Министерство образования и науки Республики Башкортостан

Государственное автономное профессиональное образовательное учреждение

Уфимский колледж статистики, информатики и вычислительной техники

|  |  |
| --- | --- |
|  | УТВЕРЖДАЮ  Заместитель директора  по учебной работе  \_\_\_\_\_\_\_\_\_\_\_З.З. Курмашева  «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_20\_\_ г. |

Веб-Сервис «MelodyBox»

Пояснительная записка к групповому проекту

МДК.09.03 Разработка кода информационных систем

|  |  |
| --- | --- |
|  | Руководитель проекта  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Файзулова.А.И  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 г. |
|  | Студенты гр. 20ВЕБ-1  А.А.Смирнов, Тихонов К.А, Гилязетдинов А.Р, Махмутов Д.Р, Ильина Н.Д.  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 г. |

2023

АННОТАЦИЯ

Пояснительная записка к групповому проекту содержит постановку и программу решения задачи «Веб-Сервис MelodyBox».

Сайт написан на языке гипертекстовой разметки HTML в среде программирования Sublime Text с использованием сервера баз данных MySQL, предназначена для работы в браузерах, отлажена на данных контрольного примера.

СОДЕРЖАНИЕ

|  |  |
| --- | --- |
|  | лист |
| ВВЕДЕНИЕ | 4 |
| 1 Проектирование информационной системы | 6 |
| 1.1 Описание предметной области | 6 |
| 1.2 Описание входной информации | 7 |
| 1.3 Описание выходной информации  1.4 UML диаграммы | 8  9 |
| 1.5 Концептуальное моделирование | 9 |
| 1.6 Логическое моделирование | 10 |
| 1.7 Описание структуры базы данных | 10 |
| 1.8 Контрольный пример | 13 |
| 1.9 Общие требования к программному продукту | 15 |
| 2 Экспериментальный раздел | 17 |
| 2.1 Описание программы | 17 |
| 2.2 Тестирование программного продукта | 19 |
| 2.3 Руководство пользователя | 36 |
| ЗАКЛЮЧЕНИЕ | 42 |
| СПИСОК ЛИТЕРАТУРЫ  Приложение А Выходные данные  Приложение Б Исходный код  СПИСОК СОКРАЩЕНИЙ | 43  44  45  55 |

ВВЕДЕНИЕ

В эпоху цифровых технологий и широкого распространения интернета, аудиохостинги стали неотъемлемой частью современной культуры и развлечений. Они обеспечивают пользователей доступом к разнообразному музыкальному контенту, поддерживая удобство прослушивания и обмена музыкой.

Цель данной групповой работы заключается в создании основных аспектов проектирования и реализации аудиохостинга, включая структуру базы данных, механизмы авторизации, управление контентом, а также обеспечение безопасности данных. Так же, рассматриваются аспекты пользовательского опыта, такие как создание плейлистов и взаимодействие с артистами.

Кроме того, в процессе выполнения групповой работы будут рассмотрены исключительные аспекты, связанные с взаимодействием музыкантов и аудитории на аудиохостинге. Это может включать в себя создание профилей для артистов, механизмы публикации их музыкальных работ.

Обобщение результатов исследования и практической работы позволит сформулировать выводы относительно эффективности разработанного аудиохостинга и предложить рекомендации для дальнейшего совершенствования подобных платформ в контексте современной цифровой музыкальной индустрии.

1. Проектирование информационной системы
   1. Описание предметной области

Требуется разработать веб-сервис аудиохостинг «MelodyBox».

Аудиохостинг «MelodyBox» это веб-платформа, сервис, предназначенный для хранения, управления и распространения аудио-контента в цифровом формате.

В системе предусмотрены роли:

* пользователь;
* артист/музыкант;
* администратор.

При входе на сайт клиент увидит главную страницу, на которой находятся популярные артисты и их музыка о которой известно:

* название;
* имя артиста;
* название альбома;
* длительность.

Для того чтобы добавить музыку, артиста в понравившиеся необходимо пройти регистрацию, в которой требуется указать:

* имя;
* фамилия;
* e-mail;
* номер телефона.

Пользователь имеет следующие возможности:

* просматривать основную информацию сайта;
* создать свой плейлист;
* отметить понравившуюся ему музыку, артиста;
* оформить заявку на карту музыканта.

Артист имеет следующие функции:

* добавлять музыку;
* удалять музыку;
* создавать альбом/ы;
* удалять альбом/ы;
* редактировать карту музыканта.

Администратор имеет следующие функции:

* управлять статусами пользователя.

Ограничения системы:

Строится по принципу «что не разрешено, то запрещено».

* 1. Описание входной информации

Входным документом аудиохостинга «MelodyBox» является пользовательская учётная запись в которой указано:

* имя;
* фамилия;
* e-mail;
* номер телефона;
* дата рождения.
  1. Описание выходной информации

Описание выходных документов представлено на таблице 1.3.1.

Таблица 1.3.1 – описание выходных документов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование документа (шифр) | Периодичность выдачи документа | Кол-во экз. | Куда передаются | Итоги |
| Плейлист | При создании плейлиста | Не ограничено | Пользователь | Плейлист |

Форма выходного документа «Плейлист» представлен на рисунке A.1 в приложении A.

* 1. Диаграмма прецедентов

Диаграмма прецедентов – это диаграмма, которая позволяет визуализировать различные типы ролей и их взаимодействия с системой.

Диаграмма прецедентов представлена на рисунке 1.4.1

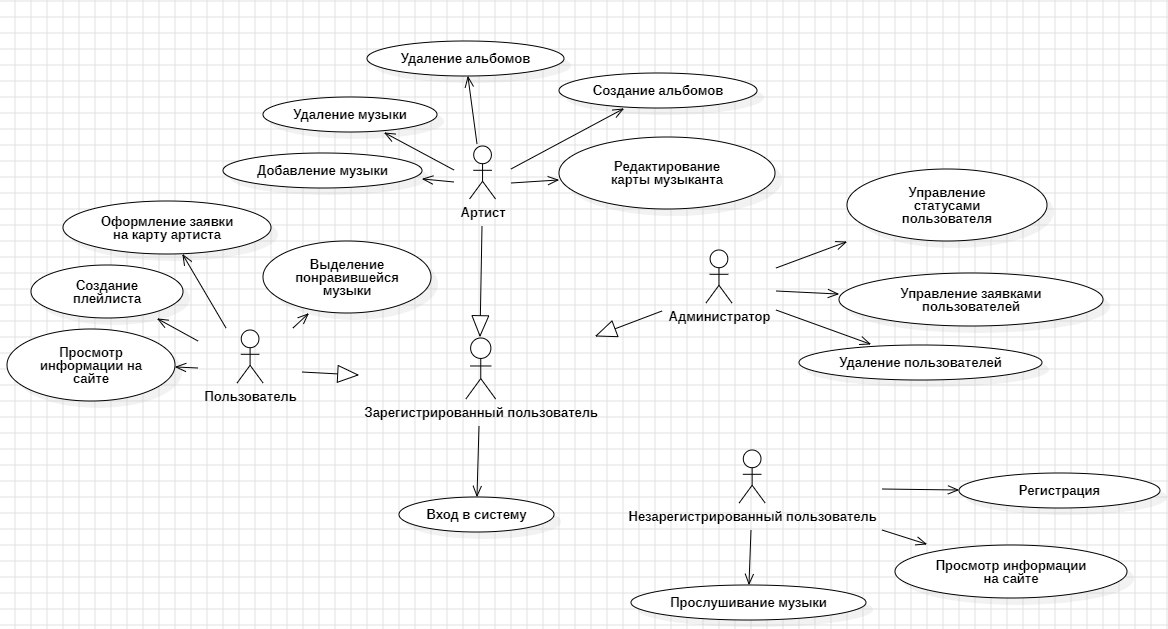


Рисунок 1.4.1 – Диаграмма прецедентов

* 1. Концептуально проектирование

Концептуальная модель – это отражение предметной области, для которой разрабатывается база данных.

Пример ER диаграммы представлен на рисунке 1.5.1.

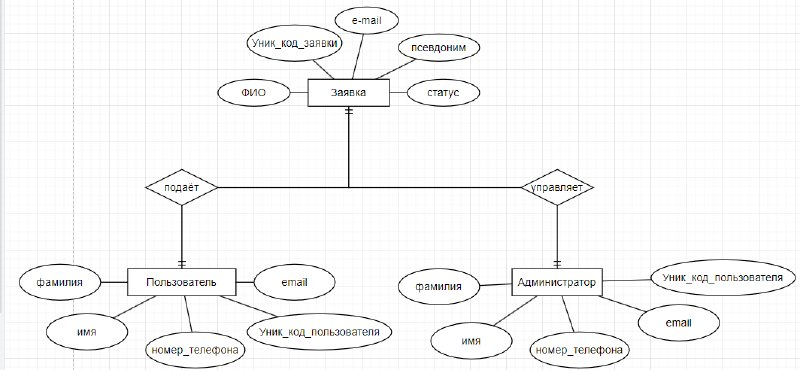


Рисунок 1.5.1 – Концептуальное моделирование

* 1. Логическое проектирование

Логическое моделирование используется для построения модели данных.

Пример ERD - диаграммы представлен на рисунке 1.6.1.

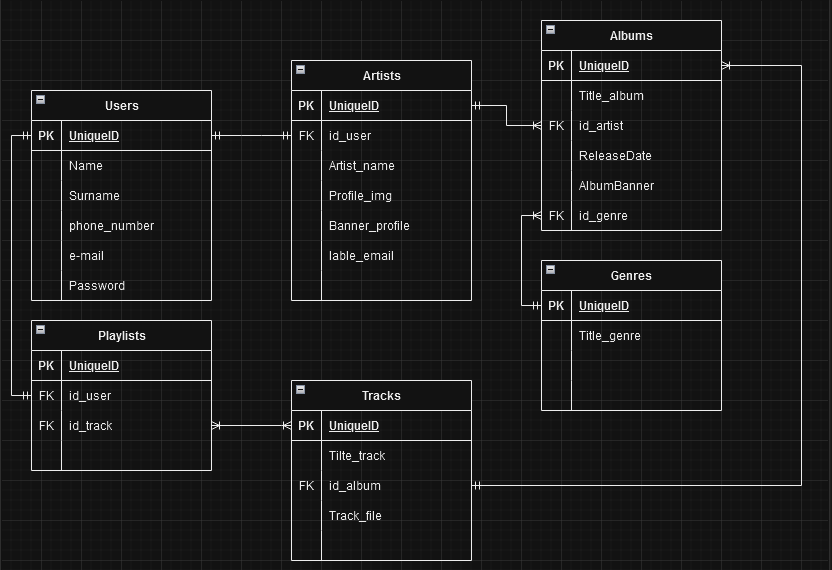


Рисунок 1.6.1 – Логическое моделирование

* 1. Описание структуры базы данных

Описание структуры базы данных представлена в таблицах 1.7.1 – 1.7.6

Таблица 1.7.1 –«users» (пользователи)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Описания поля | Тип данных | Размер поля | Тип ключа (  PK – первичный, FK – внешний) |
| 1 | 2 | 3 | 4 | 5 |
| id\_user | Первичный ключ | INT | 4 | PK |
| Name | Имя | VARCHAR | 255 |  |
| Surname | Фамилия | VARCHAR | 255 |  |
| Phone\_number | Номер телефона | VARCHAR | 255 |  |
| e-mail | почта | VARCHAR | 255 |  |
| Password | пароль | VARCHAR | 255 |  |

Таблица 1.7.2 –«Artists» (артисты)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Описания поля | Тип данных | Размер поля | Тип ключа (  PK – первичный, FK – внешний) |
| 1 | 2 | 3 | 4 | 5 |
| id\_artist | Первичный ключ | INT | 4 | PK |
| id\_user | Вторичный ключ | INT | 4 | FK |
| Artist\_name | Псевдоним | VARCHAR | 255 |  |
| Prodile\_img | Фото профиля | VARCHAR | 255 |  |
| Banner\_profile | Изображение баннера | VARCHAR | 255 |  |
| Lable\_email | Почта компании | VARCHAR | 255 |  |
| Status | Cтатус | VARCHAR | 255 |  |

Таблица 1.7.3 –«Albums» (альбомы)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Описания поля | Тип данных | Размер поля | Тип ключа (  PK – первичный, FK – внешний) |
| 1 | 2 | 3 | 4 | 5 |
| id\_album | Первичный ключ | INT | 4 | PK |
| Title\_album | Название альбома | VARCHAR | 255 |  |
| id\_artist | Вторичный ключ | INT | 4 | FK |
| ReleaseDate | Дата выхода | DATE | 3 |  |
| AlbumBanner | Изображение альбома | VARCHAR | 255 |  |
| id\_album | Вторичный ключ | INT | 4 | FK |

Таблица 1.7.4 –«Genres» (жанры)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Описания поля | Тип данных | Размер поля | Тип ключа (  PK – первичный, FK – внешний) |

Продолжение таблицы 1.7.4 –«Genres» (жанры)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| id\_genre | Первичный ключ | INT | 4 | PK |
| Title\_genre | Название жанра | VARCHAR | 255 |  |

Таблица 1.7.5 –«Tracks» (трэки)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Описания поля | Тип данных | Размер поля | Тип ключа (  PK – первичный, FK – внешний) |
| 1 | 2 | 3 | 4 | 5 |
| id\_track | Первичный ключ | INT | 4 | PK |
| Title\_track | Название трека | VARCHAR | 255 |  |
| id\_album | Вторичный ключ | INT | 4 | FK |
| Track\_file | Файл трека | VARCHAR | 255 |  |

Таблица 1.7.6 –«Playlists» (плэйлист)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Описания поля | Тип данных | Размер поля | Тип ключа (  PK – первичный, FK – внешний) |
| 1 | 2 | 3 | 4 | 5 |
| id\_playlist | Первичный ключ | INT | 4 | PK |
| id\_user | Вторичный ключ | INT | 4 | FK |
| id\_track | Вторичный ключ | INT | 4 | FK |

* 1. Контрольный пример

Таблица 1.8.1 – таблица «users» (пользователи)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Фамилия | Номер телефона | Почта | Пароль |
| Данил | Махмутов | 88005553535 | dmahmutov@mail.ru | admin |
| Альфир | Рахимов | 88005553534 | arahimov@mail.ru | 12345qwerty |
| Арслан | Заиткулов | 88005553533 | arzaitkulov@mail.ru | 12345qwerty |

Таблица 1.8.2 – таблица «Artists» (артисты)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Пользователь | Псевдоним | Фото профиля | Изображение баннера | почта |
| Данил | Cmonia | File1.png | Filebanner1.png | cmonia@mail.ru |
| Арслан | C-Moon | File2.png | Filebanner2.png | cmoon@mail.ru |
| Альфир | Melodian | File3.png | Filebanner3.png | melodian@mail.ru |

Таблица 1.8.3 – таблица «Albums» (альбомы)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название Альбома | Артист | Дата создания | Изображение альбома | Название жанра |
| Bones | Melodian | 24.09.2023 | Bannerimg1.png | Блюз |
| Crimson | Cmonia | 1.10.2023 | Bannerimg2.png | Метал |
| Up in the sky | C-Moon | 2.12.2023 | Bannerimg3.png | Рок |

Таблица 1.8.4 – таблица «Genres» (жанры)

|  |
| --- |
| Название жанра |
| Рок |
| Метал |
| Блюз |

Таблица 1.8.5 – таблица «Tracks» (трэки)

|  |  |  |
| --- | --- | --- |
| Название трека | Альбом | Файл трек |
| Bones | Bones | Bones.mp3 |
| Hero | Up in the sky | Hero.mp3 |
| Hellsinger | Crimson | Hellsinger.mp3 |

Таблица 1.8.6 – таблица «Playlists» (плэйлист)

|  |  |
| --- | --- |
| Пользователь | Трек |
| Данил | Hero |
| Альфир | Hero |
| Арслан | Hellsinger |

* 1. Общие требования к программному продукту
     1. Введение

1. Краткое описание проекта

MelodyBox предназначена для загрузки и прослушивания аудио в браузере через встроенный медиа-плеер.

Основным назначением MelodyBox является хранение и управление аудио-контентом.

1. Цели и задачи разработки

MelodyBox создается с целью:

- предоставления услуг аудио контента пользователю;

- повышения качества (полноты, точности, достоверности, своевременности, согласованности) информации;

1. Ссылки на связанные документы или проекты

ТЗ

* + 1. Функциональные требования

1. Подробное описание функциональности, которую должен предоставлять программный продукт.
2. Входы и выходы системы.

Пользовательская учётная запись, плейлист.

1. Взаимодействие с другими системами.

Отсутствует

* + 1. Требования к интерфейсу

1. Описание пользовательского интерфейса.

- Главная страница:

- Новые релизы;

- Жанры;

- Новые альбомы

- Страница Избранного Плейлиста:

- Треки, добавленные пользователем;

- Информация о треке

- Страница Альбома:

- Баннер альбома;

- Треки, добавленные артистом;

- Страница с музыкой по жанрам:

- Название жанра;

- Треки, относящиеся к этому жанру;

1. Требования к навигации и взаимодействию пользователя с системой.

- Навигационное меню:

- Фиксированное меню для быстрого доступа ко всем разделам (Главная, Новые релизы, Жанры, Авторы, Новые альбомы, Поиск, Авторизация и Регистрация).

- Поиск:

- Возможность быстрого поиска, по ключевым словам, брендам и жанрам музыки.

- Отображение результатов поиска

- Добавление в плейлист:

- Интуитивно понятная кнопка «Лайк» на треке.

- Визуальное подтверждение добавления трека в плейлист.

- Плейлист:

- Возможность прослушивания и редактирования плейлиста.

- Адаптивность:

- Полная поддержка мобильных устройств с удобным интерфейсом для сенсорных экранов.

- Отзывчивый дизайн для различных разрешений экрана.

- Аутентификация и профили:

- Возможность регистрации и авторизации пользователя.

- Личный кабинет с возможностью изменения персональных данных.

- Уведомления:

- Оповещение пользователя о добавлении в корзину, успешном заказе и других важных событиях.

* + 1. Требования к производительности

1. Ожидаемые показатели производительности.

Пропускная способность: 100;

Время ответа сервера: 2мс;

Время отображения: 2мс;

1. Требования к загрузке системы при различных нагрузках.

Запросов в секунду: не более 1000 в секунду;

Пользователей в системе:

Активных – 8000 человек.

Зарегистрированных – 57000 человек.

Одновременно работающих – 6000 человек.

* + 1. Требования к безопасности

1. Требования к управлению доступом и аутентификации.

Разграничение прав доступа пользователей и администраторов Системы должно строиться по принципу "что не разрешено, то запрещено".

1. Меры безопасности для защиты данных.

Клиент должен взаимодействовать с сервером посредством протокола SSL.

1. Требования к обеспечению конфиденциальности.

Требования к обеспечению конфиденциальности основываются на: Закон РФ от 27.12.1991 N 2124-1(ред. от 13.06.2023)"О средствах массовой информации"

1. Экспериментальный раздел
   1. Описание программы

При запуске программного продукта выполняется проект «MelodyBox», после которого выполняются следующие сценарии:

Карта сайта представлена на рисунке 2.1.1.



Рисунок 2.1.1 – Карта сайта

Таблица 2.1.1 – Описание файлов

|  |  |  |
| --- | --- | --- |
| Модуль | Метод | Назначение |
| AdminController.php | index | Открытие главной страницы администратора |
| OrderNew | Вывод новых заявок |
| OrderDeny | Вывод отклонённых заявок |
| OrderAccept | Вывод принятых заявок |
| AcceptApp | Обновление статуса на «Принято» |
| DenayApp | Обновление статуса на «Отклонено» |
| Delete | Удаление артиста |

Продолжение таблицы 2.1.1– Описание файлов

|  |  |  |
| --- | --- | --- |
| IndexController.php | index | Открытие главной страницы сайта |
| TracController | showTrack | Вывод треков |
| addTracks | Добавление трека |
| LikeTrack | Добавление трека в плейлист |
| addLike | Проверка существует ли лайк |
| genresShow | Вывод жанров |
| AuthorShow | Вывод авторов |
| UserController | registration | Регистрация пользователя |
| login | Авторизация пользователя |
| logout | Выход из аккаунта |
| application\_artist | Заявка на артиста |
| updateUser | Редактирование информации пользователя |

* 1. Тестирование программного продукта

Цель тестирования – проверка соответствия программного обеспечения к предъявляемым требованиям, обеспечение уверенности в качестве программного обеспечения, поиск ошибок.

Для выявления ошибок были проведены тесты.

Таблица 2.2.1 – Общая информация о тестировании

|  |  |
| --- | --- |
| Общая информация о тестировании | |
| Название теста | Tests\_MelodyBox |
| Номер версии | 1 |
| Имя тестера | Смирнов Алексей |
| Дата тестирования | 28.06.2023 |

Таблица 2.2.2 - тестирование регистрации при вводе верных данных

|  |  |
| --- | --- |
| Наименование | Описание |
| 1 | 2 |
| Test Case # | T\_REGISTRATION\_1 |
| Приоритет тестирования | Высокий |
| Название тестирования | Проверка регистрации с верными пользовательскими данными |
| Резюме испытания | Необходимо добиться корректного поведения системы, при верных данных при регистрации пользователя |
| Шаги тестирования | 1. Открыть модульное окно регистрации пользователя; 2. Ввести в поля «Имя», «Фамилия», «Номер телефона», «Электронная почта», «Пароль», «Подтверждение пароля» данные; 3. Нажать на кнопку «Создать аккаунт». |
| Данные тестирования | Имя: Кирилл;  Фамилия: Тихонов;  Номер телефона: 8 800 555 35 35;  Электронная почта: example@mail.ru;  Пароль: password;  Подтверждение пароля: password. |

Продолжение таблицы 2.2.2 – тестирование регистрации при вводе верных данных

|  |  |
| --- | --- |
| 1 | 2 |
| Ожидаемый результат | Сайт должен вывести сообщение о верном вводе данных |
| Фактический результат | Вывелось сообщение «Успешная регистрация»  Результат работы показан на рисунке 2.2.1 |
| Предпосылки | Открытие сайта |
| Постусловия | Работа системы происходит корректно |
| Статус (Pass/Fall) | Pass |

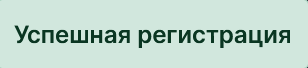


Рисунок 2.2.1 – Результат работы регистрации пользователя при вводе верных данных

Таблица 2.2.3 - тестирование регистрации при вводе неверных данных

|  |  |
| --- | --- |
| Наименование | Описание |
| 1 | 2 |
| Test Case # | T\_ REGISTRATION \_ERROR\_1 |
| Приоритет тестирования | Высокий |
| Название тестирования | Проверка регистрации с неверными пользовательскими данными |
| Резюме испытания | Необходимо добиться корректного поведения системы, при неверных данных при регистрации пользователя |
| Шаги тестирования | 1. Открыть модульное окно регистрации пользователя; 2. Ввести в поля необходимы данные 3. Нажать на кнопку «Создать аккаунт». |

Продолжение таблицы 2.2.3 тестирование регистрации при вводе неверных данных

|  |  |
| --- | --- |
| 1 | 2 |
| Данные тестирования | Имя: NULL;  Фамилия: NULL;  Номер телефона: NULL;  Электронная почта: NULL;  Пароль: NULL;  Подтверждение пароля: NULL. |
| Ожидаемый результат | Сайт должен вывести сообщение о незаполненных полях |
| Фактический результат | Вывелось сообщение «Заполните поле»  Результат работы показан на рисунке 2.2.2 |
| Предпосылки | Открытие сайта |
| Постусловия | Работа системы происходит корректно |
| Статус (Pass/Fall) | Pass |

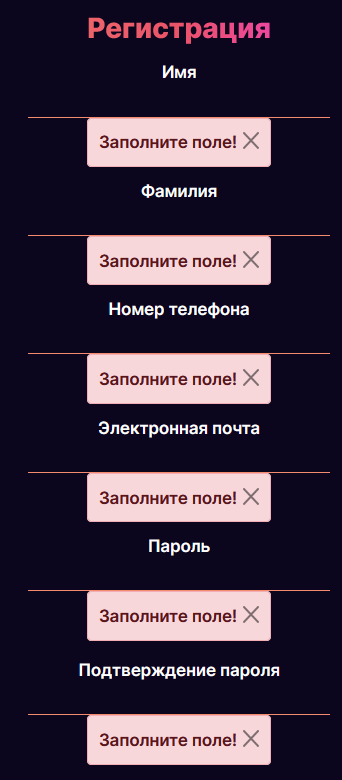


Рисунок 2.2.3 – Результат работы регистрации пользователя при вводе не вводе данных

Таблица 2.2.4 - Тестирование авторизации пользователя.

|  |  |
| --- | --- |
| Наименование | Описание |
| 1 | 2 |
| Test Case # | T\_AUTHORIZATION \_1 |
| Приоритет тестирования | Высокий |
| Название тестирования | Проверка авторизации пользователя при вводе верных данных |
| Резюме испытания | Необходимо добиться корректного поведения системы, при авторизации пользователя |
| Шаги тестирования | 1. Нажать на кнопку «Вход»; 2. Ввести в поля данные; 3. Нажать на кнопку «Войти». |
| Данные тестирования | Электронная почта: example@mail.ru;  Пароль: password; |
| Ожидаемый результат | Сайт должен вывести сообщение об успешной авторизации |
| Фактический результат | Авторизация |
| Предпосылки | Открытие сайта |
| Постусловия | Работа системы происходит корректно |
| Статус (Pass/Fall) | Pass |

Таблица 2.2.5 - тестирование авторизации при вводе неверных данных.

|  |  |
| --- | --- |
| Наименование | Описание |
| 1 | 2 |
| Test Case # | T\_ AUTHORIZATION \_ERROR\_1 |
| Приоритет тестирования | Высокий |
| Название тестирования | Проверка авторизации пользователя при вводе неверных данных |

Продолжение таблицы 2.2.5 тестирование авторизации при вводе неверных данных

|  |  |
| --- | --- |
| 1 | 2 |
| Резюме испытания | Необходимо добиться корректного поведения системы, при авторизации пользователя |
| Шаги тестирования | 1. Нажать на кнопку «Вход»; 2. Ввести в поля данные; 3. Нажать на кнопку «Войти». |
| Данные тестирования | Электронная почта: Null;  Пароль: Null; |
| Ожидаемый результат | Сайт должен вывести сообщение об ошибке при авторизации |
| Фактический результат | Ошибка авторизация |
| Предпосылки | Открытие сайта |
| Постусловия | Работа системы происходит корректно |
| Статус (Pass/Fall) | Pass |

Таблица 2.2.6 - Тестирование отправления заявки на артиста.

|  |  |
| --- | --- |
| Наименование | Описание |
| 1 | 2 |
| Test Case # | T\_ARTIST\_FORM \_1 |
| Приоритет тестирования | Высокий |
| Название тестирования | Проверка отправления заявки на Артиста |
| Резюме испытания | Необходимо добиться корректного поведения системы, при отправлении заявки |
| Шаги тестирования | 1. Нажать на кнопку «Стать автором» 2. Заполнить форму |
| Данные тестирования | Псевдоним: BimBimBomBom;  Электронная почта:example1@mail.ru. |

Продолжение таблицы 2.2.6 тестирование отправления заявки на артиста

|  |  |
| --- | --- |
| 1 | 2 |
| Ожидаемый результат | Сайт должен вывести сообщение об успешном отправлении заявки |
| Фактический результат | Заявка отправлена |
| Предпосылки | Открытие сайта |
| Постусловия | Работа системы происходит корректно |
| Статус (Pass/Fall) | Pass |

Таблица 2.2.7 - Тестирование отправления заявки на артиста при неправильном вводе данных.

|  |  |
| --- | --- |
| Наименование | Описание |
| 1 | 2 |
| Test Case # | T\_ ARTIST \_FORM\_ERROR \_1 |
| Приоритет тестирования | Высокий |
| Название тестирования | Проверка отправления заявки на Артиста |
| Резюме испытания | Необходимо добиться корректного поведения системы, при отправлении неправильной заявки |
| Шаги тестирования | 1. Нажать на кнопку «Стать автором» 2. Заполнить форму |
| Данные тестирования | Псевдоним: Null;  Электронная почта: Null. |
| Ожидаемый результат | Сайт должен вывести сообщение об ошибке заявки |
| Фактический результат | Ошибка отправления |
| Предпосылки | Открытие сайта |
| Постусловия | Работа системы происходит корректно |
| Статус (Pass/Fall) | Pass |

Таблица 2.2.8 - Тестирование отклонения заявки.

|  |  |
| --- | --- |
| Наименование | Описание |
| 1 | 2 |
| Test Case # | T\_DENY \_APPLICATION \_1 |
| Приоритет тестирования | Высокий |
| Название тестирования | Проверка отклонения заявки пользователя админом |
| Резюме испытания | Необходимо добиться корректного отклонении заявки пользователя админом |
| Шаги тестирования | 1. Зайти в аккаунт админа 2. На странице «Новые заявки» нажать «Отклонить» |
| Данные тестирования | ФИО: Смирнов Алексей Андреевич  Электронная почта: example@mail.ru  Псевдоним: Quard |
| Ожидаемый результат | Сайт должен вывести сообщение об успешном отклонении заявки |
| Фактический результат | Заявка отклонена |
| Предпосылки | Открытие сайта |
| Постусловия | Работа системы происходит корректно |
| Статус (Pass/Fall) | Pass |

Таблица 2.2.9 – Тестирование принятия заявки

|  |  |
| --- | --- |
| Наименование | Описание |
| 1 | 2 |
| Test Case # | T\_ACCEPT \_ APPLICATION \_1 |
| Приоритет тестирования | Высокий |
| Название тестирования | Проверка принятия заявки пользователя админом |

Продолжение таблицы 2.2.9 - тестирование принятия заявки

|  |  |
| --- | --- |
| 1 | 2 |
| Резюме испытания | Необходимо добиться корректного принятии заявки пользователя админом |
| Шаги тестирования | 1. Зайти в аккаунт админа 2. На странице «Новые заявки» нажать «Принять» |
| Данные тестирования | ФИО: Смирнов Алексей Андреевич  Электронная почта: example@mail.ru  Псевдоним: Quard |
| Ожидаемый результат | Сайт должен вывести сообщение об успешном принятии заявки |
| Фактический результат | Заявка принята |
| Предпосылки | Открытие сайта |
| Постусловия | Работа системы происходит корректно |
| Статус (Pass/Fall) | Pass |

Таблица 2.2.10 – Тестирование удаление артиста

|  |  |
| --- | --- |
| Наименование | Описание |
| 1 | 2 |
| Test Case # | T\_ARTIST \_DELETE \_1 |
| Приоритет тестирования | Высокий |
| Название тестирования | Удаление профиля артиста админом |
| Резюме испытания | Необходимо добиться корректного принятии заявки пользователя админом |
| Шаги тестирования | 1. Зайти в аккаунт админа 2. На странице «Авторы» нажать «Удалить» |
| Данные тестирования | ФИО: Смирнов Алексей Андреевич  Электронная почта: example@mail.ru  Псевдоним: Quard |

Продолжение таблицы 2.2.10 – тестирование удаления артиста

|  |  |
| --- | --- |
| Ожидаемый результат | Сайт должен вывести сообщение об успешном удалении артиста |
| Фактический результат | Артист удалён |
| Предпосылки | Открытие сайта |
| Постусловия | Работа системы происходит корректно |
| Статус (Pass/Fall) | Pass |

Таблица 2.2.11– Тестирование выхода из аккаунта

|  |  |
| --- | --- |
| Наименование | Описание |
| 1 | 2 |
| Test Case # | T\_SIGN \_OUT \_1 |
| Приоритет тестирования | Высокий |
| Название тестирования | Удаление профиля артиста админом |
| Резюме испытания | Необходимо добиться корректного принятии заявки пользователя админом |
| Шаги тестирования | 1. Зайти в аккаунт 2. В выпадающем меню нажать кнопку «Выход» |
| Данные тестирования | Электронная почта: example@mail.ru;  Пароль: password. |
| Ожидаемый результат | Сайт должен вывести сообщение об успешном удалении артиста |
| Фактический результат | Артист удалён |
| Предпосылки | Открытие сайта |
| Постусловия | Работа системы происходит корректно |
| Статус (Pass/Fall) | Pass |

* 1. Руководство пользователя

Главной целью руководства является обучение работы с системой.

Данный продукт предназначен для управления информацией, которая хранится в базе данных, добавлении новых пользователей, новых услуг и управлении заявками пользователей.

Минимальные требования к аппаратному обеспечению:

* oc: Windows 8/8.1/10/11 (64 bit);
* процессор: Intel Core I3 10100 или AMD Ryzen 3 3200 (2.7 GHz);
* озу: 8 ГБ;
* видеокарта: UHD 630 или Vega 8 и выше;
* жёсткий диск: 30 ГБ свободного места и выше.

Для запуска программного обеспечения пользователя нужно:

1. открыть сайт;
2. войти в систему (администратор/менеджер);
3. нажать на кнопку «Авторизоваться».

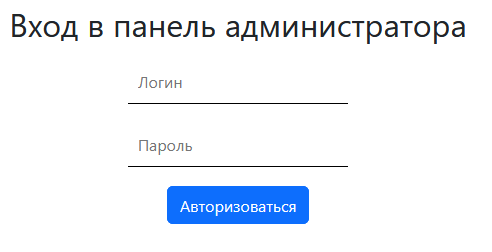
Описание возможности администратора: 

Рисунок 2.3.1 – Страница авторизации

* добавление, удаление, редактирование услуг:

1. после авторизации в меню слева нажать «Редактировать услуги»;
2. на странице для добавления услуги нужно заполнить форму и нажать «Добавить услугу»;

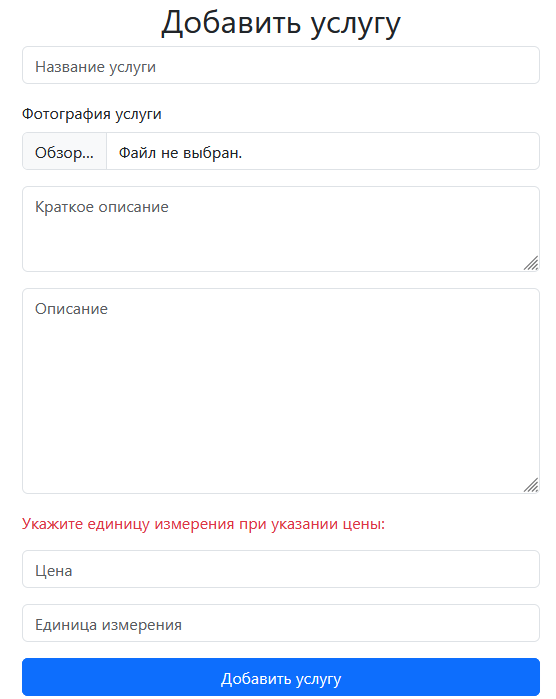


Рисунок 2.3.2 –Добавление услуги

1. для редактирования нужно нажать кнопку «Редактировать»;
2. информация перенесётся в форму заполнения;
3. заменить её;

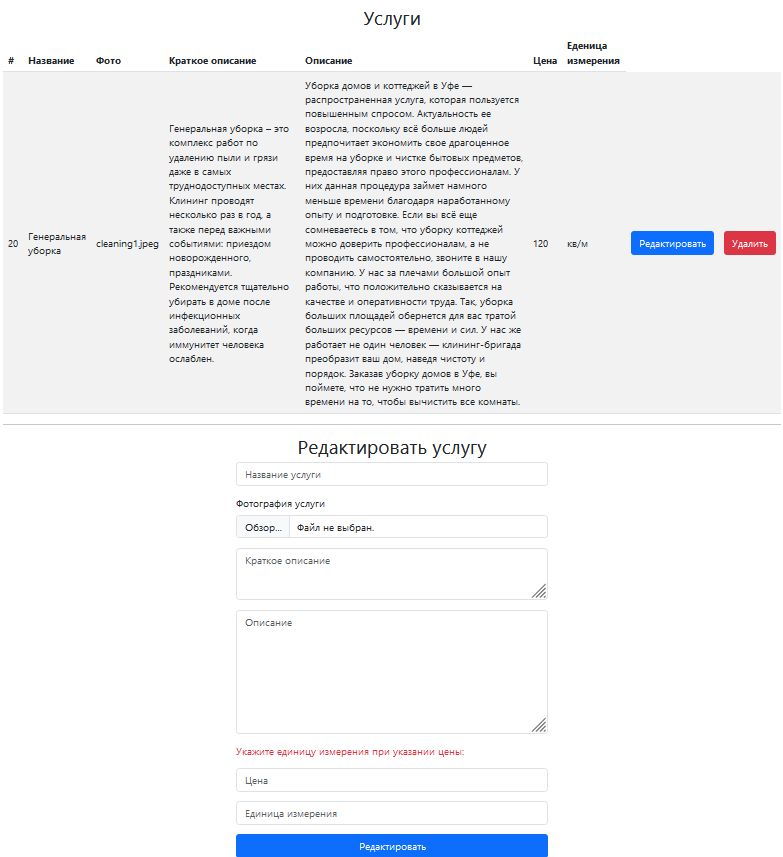


Рисунок 2.3.3 – Редактирование услуги

1. для удаления товара нажать на кнопку «Удалить».

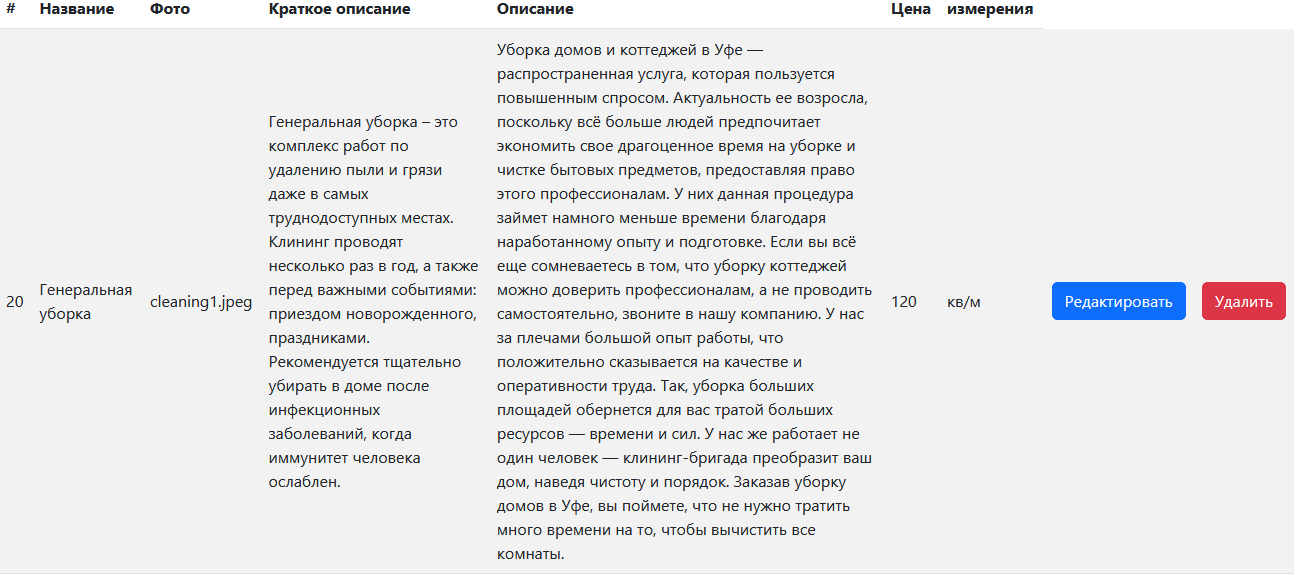


Рисунок 2.3.4 – Удаление услуги

* добавление и удаление сотрудников:

1. для добавления пользователя нужно нажать на меню слева «Редактировать пользователя»;
2. Заполнить форму и добавить пользователя;

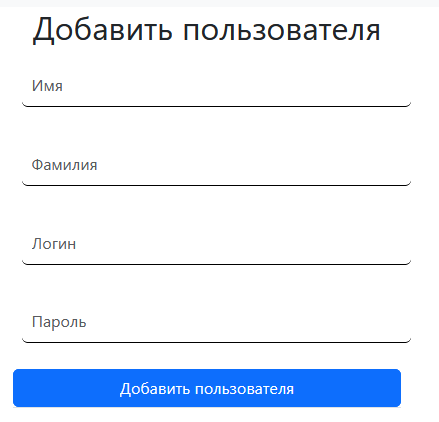


Рисунок 2.3.5 – Добавление пользователя

1. Для редактирования пользователя на странице нажать на кнопку «Редактировать» и заменить данные в форме;

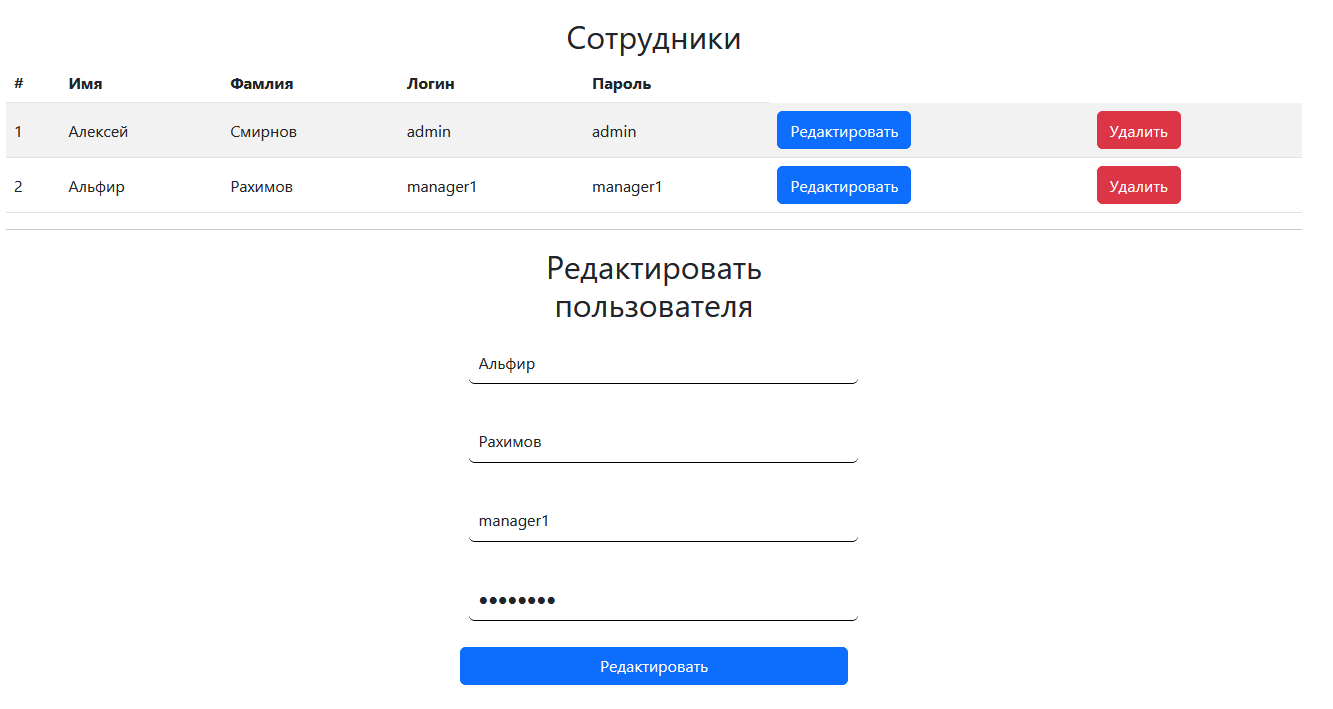


Рисунок 2.3.6 – Редактирование пользователя

1. Для удаления пользователя нажать на кнопку «Удалить».

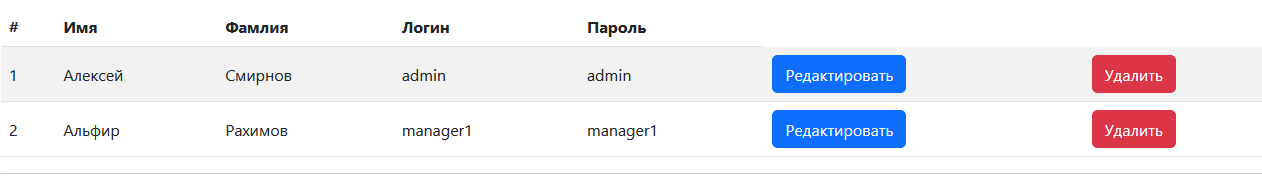


Рисунок 2.3.7 – Удаление пользователя

* управление заявками;

1. для просмотра заявок нужно нажать в левом меню на ссылку «Новые» в разделе «Заявки»;

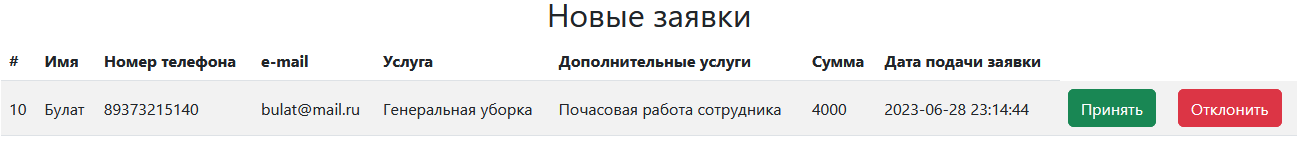
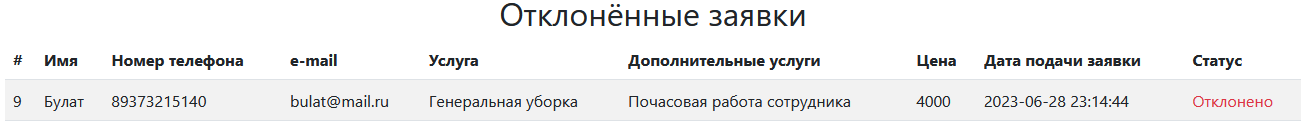


Рисунок 2.3.7 – Новые заявки

1. для того чтобы принять заявку нужно нажать на кнопку «Принять»;
2. для того чтобы отклонить заявку нужно нажать на кнопку «Отклонить»;
3. При нажатии на кнопку «Отклонить» заявка переносится во вкладку «Отклонено»;
4. Статус заявки меняется на «Отклонено»;

 Рисунок 2.3.8 – Отклонённые заявки

1. При нажатии на кнопку «Принять» заявка переносится во вкладку «Выполняется»;

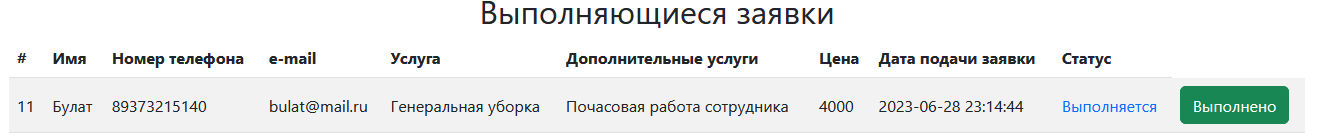


Рисунок 2.3.9 – Выполняющиеся заявки

1. При нажатии на кнопку «Выполнено» заявка переносится во вкладку «Выполнено».

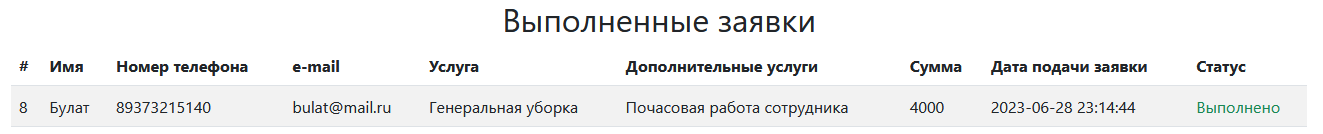


Рисунок 2.3.10 – Выполненные заявки

Заключение

В процессе выполнения курсового проекта был изучен процесс создания веб-приложения с помощью языка разметки гипертекста HTML, сформированы общие навыки работы с LARAVEL, CSS, JS, а также изучены особенности разработки дизайна веб-приложения.

Была проведена эксплуатация и отладочное тестирование программного обеспечения на данных контрольного примера, которые завершились успешно. В ходе работы над групповым проектом были решены все задачи, поставленные цели достигнуты.

СПИСОК ЛИТЕРАТУРЫ

1. ГОСТ 19.001-77. ЕСПД. Общие положения;
2. ГОСТ 19.003-80. ЕСПД. Схемы алгоритмов и программ. Обозначения условные графические. - Заменен на ГОСТ 19.701-90;
3. ГОСТ 19.404-79. ЕСПД. Пояснительная записка. Требования к содержанию и оформлению;
4. BOOTSTRAP 5 [Электронный ресурс] – Режим доступа: https://bootstrap-5.ru/ свободный;
5. Хабр [Электронный ресурс]. - Режим доступа: https://habr.com/ru/all/, свободный;
6. PHP [Электронный ресурс]. - Режим доступа: https://www.php.net/, свободный;
7. Skillbox [Электронный ресурс]. - Режим доступа: https://skillbox.ru/ , свободный;
8. JQuery [Электронный ресурс]. - Режим доступа: https://jquery.com/ свободный;
9. Stackoverflow [Электронный ресурс]. - Режим доступа: https://stackoverflow.com/ свободный;
10. Википедия [Электронный ресурс] // Свободная энциклопедия. - Режим доступа: http://ru.wikipedia.org/wiki/ , свободный;
11. JavaScript [Электронный ресурс]. - Режим доступа: https://learn.javascript.ru/first-steps/ , свободный;
12. Html5 book [Электронный ресурс]. - Режим доступа: https://html5book.ru/ , свободный;
13. MDN [Электронный ресурс]. - Режим доступа: https://developer.mozilla.org/ru/ свободный;
14. ДОКА HTML [Электронный ресурс]. - Режим доступа: https://doka.guide/html/ свободный;
15. PHP RU [Электронный ресурс]. - Режим доступа: https://php.ru/manual/ свободный;
16. JQuery ru [Электронный ресурс]. - Режим доступа: https://jquery-docs.ru/ свободный;
17. w3school [Электронный ресурс]. - Режим доступа: https://www.w3schools.com/ свободный;
18. Metanit [Электронный ресурс]. - Режим доступа: https://metanit.com/ , свободный;
19. DelftStack [Электронный ресурс]. - Режим доступа: https://www.delftstack.com/ru/ , свободный;
20. itProger [Электронный ресурс]. - Режим доступа: https://itproger.com/ , свободный.

ПРИЛОЖЕНИЕ А

Выходные данные

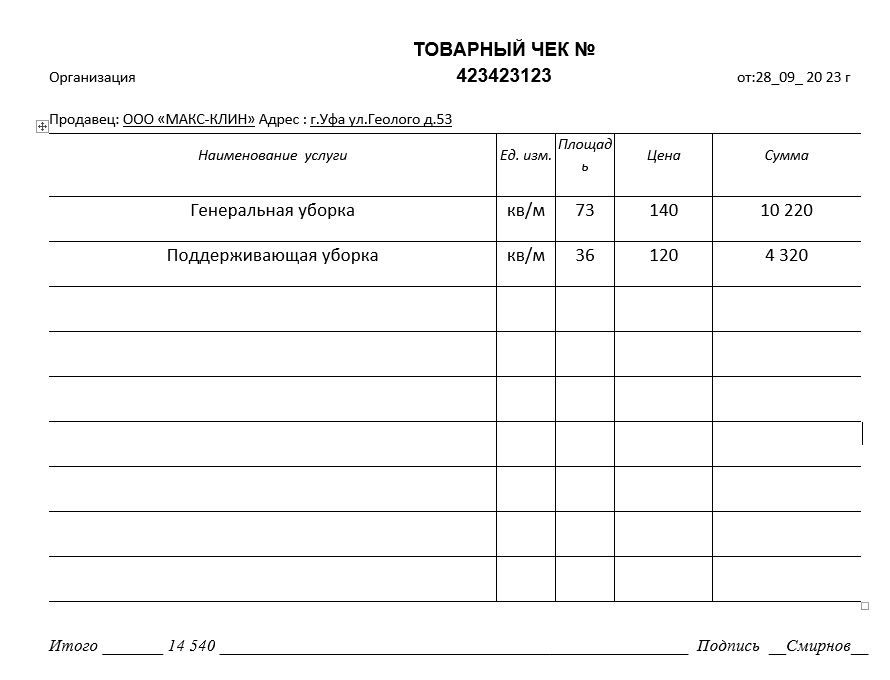


Рисунок А.1 – Товарный чек

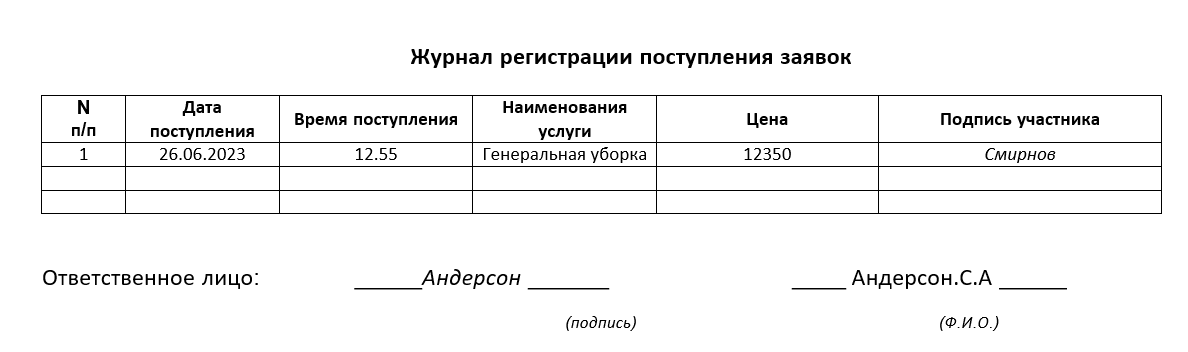


Рисунок A.2 – Журнал заявок

Приложение Б

Исходный код

Главная страница

<?php

include 'layout/header.php';

include 'functions/connect.php';

if (!empty($\_SESSION['id\_user'])) {

include 'admin/sidebar.php';

}

$additional\_service\_query = "SELECT \* FROM `additional\_services`";

$additional\_service\_result = mysqli\_query($con,$additional\_service\_query);

$additional\_service\_result1 = mysqli\_query($con,$additional\_service\_query);

$service\_query = "SELECT \* FROM `services`";

$service\_result = mysqli\_query($con,$service\_query);

$service\_user\_result = mysqli\_query($con,$service\_query);

?>

Страница товара

<?php

include 'layout/header.php';

include 'functions/connect.php';

if (!empty($\_SESSION['id\_user'])) {

include 'admin/sidebar.php';

}

$service\_query = "SELECT \* FROM `services`";

$service\_result = mysqli\_query($con,$service\_query);

$additional\_service\_query = "SELECT \* FROM `additional\_services`";

$additional\_service\_result = mysqli\_query($con,$additional\_service\_query);

$additional\_service\_result1 = mysqli\_query($con,$additional\_service\_query);

$service\_query = "SELECT \* FROM `services`";

$service\_result = mysqli\_query($con,$service\_query);

$service\_user\_result = mysqli\_query($con,$service\_query);

?>

Отправление заявки

<?php

include 'connect.php';

if(!empty($\_POST)){

$name = $\_POST["name"];

$phone\_number = $\_POST['number'];

$email = $\_POST['e-mail'];

$service\_select = $\_POST['service\_select'];

$date = gmdate('Y-m-d h:i:s', time());

ПРОДОЛЖЕНИЕ ПРИЛОЖЕНИЯ Б

$query = "INSERT INTO `orders`(`id\_order`, `name\_client`, `phone\_client`, `e-mail\_client`, `date`, `services`, `additional\_services`, `cost\_order`, `status`) VALUES (NULL,'$name','$phone\_number','$email','$date','$service\_select','$add\_service','-',1)";

$result = mysqli\_query($con, $query);

if($result){

echo "<script>alert('Заявка отправлена');location.href='/';</script>";

}

else{

echo "<script>alert('Ошибка отправления. Попробуйте снова.')</script>";

echo mysqli\_error($con);

echo "<br>",$query;

}

}

?>

Изменение статуса заявки

<?php

include 'connect.php';

$id\_order = $\_GET['id\_order'];

$del\_query = "UPDATE `orders` SET `status` = '3' WHERE `orders`.`id\_order` = '$id\_order'";

$del\_result = mysqli\_query($con, $del\_query);

if (!empty($del\_result)){

echo "<script> alert('Статус заявки изменён на Выполнено'); location.href='../admin/ordersProg.php' </script>";

}

else{

echo "<script> alert('Ошибка! Попробуйте снова.')</script>";

echo mysqli\_error($con);

}

?>

Редактирование пользователя

<?php

include 'connect.php';

if(!empty($\_POST)){

$id\_user = $\_POST["id"];

$name = $\_POST["name"];

$surname = $\_POST['surname'];

$login = $\_POST['login'];

$password = $\_POST["password"];

$query = "UPDATE `users` SET `name`='$name',`surname`='$surname',`login`='$login',`password`='$password'

WHERE `id\_user`='$id\_user'";

ПРОДОЛЖЕНИЕ ПРИЛОЖЕНИЯ Б

$result = mysqli\_query($con, $query);

if($result){

echo "<script>alert('Пользователь отредактирован успешно');location.href='../admin/userRedact.php';</script>"

}

else{

echo "<script>alert('Ошибка редактирования. Попробуйте снова.')</script>";

echo mysqli\_error($con);

echo "<br>",$query;

}

}

?>

Редактирование услуги

<?php

$con = mysqli\_connect("localhost","root", "","cleaning");

if ($\_FILES && $\_FILES["photo"]["error"] == UPLOAD\_ERR\_OK) {

$photo1 = "../media/images/" . $\_FILES["photo"]["name"];

move\_uploaded\_file($\_FILES["photo"]["tmp\_name"],$photo1);

}

if(!empty($\_POST)){

$id\_service = $\_POST['id'];

$name\_service = $\_POST["title"];

$cost\_service = $\_POST['cost'];

$photo=$\_FILES["photo"]["name"];

$short\_description = $\_POST['short\_description'];

$unit = $\_POST['unit'];

$description = $\_POST['description'];

$query = "UPDATE `services` SET `name\_service`=' $name\_service',`photo`='$photo',`cost\_service`='$cost\_service',`unit`='$unit',`short\_description`='$short\_description',`description`='$description' WHERE `id\_service`=$id\_service"; $result = mysqli\_query($con, $query);

if($result){

echo "<script>alert('Услуга редактирована успешно');location.href='../admin/serviceRedact.php';</script>";

}

else{

echo "<script>alert('Ошибка добавления. Попробуйте снова.')</script>";

echo mysqli\_error($con);

echo "<br>",$query;

}

}

?>

ПРОДОЛЖЕНИЕ ПРИЛОЖЕНИЯ Б

Отклонение заявки

<?php

include 'connect.php';

$id\_order = $\_GET['id\_order'];

$del\_query = "UPDATE `orders` SET `status` = '4' WHERE `orders`.`id\_order` = '$id\_order'";

$del\_result = mysqli\_query($con, $del\_query);

if (!empty($del\_result)){

echo "<script> alert('Заявка отклонена успешно!'); location.href='../admin/ordersNew.php' </script>";

}

else{

echo "<script> alert('Ошибка! Попробуйте снова.')</script>";

echo mysqli\_error($con);

}

?>

Удаление пользователя

<?php

include 'connect.php';

$id\_user = $\_GET['id\_user'];

$del\_query = "delete from `users` where id\_user = $id\_user";

$del\_result = mysqli\_query($con, $del\_query);

if (!empty($del\_result)){

echo "<script> alert('Запись удалена успешно!'); location.href='../admin/userRedact.php' </script>";

}

else{

echo "<script> alert('Ошибка удаления! Попробуйте снова.')</script>";

echo mysqli\_error($con);

}

?>

Удаление услуг

<?php

include 'connect.php';

$id\_service = $\_GET['id\_service'];

$del\_query = "delete from `services` where id\_service = $id\_service";

$del\_result = mysqli\_query($con, $del\_query);

if (!empty($del\_result)){

echo "<script> alert('Услуга удалена успешно!'); location.href='../admin/serviceRedact.php' </script>";

}

else{

echo "<script> alert('Ошибка удаления! Попробуйте снова.')</script>";

echo mysqli\_error($con);

}

?>

Удаление дополнительной услуги

<?php

include 'connect.php';

$id\_addservice = $\_GET['id\_addservice'];

$del\_query = "delete from `additional\_services` where id\_addservice = $id\_addservice";

$del\_result = mysqli\_query($con, $del\_query);

if (!empty($del\_result)){

echo "<script> alert('Дополнительная услуга удалена успешно!'); location.href='../admin/serviceRedact.php' </script>";

}

else{

echo "<script> alert('Ошибка удаления! Попробуйте снова.')</script>";

echo mysqli\_error($con);

}

ПРОДОЛЖЕНИЕ ПРИЛОЖЕНИЯ Б

?>

Подключение

<?php

$con = mysqli\_connect("localhost","root", "","cleaning");

?>

Авторизация

<?php

include '../functions/connect.php';

?>

<?php

session\_start();

if(!empty($\_POST)){

$login = $\_POST["login"];

$password = $\_POST["password"];

$query = "select \* from users where login = '$login' and password = '$password'";

$user\_info = mysqli\_fetch\_array(mysqli\_query($con, $query));

if ($user\_info){

$\_SESSION["role\_user"] = $user\_info['role'];

$\_SESSION["id\_user"] = $user\_info['id\_user'];

echo "<script>alert('Вы успешно зашли в аккаунт');location.href='../index.php';</script>";

}

else{

echo "<script>alert('Ошибка входа');location.href='../admin/index.php';</script>";

}

}

?>

Добавление пользователя

?php

include 'connect.php';

if(!empty($\_POST)){

$name = $\_POST["name"];

$surname = $\_POST['surname'];

$login = $\_POST['login'];

$password = $\_POST["password"];

$query = "INSERT INTO `users` (`id\_user`, `name`, `surname`, `login`, `password`, `role`) VALUES (NULL, '$name', '$surname', '$login', '$password', '2');";

$result = mysqli\_query($con, $query);

if($result){

echo "<script>alert('Пользователь успешно зарегестрирован');location.href='../admin/userRedact.php';</script>";

}

else{

echo "<script>alert('Ошибка добавления. Попробуйте снова.')</script>";

echo mysqli\_error($con);

echo "<br>",$query;

}

}

?>

<?php

$con = mysqli\_connect("localhost","root", "","cleaning");

if ($\_FILES && $\_FILES["photo"]["error"] == UPLOAD\_ERR\_OK) {

$photo1 = "../media/images/" . $\_FILES["photo"]["name"];

move\_uploaded\_file($\_FILES["photo"]["tmp\_name"],$photo1);

}

if(!empty($\_POST)){

$name\_service = $\_POST["title"];

ПРОДОЛЖЕНИЕ ПРИЛОЖЕНИЯ Б

$cost\_service = $\_POST['cost'];

$photo=$\_FILES["photo"]["name"];

$short\_description = $\_POST['short\_description'];

$description = $\_POST['description'];

$unit = $\_POST['unit'];

$query = "INSERT INTO `services`(`id\_service`, `name\_service`,`photo`, `cost\_service`, `unit`, `short\_description`, `description`) VALUES (NULL,'$name\_service', '$photo', '$cost\_service','$unit','$short\_description','$description')";

// print\_r($query);

$result = mysqli\_query($con, $query);

if($result){

echo "<script>alert('Услуга добавлена успешно');location.href='../admin/serviceRedact.php';</script>";

}

else{

echo "<script>alert('Ошибка добавления. Попробуйте снова.')</script>";

echo mysqli\_error($con);

echo "<br>",$query;

}

}

?>

Добавление дополнительноц услуги

<?php

include 'connect.php';

if(!empty($\_POST)){

$name\_addservice = $\_POST["name\_addservice"];

$cost\_addservice = $\_POST['cost\_addservice'];

$unit\_addit = $\_POST['unit\_addit'];

$query = "INSERT INTO `additional\_services`(`id\_addservice`, `name\_addservice`, `cost\_addservice`,`unit\_addit`) VALUES (NULL,'$name\_addservice','$cost\_addservice','$unit\_addit')";

$result = mysqli\_query($con, $query);

if($result){

echo "<script>alert('Дополнительная услуга добавлена успешно');location.href='../admin/serviceRedact.php';</script>";

}

else{

echo "<script>alert('Ошибка добавления. Попробуйте снова.')</script>";

echo mysqli\_error($con);

echo "<br>",$query;

}

}

?>

Подтверждение заявки

<?php

include 'connect.php';

$id\_order = $\_GET['id\_order'];

$del\_query = "UPDATE `orders` SET `status` = '2' WHERE `orders`.`id\_order` = '$id\_order'";

$del\_result = mysqli\_query($con, $del\_query);

if (!empty($del\_result)){

echo "<script> alert('Заявка принята успешно!'); location.href='../admin/ordersNew.php' </script>";

}

else{

echo "<script> alert('Ошибка! Попробуйте снова.')</script>";

echo mysqli\_error($con);

}

?>

ПРОДОЛЖЕНИЕ ПРИЛОЖЕНИЯ Б

Страница редактирования пользователя

<?php

include 'connect.php';

$id\_order = $\_GET['id\_order'];

$del\_query = "UPDATE `orders` SET `status` = '2' WHERE `orders`.`id\_order` = '$id\_order'";

$del\_result = mysqli\_query($con, $del\_query);

if (!empty($del\_result)){

echo "<script> alert('Заявка принята успешно!'); location.href='../admin/ordersNew.php' </script>";

}

else{

echo "<script> alert('Ошибка! Попробуйте снова.')</script>";

echo mysqli\_error($con);

}

?>

Редактирование услуг

<?php

include '../layout/header.php';

include "../functions/connect.php";

include 'sidebar.php';

$service\_query = "SELECT \* FROM `services`";

$service\_result = mysqli\_query($con,$service\_query);

$additional\_query = "SELECT \* FROM `additional\_services`";

$additional\_result = mysqli\_query($con,$additional\_query);

///

?>

<?php

$service\_get = !empty($\_GET['id\_service'])?$\_GET["id\_service"]:false;

$query\_edit = "select \* from services where id\_service ='$service\_get'";

$service\_edit = mysqli\_fetch\_array(mysqli\_query($con,$query\_edit));

?>

Заявка принята

<?php

include '../layout/header.php';

include '../functions/connect.php';

include 'sidebar.php';

$order\_query = "SELECT \* FROM `orders`, `order\_status`, `services`,`additional\_services` WHERE orders.status=order\_status.id\_orderstatus and orders.services=services.id\_service and orders.additional\_services=additional\_services.id\_addservice AND id\_orderstatus = 3; ";

$order\_result = mysqli\_query($con,$order\_query);

?>

ПРОДОЛЖЕНИЕ ПРИЛОЖЕНИЯ Б

Заявка выполняется

<?php

include '../layout/header.php';

include '../functions/connect.php';

$order\_query = "SELECT \* FROM `orders`, `order\_status`, `services`,`additional\_services` WHERE orders.status=order\_status.id\_orderstatus and orders.services=services.id\_service and orders.additional\_services=additional\_services.id\_addservice AND id\_orderstatus = 2; ";

$order\_result = mysqli\_query($con,$order\_query);

?>

Новые заявки

<?php

include '../layout/header.php';

include '../functions/connect.php';

include 'sidebar.php';

$order\_query = "SELECT \* FROM `orders`, `order\_status`, `services`,`additional\_services` WHERE orders.status=order\_status.id\_orderstatus and orders.services=services.id\_service and orders.additional\_services=additional\_services.id\_addservice AND id\_orderstatus = 1 ";

$order\_result = mysqli\_query($con,$order\_query);

?>

Заявка отклонена

<?php

include '../layout/header.php';

include '../functions/connect.php';

$order\_query = "SELECT \* FROM `orders`, `order\_status`, `services`,`additional\_services` WHERE orders.status=order\_status.id\_orderstatus and orders.services=services.id\_service and orders.additional\_services=additional\_services.id\_addservice AND id\_orderstatus = 4; ";

$order\_result = mysqli\_query($con,$order\_query);

?>

Кнопка вверх

//Достаём кнопку

let mybutton = document.getElementById("btn-back-to-top");

// Когда пользователь прокрутит страницу вниз на 20 пикселей от верхней части документа, отобразите кнопку

window.onscroll = function () {

scrollFunction();

};

function scrollFunction() {

if (

document.body.scrollTop > 20 ||

document.documentElement.scrollTop > 20

ПРОДОЛЖЕНИЕ ПРИЛОЖЕНИЯ Б

) {

mybutton.style.display = "block";

} else {

mybutton.style.display = "none";

}

}

// Когда пользователь нажимает на кнопку, прокрутите страницу до верхней части документа

mybutton.addEventListener("click", backToTop);

function backToTop() {

document.body.scrollTop = 0;

document.documentElement.scrollTop = 0;

}

//Calculator

$('#priceSelect,#price, #square, #flexSwitchCheckDefault').on('change',function(){

sum();

})

function sum (){

var priceSelect = $('#priceSelect').data('value');

var square = $('#square').val();

var total = 0;

$('#checkboxer input:checked').each(function(){

total += parseInt($(this).data('value'),0);

})

var sum = (priceSelect \* square) + total ;

console.log(sum);

$('.sum').text(sum);

СПИСОК СОКРАЩЕНИЙ

1. БД - база данных;
2. СУБД - система управления базами данных;
3. ОС - операционная система;