|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ  **ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**  **ВЫСШЕГО ОБРАЗОВАНИЯ**  **“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”**  Факультет *компьютерных наук*  Кафедра *программирования и информационных систем*  *Продажа электронных билетов на концерты на различных площадках.*  *Курсовая работа*  09.03.02 Информационные системы и технологии  Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Е.П. Булавина, Ю.В. Петров, В.А. Горбушин, 3 курс  Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*В.С. Тарасов, преподаватель*  Воронеж 2021 |  |  |  |

Содержание

[Содержание 2](#_Toc74947665)

[Введение 3](#_Toc74947666)

[1. Постановка задачи 4](#_Toc74947667)

[2. Анализ предметной области 6](#_Toc74947668)

[2.1. Глоссарий 6](#_Toc74947669)

[2.2. Анализ аналогов 8](#_Toc74947670)

[2.3. Анализ потребности 9](#_Toc74947671)

[3. Продуктовые воронки 10](#_Toc74947672)

[4. Диаграммы 11](#_Toc74947673)

[4.1. Диаграмма прецедентов 11](#_Toc74947674)

[4.2. Диаграмма классов 12](#_Toc74947675)

[4.3. Диаграмма объектов 13](#_Toc74947676)

[4.4. Диаграмма последовательностей 13](#_Toc74947677)

[4.5. Диаграмма взаимодействий 18](#_Toc74947678)

[4.6. Диаграмма состояний 19](#_Toc74947679)

[4.7. Диаграмма активности 21](#_Toc74947680)

[4.8. Диаграмма развертывания 23](#_Toc74947681)

[4.9. Модель IDEF0 23](#_Toc74947682)

[4.10. ER-диаграмма 25](#_Toc74947683)

[5. Реализация 27](#_Toc74947684)

[5.1. Метод решения задачи 27](#_Toc74947685)

[5.1.1. Фронтенд 28](#_Toc74947686)

[5.1.2. Бэкенд 28](#_Toc74947687)

[5.2. Инструментальные средства, использованные для разработки приложения. 28](#_Toc74947688)

[5.3. Описание функционала приложения и интерфейса пользователя. 29](#_Toc74947689)

[6. Тестирование 34](#_Toc74947690)

[6.1. Дымовое тестирование 34](#_Toc74947691)

[6.2. Интеграционное тестирование серверной части 34](#_Toc74947692)

[6.3. Проверка эргономичности 35](#_Toc74947693)

[Заключение 36](#_Toc74947694)

[Список используемой литературы: 37](#_Toc74947695)

[Приложение 39](#_Toc74947696)

Введение

С ростом прогресса в области информационных систем, у человека появилась возможность автоматизировать некоторые элементы обыденной жизни. С каждым днем возможность автоматизации обретает всё больше процессов.

Если говорить точнее, то иметь возможность продать что-то, не задействовав человека, а с помощью автоматизации более выгодно. Особенно, если это не какое-то глобальное производство, а веб-приложение.

Существуют веб-приложения различного плана, которые могут ограничиваться лишь предоставлением мультимедийного контента, а могут являться «посредниками» в предоставлении реальных товаров и услуг.

Важной частью поставляемых товаров и услуг является продажа билетов, их предоставление онлайн позволяет клиенту сохранить время и силы для наслаждения непосредственно мероприятием.

В связи с этим, данная курсовая работа обозревает процесс создания веб-приложения, специализацией которого является деятельность в сфере продажи билетов на концерты. Здесь и далее будет рассмотрены цели, задачи, специфика и результаты работы по созданию приложения.

1. Постановка задачи

Развлечения всегда были неотъемлемой частью жизни человека. Они позволяют людям отвлечься от повседневной рутины, поднимают настроение, расслабляют. К этой сфере относится и музыка, которая

Цель данной курсовой работы – разработка веб-приложения, позволяющего его посетителям и пользователям приобретать билеты на музыкальные концерты.

Задачи, которые необходимо решить в ходе развития проекта:

* Сбор необходимой информации об исследуемой предметной области;
* Проведение анализа полученных данных;
* Создание диаграмм, отражающих основные аспекты приложения;
* Разработка веб-приложения;
* Проведение тестирования веб-приложения;
* Размещение приложения на сервере.

Разработка веб-приложения состоит из двух основных частей, к ним относятся разработка серверной и клиентской частей.

Разработка серверной части:

* Проработка и реализация базы данных;
* Реализация работы с используемыми API;
* Формирование запросов к базе данных.

Разработка клиентской части:

* Проработка дизайна веб-приложения;
* Формирование веб-страниц приложения;
* Реализация взаимодействия с серверной частью.

Приложение должно позволить пользователям следующее:

* Просмотр информации о грядущих концертах (название концерта, место проведения концерта, дата и время проведения концерта, статус концерта);
* Ознакомление с исполнителем посредством представленной на сайте информации о нем;
* Приобретение билетов на выбранные концерты;
* Написание отзывов об артистах и концертах и выставление оценок, в том случае, если пользователь авторизован;
* Получение контактных данных администрации приложения для связи (к примеру, для обсуждения о внесении данных какого-либо концерта в приложение).

1. Анализ предметной области
   1. Глоссарий

В курсовой работе будут использоваться следующие термины:

Аватар - Фотография или другое графическое изображение, используемое в учетной записи для персонализации пользователя.

Администратор - Специалист, уполномоченный управлять и администрировать определенный участок IT-среды.

Веб-приложение (сайт) - Клиент-серверное приложение, в котором клиент взаимодействует с веб-сервером при помощи браузера.

Карусель - Компонент слайд-шоу для цикличного повторения элементов.

Модератор - Административное лицо сервиса, которое действует от лица представителя веб-ресурса и выполняет определенный ряд задач, направленных на соблюдение пользователями правил данного ресурса.

Пользователь - Лицо, которое использует приложение для выполнения конкретной функции.

СУБД - Система управления базами данных. Комплекс программно-языковых средств, позволяющих создать базы данных и управлять данными.

Android - Операционная система для мобильных устройств.

API - Программный интерфейс приложения. Описание способов, которыми одна компьютерная программа может взаимодействовать с другой программой.

Bootstrap - Свободный набор инструментов для создания сайтов и веб-приложений.

CSS - Формальный язык, служащий для описания оформления внешнего вида документа, созданного с использованием языка разметки (HTML, XHTML, XML).

Flask - Фреймворк для создания веб-приложений на языке программирования Python, использующий набор инструментов Werkzeug, а также шаблонизатор Jinja2.

Genius - Сайт, позволяющий пользователям предоставлять аннотации и интерпретации текстов песен, новости, источники, стихи, документы.

Google Календарь - Сервис для планирования встреч, событий и дел, разработанный компанией Google.

HTML - Стандартизированный язык разметки веб-страниц воВсемирной паутине.

JavaScript - Мультипарадигменный язык программирования, используется как встраиваемый язык для программного доступа к объектам приложений.

JQuery - Набор функций JavaScript, фокусирующийся на взаимодействии JavaScript и HTML.

Linux - Семейство Unix-подобных операционных систем на базе ядра Linux.

PostgreSQL - Свободная объектно-реляционная система управления базами данных.

Python - Высокоуровневый язык программирования общего назначения.

REST - Архитектурный стиль взаимодействия компонентов распределённого приложения в сети.

Songkick - Находящаяся в США служба поиска концертов, принадлежащая Warner Music Group.

Spotify - Интернет-сервис потокового аудио (стриминговый), позволяющий легально прослушивать музыкальные композиции, аудиокниги и подкасты, не скачивая их на устройство.

SQLAlchemy - Программная библиотека на языке Python для работы с реляционными СУБД с применением технологии ORM.

Windows 10 - Операционная система для персональных компьютеров и рабочих станций, разработанная корпорацией Microsoft в рамках семейства Windows NT.

* 1. Анализ аналогов

Для формулирования более конкретных требований к веб-приложению необходимо было провести анализ аналогов. К ним можно отнести такие сайты-агрегаторы как, например, Яндекс.Афиша, Афиша и сайт концертного зала «Event-Hall».

Яндекс.Афиша - это билетный сервис, работающий с 2005 года, созданный компанией Яндекс.

Сильные стороны сервиса:

* Выбор места;
* Наличие рекомендательной системы;

Слабые:

* Наличие назойливой рекламы, из-за которой просмотр содержимого сайта замедляется и становится менее комфортным.

Афиша – это медийно-сервисная платформа о развлечениях и культурном досуге, основанная в апреле 1999 года.

Сильные стороны сервиса:

* Возможность оставлять отзывы;

Слабые:

* Интерфейс покупки билета для разных мероприятий различается;

«Event-Hall» – сервис для продажи билетов в рамках одной торговой сети.

Сильные стороны сервиса:

* Сервис позволяет приобретать билеты напрямую у владельцев зала, в котором проходит концерт.

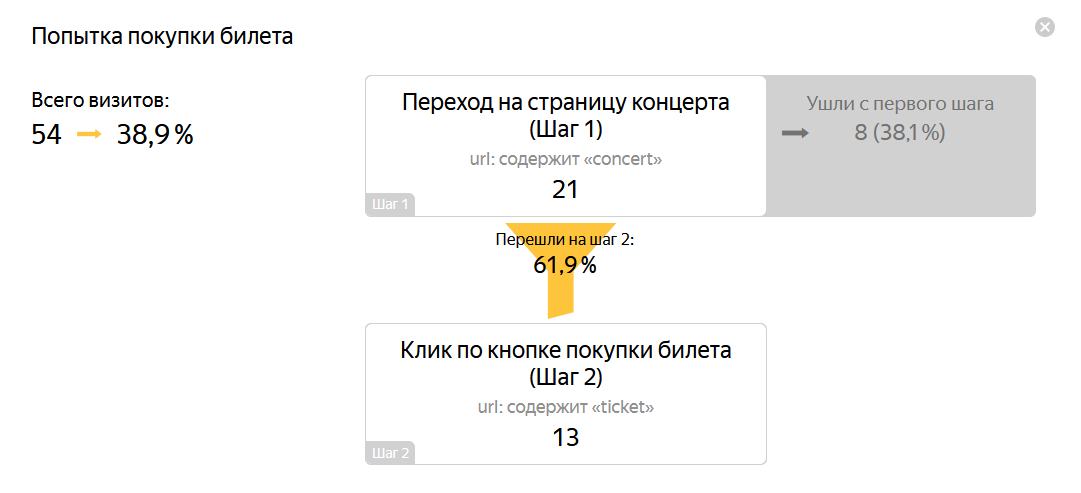
Слабые:

* Сервис предназначен только для одного концертного зала, т.е. мало количество доступных концертов, ограниченность по городу;
  1. Анализ потребности

После изучения аналогов разрабатываемого проекта можно сделать вывод, что ни один из них не является идеальным решением поставленной задачи. Тем не менее, само большое количество различных сервисов, позволяющих приобретать билеты на музыкальные концерты, дает понять, что у людей есть потребность в этом. В связи с этим разработка проекта Concert Hall является актуальной, потому необходимо будет учесть недостатки аналогов и не допустить их.

1. Продуктовые воронки

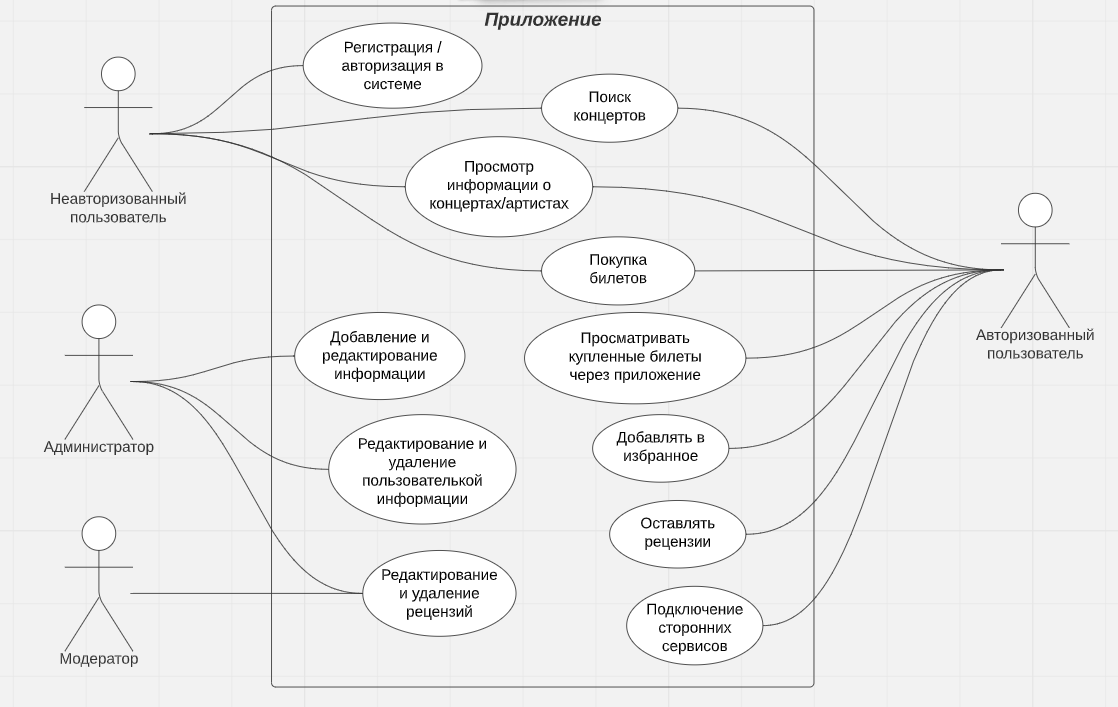
Главной задачей приложения является продажа билетов на концерты. В связи с этим при помощи сервиса Яндекс.Метрика были рассмотрены продуктовые воронки.



1. Продуктовая воронка для попытки покупки билета
2. Диаграммы

Диаграммы необходимы для на этапе моделирования приложения. Они используются для визуализации, спецификации, конструирования и документирования программных систем.

* 1. Диаграмма прецедентов



1. Диаграмма прецедентов (use-case)

На данной диаграмме представлены функции актеров (пользователей системы) в системе.

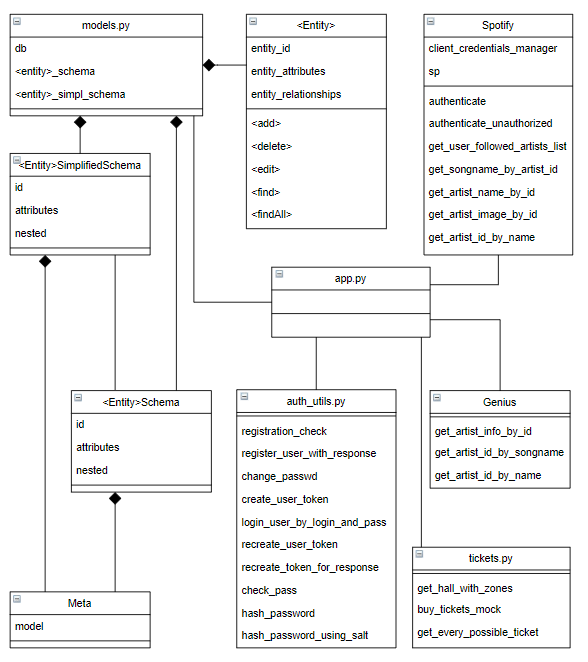
Неавторизованный пользователь: регистрация/авторизация в системе, поиск концертов, просмотр информации о концертах/артистах, покупка билетов;

Авторизованный пользователь (дополнительно к функциям неавторизованного пользователя): просмотр купленных билетов через приложение, возможность добавлять в избранное концерты и артистов, возможность оставлять рецензии на концерты и артистов, подключение сторонних сервисов;

Модератор: редактирование и удаление рецензий;

Администратор (дополнительно к функциям модератора): добавление и редактирование информации, редактирование и удаление пользовательской информации.

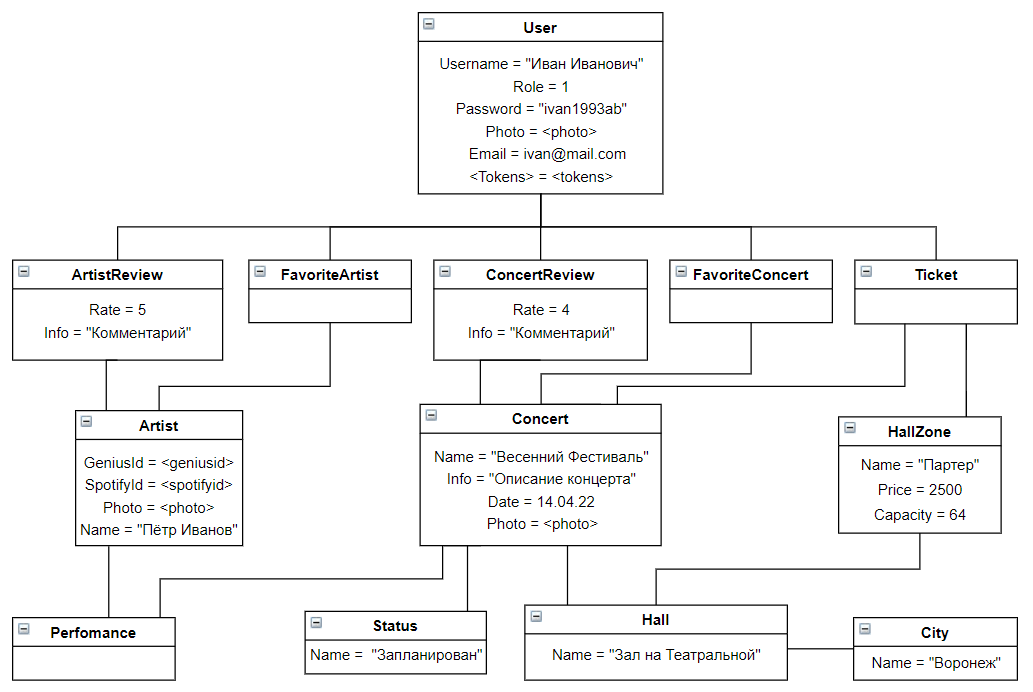
* 1. Диаграмма классов



1. Диаграмма классов приложения

В данной диаграмме представлены классы системы, их свойства, атрибуты и функции. Взаимодействие между модулями производится посредством контроллера.

* 1. Диаграмма объектов

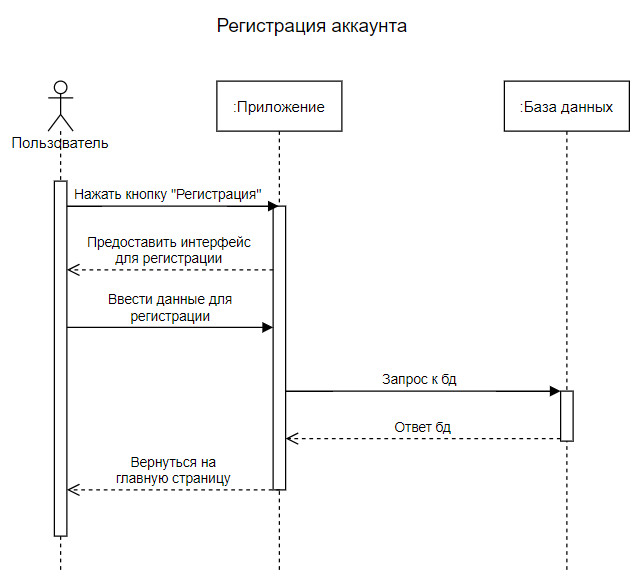


1. Диаграмма объектов проекта

Диаграмма объектов показывает множество объектов - экземпляров классов и отношений между ними в какой-то момент времени. Выступает в роли дополнения к диаграмме классов.

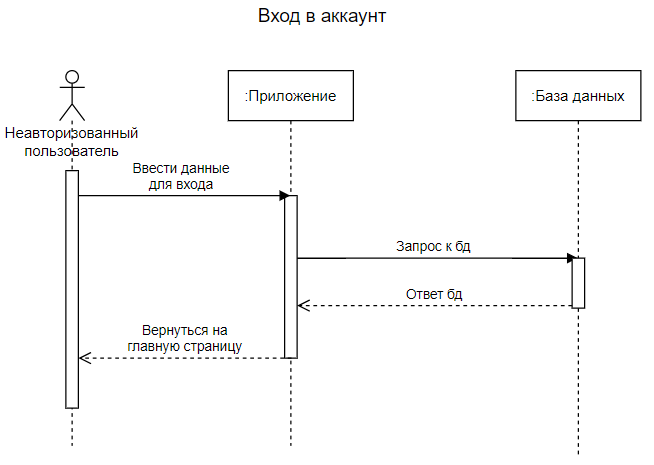
* 1. Диаграмма последовательностей

Диаграмма последовательностей служит для описания взаимодействия объектов системы по мере их проявления в сценарии. В данном случае имеем 5 сценариев.



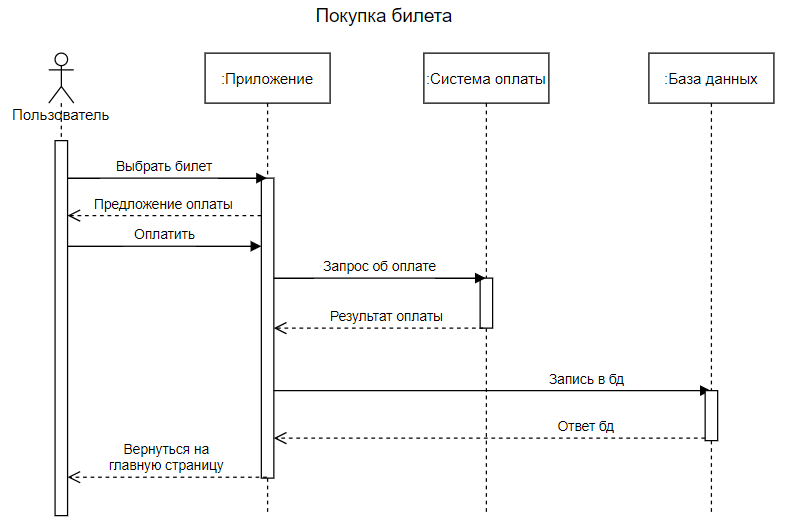
1. Диаграмма последовательности для регистрации

Незарегистрированный пользователь после нажатия на кнопку регистрации получает от приложения форму для ввода данных регистрации. После отправки этих данных приложение делает запрос к БД и в случае успешного ответа (т.е. данные введены корректно и не совпадают с существующим) пользователь получает зарегистрированный аккаунт.



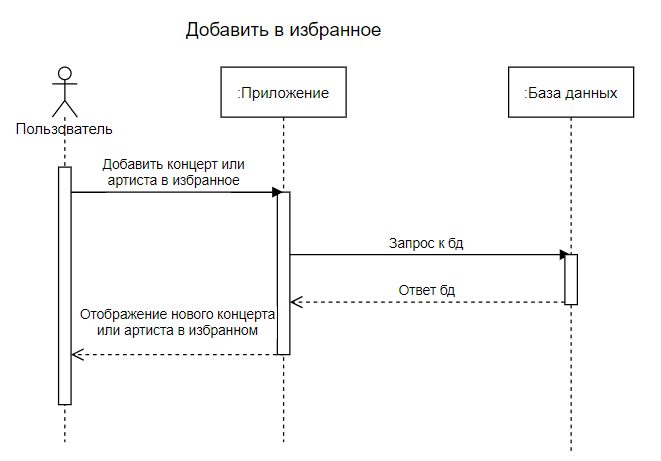
1. Диаграмма последовательностей для авторизации

Неавторизованный пользователь вводит данные для входа (логин и пароль) в форму приложения, которое отправляет запрос в БД и в случае положительного ответа позволяет пользователю войти в приложение (с возвратом на главную страницу), в противном случае будет предложено ввести данные заново.



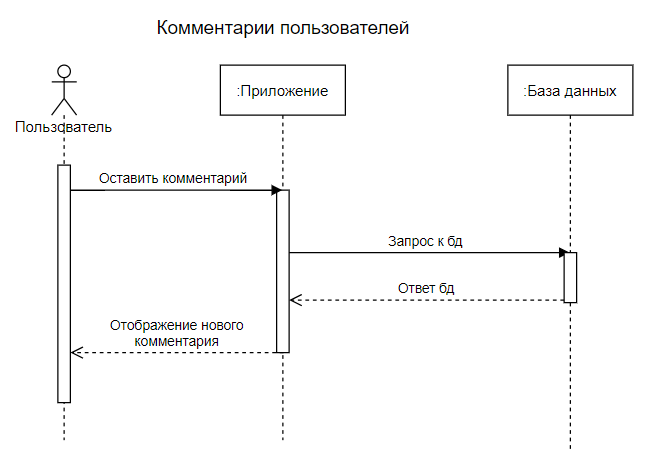
1. Диаграмма последовательностей для покупки билета

Пользователь выбирает необходимые билеты, после чего приложение предлагает пользователю оплатить покупку. Пользователь вводит данные для оплаты в приложение, которое посылает запрос в систему оплаты. В случае положительного ответа в БД записывается информация о приобретенных билетах.



1. Диаграмма последовательностей для добавления элемента в избранное

Пользователь выбирает концерт или артиста, чтобы добавить в список избранного. После нажатия на соответствующую кнопку приложения производится запрос на запись данной информации в БД в случае удачного ответа запись производится.

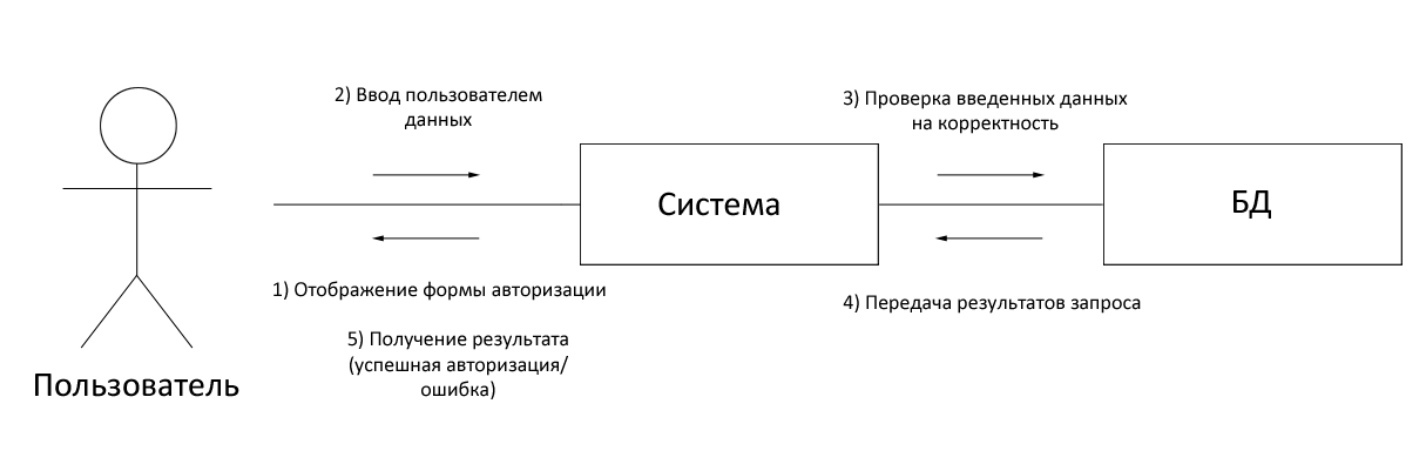


1. Диаграмма последовательностей для добавления комментариев

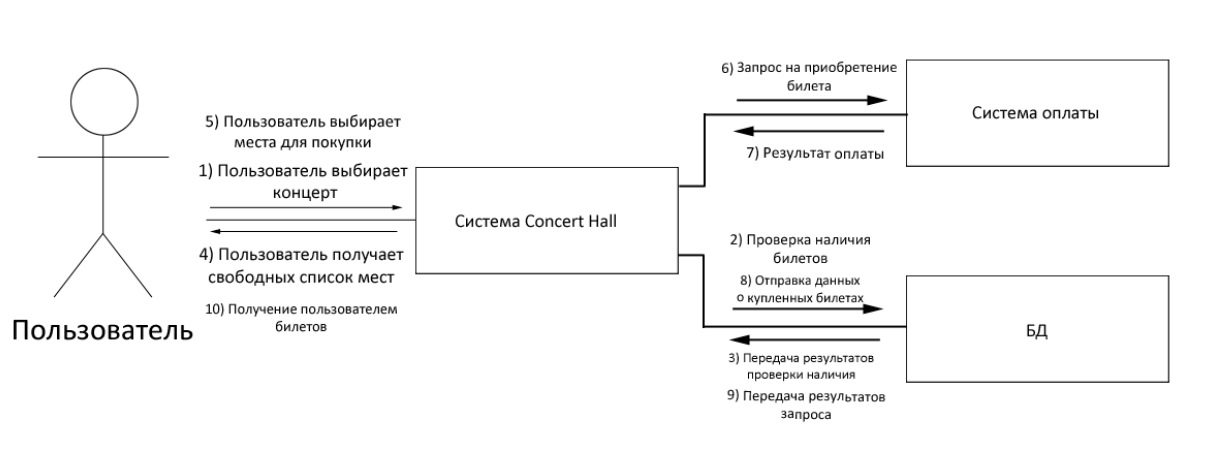
Пользователь вводит текст комментария в соответствующем поле приложения и выбирает оценку для выбранного концерта/артиста. После нажатия на кнопку отправки комментария приложение делает запрос на запись в БД и в случае успешного ответа комментарий добавляется.

* 1. Диаграмма взаимодействий

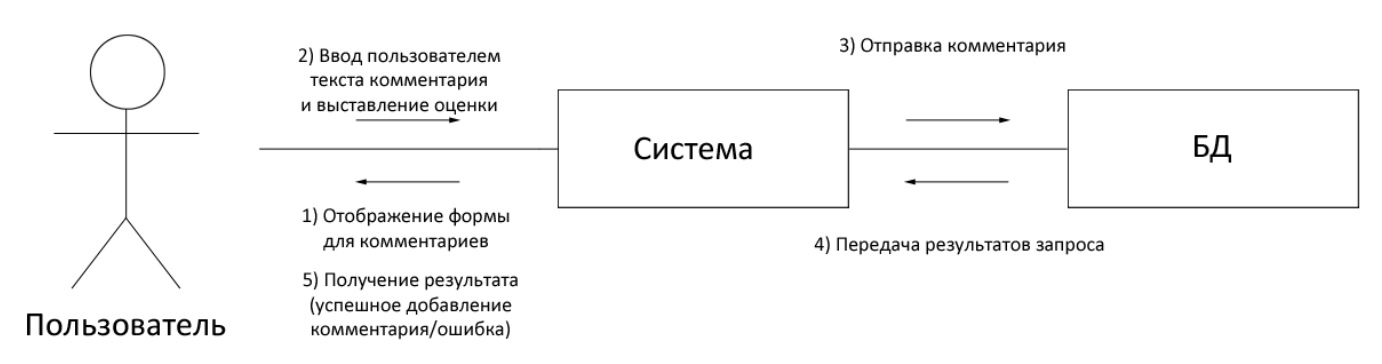
Данная диаграмма определяет возможные способы взаимодействия пользователей с системой. Позволяет видеть все взаимодействия запросов в системе, служит дополнением к диаграмме последовательностей



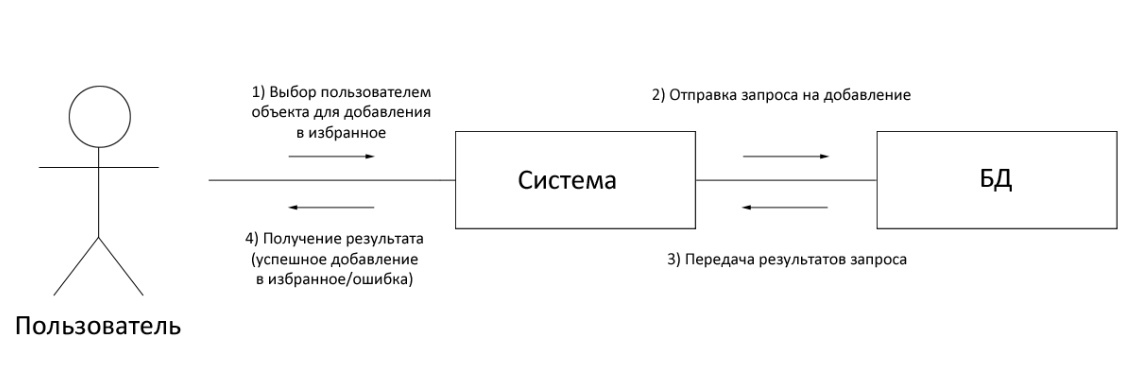
1. Диаграмма взаимодействия для авторизации



1. Диаграмма взаимодействия для покупки билета

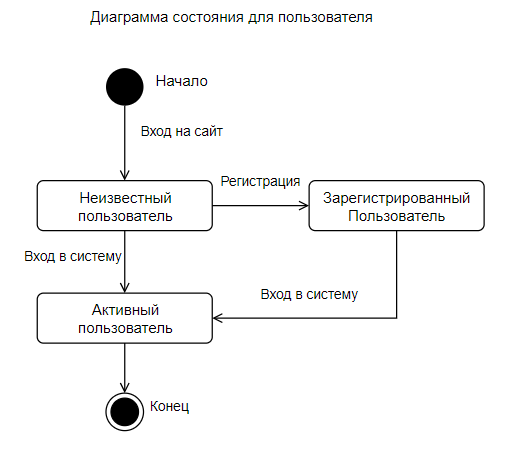


1. Диаграмма взаимодействия для добавления комментария

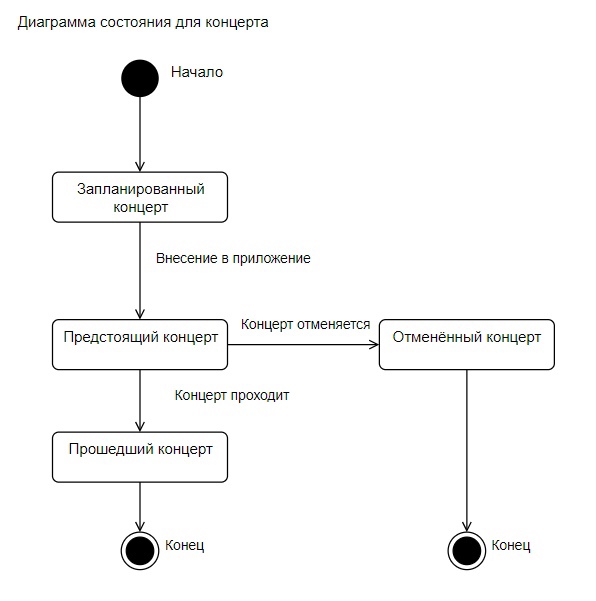


1. Диаграмма взаимодействия для добавления элемента в избранное
   1. Диаграмма состояний

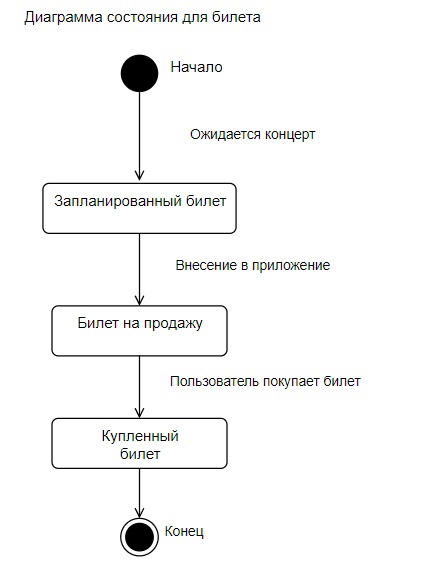
Данные диаграммы описывают переход объектов из одних состояний в другие.



1. Диаграмма состояний для авторизации

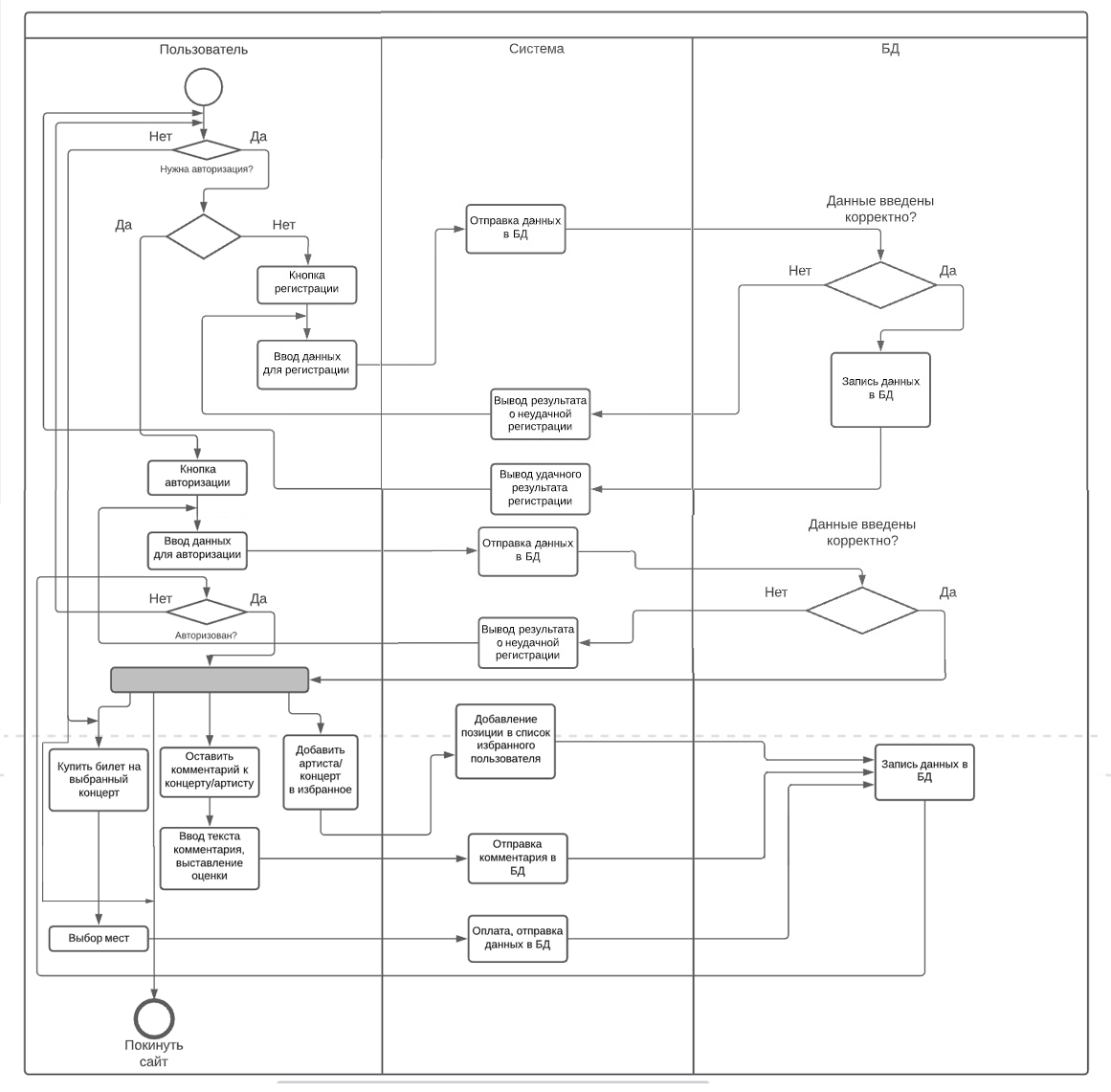


1. Диаграмма состояний для концерта

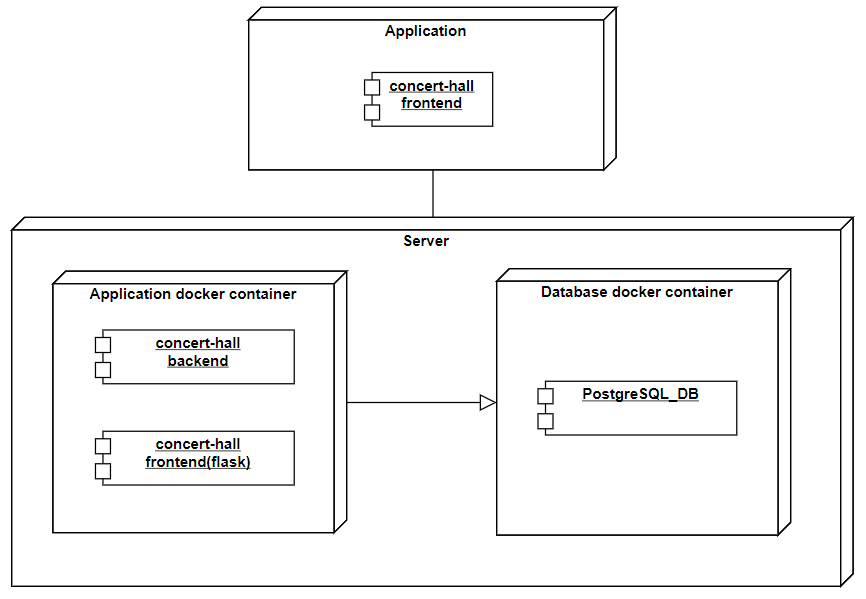


1. Диаграмма состояний для билета
   1. Диаграмма активности

Диаграмма активности описывает возможную последовательность объектов системы. Например, в данной диаграмме видно, что пользователь может, не авторизовавшись, либо покинуть приложение, либо купить билет, либо авторизоваться. Авторизованный пользователь может помимо покупки билетов также оставлять комментарии к концертам и артистам, а также добавлять их в свой список избранного.



1. Диаграмма активности для приложения
   1. Диаграмма развертывания

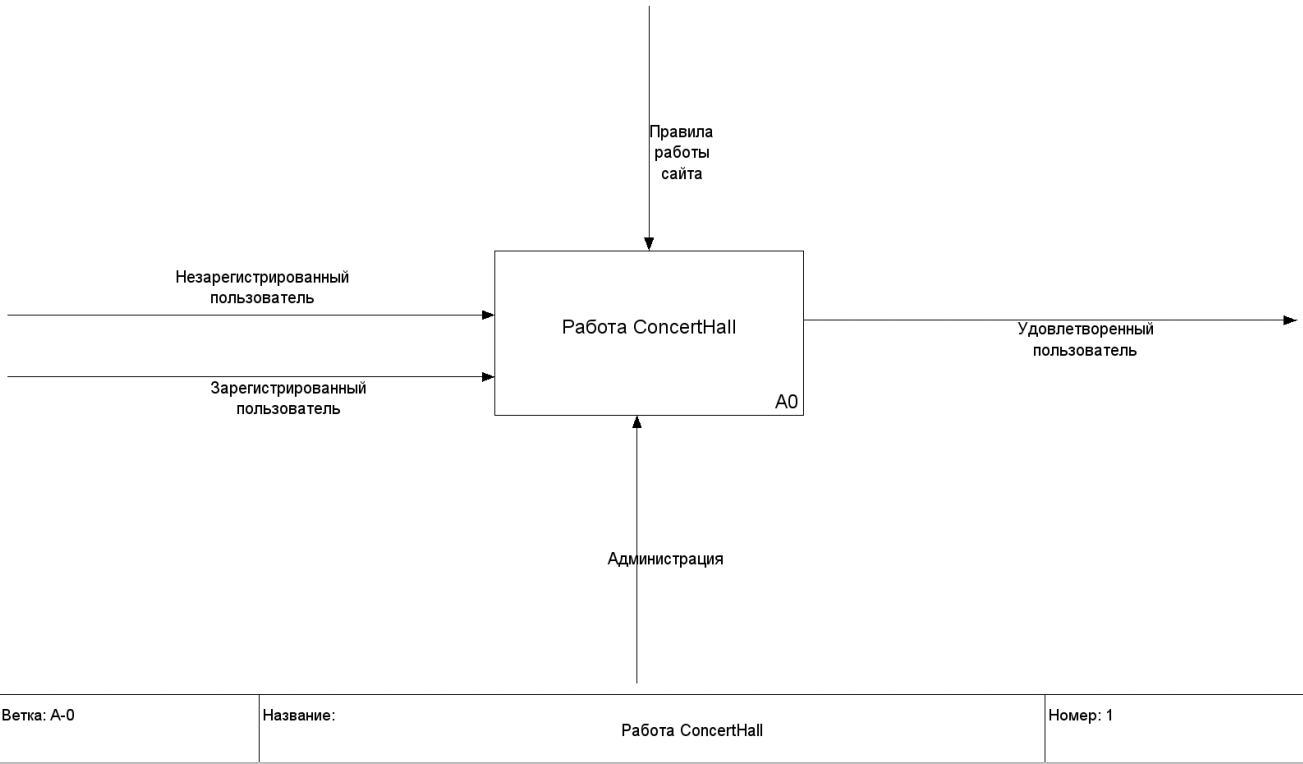


1. Диаграмма развертывания для приложения

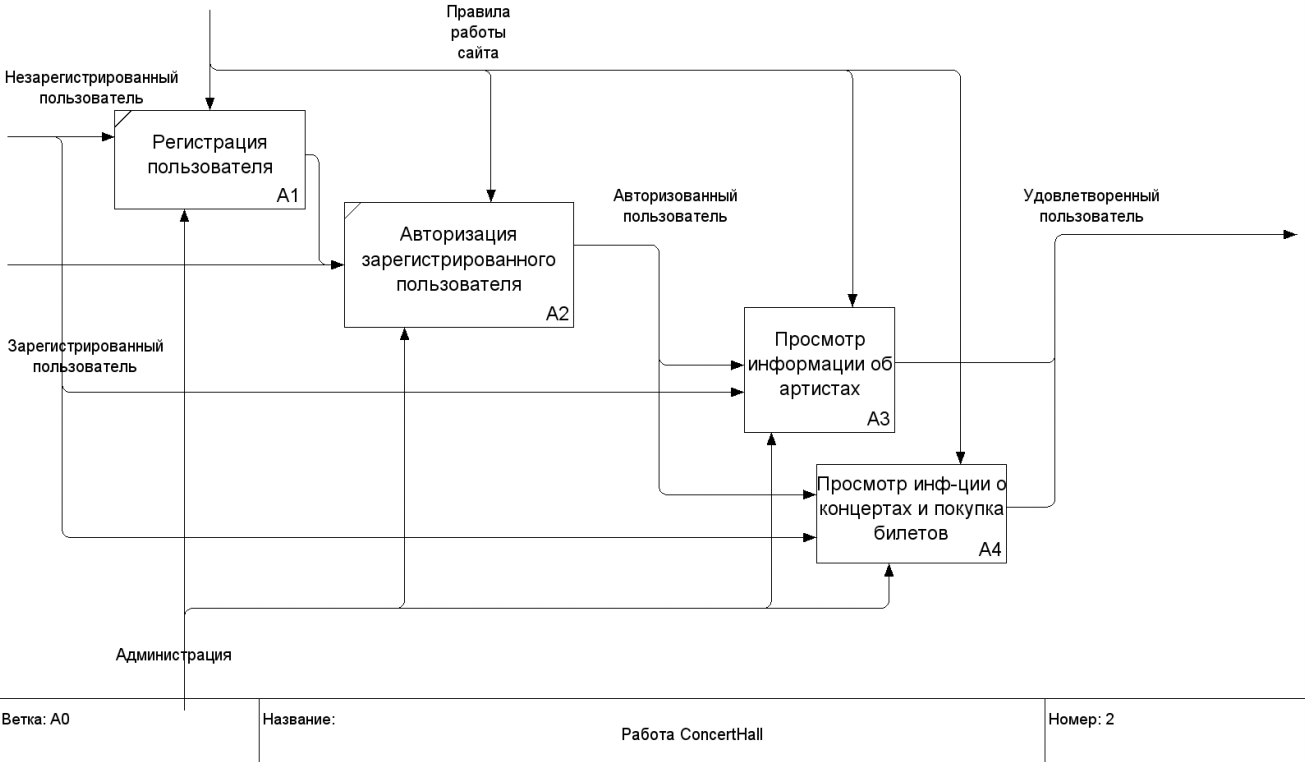
Данная диаграмма определяет аппаратные компоненты, их функционирование, а также их взаимодействие друг с другом.

* 1. Модель IDEF0

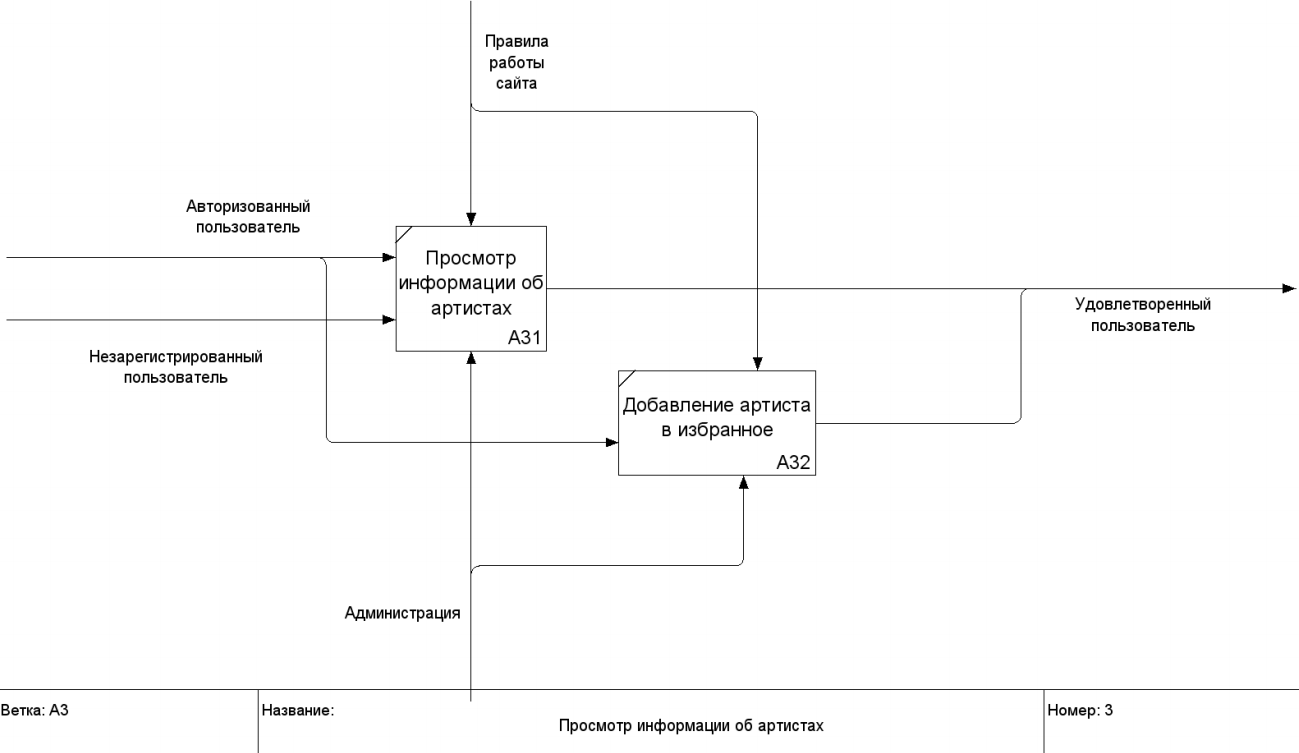
IDEF0 модель отображает структуру и функции приложения, а также потоки информации и материальных объектов, преобразуемые этими функциями.



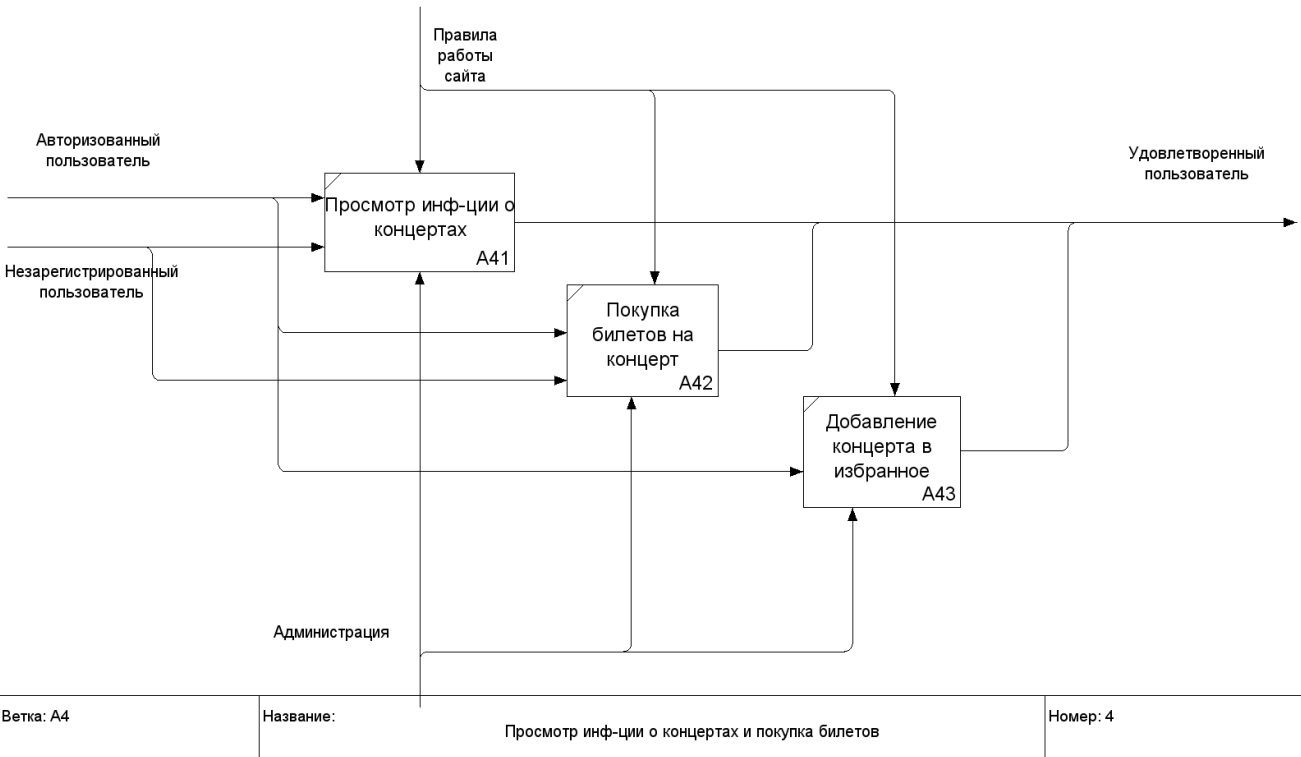
1. Модель IDEF0 – уровень A-0



1. Модель IDEF0 – уровень A0

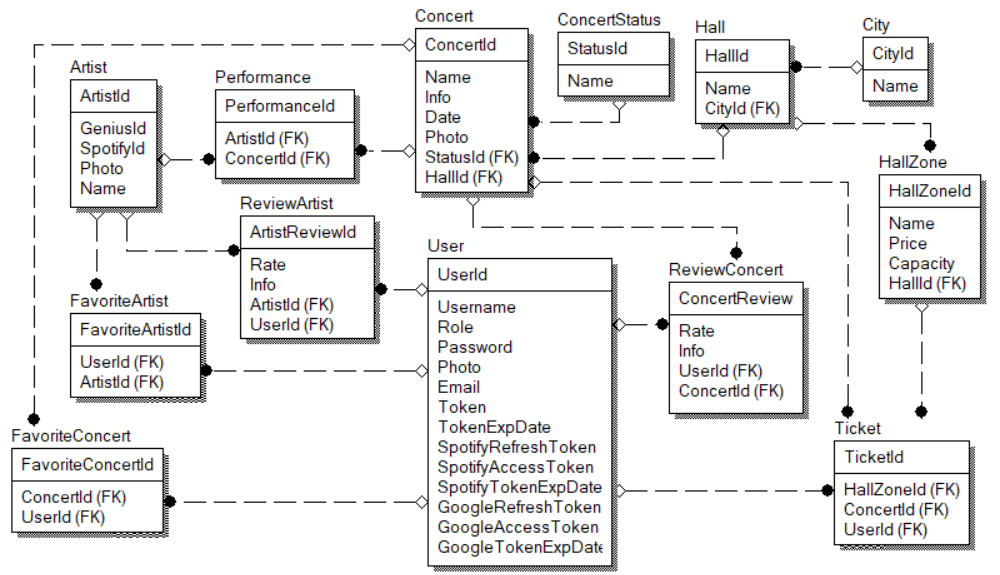


1. Модель IDEF0 – уровень A3 (просмотр информации об артистах)



1. Модель IDEF0 – уровень A4 (просмотр информации о концертах и покупка билетов)
   1. ER-диаграмма

Данная диаграмма является представлением базы данных в виде наглядных графических диаграмм.



1. ER-диаграмма проекта
2. Реализация
   1. Средства реализации

Использованы следующие элементы:

* Язык программирования Python;
* Flask Framework;
* База данных PostgreSQL;
* Шаблонизатор Jinja2;
* Google OAuth 2.0,

а также дополнительные библиотеки.

Существуют следующие алгоритмы работы программы:

* Авторизация на сайте;
* Поиск концертов;
* Просмотр информации о концертах и артистах;
* Редактирование информации о концертах и артистах;
* Редактирование информации о рецензиях;
* Редактирование информации о пользователях.

Идея состоит в отклике приложения на действия пользователя, в некоторых случаях (просмотр данных о концерте, о местах и т.д.) обращение к базе данных, в некоторых – использование сторонних api (Spotify, Genius – при просмотре информации о артистах), а также отображении запрашиваемой информации на странице. Отображение производится с помощью фреймворка Flask.

Проект поделён на 2 части – фронтенд и бэкенд.

* + 1. Фронтенд

Фронтенд содержит в себе более 15 шаблонов, которые отражают визуальную составляющую сайта, также в этих шаблонах при работе приложения отображаются данные. Предусмотрены специальные шаблоны для ситуаций, когда человек пытается ввести не ту ссылку.

Важной частью фронтенда является app.py, элемент приложения, содержащий в себе скрипты для обработки действий пользователя, которые помогают пользователю как управлять отображаемыми данными, так и вообще позволяют приложению работать на стороне клиента [1].

Отдельными элементами являются google\_front.py и spotify\_front.py, которые с помощью реализации методов интегрируют возможности сторонних сервисов в текущее приложение на стороне клиента.

* + 1. Бэкенд

Следующая часть приложения располагается на сервере, обрабатывает запросы со стороны клиента и отвечает за удалённый доступ и хранение данных о концертах, артистах, залах и так далее. Состоит из непосредственно базы данных, также имеется конфигурационные файлы доступа к базе данных.

Файл models.py предоставляет объекты [2], на которые маппятся записи базы данных, и с которыми можно взаимодействовать из приложения. В этой части приложения используется ORM SQLAlchemy, которая позволила абстрагироваться от специфики баз данных и управлять ей непосредственно конструкциями языка Python.

Есть также элементы, добавляющие поддержку сторонних сервисов – Spotify, Genius, Google Calendar. auth\_utils.py содержит методы, связанные с авторизацией пользователя, tickets.py – с покупкой билетов.

* 1. Инструментальные средства, использованные для разработки приложения.

При разработке были задействованы следующие средства:

Текстовые процессоры: Стандартный блокнот Windows, Notepad++.

Веб-браузеры: Google Chrome, Microsoft Edge, Mozilla Firefox, Yandex Browser, Chromium.

Для работы с PostgreSQL был использован графический компонент

pgAdmin4.

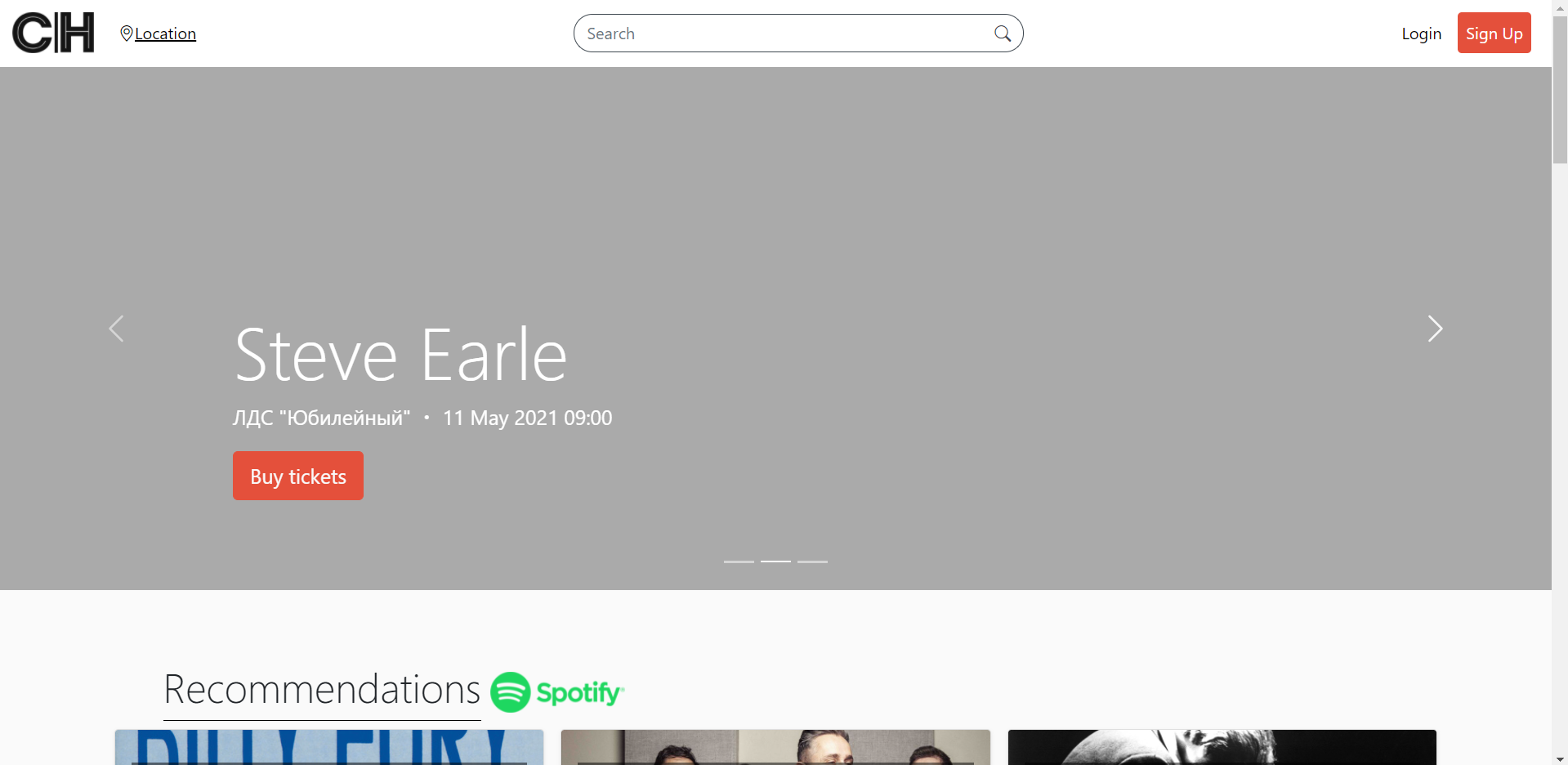
IDE: PyCharm Professional с активной лицензией для студентов и преподавателей.

Для развёртывания сайта используется платформа Heroku.

* 1. Описание функционала приложения и интерфейса пользователя.

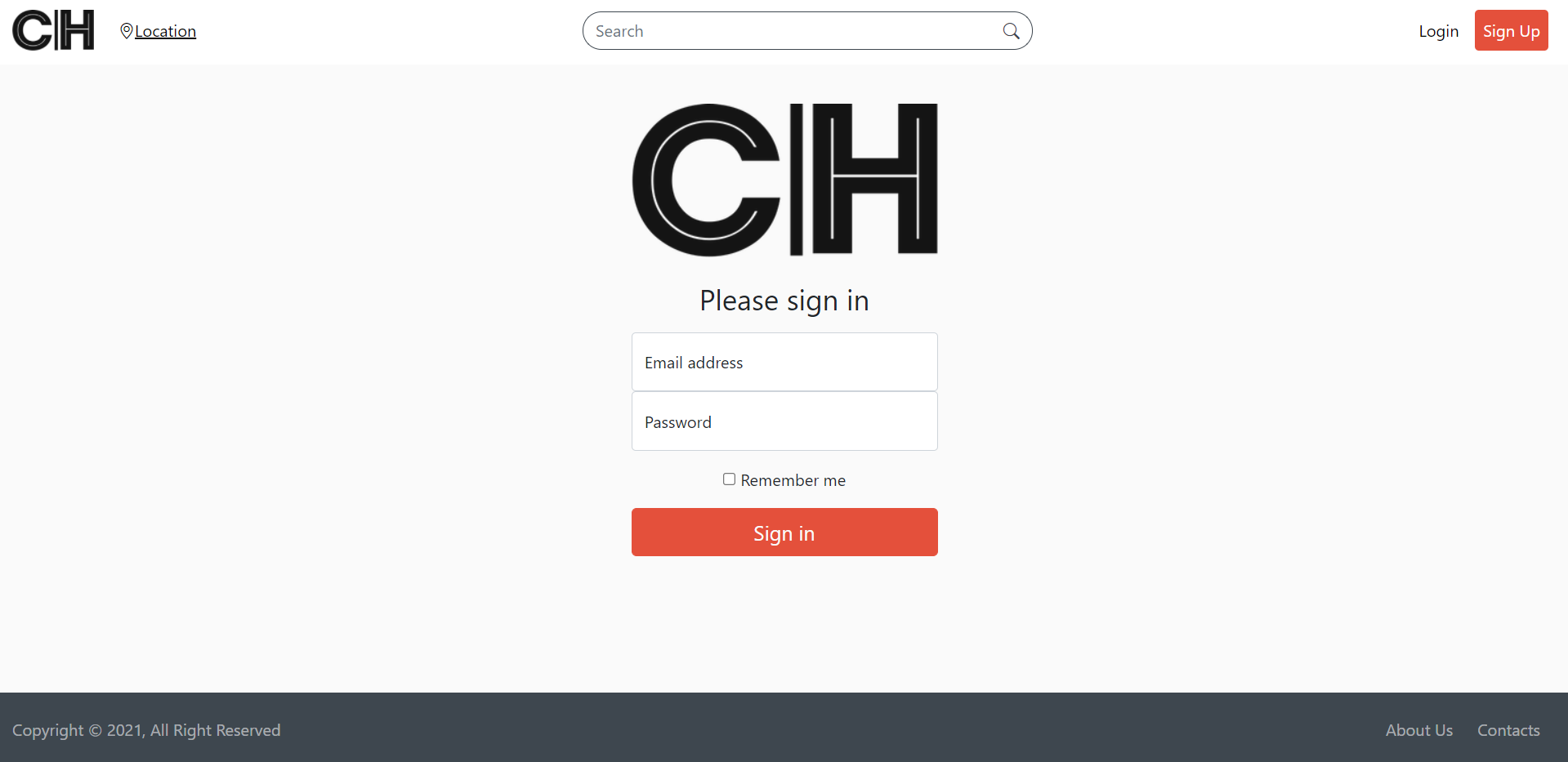
Реализованный функционал соответствует поставленной задаче – реализован функционал и интерфейс, позволяющий произвести поиск и покупку билета.

На следующем рисунке представлен изначальный экран, на который попадает возможный пользователь сайта.



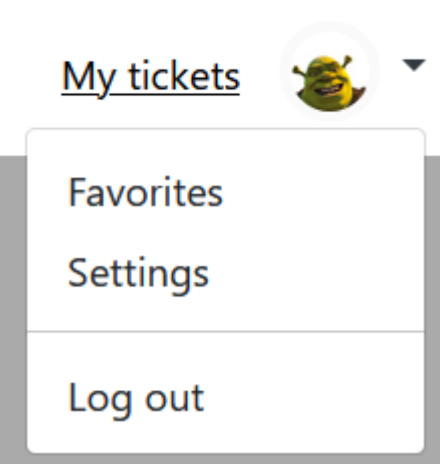
1. Начальное меню

Данное меню позволяет перейти на страницы, обозреваемые далее, одна из них – страница авторизации:

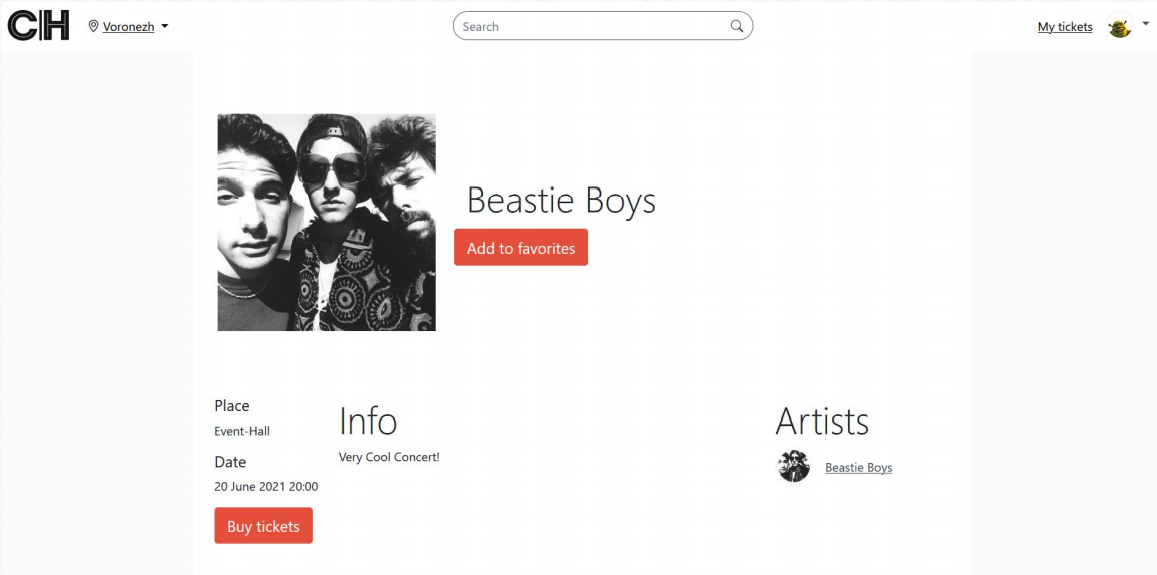


1. Страница авторизации

с помощью которой можно войти в личный кабинет. Страница регистрации имеет схожий интерфейс, и предоставляет возможность создать свой аккаунт в приложении.

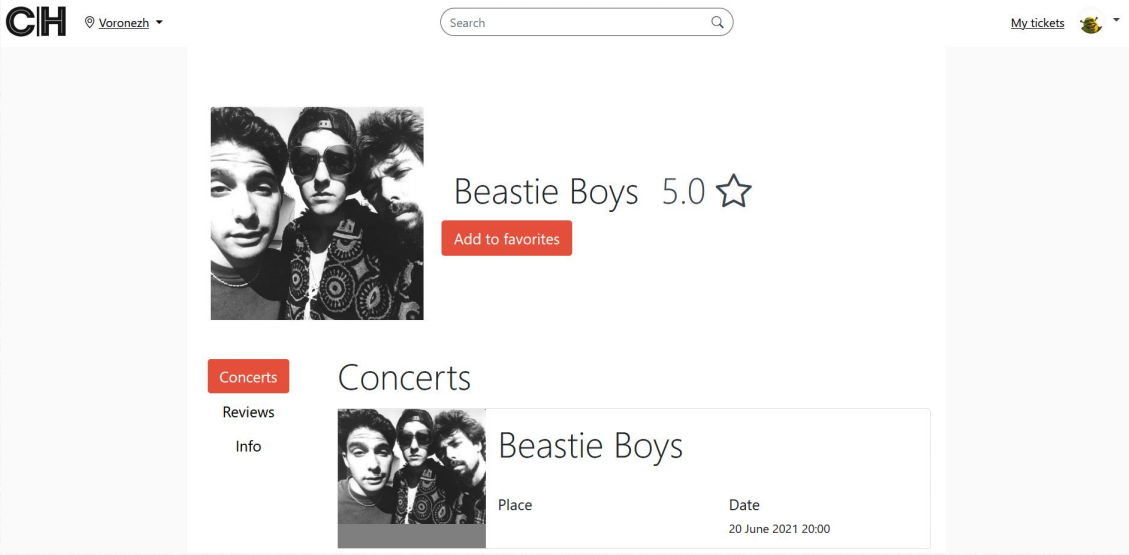


1. Выпадающий список для авторизованного пользователя

У авторизованного пользователя имеется выпадающий список, который ведёт в настройки и к списку избранного, также здесь можно выйти из аккаунта. 

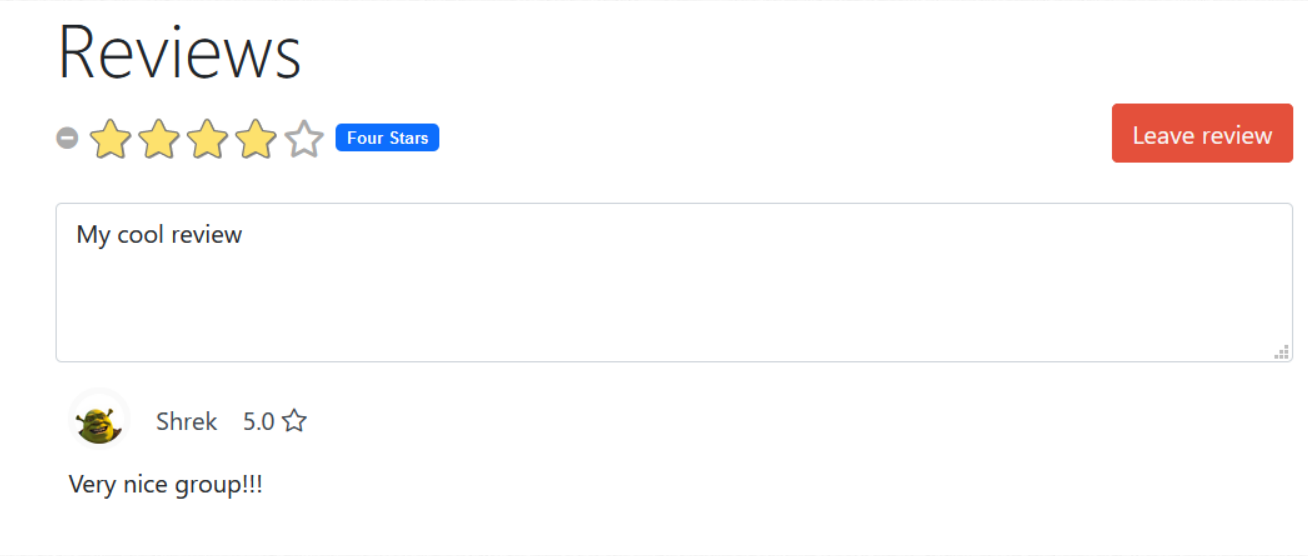
1. Страница концерта

С главной страницы имеется возможность также перейти на страницу с концертом, где можно посмотреть информацию о нём, в том числе и отзывы.



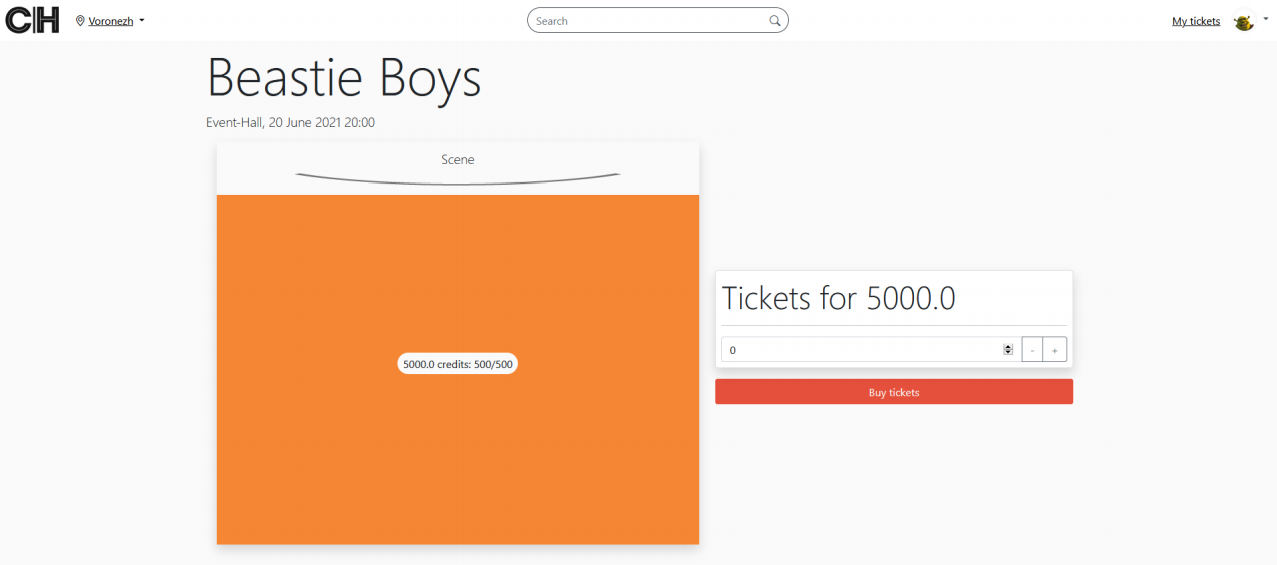
1. Страница артиста со списком концертов

В концерте отмечены участвующие артисты, если перейти по ссылке конкретного артиста, можно увидеть информацию о нём, а также список концертов, в которых он участвует.



1. Блок комментариев

Концерты и артистов можно оценивать и оставлять комментарии, более подробно описывающие мнения пользователей. Таким образом составляется оценка концерта или артиста публикой.



1. Страница выбора и покупки билетов

Со страницы концерта можно перейти в меню покупки, где можно приобрести билеты, в интерфейсе зона посадки, цена и поле для выбора количества билетов.

Страница настроек пользователя необходима для кастомизации аккаунта пользователя в системе, также позволяет настроить некоторые пункты под собственные нужды

Страница избранного пользователя нужна для отображения концертов и артистов, которые понравились пользователю. Таким образом он может отобрать понравившееся, которое в последствии не нужно будет снова искать.

Страница билетов пользователя отражает все билеты, купленные пользователем, чтобы он мог посмотреть и проверить их наличие.

1. Тестирование

Для определения работоспособности приложения над приложением было проведено несколько видов тестов.

* 1. Дымовое тестирование

Первым по счёту идёт дымовое тестирование, суть которого заключается в минимальном наборе тестов на явные ошибки.

На данном этапе были проведены и успешно пройдены следующие тесты:

* Авторизация пользователя;
* Получение информации о концертах;
* Получение информации о артистах;
* Получение информации о имеющихся билетах пользователя;
* Оставление рецензии о концерте;
* Покупка билета.
  1. Интеграционное тестирование серверной части

Интеграционное тестирование подразумевает собой проверку объединённых в группу программных модулей приложения. Здесь проводились следующие тесты:

* Регистрация пользователя;
* Удаление пользователя;
* Изменение информации пользователя;
* Получение информации о всех актёрах;
* Добавление информации о актёре;
* Удаление актёра;
* Изменение информации о актёре;
* Авторизация пользователя;
* Создание концерта админом;
* Изменение концерта админом;
* Удаление концерта админом;
* Получение информации о всех концертах;
* Получение детальной информации концерта;
* Оплата билета;

Которые также были пройдены.

* 1. Проверка эргономичности

Последним этапом тестирования будет проверка функций приложения уже живым человеком. Включает следующие тесты:

* Регистрация пользователя;
* Авторизация пользователя;
* Просмотр настроек пользователя;
* Покупка билета;
* Просмотр информации о концерте;
* Просмотр информации о актёре;
* Получение информации о билетах на текущий концерт;
* Оценка актёра;
* Оценка концерта.

Заключение

Подводя итоги, можно сказать, что в ходе работы были выполнены следующие пункты:

* Собрана необходимая информация об исследуемой предметной области;
* Проведён анализ полученных данных;
* Разработаны концепции работы приложения;
* Выбран наилучший вариант и создана модель программы;
* Написан код приложения, отлажен, откорректирован до стабильной работоспособности.

Тема актуальна по целому ряду причин, вот некоторые из них:

* Активное развитие информационных технологий;
* Повышение качества жизни людей, и, как следствие, увеличение их покупательной способности;
* Возможная реализация перспектив, которые приведены далее.

Несомненно, можно продолжить развивать данную тему, курсовая работа имеет перспективы, которые заключаются в следующих пунктах:

* Дальнейшее исследование темы;
* Развитие функционала приложения;
* Использование новых стеков технологий;
* Более глубокая оптимизация приложения.

Список используемой литературы:

1. McConnell S. Code Complete: A Practical Handbook of Software Construction / Steve McConnell. – 2nd Edition, QA75.5-76.95.
2. Swaroop C. A Byte of Python / C.H. Swaroop. – 2013.
3. Shaw Z. Learn Python the Hard Way: A Very Simple Introduction to the Terrifyingly Beautiful World of Computers and Code (Zed Shaw's Hard Way Series) / Zed Shaw. – 3rd Edition, September 19, 2013.
4. Percival H. Test-Driven Development with Python: Obey the Testing Goat: Using Django, Selenium, and JavaScript / Harry Percival – 1st Edition, June, 2014.
5. Grinberg M. Flask Web Development: Developing Web Applications with Python / Miguel Grinberg. – 2nd Edition, 2014.
6. Maia I. Building Web Applications with Flask / Italo Maia. – June 19, 2015.
7. Cunningham K. Python in 24 Hours, Sams Teach Yourself / Katie Cunningham. – Pearson Education, 2013. – 320 p.
8. Merkel D. Docker: lightweight Linux containers for consistent development and deployment / Dirk Merkel. – 1st Edition, 2014.
9. Mouat A. Using Docker: Developing and Deploying Software with Containers / Adrian Mouat – 1st Edition, December 9, 2015.
10. Copperwaite M., Leifer C. Learning Flask Framework. Build dynamic, data driven websites and modern web applications with Flask / Matt Copperwaite. – Packt Publishing, 2015.
11. Ramalho L. Fluent Python: Clear, Concise, and Effective Programming / Luciano Ramalho. – 1st Edition, 2015.
12. Reitz K. The Hitchhiker's Guide to Python: Best Practices for Development / Kenneth Reitz. – 1st Edition, 2016.
13. Beazley D. Python Essential Reference / David Beazley. – 4th Edition, Addison-Wesley Professional, 2009.
14. Larman C. Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development / Craig Larman. - 3rd Edition, 1997.
15. Booch G. The Unified Modeling Language User Guide (Object Technology Series) / Grady Booch. - 2nd Edition, 1998.