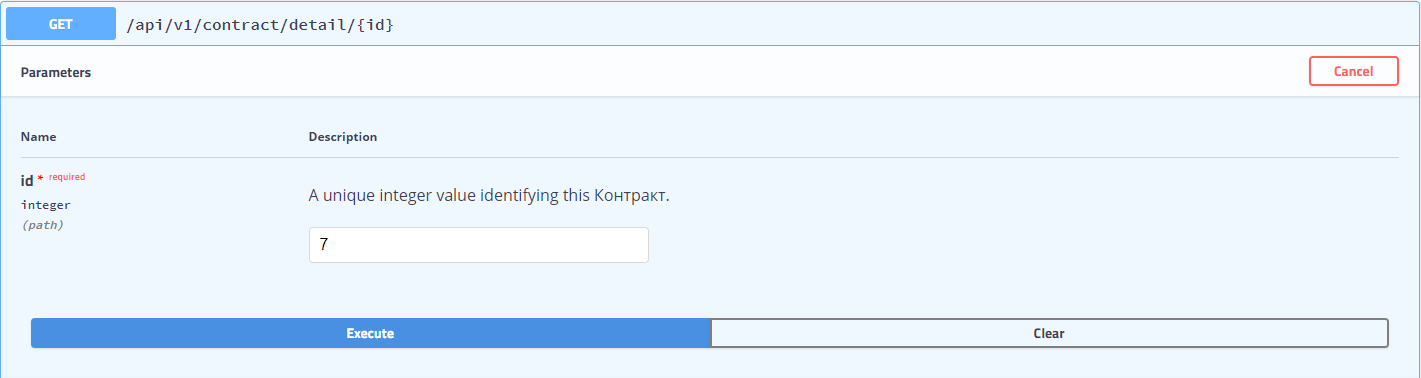
Swagger

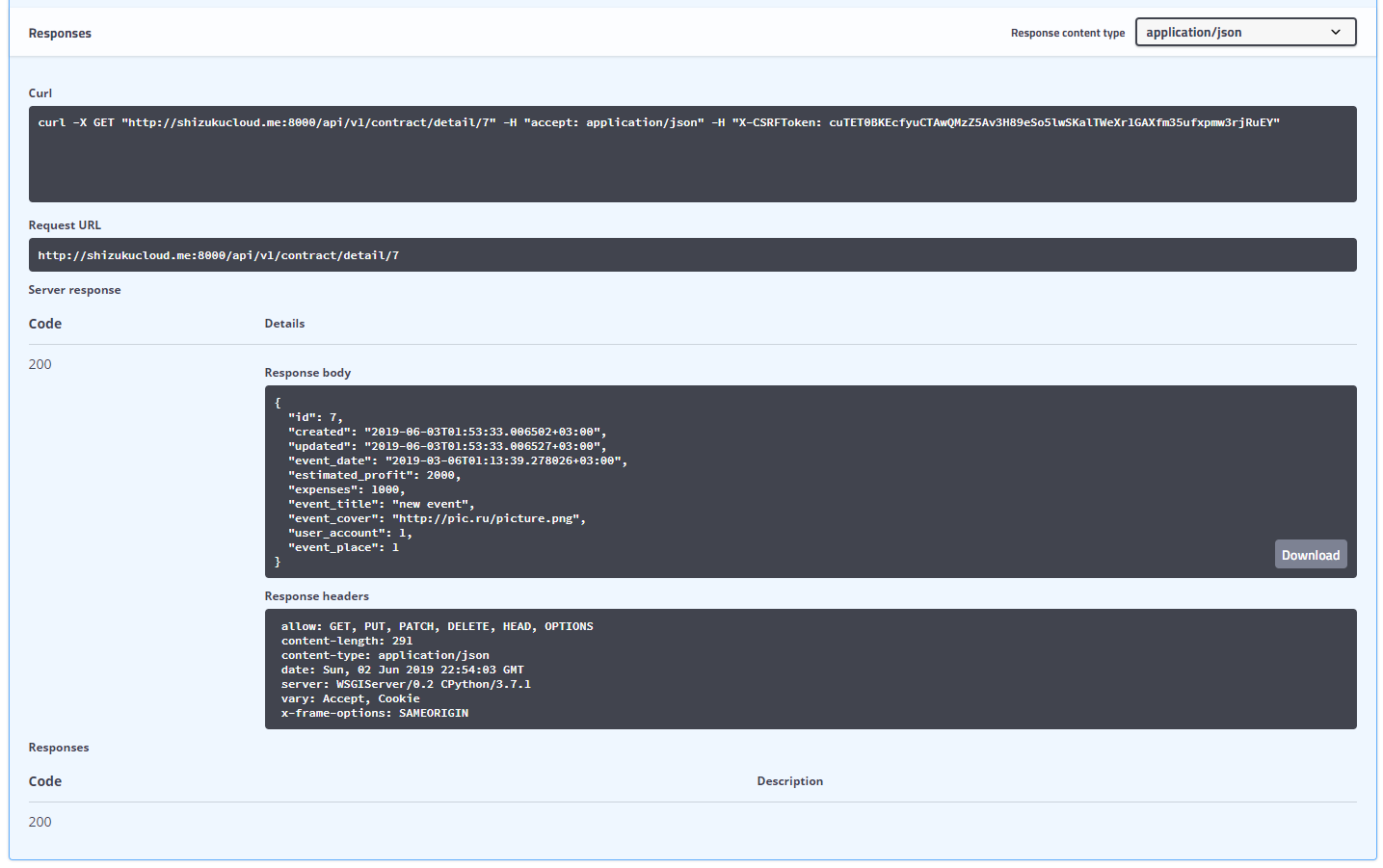
Для тестирования и документирования API нашего веб-приложения был использован swagger UI. С помощью данного приложения можно визуализировать процесс отправки HTTP запросов и получения ответа, а также получить исчерпывающую документацию о всех доступных методах в API.

На скриншотах ниже показан процесс тестирования методов POST (создания контракта) и GET (получения информации о контракте)



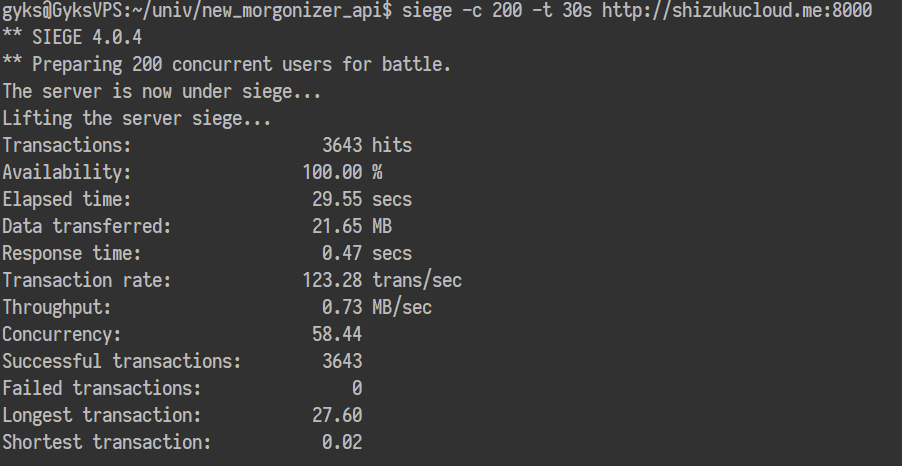






Siege нагрузочное тестирование

Siege – это утилита для нагрузочного тестирования веб-серверов. Она была создана для того чтоб дать разработчикам возможность проверить ресурсоёмкость своего кода в условиях, максимально приближенных к реальным. Так же Siege может имитировать обращения к сайту сразу нескольких пользователей. Это позволяет держать сервер под нагрузкой долгое время. Результат тестирования на скриншоте ниже.



Расшифровка скриншота:

Transactions – количество обращений к серверу. В примере это число высчитывается из 25 пользователей [ -c25 ] запустивших по 10 обращений [-r10], что в общей сумме составляет 250.

Elapsed time – общая продолжительность тестирования. Она высчитывается начиная с первого обращения к серверу и кончая получением ответа на последний запрос. В примере тест занял 14.67 секунд.

Data transferred – суммарное количество данных переданное всеми имитируемыми пользователями. Оно включает в себя как тела запросов, так и их заголовки.

Response time – среднее время за которое сервер успел ответить клиенту.

Transaction rate – среднее число обращений которые сервер успел обработать за секунду. Оно получается путём деления общего числа запросов на затраченное время.

Throughput – среднее число данных передаваемых ежесекундно от сервера к пользователям.

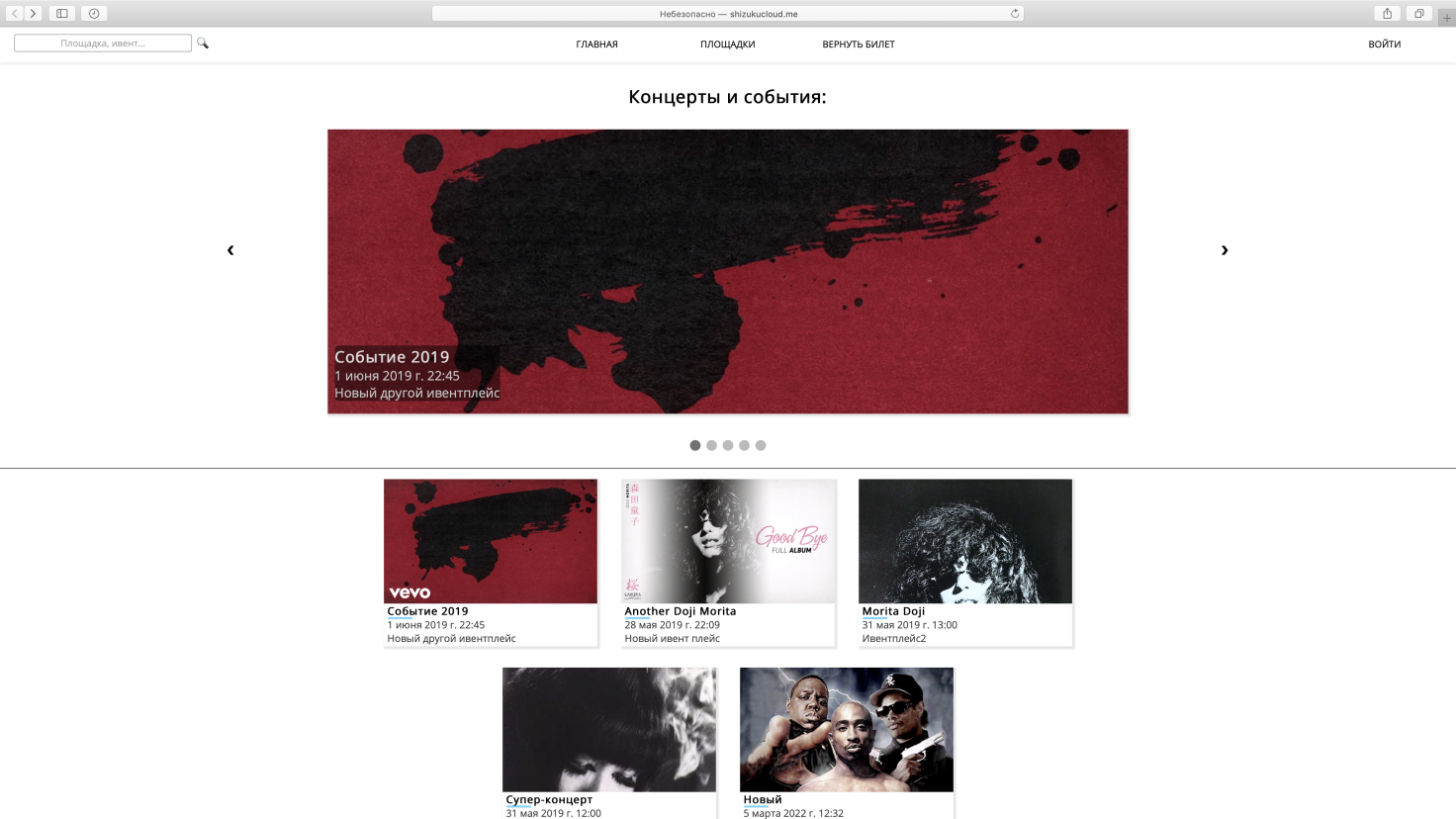
Concurrency – количество одновременных подключений при которых сервер отвечает без задержек.

Successful transactions – количество запросов на которые сервер ответил кодом меньше 400.

Availability – отношение успешных запросов к общему числу запросов. 1 или 100% означает, что сервер полностью справляется с нагрузкой для данной конфигурации.

Реализация

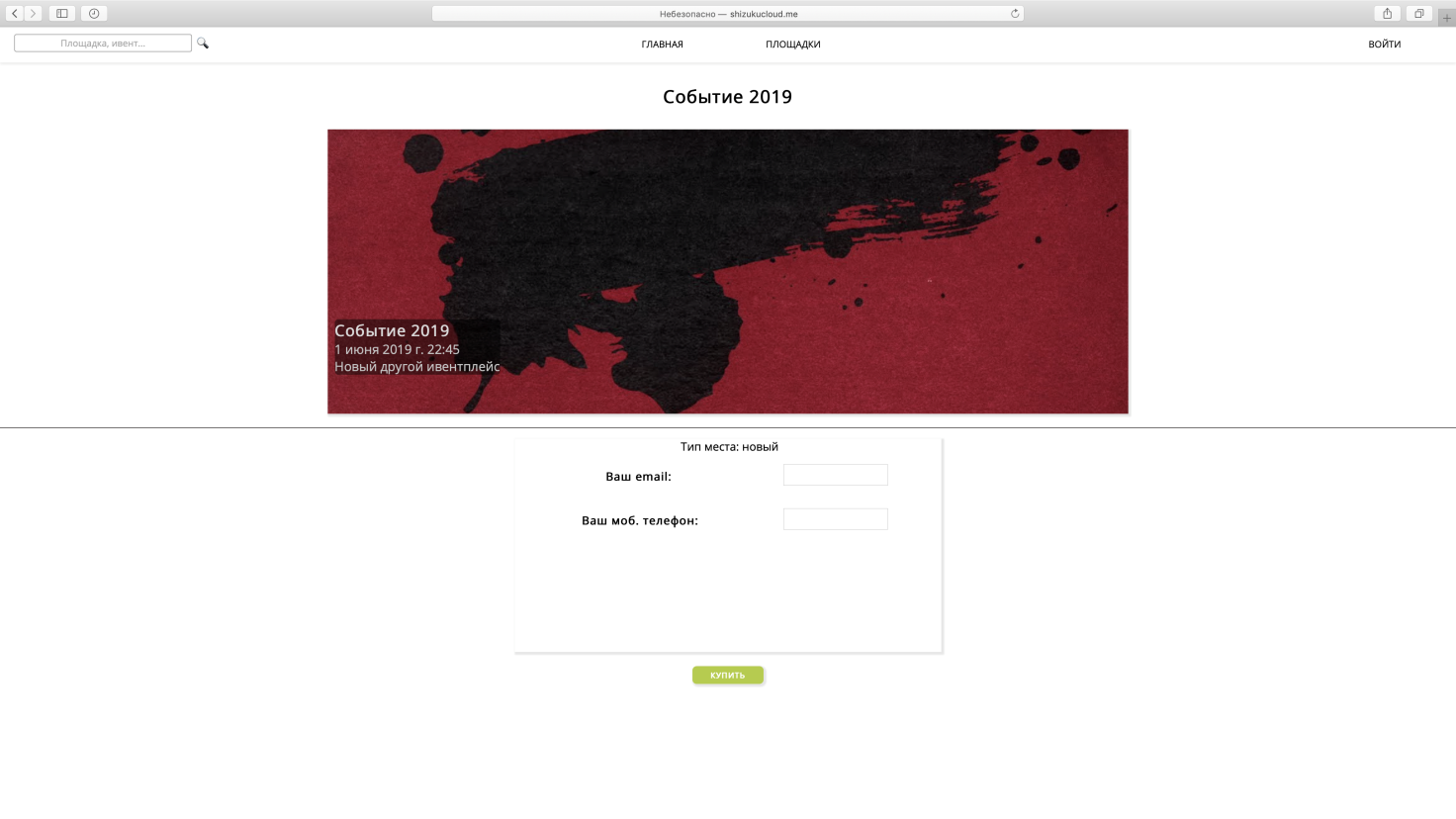
Главная страница



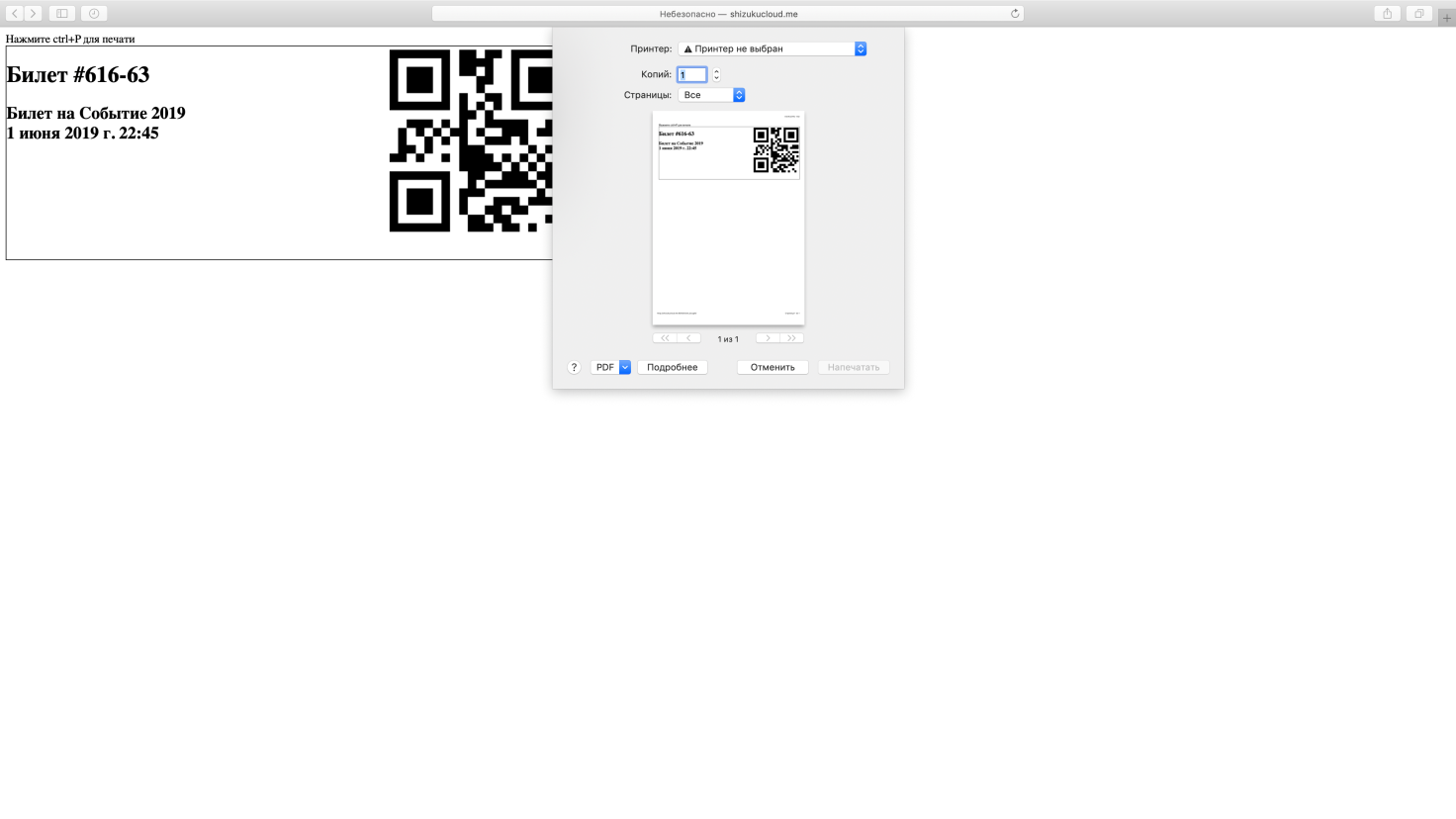
Просмотр информации о событии



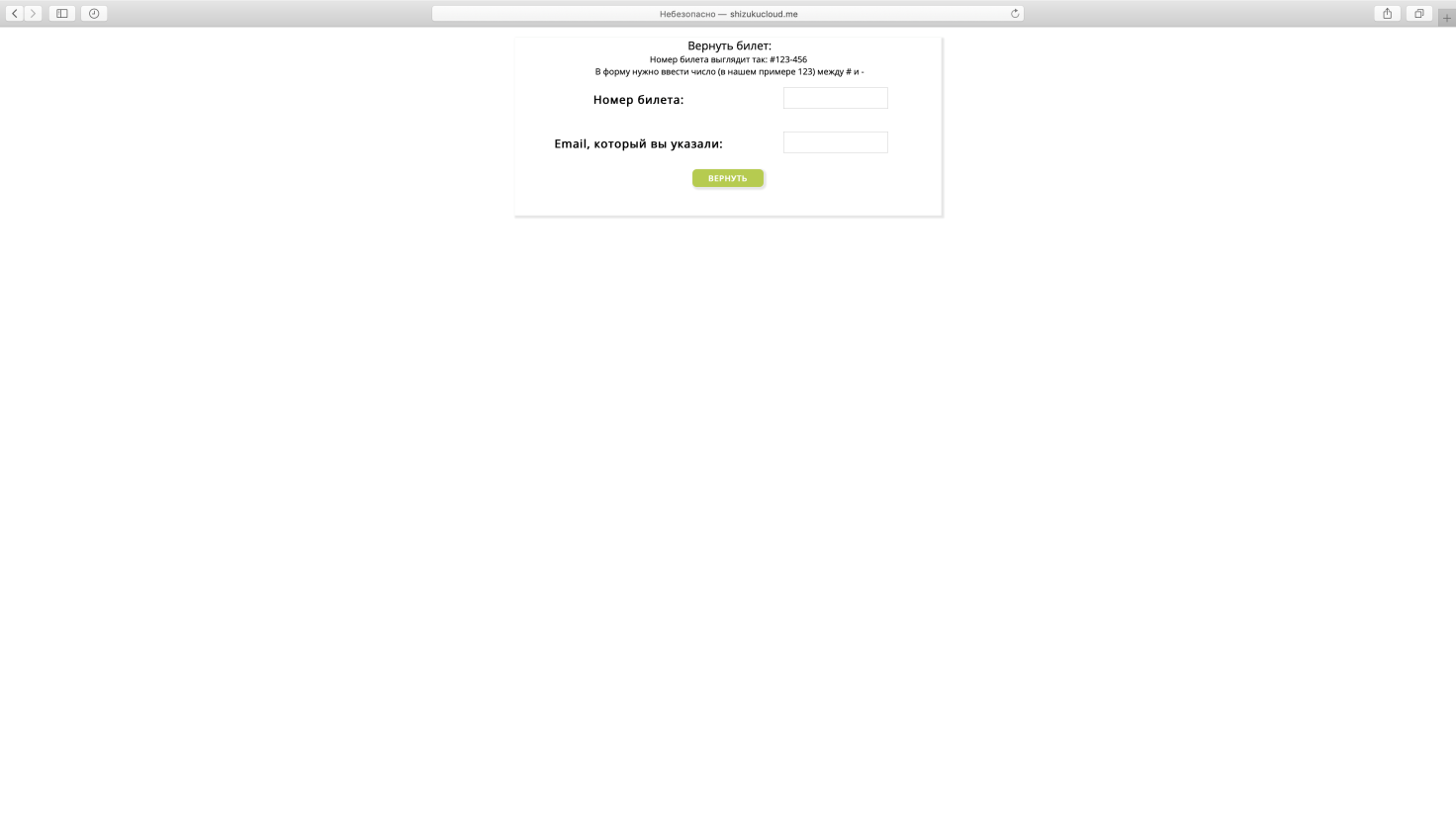
Покупка билета



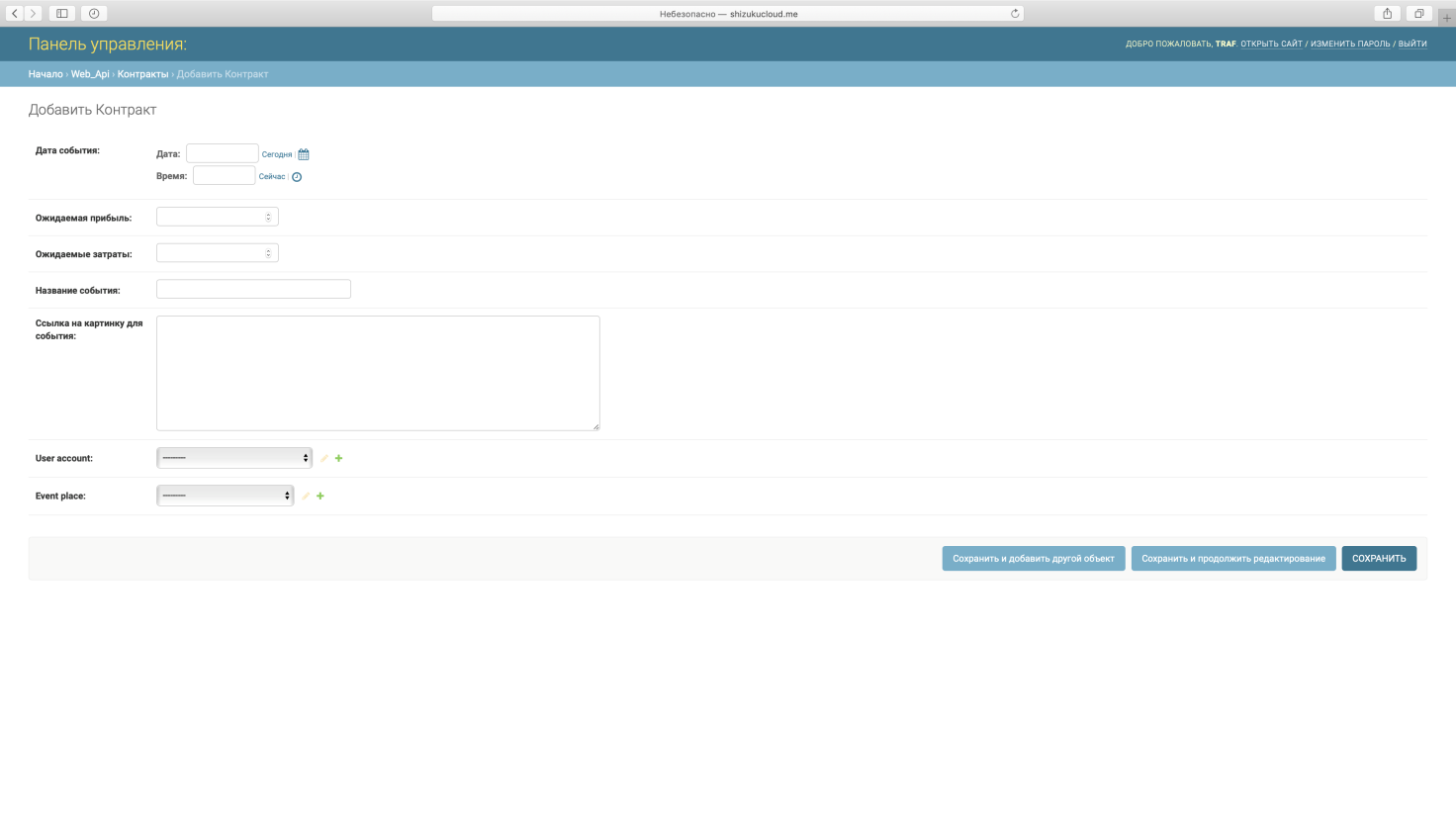
Купленный билет и его печать



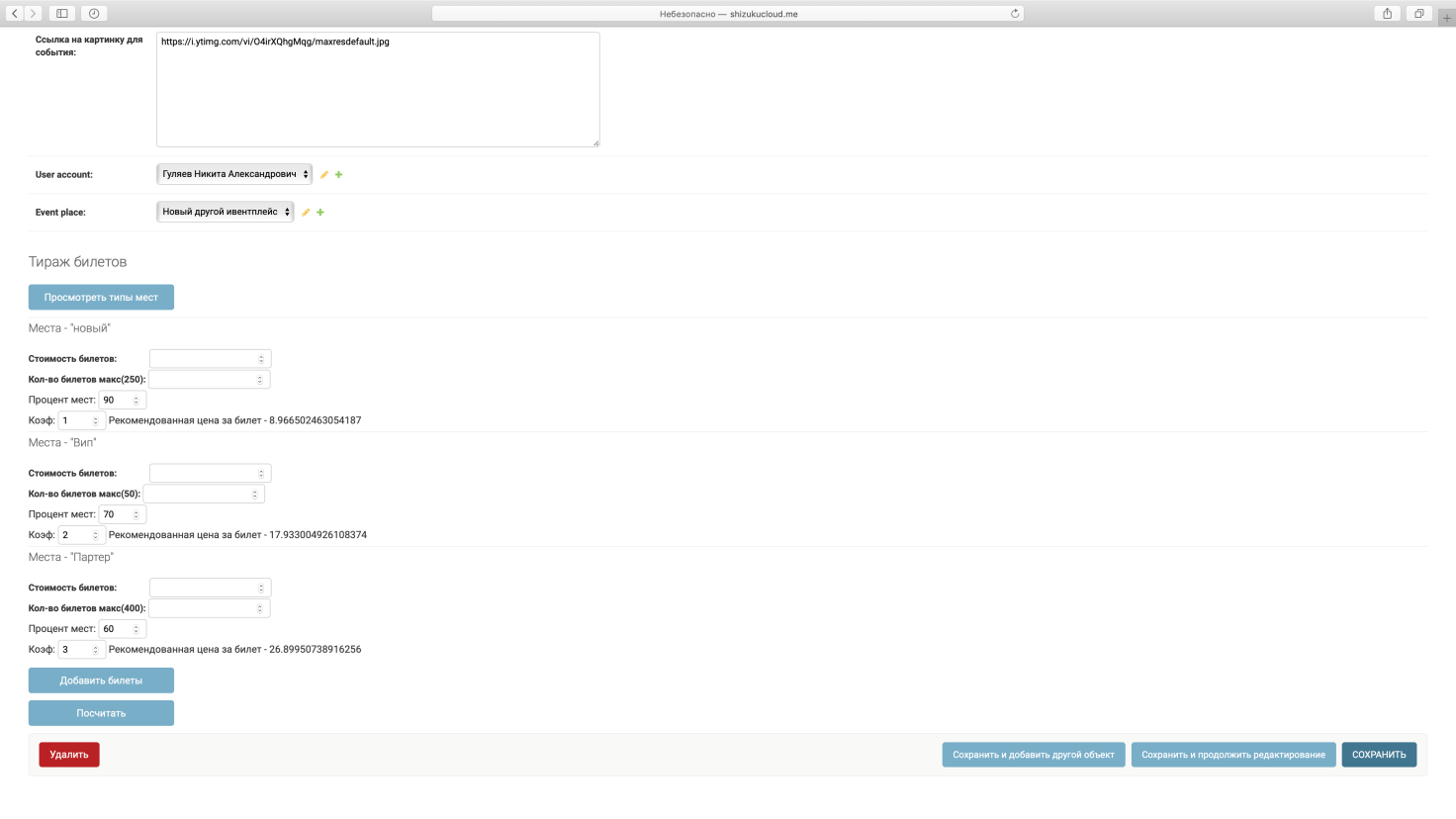
Возврат билета



Создание контракта



Добавление билетов



Работающее приложение можно посмотреть по ссылке

<http://shizukucloud.me:8000>

Заключение

В ходе данной работы было реализовано приложение для проведения концертов на различных площадках и продажи электронных билетов на эти концерты. Приложение предоставляет весь функционал, указанный в постановке задачи, а именно:

* Менеджер может создавать события, редактировать их и удалять.
* Незарегистрированный пользователь может смотреть информацию о событиях, покупать и возвращать билеты.

Отчет по команде:

* Трофимов: диаграмма прецедентов, диаграмма классов, диаграмма последовательностей, диаграмма деятельности, диаграммы взаимодействия, план мануального тестирования, модели для ORM, веб-API, просмотр информации о событиях/площадках, поиск по событиям/площадкам, Swagger.
* Гуляев: ER-диаграмма, диаграмма развертывания, диаграмма объектов, диаграмма последовательностей, макет главной страницы, скелет приложения, интерфейсно реализованные классы, добавления билетов для события, возврат/покупка билетов, форма для создания события/площадки, просмотр информации о событиях/площадках.