МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”

Факультет *компьютерных наук*

Кафедра *технологий обработки и защиты информации*

Сайт по поиску музыкантов и групп «Believe»

Курсовой проект

09.03.02 Информационные системы и технологии

Обработка информации и машинное обучение

Допущен к защите

Обучающийся _____М.О. Саввин, *3 курс, д/о*

Обучающийся _____Е.А. Стеблева, *3 курс, д/о*

Обучающийся _____М.А. Дынин, *3 курс, д/о*

Воронеж 2019

Содержание

Введение

В современном интернете есть множество различных сайтов и сервисов, которые упрощают людям поиск работы, недвижимости, автомобиля, вещей и многого другого. Но любому человеку нужен отдых, и самое очевидное и первостепенное, о чем думает каждый — это хобби. По статистике 28% людей всего мира считают своим основным хобби музыку. В настоящее время существуют группы и/или паблики в социальных сетях, форумы, посвященные поиску музыкантов в каком-либо конкретном городе, но у каждого из таких форумов различные функциональные возможности, разный интерфейс.

Желаемый сайт должен облегчить пользователям поиск музыкантов в группу и наоборот в любом городе, который их интересует. Тем самым необходимости в просмотре большого количества форумов и сайтов больше не будет. Желаемый сайт должен предоставлять основную необходимую функциональность:

- Поиск музыкантом группы
- Поиск группой музыканта
- Ненагруженный, интуитивно понятный даже низкоуровневому пользователю интерфейс.

Данный курсовой проект посвящен разработке именно такого, простого в освоении, но в то же время выполняющего самые необходимые функции, сайта, способного уменьшить временные затраты каждого человека на поиск необходимых музыкантов и/или групп.

1. Постановка задачи

Цель курсовой работы: реализовать сайт, который отвечает следующим требованиям:

- Интуитивный пользовательский интерфейс;
- Отсутствие нагромождений;
- Отсутствие броских цветов;
- Возможность выполнения основных задач сайта:
  - Просмотр заявок музыкантов на поиск группы и возможность откликнуться на каждую;
  - Просмотр заявок групп на поиск музыканта и возможность откликнуться на каждую;
  - Создание заявки на поиск музыканта/группы с возможность ее вывода в топ;
  - Редактирование личных данных и созданных заявок;
- Возможность перехода на все страницы сайта с главного экрана;

Для достижения данной цели были выделены следующие задачи:

1. Разработка Front-end части сайта, находящиеся на телефоне/компьютере пользователя;
2. Разработка Back-end части сайта, развернутой на удаленном сервере сайта;
3. Создание связи между Front-end и Back-end частями сайта;
4. Разработка базы данных, расположенной на удаленном сервере.

2. Анализ предметной области

2.1 Глоссарий

VIP-код — последовательность символов, вводимая при создании заявки, с помощью которой пользователь может вывести ее (заявку) в топ.

Заявка — заполненная пользователем информация, необходимая и достаточная для поиска и отбора музыкантов и групп.

Топ — расположение заявок в начале списка всех заявок.

Исполнитель — сольный артист (музыкант) или группа.

2.2 Анализ существующих решений

1. https://myband.ru/

Достоинства:

- Большое разнообразие функций и возможностей
- Красивое оформление
- Возможность загрузки своих аудио-работ
- Синхронизация с различными соц. сетями

Недостатки:

- Ограниченное количество критериев для фильтрации
- Отсутствие структуры составления заявки
- Отсутствие кроссбраузерности

2. https://muzlk.com/

Достоинства:

- Присутствие личных сообщений
- Возможность добавления своих работ в формате аудио и видео
- Формат соц. сети

Недостатки:

- Плохая оптимизация и проблемы с входом на сервис
- Неоднозначный дизайн и подбор цветов
- Перебои в работе сайта
- Отсутствие модерирования
- Отсутствие возможности редактирования личного кабинета

3. https://www.realrocks.ru/

Достоинства:

- Приятный дизайн
- Большое разнообразие жанров
- Синхронизация с различными соц. сетями

Недостатки:

- Отсутствие возможности поиска группы

- Присутствие чужой рекламы на странице в случае обычного аккаунта
- Ограниченное количество критериев для фильтрации

2.3 Анализ задачи

2.3.1 Варианты использования приложения

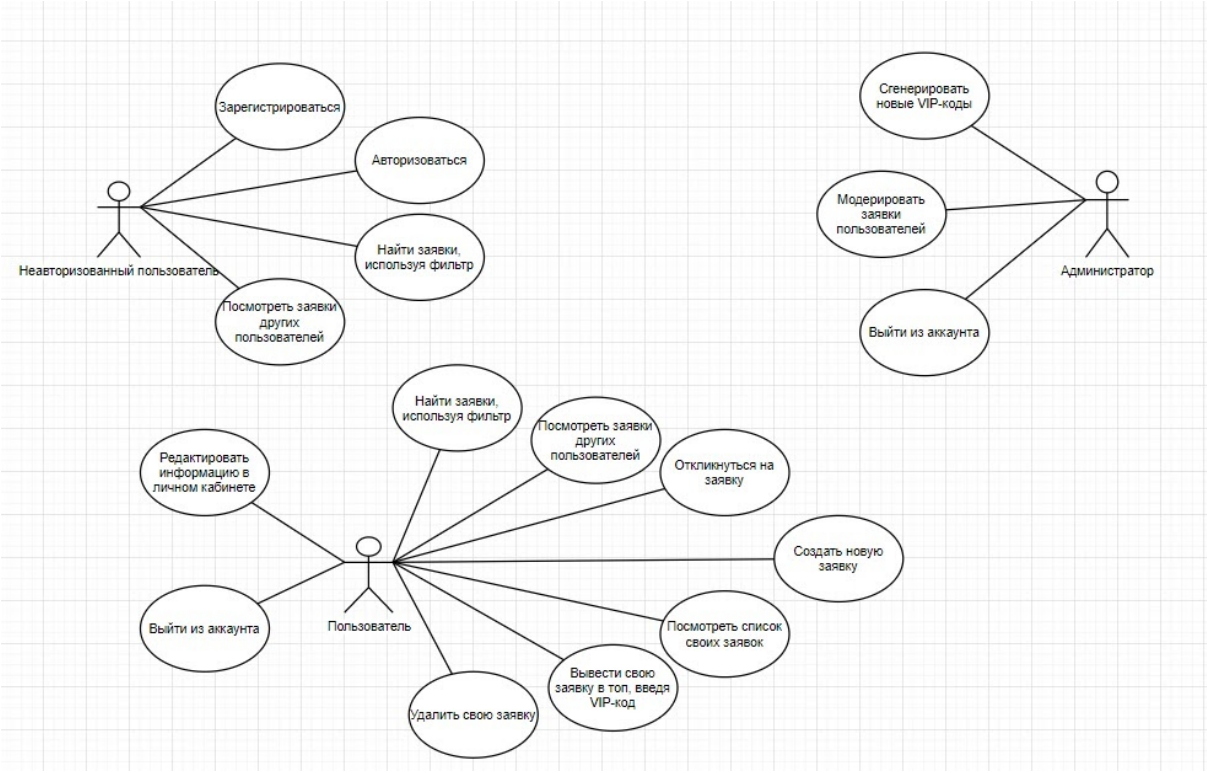

Рисунок 1. Диаграмма прецедентов.

При взаимодействии с сайтом у пользователя есть определенный список возможностей, который более наглядно изображен на рисунке 1:

- Поиск заявок, используя фильтр
- Просмотр заявок других пользователей
- Отклик на заявку
- Создание заявки
- Просмотр списка своих заявок
- Вывод заявки в топ
- Удаление заявки

- Выход из аккаунта
- Редактирование информации в личном кабинете

Неавторизованный пользователь имеет возможность:

- Зарегистрироваться
- Авторизоваться
- Найти заявки, используя фильтр
- Посмотреть заявки других пользователей

Администратор сервиса может:

- Сгенерировать VIP-коды
- Модерировать заявки пользователей
- Выйти из аккаунта

2.3.2 Взаимодействие компонентов системы

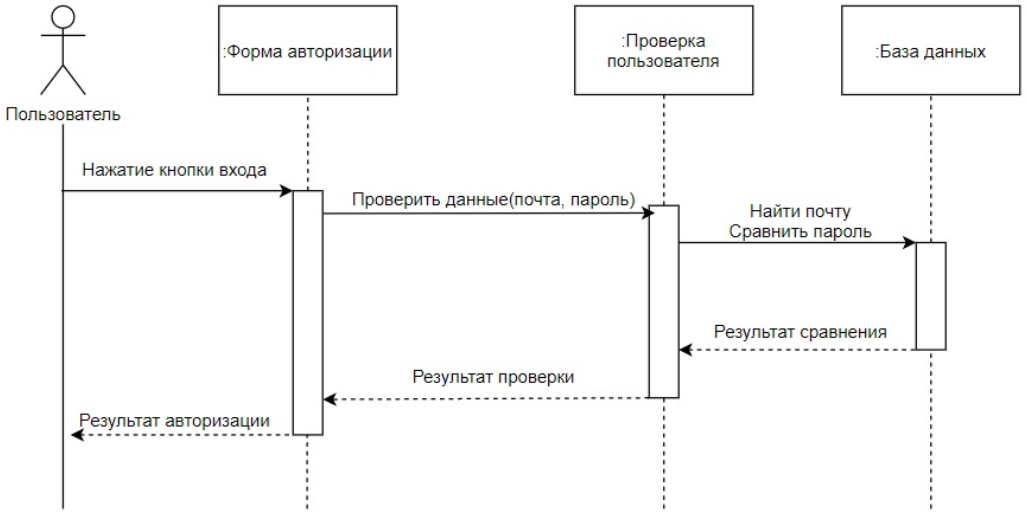

Рисунок 2. Диаграмма последовательностей.

На рисунке 2 показана диаграмма последовательности, на которой изображено упорядоченное во времени взаимодействие объектов при авторизации пользователя.

Для авторизации пользователь обращается к форме авторизации, которая передаёт введённые данные на проверку в модуль проверки пользователя. Тот в свою очередь проверяет существование данного пользователя в базе данных и совпадение введённого пароля с паролем, хранящимся в базе данных. Модуль проверки пользователя посылает статус проверки на форму авторизации, которая выводит пользователю результат авторизации.

2.3.3 Варианты состояния системы

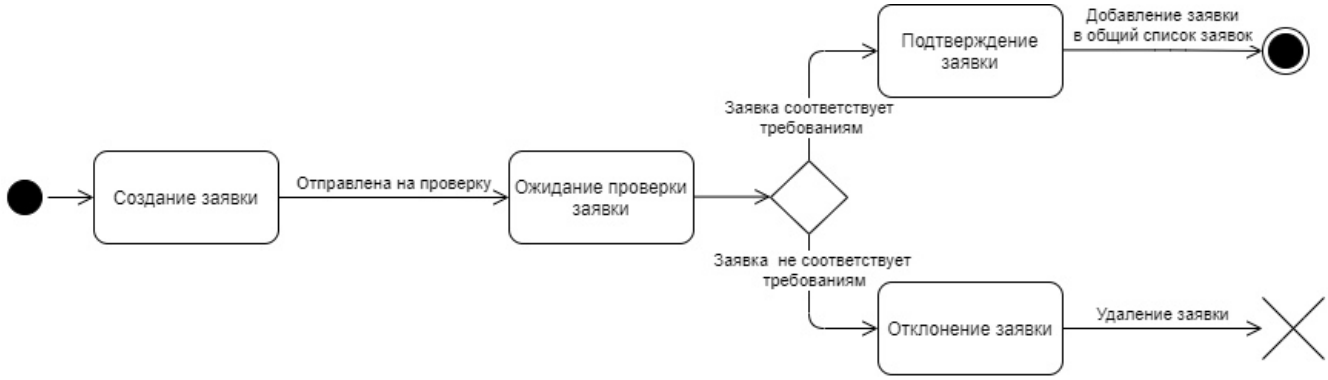

Рисунок 3. Диаграмма состояний.

Диаграмма состояний, изображенная на Рисунке 3, отражает возможные состояния заявки. После создания заявки она отправляется на проверку администратором и переходит в состояние ожидания проверки. Если заявка соответствует требованиям (проходит проверку), то она переходит в состояние подтверждения (является подтвержденной) и добавляется в общий список всех заявок нужной категории. Если заявка не соответствует требованиям (не проходит проверку), то она переход в состояние отклонения (является отклоненной) и удаляется из списка ожидающих проверку заявок.

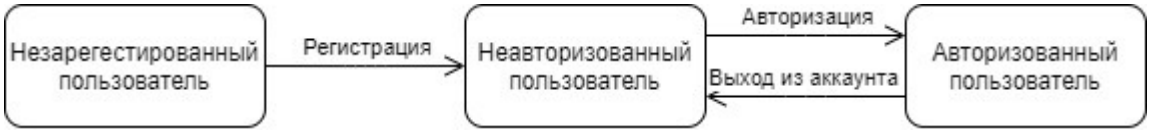

Рисунок 4. Диаграмма состояний.

Диаграмма состояний, изображенная на Рисунке 4, отражает возможные состояния пользователя. Изначально любой пользователь, не прошедший регистрацию, находится в состоянии незарегистрированного пользователя. После прохождения регистрации пользователь переходит в состояние неавторизованного пользователя. После авторизации пользователь переходит в состояние авторизованного пользователя. Если пользователь выходит из аккаунта он возвращается к состоянию неавторизованного пользователя и может авторизоваться снова.

2.3.4 Варианты действия в системе

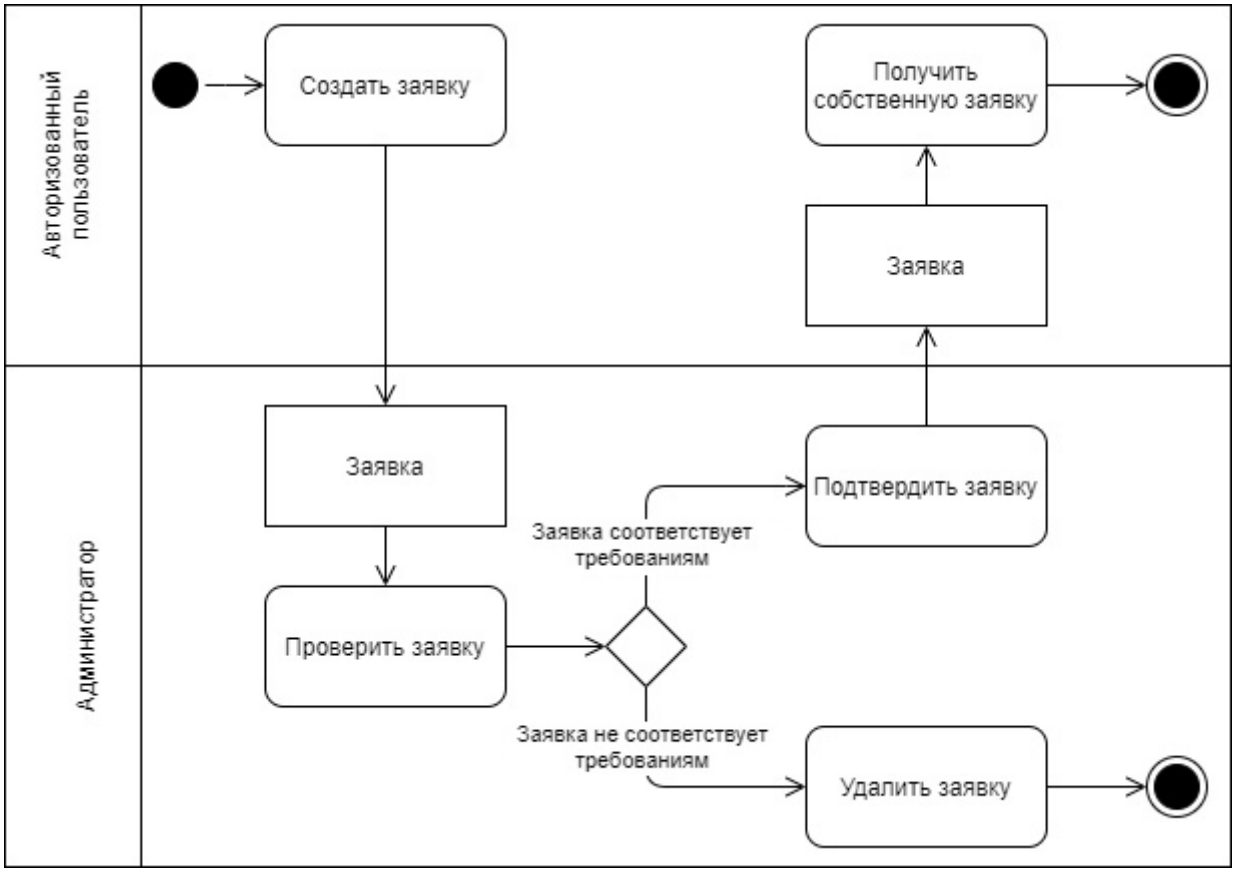

Рисунок 5. Диаграмма активности.

Диаграмма активности, изображенная на Рисунке 5, отражает действия авторизованного пользователя и администратора при создании заявки. После того, как авторизованный пользователь создает заявку, она переходит к администратору. При получении заявки администратор должен ее проверить. Если заявка не соответствует требованиям, то администратор удаляет ее из списка ожидающий проверку заявок. Если заявка соответствует требованиям, то администратор подтверждает ее, и она (заявка) переходит тому авторизованному пользователю, который ее создал. Авторизованный пользователь получает собственную заявку, и она в свою очередь закрепляется за ним.

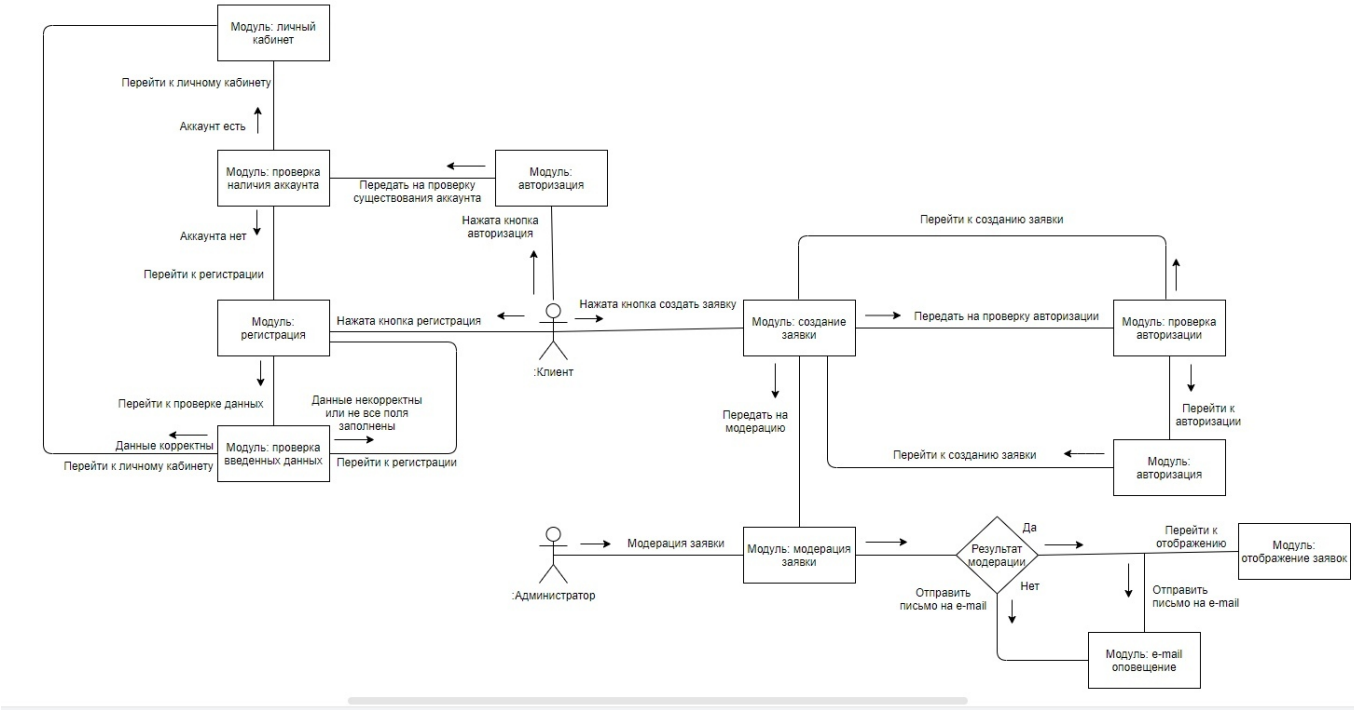

Рисунок 6. Диаграмма взаимодействий.

На Рисунке 6 представлена диаграмма взаимодействий. Она отражает возможные действия пользователя и системы.

Если пользователь захочет оставить заявку, заполнив анкету, система проверит статус авторизации клиента, если он не авторизирован, то система предупредит его об этом и предоставит возможность авторизоваться. Если пользователь авторизован, заявка попадает на модерацию, где администратор проверяет корректность предоставленных пользователем данных и выносит вердикт. Если заявка одобрена, то клиент получает уведомление об одобрении заявки, и она появляется в списке всех заявок, в случае отказа, пользователь получает уведомление об отказе с указанием причины.

Если пользователь захочет пройти процесс регистрации, заполнив необходимые поля, система проверит входящие данные на корректность и в случае ошибки выдаст предупреждение. Если данные корректны, пользователь перейдет в личный кабинет.

При желании пользователя авторизоваться, заполнив необходимые поля,

система так же проверит корректность введенных данных, в случае, если пользователя с такими данными нет, система предложит зарегистрироваться.

2.3.5 Развертывание приложения

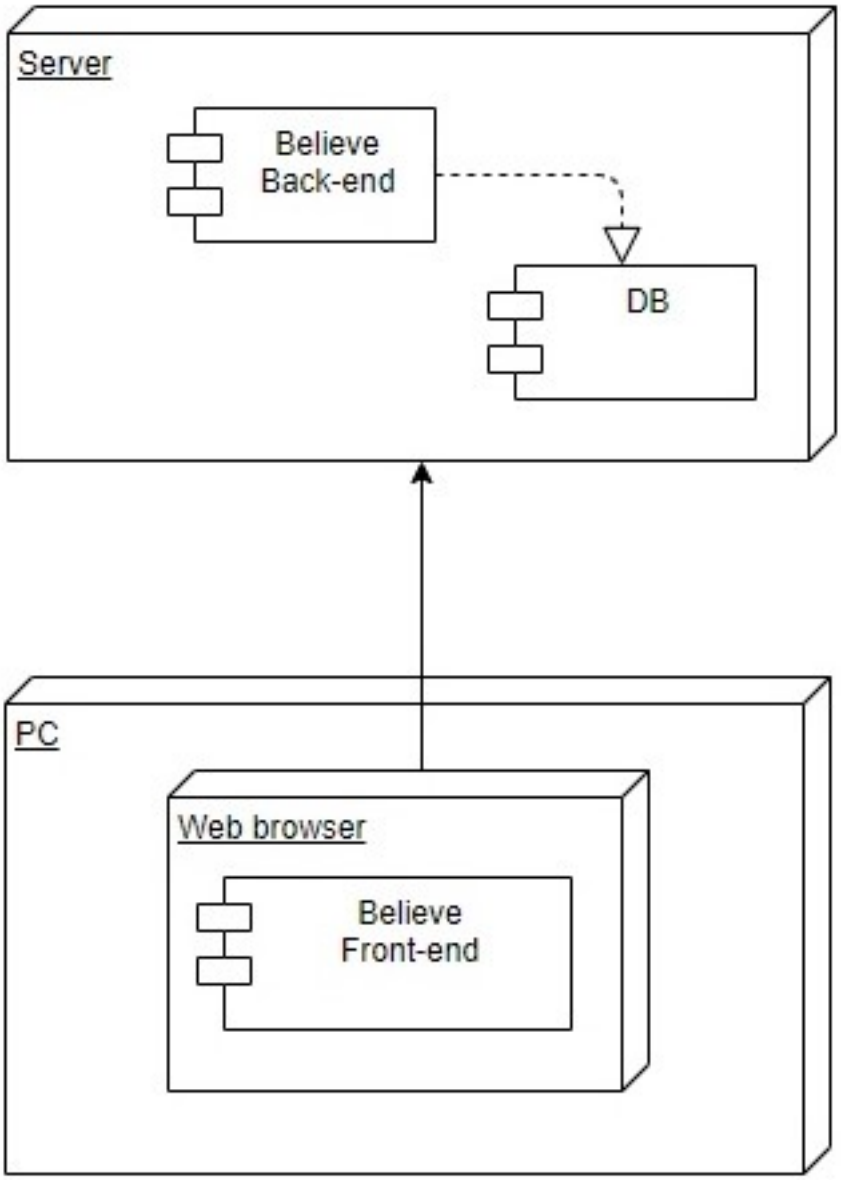

Рисунок 7. Диаграмма развертывания.

На Рисунке 7 представлена диаграмма развертывания, чтобы определить какие аппаратные компоненты («узлы») существуют, какие программные компоненты работают на каждом узле и как различные части этого комплекса соединяются друг с другом. Для разрабатываемого web-приложения узлом устройства является персональный компьютер и сервер, а в качестве узла среды выполнения выступает web-браузер. В браузере развернут front-end приложения, а на серверной части back-end и база данных.

3. Анализ средств реализации

В качестве средств реализации приложения были выбраны следующие технологии:

- HTML, CSS, JS и сторонние библиотеки JS – web-ориентированные языки HTML и CSS необходимы для разработки front-end части. JS и сторонние библиотеки дают доступ к огромному количеству инструментов, заточенных под разные задачи и упрощающие процесс разработки.
- В качестве СУБД была выбрана MySQL. Она является хорошо масштабируемой, в равной степени легко может быть использована для работы, как с малыми, так и с большими объемами данных. А за счет упрощения некоторых используемых в ней стандартов система имеет высокую производительность.
- В качестве языка разработки в back-end использовался язык PHP и библиотека Redbean.

4. Реализация

4.1 Сущности

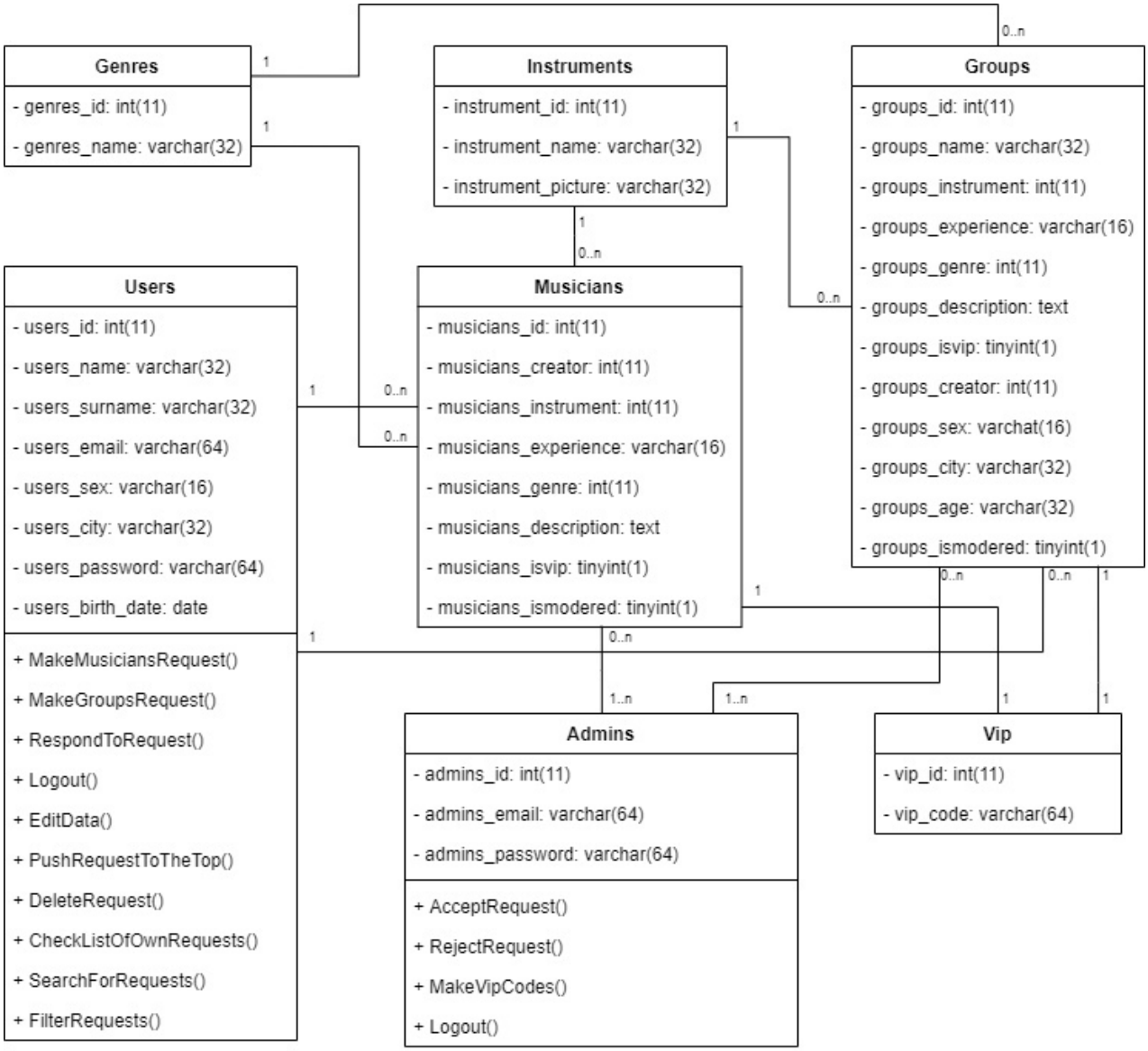

Рисунок 8. Диаграмма классов.

На Рисунке 8 изображена диаграмма классов, отражающая их отношения.

1. Класс «Genres» – представляет собой отражение сущности жанр. Класс имеет следующие свойства:
   • «genres_id» – уникальный идентификатор.
   • «genres_name» – название жанра.

   2. Класс «Instruments» – представляет собой отражение сущности инструмент. Класс имеет следующие свойства:

• «instrument_id» – уникальный идентификатор.
• «instrument_name» – название инструмент.
• «instrument_picture» – название картинки инструмента.

3. Класс «Vip» – представляет собой отражение сущности вип-код. Класс имеет следующие свойства:
• «vip_id» – уникальный идентификатор.
• «vip_code» – вип-код.
4. Класс «Musicians» – представляет собой отражение сущности заявка музыканта. Класс имеет следующие свойства:
• «musicians_id» – уникальный идентификатор.
• «musicians_creator» – создатель заявки музыканта (FK).
• «musicians_instrument» – выбранный инструмент в оставляемой им заявке музыканта (FK).
• «musicians_experience» – опыт игры на выбранном инструменте.
• «musicians_genre» – предпочитаемый жанр (FK).
• «musicians_description» – дополнительное описание.
• «musicians_isvip» – наличие вип-статуса заявки музыканта.
• «musicians_ismodered» – находится ли заявка в модерации.

5. Класс «Groups» – представляет собой отражение сущности заявка группы. Класс имеет следующие свойства:
• «groups_id» – уникальный идентификатор.
• «groups_name» – название группы.
• «groups_instrument» – требуемый группе исполнитель, играющий на данном инструменте(FK).
• «groups_experience» – желаемый опыт игры на выбранном инструменте.
• «groups_genre» – играемый жанр (FK).
• «groups_description» – дополнительное описание.
• «groups_creator» – создатель заявки группы (FK).

• «groups_sex» – желаемый группе исполнитель данного пола.

• «groups_city» – желаемый группе исполнитель, проживающий в данном городе.

• «groups_age» – желаемый группе исполнитель данного возраста.

• «groups_isvip» – наличие вип-статуса заявки группы.

• «groups_ismodered» – находится ли заявка в модерации.

6. Класс «Users» – представляет собой отражение сущности пользователь. Класс имеет следующие свойства:

• «users_id» – уникальный идентификатор.

• «users_name» – имя пользователя.

• «users_surname» – фамилия пользователя.

• «users_email» – email пользователя.

• «users_sex» – пол пользователя.

• «users_city» – город пользователя.

• «users_password» – пароль пользователя.

• «users_birth_date» – дата рождения пользователя.

7. Класс «Admins» – представляет собой отражение сущности администратор. Класс имеет следующие свойства:

• «admins_id» – уникальный идентификатор.

• «admins_email» – email администратора.

• «admins_password» – пароль администратора.

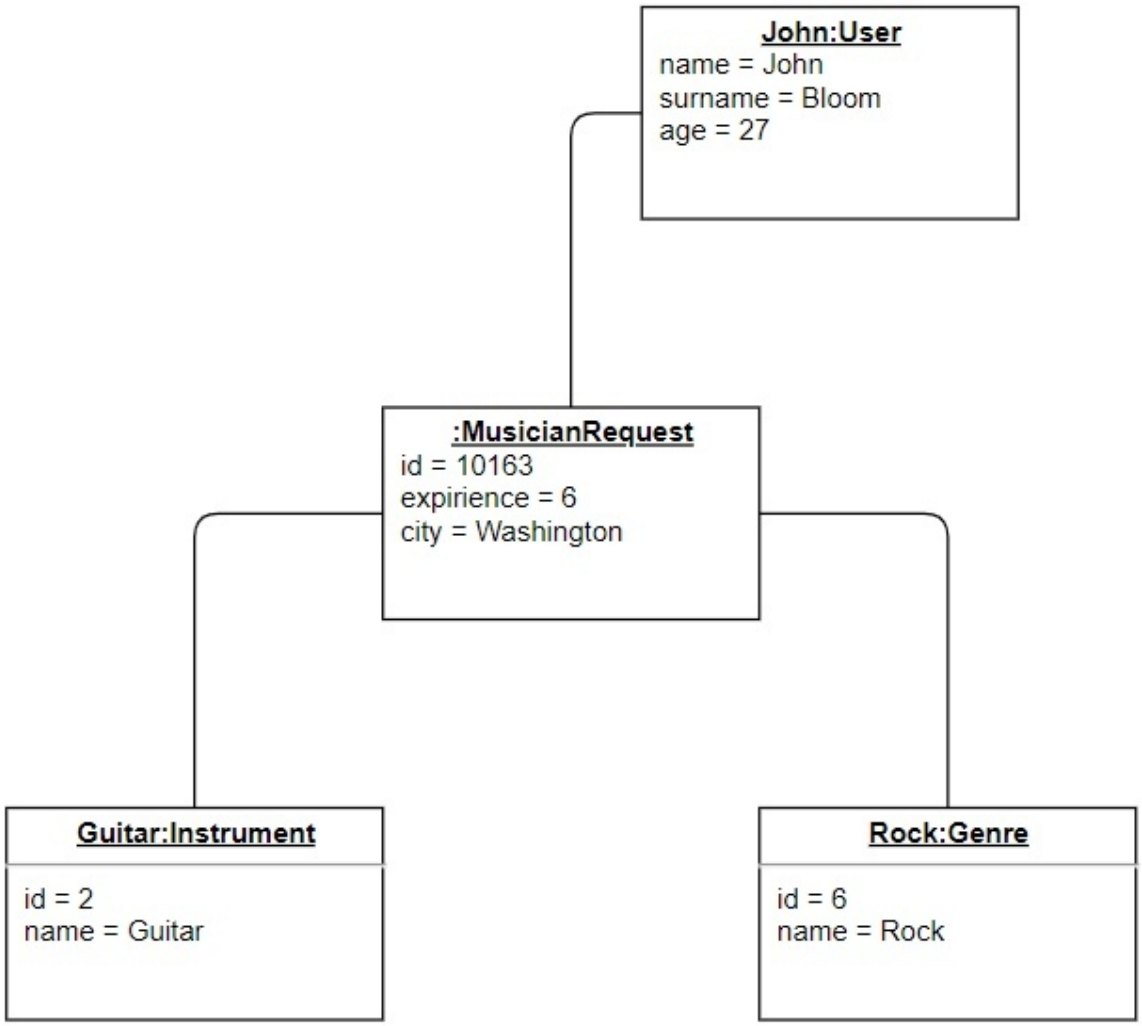

Рисунок 9. Диаграмма объектов.

На Рисунке 9 изображена диаграмма объектов, которая отражает множество экземпляров классов и отношений между ними в некоторый момент времени. На ней изображён экземпляр класса “Авторизованный пользователь”, экземпляр класса “Заявка музыканта”, которая была создана пользователем и хранит в себе ссылку на него. В свою очередь экземпляр класса “Заявка музыканта” содержит в себе поле Инструмент, являющееся экземпляром класса “Инструмент” и поле Жанр, являющееся экземпляром класса “Жанр”, которые также отражены на диаграмме, и показана их связь с другими объектами.

4.2 Сценарии воронок конверсии

1) Посетил главную страницу - Авторизовался - Перешел на страницу создания заявки музыканта - Нажал кнопку "Добавить".
2) Посетил главную страницу - Авторизовался - Перешел на страницу создания заявки группы - Нажал кнопку "Добавить".
3) Посетил главную страницу - Авторизовался - Выбрал нужные фильтры - Нажал кнопку "Найти".