Техническое задание 3

1. Общие сведения 3

1.1. Наименование информационной системы 3

1.2. Основания для выполнения работ 3

1.3. Наименование организаций – Заказчик и Исполнитель 3

1.4. Порядок оформления и сдачи работ 3

2. Назначение и цели системы 3

2.1. Назначение системы 3

2.2. Цели создания системы 3

3. Требования к системе 4

3.1. Требования к системе в целом 4

3.2. Управление системой 4

3.3. Отчеты 5

4. Инструменты разработки 5

4.1. Клиентская часть 5

4.2. Серверная часть 5

5. Состав и содержание работ по созданию системы 6

5.1. Стадии разработки 6

5.2. Этапы разработки 6

6. Согласование и утверждение технического задания 6

*7.* Порядок контроля и приемки системы 7

8. Требования к документированию 9

База данных 10

1. Предметная область 10

2. Логическая модель базы данных 10

3. Физическая модель базы данных 11

*4.* Описание основных сущностей и типы полей 12

5. Содержимое базы данных 12

5.1. Таблица «Пользователь» 12

5.2. Таблица «Награды» 13

5.3. Таблица «Мероприятия» 13

5.4. Таблица «Мероприятия пользователя» 13

5.5. Таблица «Награды пользователя» 13

6. Выводы 13

Интерфейс 14

1. Проектирование интерфейса страниц системы 14

1.1. Страница авторизации 14

1.2. «Домашняя» страница 15

1.3. Страница «Панель управления» 15

1.4. Страница «Отчет» 16

1.5. Модальное окно с ошибкой 16

2. Выводы 16

Диаграмма классов 17

1. Проектирование диаграммы классов 17

1.1. Диаграмма классов 17

1.2. Структура классов 17

2. Вывод 19

Реализация программы 20

1. Разработка программы 20

1.1. Создание UI-компонентов 20

1.2. Инициализация и создание хранилища состояний 20

1.3. Вёрстка страниц 20

1.4. Проработка логики страниц 21

1.5. Добавление реакции на ошибочный ввод 21

2. Выводы 23

Тестирование 24

1. Проверка описанных функций 24

1.1. Авторизация 24

1.3. Страницы с таблицами 25

2. Пользовательский интерфейс 27

3. Выводы 27

Вывод 28

Техническое задание

1. Общие сведения
   1. Наименование информационной системы

Корпоративная система учета ППО ЕвроХим

* 1. Основания для выполнения работ

Работа выполняется на основании договора №1 от 02.02.2023 г.

* 1. Наименование организаций – Заказчик и Исполнитель
     1. Заказчик

ППО «Профсоюз усольских калийщиков»;

Адрес фактический: г. Пермь, ул. Дзержинского, д. 47;

Телефон: +7 (342) 270-01-80;

* + 1. Исполнитель

ИП «БМ Технологии»

Адрес фактический: г. Пермь, ул. Профессора Поздеева, д. 13;

Телефон: +7 (982) 446-71-90;

* 1. Порядок оформления и сдачи работ

Работы по разработке «Корпоративная система учета ППО ЕвроХим» сдаются разработчиком поэтапно в соответствии с календарным планом проекта.

1. Назначение и цели системы
   1. Назначение системы

Система предназначена для улучшения условий администрирования ППО «ЕвроХим». Также предназначена для повышения уровня ознакомления с актуальными новостями и мероприятиями, связанными с ППО.

* 1. Цели создания системы

«Корпоративная система учета ППО ЕвроХим» создается с целью:

1. Удобного процесса администрирования и учета сотрудников ППО ЕвроХим;
2. Просмотра и редактирования данных, связанных с ППО ЕвроХим и его участниками;
3. Удобного процесса ознакомления с актуальными новостями и мероприятиями.

Все поставленные цели должны быть выполнены и проверены в соответствии с планом.

1. Требования к системе
   1. Требования к системе в целом
      1. Требования к структуре и функционированию системы

Автоматизированная система должна обеспечивать возможность выполнения следующих функций:

1. Функция аутентификации;
2. Функция, обеспечивающая добавление, изменение и удаление данных о пользователях системы;
3. Функция, предоставляющая сведения об участниках ППО «ЕвроХим»;
4. Функция, предоставляющая сведения о наградах и мероприятиях, которые связаны с участниками ППО «ЕвроХим»;
5. Функция просмотра актуальных новостей и мероприятий, связанных с ППО «ЕвроХим»;
6. Функция генерации отчетов;
7. Формирование отчетов в формате xls (Microsoft Excel);
8. Реакция системы на ошибочный ввод;
9. Пользовательский интерфейс должен быть доступным и понятным Пользователю.
   * 1. Требования к эргономике и технической эстетике

Система должна обеспечивать удобный для конечного пользователя интерфейс, отвечающий следующим требованиям:

1. Поддержка русского языка;
2. Удобная, яркая цветовая палитра;
3. При возникновении ошибок в работе системы на экране должно появляться всплывающее окно, которое оповестит конечного пользователя об ошибке.
   1. Управление системой

В системе должно быть разграничение доступа по роли пользователя. Всего роли две:

1. Пользователь;
2. Администратор.

На каждую роль в системе будет встроен необходимый набор функций согласно их должностным обязанностям.

* + 1. Пользователь

Пользователь имеет доступ к данным, связанным непосредственно с ним, а также к новостной ленте и мероприятиям.

* + 1. Администратор

Администратор имеет доступ ко всем функция, доступным Пользователю, а также:

1. Изменение, удаление и добавление участников ППО «ЕвроХим»;
2. Экспорт отчетов по перечисленным направлениям отчетности;
3. Просмотр данных о конкретных пользователях системы
   1. Отчеты

Отчетные формы составляются как конкретные данные на одну из поставленных тем. Отчет должен быть представлен в формате xls.

* + 1. Отчет об участниках ППО «ЕвроХим»

Отчет содержит следующий столбцы:

* Логин пользователя;
* Роль пользователя;
* Пол пользователя;
* Возраст пользователя.
  + 1. Отчет о награждениях

Отчет содержит следующие столбцы:

* Название мероприятия;
* Описание награды;
* Тип награды;
* Количество очков.
  + 1. Отчет о мероприятиях

Отчет содержит следующие столбцы:

* Название мероприятия;
* Описание мероприятия;
* Адрес;
* Дата;
* Вид награды.

1. Инструменты разработки

Предполагается использование RestAPI для построения архитектуры Web-приложения.

4.1. Клиентская часть

Для стороны «Клиент» предлагаются следующие технологии:

* Вёрстка: HTML5, CSS3;
* Язык программирования: JavaScript, TypeScript;
* Правила: ES6+;
* Фреймворк: React;
* Менеджер состояния: Redux. Redux Toolkit.

4.2. Серверная часть

Для стороны «Сервер» предлагаются следующие технологии:

* База данных: PostgreSQL 11;
* Язык программирования: Java, Kotlin;
* Фреймворк: Spring Boot.
  1. 5. Состав и содержание работ по созданию системы

5.1. Стадии разработки

Разработка системы должна быть проведена в три этапа:

1. Разработка технического задания;
2. Рабочее проектирование;
3. Внедрение системы.

5.2. Этапы разработки

* + 1. Этапы разработки технического задания

На стадии разработки технического задания должны быть выполнены следующие этапы:

1. Разработка технического задания;
2. Согласование технического задания;
3. Утверждение технического задания.
   * 1. Этапы разработки рабочего проектирования системы

На стадии рабочего проектирования системы должны быть выполнены следующие этапы:

1. Разработка программы;
2. Тестирование программы.
3. Согласование и утверждение технического задания

На этапе разработки программной документации должна быть выполнена разработка программных документов в соответствии с требованиями к составу документации.

Этап тестирования автоматизированной системы должен осуществляться по следующему плану:

1. Проверка реакции системы на ввод некорректных данных;
2. Проверка корректности внесения изменений в базу данных.

Таблица 1 – Календарный план

|  |  |  |  |
| --- | --- | --- | --- |
| **Стадии разработки** | **Этапы работ** | **Содержание работ** | **Время выполнения** |
| Проектирование БД | Проектирование и разработка логической и физической модели БД | Описание предметной области с выделением основных сущностей и связей между ними | 10.02.2023-16.02.2023 |
| Проектирование интерфейса | Проектирование интерфейса | Проектирование интерфейса согласно обозначенным функциональным требованиям | 17.02.2023-23.02.2023 |
| Проектирование структуры классов | Разработка структур классов | Разработка структуры классов для корректной и оптимизированной работы всей системы | 24.02.2023-02.03.2023 |
| Реализация | Программная реализация интерфейса и функциональных частей | Написание кода для корректной работы системы, а также учет всех функциональных требований | 03.03.2023-24.03.2023 |
| Тестирование | Тестирование системы | Проведение тестирований разработанной системы | 25.03.2023-01.04.2023 |

1. Порядок контроля и приемки системы

После передачи Исполнителем отдельного функционального модуля программы Заказчику, последний имеет право тестировать модуль в течение 7 дней.

После тестирования Заказчик должен принять работу по данному этапу или в письменном виде изложить причину отказа от принятия.

В случае обоснованного отказа Исполнитель обязуется доработать модуль. В противном случае после проведения испытаний Исполнитель совместно с Заказчиком подписывает Акт приемки-сдачи автоматизированной системы в эксплуатацию.

Таблица 2. Виды, состав, объем и методы испытаний системы

|  |  |  |  |
| --- | --- | --- | --- |
| **Контрольные вопросы** | **Показатель качества** | **Код** | **Вид тестирования** |
| Соответствует ли реализация функций программного обеспечения задачам пользователя? Насколько полно автоматизированы задачи пользователя? | Пригодность | 1.1 | Функциональное |
| Насколько функционирование программного обеспечения и получаемые результаты (число десятичных знаков, округление) соответствуют требованиям приложения? | Правильность | 1.2 | Функциональное |
| Насколько легко и эффективно осуществляется взаимодействие с другим программным обеспечением в среде пользователя? | Способность к взаимодействию | 1.3 | Интеграционное |
| Обеспечивает ли программное обеспечение средства санкционирования доступа и выполняет ли требования приложения? | Защищенность | 1.4. | Функциональное, Стресс-тестирование |
| Функционирует ли система надежно в соответствии с требованиями поддержки приложения и технологичности, включая управление аномалиями (с оценкой средств управления аномалиями: определение ошибочных ситуаций системы и условий, требующих специальной обработки для подтверждения целостности системы; особенности восстановления и работы в условиях неполной работоспособности)? | Надежность | 2 | Функциональное, Стресс-тестирование |
| Применимо ли программное обеспечение в заданной операционной и поддерживающей среде? | Практичность | 3 | Интеграционное |
| Функционирует ли система эффективно, минимизируя издержки, с минимальным временем отклика и максимальной производительностью системы (с оценкой использования данных, оценкой эффективности по памяти, оценкой выполнения итераций и проверкой требований технологичности)? | Эффективность | 4 | Функциональное |
| Насколько легко исправлять ошибки и устранять недостатки? Насколько легко расширять возможности или технологию путем развития существующих функций или добавления новых функций или данных? | Изменяемость | 5.1 | Стресс-тестирование |
| Насколько легко переносить программное обеспечение для использования в другой среде (конфигурация КТС и/или среда программной системы)? | Мобильность | 6.1 | Интеграционное |

1. Требования к документированию

Для системы на различных стадиях создания должны быть выпущены следующие документы из числа предусмотренных в ГОСТ 34.201-89 «Информационная технология. Комплекс стандартов на автоматизированные системы».

Виды, комплектность и обозначения документов при создании Информационно-справочная система ППО ЕвроХим приведены в таблице 3.

Таблица 3. Виды, комплектность и обозначения документов

|  |  |  |
| --- | --- | --- |
| **Наименование документа** | **Код документа** | **Часть проекта** |
| Техническое задание | ТЗ | Предпроектное исследование |
| Инструкция пользователя | ИП | Рабочее проектирование |

База данных

**Цель работы:** спроектировать базу данных корпоративной информационной системы профсоюза «ЕвроХим» в рамках технического задания.

**Задачи:**

1. Выделить предметную область проектируемой базы данных;
2. Спроектировать физическую и логическую модель базы данных;
3. Обосновать выбор типов полей;
4. Заполнить базу данных тестовыми данными.
5. Предметная область

Для обеспечения эффективности ведения информации о деятельности профсоюза ЕвроХима, необходима система, основанная на современной базе данных, включающая в себя данные о пользователях и их участии в жизни профсоюза.

В базе данных необходимо хранить информацию о наградах и мероприятиях.

Деятельность направлена на работу с пользователями. Поэтому нужна возможность для ввода первичной информации о пользователях. Отчетные формы для администраторов и пользователей системы, составляются как аналитический отчет по полученным наградам и по участию в мероприятиях.

Автоматизация процесса позволит уменьшить ручной труд, погасить процент ошибок, снизить время на обработку информации и работу с документами.

1. Логическая модель базы данных

На основе предметной области была составлена логическая модель со связями. Логическая модель базы данных представлена на рисунке 1.

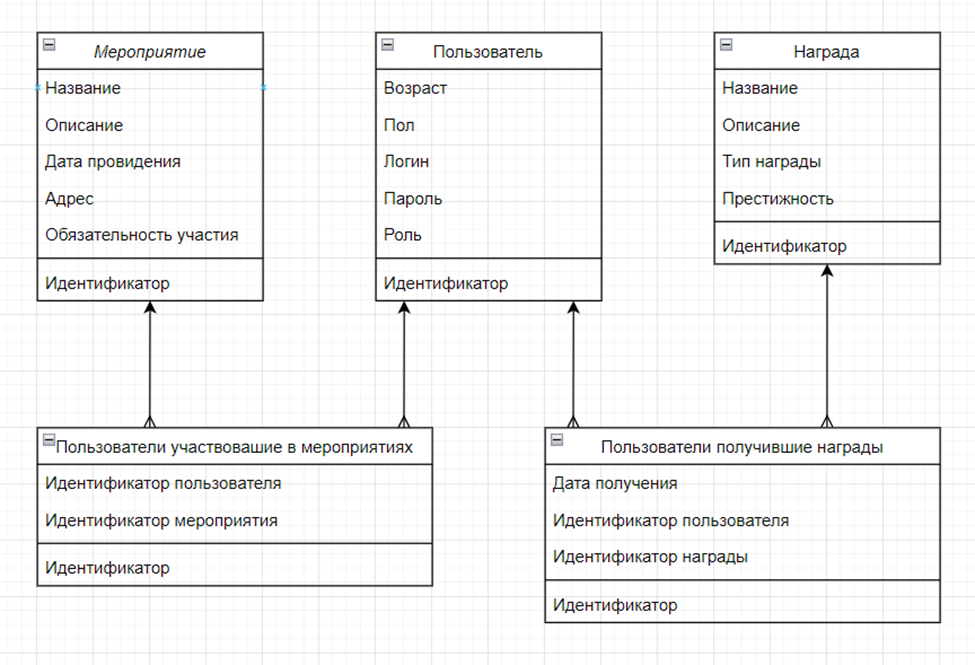


Рисунок 1 – Логическая модель базы данных

1. Физическая модель базы данных

Для разработки программного продукта была выбрана СУБД PostgreSQL. Ее функциональные возможности определены следующим образом:

1. Интерпретация баз данных в виртуальной среде;
2. Создание связей между таблицами;
3. PostgreSQL позволяет выполнять простейшие операции с данными:

* добавление в таблицу одной или нескольких записей;
* удаление из таблицы одной или нескольких записей;
* обновление значений некоторых полей в одной или нескольких записях;
* поиск одной или нескольких записей, удовлетворяющих заданному условию.

На основе предметной области и логической модели базы данных была составлена физическая модель базы данных. Физическая модель диаграммы базы данных (рис.2).

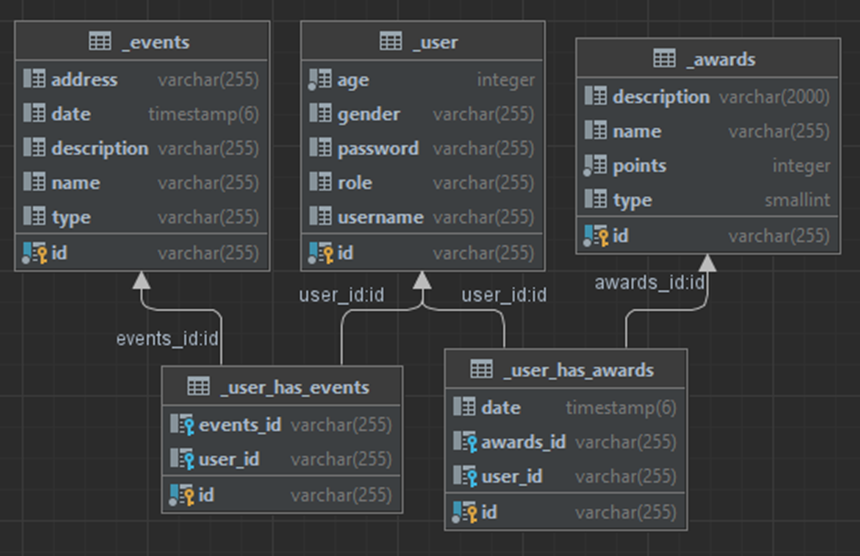


Рисунок 2 – Физическая модель базы данных

1. Описание основных сущностей и типы полей

Тип данных для идентификаторов (первичные и внешние ключи) выбран VARCHAR, который хранит UUID (англ. universally unique identifier «универсальный уникальный идентификатор»). Выбран такой тип данных т.к. записей в таблице может быть огромное количество. Для дат выбран тип данных TIMESTAMP, содержащий день, месяц, год, а также час, минуты, секунду, миллисекунду т.к. в системе есть мероприятия, у которых важна точная дата и время провидения.

Таблица 1 - Связи между сущностями в таблице базы данных

|  |  |  |
| --- | --- | --- |
| **Название таблицы 1** | **Название таблицы 2** | **Связь между таблицами** |
| \_user\_has\_events | \_users | Один ко многим идентифицирующая связь |
| \_user\_has\_events | \_events | Один ко многим не идентифицирующая связь |
| \_user\_has\_awards | \_users | Один ко многим идентифицирующая связь |
| \_user\_has\_awards | \_awards | Один ко многим идентифицирующая связь |

1. Содержимое базы данных
   1. Таблица «Пользователь»

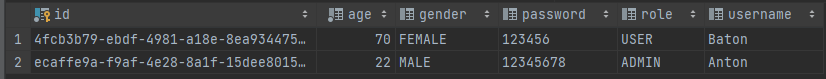


Рисунок 3 – Таблица «Пользователь»

* 1. Таблица «Награды»

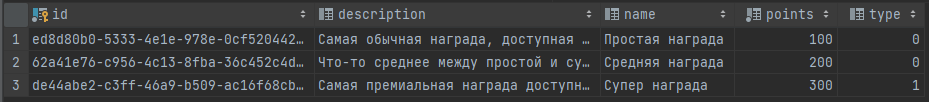


Рисунок 4 – Таблица «Награды»

* 1. Таблица «Мероприятия»

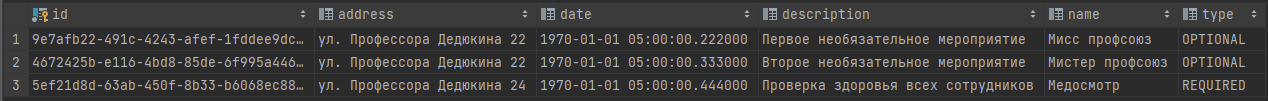


Рисунок 5 – Таблица «Мероприятия»

* 1. Таблица «Мероприятия пользователя»

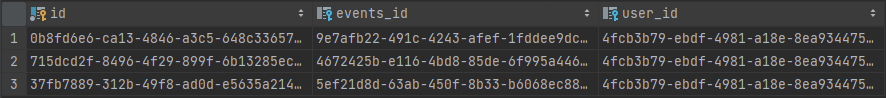


Рисунок 6 – Таблица «Меропритяия пользователя»

* 1. Таблица «Награды пользователя»

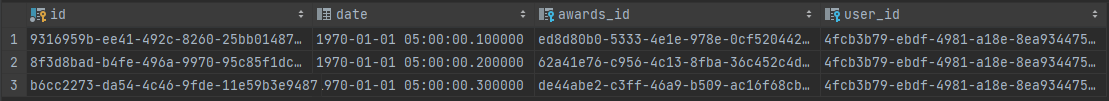


Рисунок 7 – Таблица «Награды пользователя»

1. Выводы

В данной лабораторной работе была спроектирована логическая и физическая модель данных, описаны сущности и заполнены таблицы базы данных.

Интерфейс

**Цель работы:** спроектировать интерфейс согласно обозначенному функционалу в техническом задании.

**Задачи:**

1. Разработать прототип входа для администратора и пользователей;
2. Разработать прототипы страниц с отчетами по представленным темам;
3. Разработать прототип отчетов.

Дизайн проработан с помощью сайта Figma.

Ссылка на проект:

<https://www.figma.com/proto/gLyETTXE6IlB51SOW1WWp8/%D0%9F%D0%9F%D0%9E-%22%D0%95%D0%B2%D1%80%D0%BE%D1%85%D0%B8%D0%BC%22?node-id=34-2&scaling=contain&page-id=0%3A1>

1. Проектирование интерфейса страниц системы

1.1. Страница авторизации

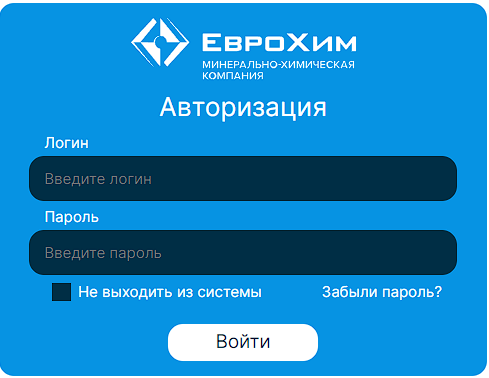


Рисунок 1 – Страница авторизации системы

1.2. «Домашняя» страница

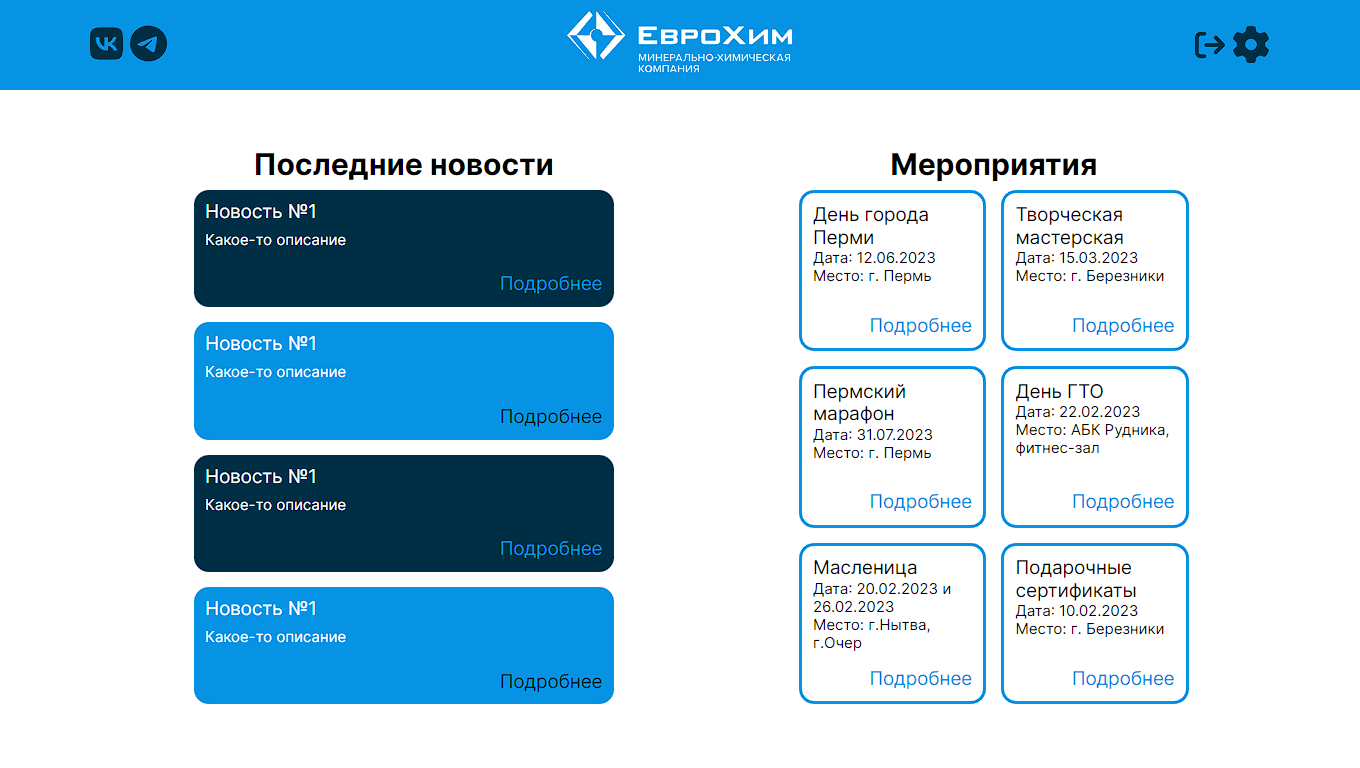


Рисунок 2 – «Домашняя» страница с правами администратора

1.3. Страница «Панель управления»



Рисунок 3 – Страница «Панель управления»

1.4. Страница «Отчет»

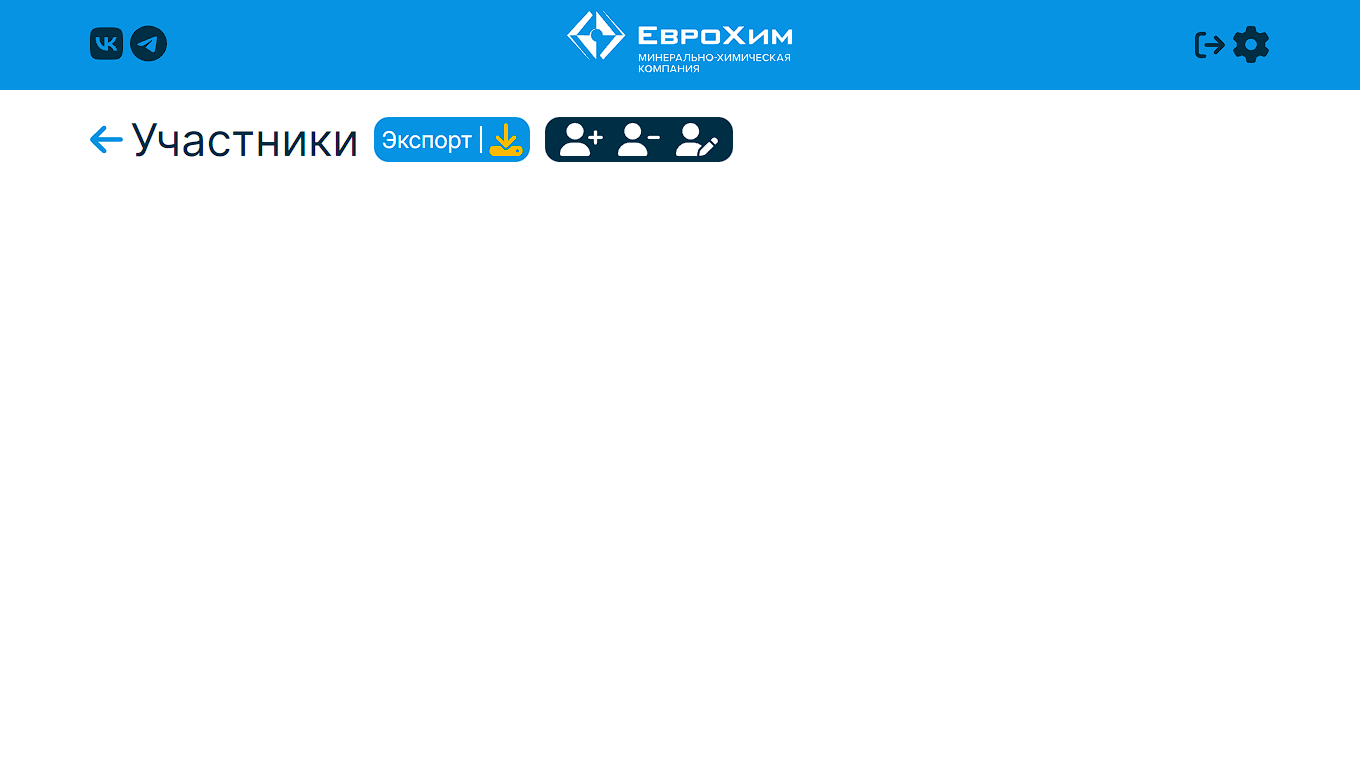


Рисунок 4 – Страница «Отчет»

1.5. Модальное окно с ошибкой

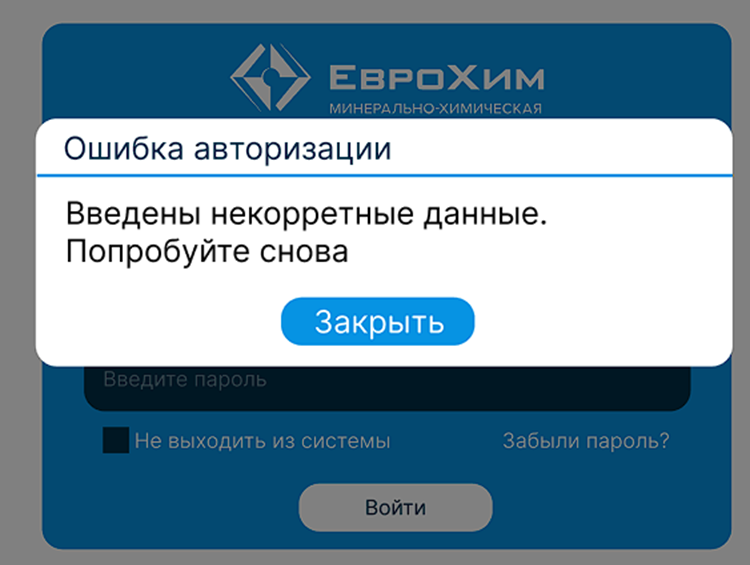


Рис. 5 – Модальное окно, которое сообщает об ошибке

1. Выводы

В данной лабораторной работе был спроектирован интерфейс и отображение отчетов согласно техническому заданию.

Диаграмма классов

**Цель работы:** разработать структуру классов согласно обозначенному функционалу в техническом задании.

**Задачи:**

1. Определить основной функционал системы;
2. Разработать структуру классов для функциональной части;
3. Описать классы и методы.
4. Проектирование диаграммы классов
   1. Диаграмма классов

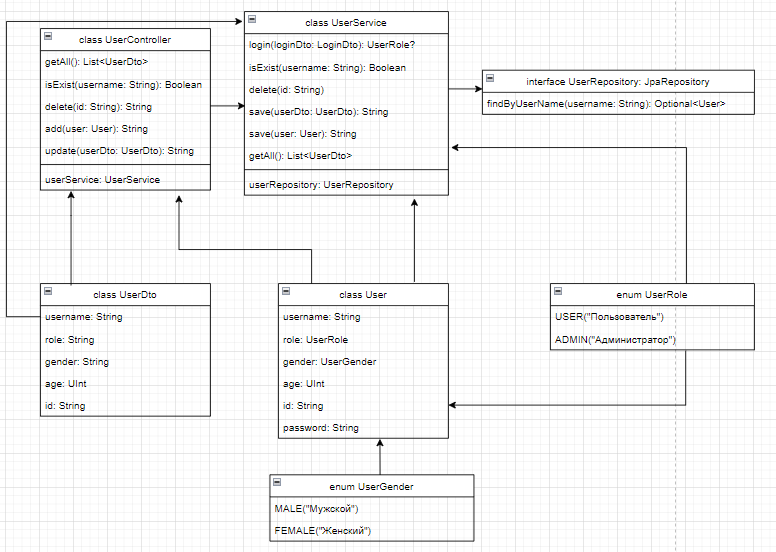
Диаграмма классов представлена на рисунке 1.

Рисунок 1 – Диаграмма классов информационной системы профсоюза ЕвроХима

* 1. Структура классов
     1. UserController

Нужен для получения запросов от «Клиент» и формирования отчетов.

Методы:

1. getAll – Получить список всех пользователей;
2. isExist – Проверить существует ли пользователь с таким логином;
3. delete – Удаление пользователя по id;
4. add – Добавление пользователя;
5. update – Обновление данных о пользователе;
6. userService – Сервис для работы с пользователями;
   * 1. UserService

Класс, в котором происходит вся основная логика работы системы.

Методы:

1. getAll – Получить список всех пользователей из базы;
2. isExist – Проверить существует ли пользователь с таким логином в базе;
3. delete – Удаление пользователя по id из базы;
4. save – Добавление/Обновление пользователя в базе;
5. login – Авторизация пользователя;
6. userRepository – Интерфейс для выполнения запросов к базе;
   * 1. UserRepository

Интерфейс для обращения к базе данных (реализует стандартные методы от JPARepository).

Методы:

1. findByUsername – Поиск пользователя в базе по логину
   * 1. User

Класс представляющий точную копию таблицы из базы.

Столбцы:

* username – Логин пользователя
* password – Пароль пользователя
* age – Возраст пользователя
* gender – Пол пользователя
* role – Роль пользователя
* id – Идентификатор пользователя
  + 1. UserRole

Класс для перечисления возможных ролей.

Возможные значения:

* USER – Роль простого пользователя
* ADMIN – Роль администратора
  + 1. UserGender

Класс для перечисления возможных полов.

Возможные значения:

* MALE – Мужской пол
* FEMALE – Женский пол
  + 1. UserDto

Класс для обмена данными о пользователе.

Столбцы:

* username – Логин пользователя
* age – Возраст пользователя
* gender – Пол пользователя
* role – Роль пользователя
* id – Идентификатор пользователя

В данном примере рассмотрен основной принцип устройства системы, для сущностей \_awards, \_events, \_user\_has\_awards, \_user\_has\_events устройство будет таким же, поменяется только название классов и тип некоторых полей/ переменный в соответствии с сущностью. Аналогичным образом устроен и экспорт, но для него нет отдельной таблицы в базе, он происходит из уже имеющихся таблиц, классов и сущностей.

1. Вывод

В данной лабораторной работу была разработана диаграмма классов и расписаны назначение классов и методов для корпоративной системы учета ППО «ЕвроХим».

Реализация программы

**Цель работы:** запрограммировать обозначенный в ТЗ функционал с учётом созданной БД и разработанных структур классов.

**Задачи:**

1. Перенести пользовательский интерфейс в код;
2. Прописать всю логику программы, описанную в техническом задании.
3. Разработка программы
   1. Создание UI-компонентов

Для правильного выполнения поставленных задач, необходим UI-пакет базовых компонентов, таких как кнопка, текстовой поле и т.д. Разработчиками было принято решение написать этот пакет самостоятельно. Были написаны следующие компоненты:

1. Button – кнопка (прил. 1);
2. ExportButton – кнопка экспорта (прил. 2);
3. InputText- поле для текстового ввода (прил. 3);
4. InputCheckBox – поле переключателя (прил. 4);
5. Modal – модальное окно с различными данными (прил. 5);
6. Table – таблица (использована библиотека react-table).
   1. Инициализация и создание хранилища состояний

В списке технологий, которые предлагались для работы над системой, был менеджер состояний Redux и его дополнение Redux Toolkit. Они и были использованы для работы над системой.

Для их корректной работы необходимо их подключить, инициализировать и создать.

* 1. Вёрстка страниц

Для работы над системой был выбран фреймворк React, который поддерживает и поощряет функциональное программирование. Исходя из чего, классовых компонентов в системы будет очень мало.

Система, на этапе проработки интерфейса, была разделена на четыре части:

1. Страница авторизации (прил. 6);
2. Домашняя страница (прил. 7);
3. Панель управления (прил. 8);
4. Страница с таблицей (прил. 9).

Каждая страница имеет свою логику, поэтому они были вынесены в отдельные компоненты, т.е. каждая страница – отдельный компонент, со своей логикой.

Переходы между страницами возможны благодаря библиотеке react-router-dom, которая позволяет без перезагрузки страницы менять используемые компонент для его отрисовки.

* 1. Проработка логики страниц

После вёрстки страниц, необходимо соединить их с серверной частью, чтобы получать данные, а также иметь возможность их изменять. В качестве технологии общения, была выбрана библиотека axios, которая позволяет отправлять асинхронные запросы на сервер. Выглядит это следующим образом:

export const fetchUserEvents = () => (dispatch: (arg0: IGetUserEvents) => void) => {

    request(RequestTypesEnum.get, RequestApiEnum.getUserEvents, null)

        .then(res => {

            const { data } = res;

            if (!data) {

                return null;

            }

            dispatch({

                type: TABLE\_GET\_USER\_EVENTS,

                userEvents: data,

            })

        })

}

То есть у нас есть встроенная функция request, которая принимает три параметра:

1. Метод запроса: get, pos, delete, put и т.д;
2. API, по которому нужно проводить запрос;
3. Тело запроса (подходит только для post).

Запрос возвращает нам объект типа Promise, из которого мы можем узнать статус запроса и забрать из него данные.

* 1. Добавление реакции на ошибочный ввод

Ошибочный ввод бывает разный, и разработчики могут упростить этап тестирования тем, что наложат правила на вводимые значения. Так, реакции можно разделить на два вида.

* + 1. Валидация полей

Один из способов добавления реакций программы на ввод некорректных данных – валидация полей.

Валидация полей – наложение правил на действия пользователя системы.

Так, например, для страницы авторизации были созданы следующие правила:

1. Логин:
   * Только английские символы;
   * Обязательно к заполнению;
   * Только строки (нельзя использовать только цифры и первый символ – буква.
2. Пароль:
   * Любые символы;
   * Обязательно к заполнению;
   * Не менее 6-ти символов.

Если валидация полей выдает ошибку, то пользователь это увидит. В случае с КИС «ППО ЕвроХим» пользователь не сможет пройти аутентификаю пока не заполнить правильно все поля. Для удобной работы с валидацией были использованы следующие библиотеки:

* React-Hook-Form;
* Yup.

Выглядит это следующим образом:

export const schema = yup.object({

    username: yup

        .string()

        .required('Поле не может быть пустым'),

    password: yup

        .string()

        .required('Поле не может быть пустым')

        .min(6, 'Не менее 6 символов')

});

* + 1. Сообщение об ошибке

Ещё одним способом оповещения пользователя об ошибке является прерывание работы системы и его информирование. С этой работой хорошо справляется модальное окно, которое блокирует всю страницу и выводит отдельное окно с сообщением об ошибке (рис. 1).

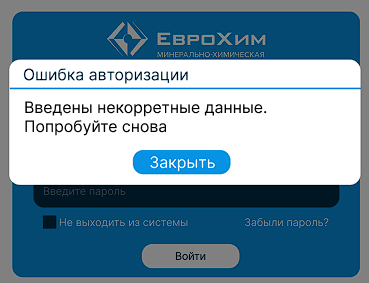


Рис. 1 – Модальное окно с ошибкой

1. Выводы

В данной лабораторной работе была разработана программа информационной системы ППО ЕвроХим. Реализован основной функционал: аутентификация, вид под разными пользователями (в зависимости от роли), редактировании базы данных, формирование отчетов, реакция системы на ошибочный ввод.

Тестирование

**Цель работы:** обозначить назначение и функционал тестирования (объект тестирования)

**Задачи:**

1. Определить объект тестирования
2. Определить виды тестирования
3. Сделать выводы о программном продукте

Этап тестирования системы включает проверку всех описанных функций, а также проверку на уязвимости пользовательского интерфейса.

* 1. 1. Проверка описанных функций

1.1. Авторизация

Были рассмотрены три случая:

1. Данные на заполнены – правила формы не пропустит нажатие кнопки (рис. 1);

2. Данные заполнены, но они неверны – выскачет модальное окно и оповестит пользователя о неправильно введенных данных (рис. 2);

3. Данные заполнены верно – пользователь попадает на «Домашнюю» страницу (рис. 3).



Рис. 1 – Форма без заполненных данных

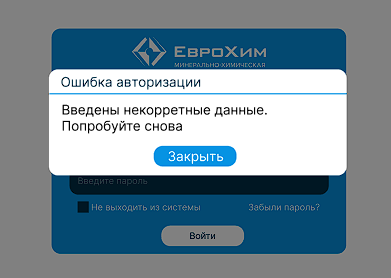


Рис. 2 – Модальное окно с ошибкой



Рис. 3 – Форма с заполненными данными

В данном блоке ошибок в тестировании не было обнаружено.

* 1. Страницы с таблицами

Так как для отображения таблиц используется один компонент, который получает разные данные, будет проверен только один из них. Для корректности был выбран самый функциональный – таблица «Участники». Помимо экспорта, данная таблица имеет управление с данными: добавление, изменение, а также удаление.

При разработке была найдена критическая ошибка, которая ломает страницу и все приложение. Связана она с быстрым открытием страницы, когда данные ещё не успели прогрузиться (рис. 4). В дальнейшем, данная ошибка будет исправлена.

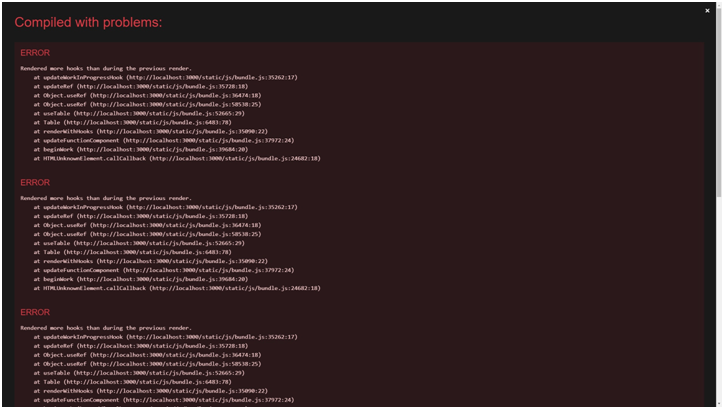


Рис. 4 – Ошибка при входе на страницу «Участники»

При нажатии на кнопку добавления, изменения или удаления пользователя появится модальное окно (рис. 5).

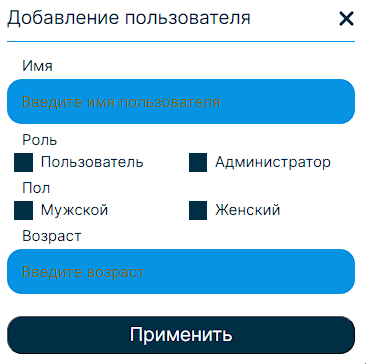


Рис. 5 – Модальное окно «Добавление пользователя»

В данной форме все поля обязательные, т.е. невозможно отправить данные без заполнения. В случае нажатия на кнопку «Применить» без заполнения данных, система оповестит пользователя об ошибках (рис. 6).

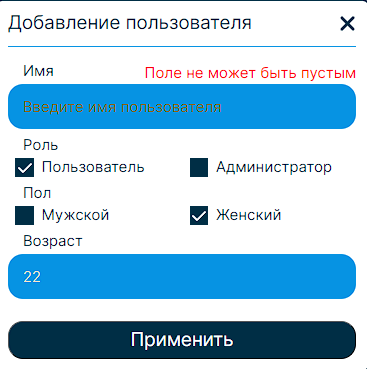


Рис. 6 – Модальное окно «Добавление пользователя» с сообщение об ошибке

После добавления пользователя данные обновляются, и таблица на странице становится актуальной (рис. 7).

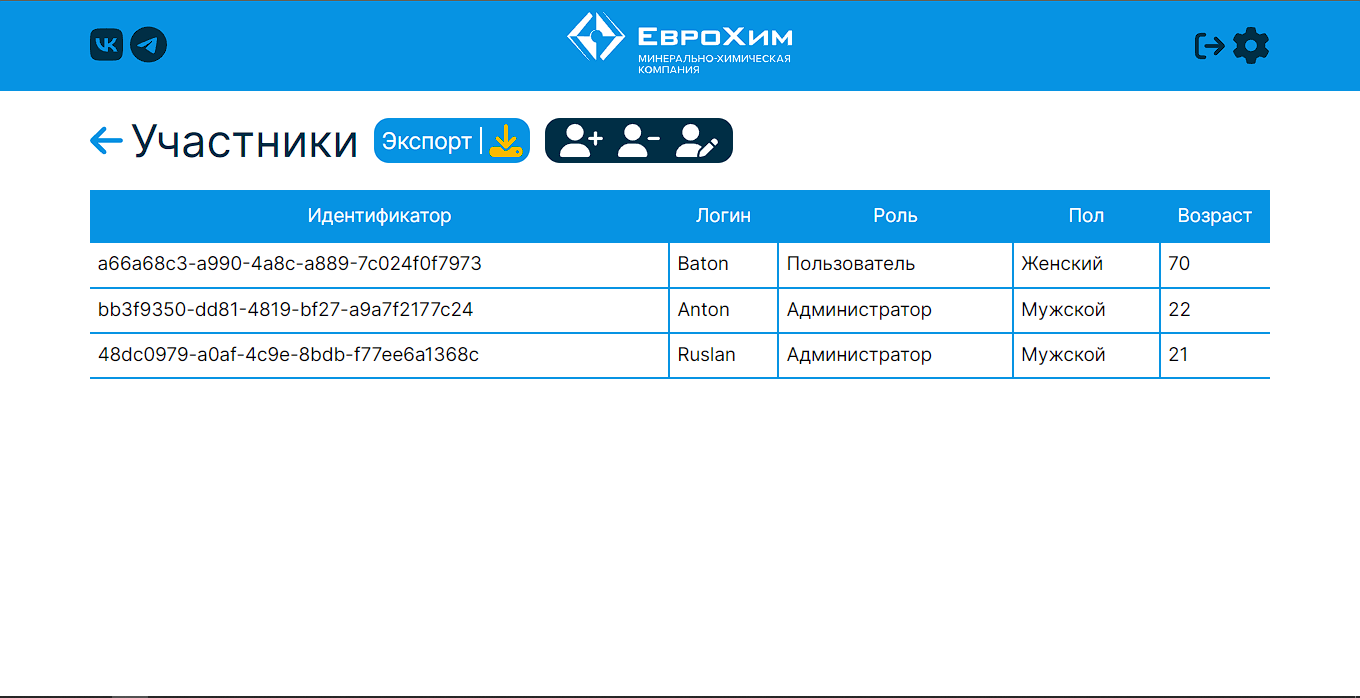


Рис. 7 – Страница «Участники» с обновленными данными

Всего, в ходе тестирования была найдена одна критическая ошибка, которая описана и в дальнейшем будет исправлена.

1. Пользовательский интерфейс

В ходе тестирования были задействованы несколько людей, которые не смогли выявить никаких проблем с пользовательским интерфейсом. В ходе работ, также не были найдены никакие проблемы.

1. Выводы

В данной лабораторной работе проведено функциональное тестирование: аутентификация, вид под разными пользователями (в зависимости от роли), редактировании базы данных, формирование отчетов, реакция системы на ошибочный ввод.

* 1. Вывод

В результате работы была спроектирована корпоративная система учета ППО «ЕвроХим». Было написано техническое задание на разработку системы, рассмотрена предметная область будущей системы, описан весь функционал – значительными функциями является функция авторизации пользователей, добавление, удаление и изменение данных, генерирование отчетов, выгрузка отчетов в формате xls, реакция системы на ошибочно введенные данные. По перечню функций спроектирована логическая и физическая модель данных с загрузкой тестовых данных в таблицы. Спроектирован прототип будущего интерфейса системы, который отражает суть функционала. Разработана диаграмма классов на основе функциональных требований и физической и логической модели базы данных. В результате работы было проведено функциональное тестирование функции авторизации, формирования отчетов, выгрузки отчетов в Excel, реакции системы на ошибки. Также было проведено нефункциональное тестирование по интерфейсу системы. Все функции в ходе тестирования отработали корректно.