

## Содержание

Введение	3
1 Назначение и цели разработки	4
2 Разработка технического проекта на основе анализа требований	5
2.1 Определение спецификаций программного обеспечения	5
2.2 Проектирование модели данных и диаграммы классов	9
2.3 Конструирование прототипа	12
3 Реализация	18
3.1 Обоснование выбора средств разработки	18
3.2 Реализация базы данных в среде СУБД	19
3.3 Описание программных модулей	21
4 Тестирование программных модулей	30
4.1 Интеграционное тестирование	30
5 Эксплуатационная документация	31
5.1 Руководство пользователя	31
5.2 Руководство программиста	32
Заключение	34
Список использованных источников	35
Приложение А Техническое задание. Требования к программным модулям	37
Приложение Б Программный код	43
Приложение В Тест-кейсы	47
Приложение Г Руководство пользователя	49
Приложение Д Руководство программиста	50

					<b>ККЭП 09.02.07 0104 ПЗ</b>			
Изм.	Лист	№ докум.	Подпись	Дата				
Разраб.	Арьков				Разработка модулей подсистемы "Управление задачами" информационной системы "FixPhoto" ООО "Зодиак-Электро". Пояснительная записка	Лит.	Лист	Листов
Провер.	Головко					ДП	2	50
Рецензент	Евтушенко					<b>Гр. 632-Д9-4ИСП</b>		
Н.контр.	Головко							
Утв.	Головко							

## Введение

В наше время очень популярны веб-сайты и веб-приложения. Они удобны для компаний, потому что являются кросс-платформенными, масштабируемыми и могут использоваться на мобильных устройствах и компьютерах.

В последнее время стали популярны сайты на различных фреймворках, т.к. они ускоряют работу сайтов и дают дополнительную защиту. Так же они позволяют пользоваться определенными преимуществами для разработчиков. Сейчас есть три самых популярных фреймворка: React, Angular, VueJS.

Компания ООО «Зодиак-Электро» занимается монтажными работами. Одна из важных проблем в данной компании является автоматизация взаимодействия менеджера и специалистов по монтажу. Менеджер должен создавать объекты, добавлять специалистов и назначать задачи на специалиста, а специалист должен получать и выполнять назначенные задачи, прикрепляя к ним фотографии для отчетности.

Тема выпускной квалификационной работы актуальна, потому что необходимо максимально автоматизировать процесс управления задачами.

## 1 Назначение и цели разработки

Заданием предусмотрена разработка модулей подсистемы "Управление задачами" информационной системы «FixPhoto» ООО «Зодиак-Электро». Разработанные модули предназначены для автоматизации работы менеджера в ООО «Зодиак-Электро».

Автоматизация позволит получить следующие преимущества:

- ускорение процесса взаимодействия менеджера и специалиста по монтажу;
- удобное и быстрое создание задач;
- удобное и быстрое обновление информации об объектах;
- добавление новых специалистов по монтажу;
- быстрое формирование отчетов по выполненным работам за выбранный менеджером период.

Данные модули позволят максимально сократить наличие человеческого фактора в ряде важных задач, таких как создание новых объектов, специалистов, задач. Помимо этого модули автоматизируют сотрудничество между менеджерами и специалистами по монтажу, а также позволят формировать отчеты за выбранный менеджером период времени.

					ККЭП 09.02.07 0104 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		4

## 2 Разработка технического проекта на основе анализа требований

### 2.1 Определение спецификаций программного обеспечения

Рассмотрим определение вариантов использования (прецедентов).

Подсистема «Управления задачами» для системы FixPhoto предназначена для менеджеров. Соответственно основные прецеденты (варианты использования) для разрабатываемой подсистемы следующие:

- добавить задачу;
- добавить объект;
- сформировать отчет;
- посмотреть фотографии.

Диаграмма вариантов использования, созданная средствами MS Visio, для проектируемой системы представлена на рисунке 1 [5].

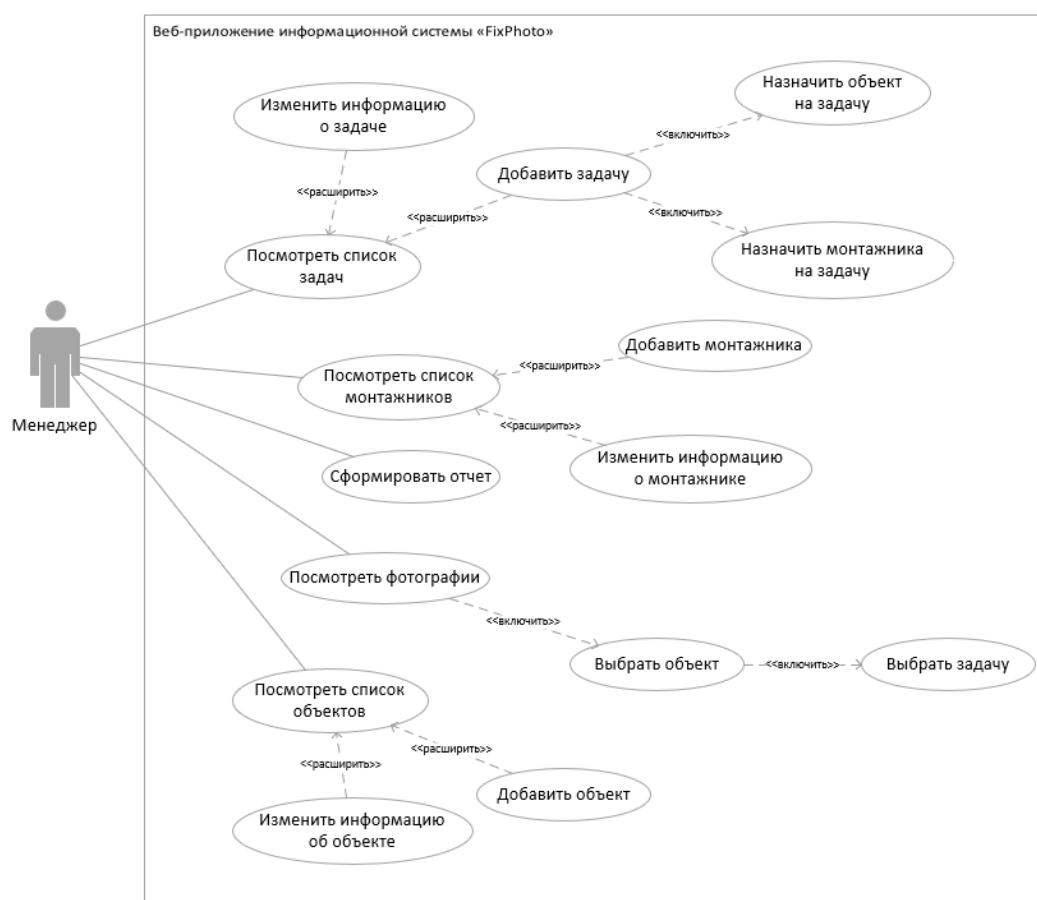


Рисунок 1 – Диаграмма вариантов использования веб-приложения информационной системы “FixPhoto”

В таблице 1 представлено описание главного раздела сценария прецедента (варианта использования) «Добавить задачу».

Таблица 1 - Главный раздел сценария «Добавить задачу»

Вариант использования	Создать задачу
Актеры	Менеджер
Краткое описание	Менеджер добавляет задачу на заданный объект компании и выбирает монтажника, который будет выполнять задачу
Цель	Успешно добавить задачу
Тип	Базовый
Ссылки на другие варианты использования	Включает в себя создание объекта или монтажника при его отсутствии в базе данных

В таблице 2 описана последовательность действий менеджера, приводящая к успешному выполнению прецедента (варианта использования) «Добавить задачу».

Таблица 2 - Сценарий успешного выполнения варианта использования «Добавить задачу»

Действия актеров	Отклик системы
1. Менеджер создает задачу Исключение 1. В списке нет необходимого объекта или монтажника	2. Система возвращает список объектов 3. Система возвращает список монтажников
4. Менеджер выбирает объект и монтажника 5. Менеджер сохраняет задачу	6. Система сохраняет задачу

В таблице 3 представлен сценарий обработки исключительных ситуаций для варианта использования «Добавить задачу».

Таблица 3 - Обработка исключительных ситуаций для варианта использования «Добавить задачу»

Действия актеров	Отклик системы
Исключение 1. В списке нет необходимого объекта или монтажника	
4. Менеджер создает новый объект или монтажника	5. Система добавляет новый объект или монтажника в базу данных
6. Менеджер сохраняет задачу	7. Система сохраняет задачу

С помощью диаграммы деятельности, представленной на рисунке 2, описан алгоритм реализации описанных сценариев для варианта использования «Добавить задачу».

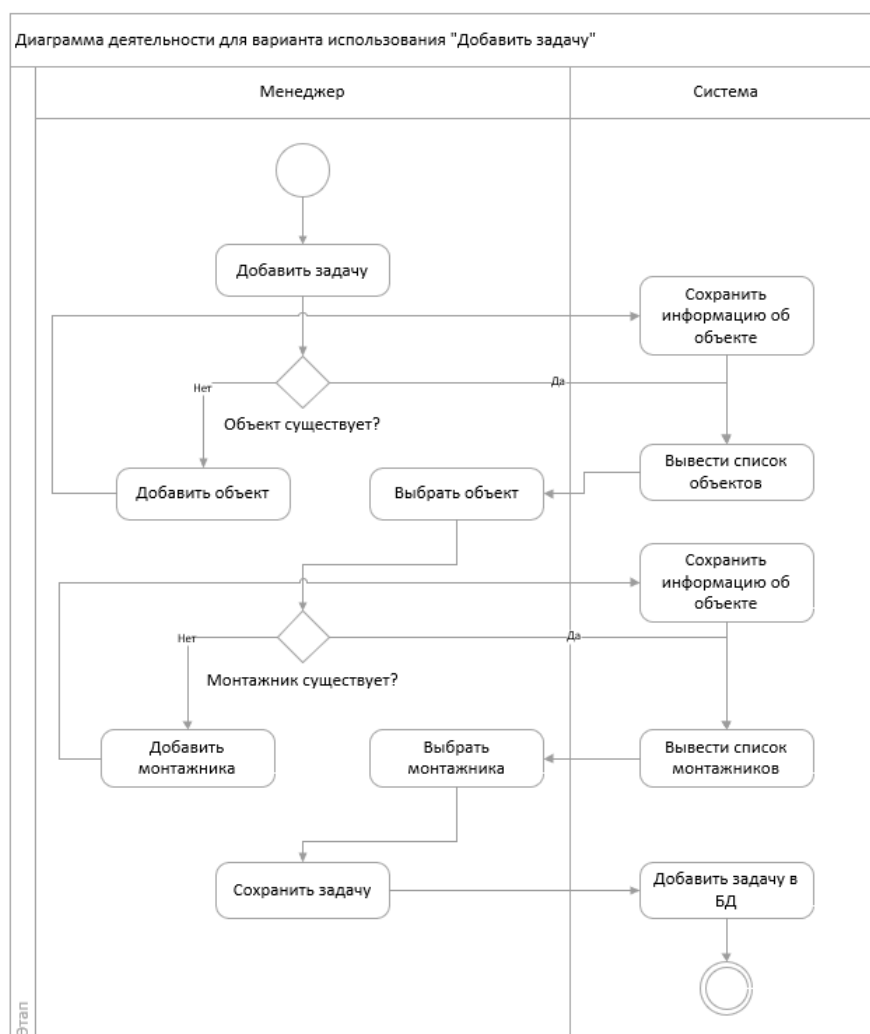


Рисунок 2 – Диаграмма деятельности для варианта использования «Добавить задачу»

```
sequenceDiagram
    actor Manager
    participant Object
    participant Assembler
    participant Task
    participant System

    alt "[Объект и монтажник есть]"
        Manager->>Task: Добавить задачу
        Task->>System: Добавить задачу в БД
        System-->>Manager: Сохранить задачу
    end

    alt "[Объекта нет]"
        Manager->>Object: Добавить объект
        Object->>System: Добавить объект в БД
        System-->>Manager: Сохранить объект
        Manager->>Task: Добавить задачу
        Task->>System: Добавить задачу в БД
        System-->>Manager: Сохранить задачу
    end

    alt "[Монтажника нет]"
        Manager->>Assembler: Добавить монтажника
        Assembler->>System: Добавить монтажника в БД
        System-->>Manager: Сохранить монтажника
        Manager->>Task: Добавить задачу
        Task->>System: Добавить задачу в БД
        System-->>Manager: Сохранить задачу
    end
```

азработанные спецификации программного обеспечения на языке UML  
ми MS Visio показывают основные действия пользователя и алгоритмы их  
ния.

## 2.2 Проектирование модели данных и диаграммы классов

На основе анализа требований заказчика к модулям подсистемы «Управление задачами», требованиям к организации входных и выходных данных, с учетом спроектированных требований к реализации функций, описанных в диаграммах вариантов использования, деятельности и последовательности, была разработана модель данных и описана в виде ER-модели, позволяющая четко описать требования к представлению логической структуры данных, на основе которой в последующем будет разработана физическая структура данных для хранения на сервере.

На рисунке 4 представлена ER–модель базы данных.

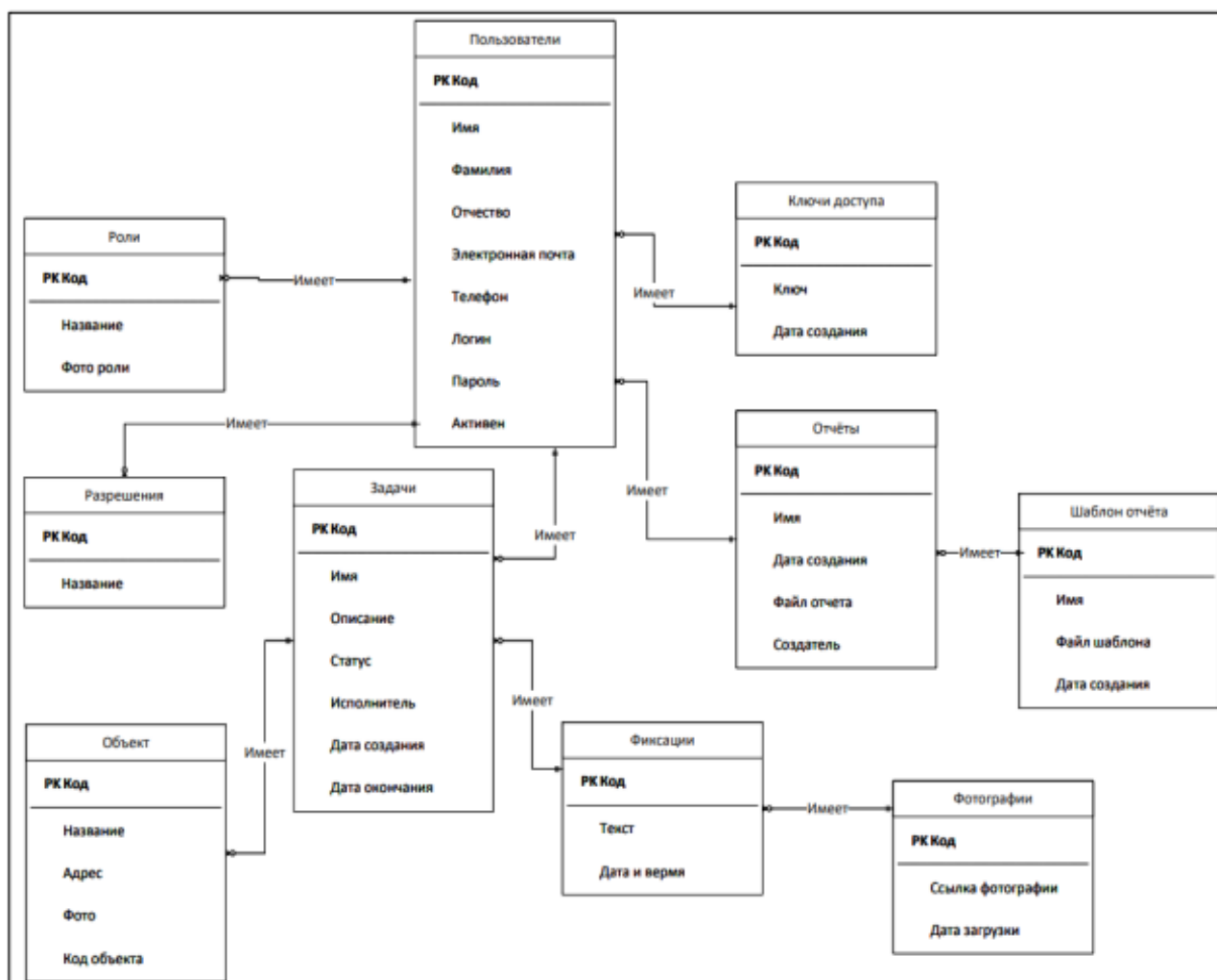


Рисунок 4 – ER-модель базы данных



На рисунке 5 представлена диаграмма классов для модулей подсистемы «Управление задачами» приложения



Данная диаграмма классов описывает объекты, которые будут использоваться в модулях подсистемы «Управление задачами», а также отображаются операции класса.

В таблице 4 представлены сведения о структуре сущностей, свойствах, а также их методах.

Таблица 4 - Характеристика диаграммы классов для локальной базы данных

Имя класса	Атрибуты класса	Методы класса
Facility	+id – идентификатор объекта; +title – название объекта; +photoPath – содержит ссылку на фотографию; +address – адрес объекта; +code – код объекта; +taskId – идентификатор задачи.	+getFacilities () – получение всех объектов; +searchFacility (String str) – поиск объекта; +getFacilityById (Integer Id) – получение объекта по идентификатору; +deleteFacilityById (Integer Id) – удаление объекта по идентификатору;
Task	+id – идентификатор задачи; +title – название задачи; +description – описание задачи; +status – статус задачи; +createTime – дата создания; +finishTime – дата завершения; +userId – идентификатор пользователя.	+getTasks() – получение всех задач; +searchTask(String str) – поиск задачи; +getTaskById(Integer Id) – получение задачи по идентификатору; +deleteTaskById(Integer Id) – удаление задачи по идентификатору; +getTaskPhotos(Integer Id) – получение фотографий по идентификатору.
Fixing	+id – идентификатор фиксации; +message – комментарий к выполненной задаче; +finishTime – дата добавления; +taskId – идентификатор задачи.	+addFixingPhoto(Integer rotate) – добавление фиксации.
Photo	+id – идентификатор фотографии; +path – ссылка на фотографию; +finishTime – дата добавления; +fixingId – идентификатор фиксации.	+addPhoto(List<String>) – добавление фотографии; +getPhotos(Integer fixId) – получение всех фотографий.
TemplateReport	+id – идентификатор шаблона отчета; +createdAt – дата создания; +file – тип отчета.	+getTemplate() – получение всех видов отчета.

## Продолжение таблицы 4

Имя класса	Атрибуты класса	Методы класса
Users	+id – идентификатор пользователя; +firstName – имя пользователя; +lastName – фамилия пользователя; +login – логин пользователя; +password – пароль пользователя; +phone – телефон пользователя; +email – электронная почта пользователя.	+getAllUsers() – получение всех пользователей; +getUserById(Integer Id) – получение одного пользователя по идентификатору; +deleteUserById(Integer Id) – удаление пользователя по идентификатору.
Report	+id – идентификатор отчета; +name – имя отчета; +status – статус отчета; +createdAt – дата создания; +userId – идентификатор пользователя; +templateReportId – идентификатор отчета.	+getAllReports() – получение всех отчетов; +downloadReportById(Integer Id) – скачивание выбранного отчета.

Данная диаграмма классов и ее описание служат для представления статической структуры модели локальной базы данных приложения.

### 2.3 Конструирование прототипа

Для построения прототипа первоначально были составлены требования к макету приложения.

Все компоненты должны иметь единый согласованный внешний вид, соответствующий руководству по стилю, а также следующим требованиям:

- разметка и дизайн (разметка должна быть масштабируема, так как устройства могут различаться размером дисплея);
- использование соответствующих элементов управления;
- расположение и выравнивание элементов;
- общая компоновка логична, понятна и проста в использовании;

- последовательный пользовательский интерфейс, позволяющий перемещаться между существующими окнами в приложении;
- соответствующий заголовок на каждом окне приложения.

Основные требования руководства по стилю:

- для приложения должна быть установлена иконка, полученная от заказчика, изображенная на рисунке 6;
- тип шрифта – Roboto;
- цветовая схема предусматривает использование в качестве основного фона – белый цвет RGB (255, 255, 225), в качестве дополнительных – синий цвет #1e5181



Рисунок 6 – Логотип приложения

С учетом требований к макету и руководству по стилю, для обеспечения требуемых функций, были разработаны макеты экранов приложения, в последствии утвержденные заказчиком. Для разработки макетов была использована Figma – онлайн-сервис для дизайнеров, разработчиков и маркетологов, предназначенный для создания прототипов сайтов или приложений, иллюстраций и векторной графики.

На рисунке 7 представлен прототип страницы авторизации для пользователей [10].

# FixPhoto

Система фотофиксации, контроля и исполнения

Войдите в свой аккаунт

Продолжить



Реализация программных решений для бизнеса

Рисунок 7 – Прототип страницы авторизации для пользователей

На рисунке 8 представлен прототип страницы с объектами, с которым пользователь будет работать и видеть все существующие объекты. На экране находится меню, расположенное сверху, основной контент в виде списка объектов, кнопки создания объекта и строки поиска [7].

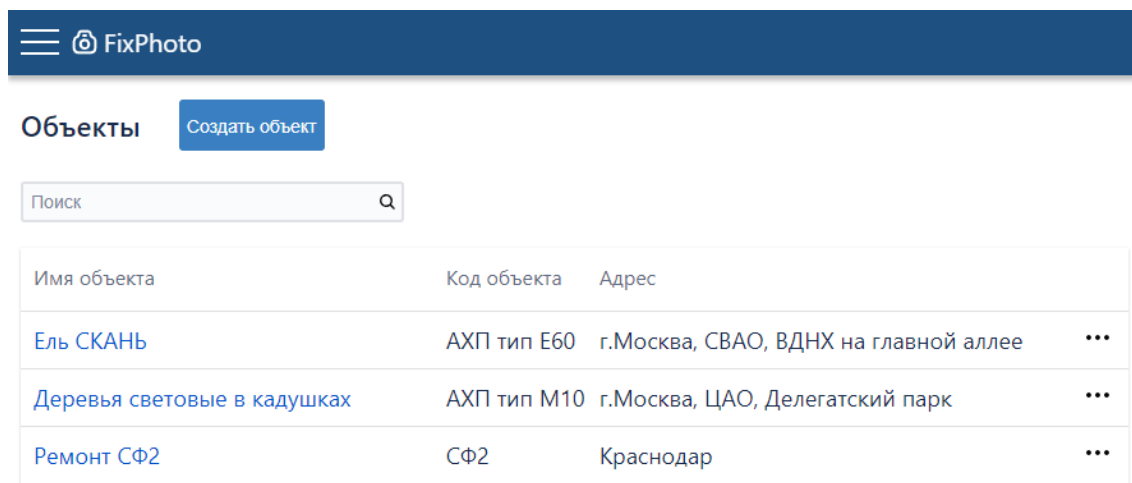


Рисунок 8 – Прототип страницы с объектами

На рисунке 9 представлен прототип страницы с задачами, с которым пользователь будет работать и видеть все существующие задачи. На экране находится меню, расположенное сверху, основной контент в виде списка задач, кнопки создания задачи.

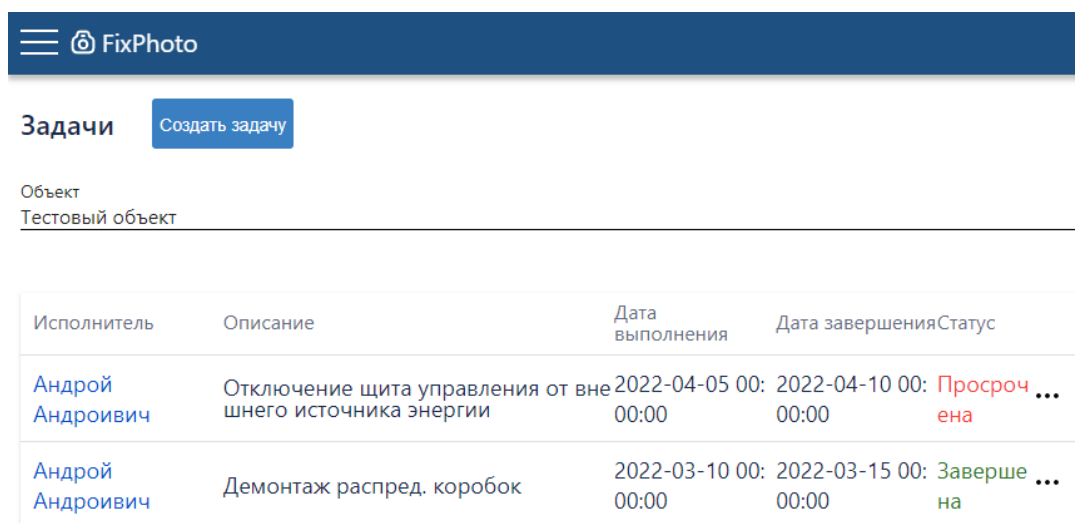


Рисунок 9 – Прототип страницы с задачами

На рисунке 10 представлен прототип страниц с фотографиями, с которым пользователь будет работать и видеть все сделанные фотографии монтажником по определенной задаче. На экране находится меню, расположенное сверху, выбор объекта и задачи, а так же основной контент в виде фотографий [8].



Рисунок 10 – Прототип страницы с фотографиями

На рисунке 11 представлен прототип страницы с отчетами, которые пользователь может установить или удалить. На экране находится меню, расположенное сверху, основной контент в виде списка отчетов.

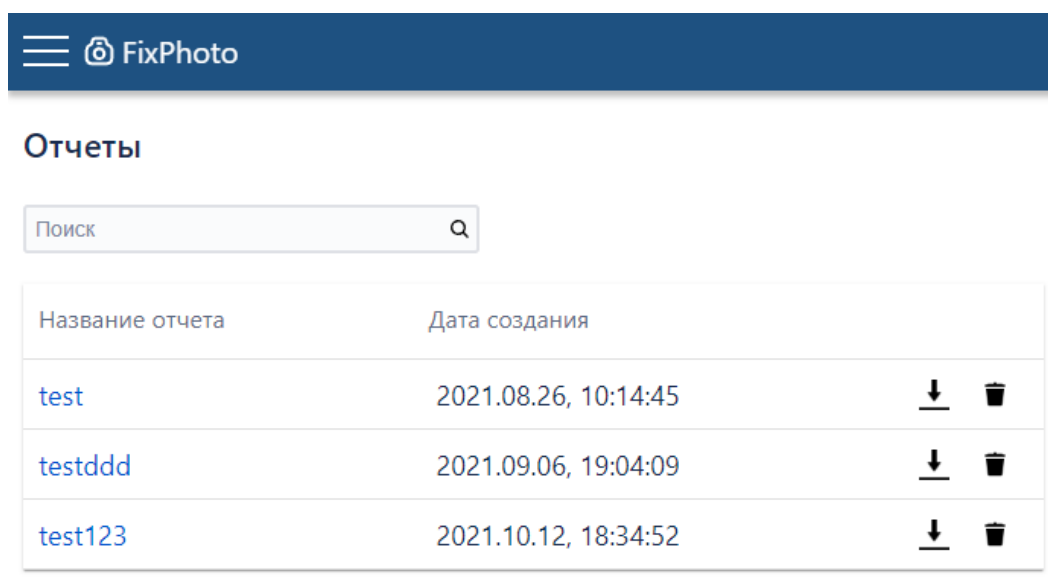


Рисунок 11 – Прототип страницы с отчетами

На рисунке 12 представлен прототип страницы с исполнителями, которые пользователь может редактировать или удалить. На экране находится меню, расположенное сверху, основной контент в виде списка исполнителей.

ФИО	Логин	
Игнатик Игнатьевич	ignatok	...
Андрой Андройвич	android	...
Ирина Алейник	irina	...

Рисунок 12 – Прототип страницы с исполнителями

Данные прототипы основных экранов отображают главный функционал для менеджеров, а именно просмотр объектов, задач, исполнителей, отчетов и возможность отредактировать или удалить, а так же просмотр фотографий по выполненным задачам. Эти макеты были утверждены заказчиком перед началом разработки.



### 3 Реализация

#### 3.1 Обоснование выбора средств разработки

Так как заказчиком была поставлена задача разработки подсистемы в виде веб-приложения, то были рассмотрены только веб-средства для разработки [3].

В таблице 5 представлены некоторые возможные средства разработки для подсистемы «Управления задачами».

Таблица 5 – Возможные средства разработки

Язык программирования и технология	Плюсы	Минусы
ReactJS	Использует virtual DOM, ленивая загрузка, рендер дочерних элементов, использует компонентно-ориентированную архитектуру, высокая масштабируемость, постоянная поддержка сообщества.	Использует только JSX, для обеспечения защиты приложения нужны знания и опыты, плохая официальная документация.
VueJS	Использует virtual DOM, использует HTML-шаблоны и JSX, архитектура подходит для более крупных приложений, подробная официальная документация, маленький размер приложения	Плохая поддержка сообщества, плохая масштабируемость, плохая гибкость
AngularJS	Средняя поддержка сообщества, высокая скорость разработки, использует MVC	Сложная архитектура, сложная структура, двунаправленное связывание данных между областями видимости

ReactJS очень хороший и новый фреймворк для разработки маленьких или средних приложений. У него очень большая поддержка пользователей. По этому есть множество библиотек, которые делают разработку более быстрой и легкой [6].

VueJS более новый фреймворк, чем ReactJS. Он имеет огромную официальную документацию, но маленькую поддержку сообщества. Подойдет больше для крупных проектов.

Фреймворк AngularJS устарел и уступает в скорости работы ReactJS и VueJS. Он по прежнему является хорошим фреймворком для определенных проектов, но большая часть веб разработчиков начали использовать ReactJS и VueJS из-за больших преимуществ.

Выбор останавливается на ReactJS, потому что остальные фреймворки не совсем подходят для разработки данных модулей «Управления задачами».

### 3.2 Реализация базы данных в среде СУБД

На основе ранее спроектированной ER-диаграммы в среде Microsoft SQL Server Management Studio 18 была разработана база данных FixPhoto. На рисунке 13 представлена часть базы, с которой непосредственно работает веб-клиент.

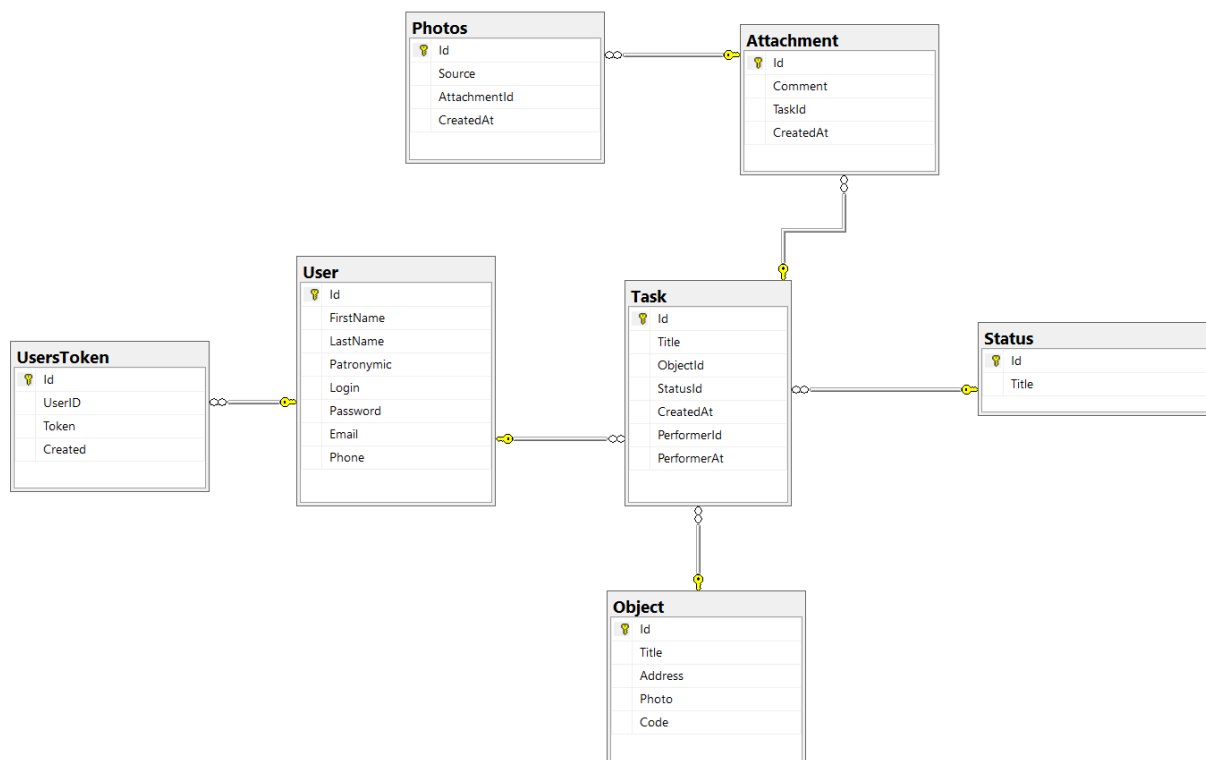


Рисунок 13 – Фрагмент базы данных для обеспечения функционирования подсистемы «Управление задачами»

В таблице 6 представлено описание таблиц базы данных FixPhoto, представленных на рисунке 13.

Таблица 6 – Возможные средства разработки

Название таблицы	Краткое описание
User	Предназначена для хранения информации о пользователях
UserTokens	Предназначена для хранения информации о ключах доступа пользователей
Attachment	Предназначена для хранения информации о фиксациях специалистов по монтажу
Task	Предназначена для хранения информации о задачах
Object	Предназначена для хранения информации об объектах
Status	Предназначена для хранения информации о статусах задач
Photos	Предназначена для хранения фотографий, прикрепленных к задаче

Для данной базы данных было разработано API, с помощью которого модули «Управления задачами» взаимодействуют с сервером.

### 3.3 Описание программных модулей

Структура проекта разделена на несколько частей: корневая папка, конфиг, папка с установленными пакетами (node\_modules), папка с иконой и файлом html (public), папка с скриптами для запуска приложения (scripts), папка содержащая весь контент приложения (src).

В папке, которая содержит весь контент приложения находятся: стили приложения, изображения, компоненты, шрифты, страницы и инструмент для управления состояниями (redux).

Так же в корневой папке приложения находятся файлы routes.jsx, index.js, App.js.

Файл routes.jsx предназначен для написания маршрутизации приложения.

Файл index.js предназначен для интеграции всего js кода в index.html, который находится в папке public.

Файл App.js предназначен для подключения маршрутизации.

На рисунке 14 представлена информация о структуре проекта приложения

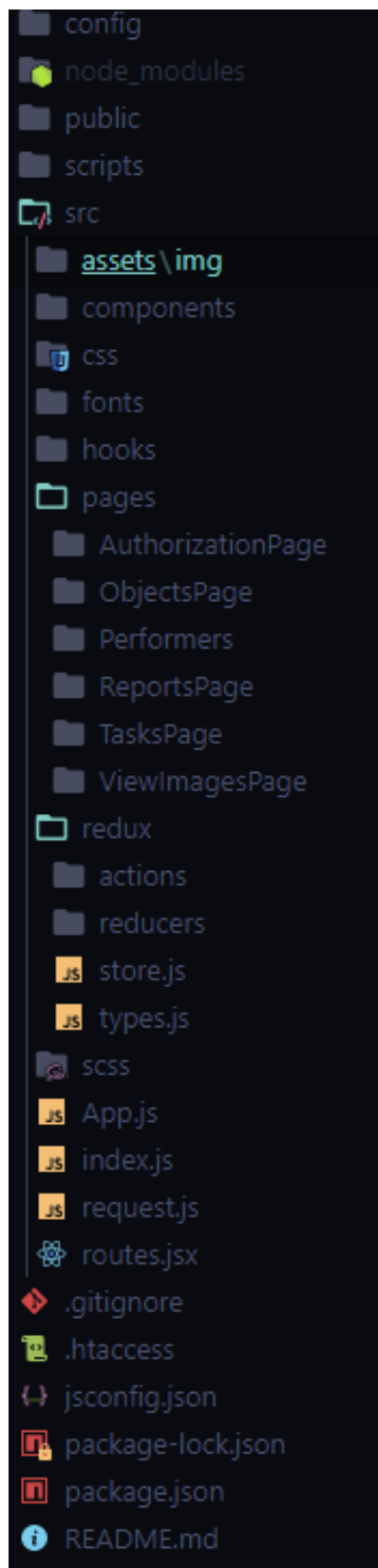


Рисунок 14 – Структура приложения

					<i>ККЭП 09.02.07 0104 ПЗ</i>	Лист
Изм.	Лист	№ докум.	Подпись	Дата		22

В таблице 7 описаны назначения директорий из проекта приложения.

Таблица 7 – Назначение директорий

Название директории	Назначение
config	Хранит конфигурационные файлы для сборки и скриптов
node_modules	Хранит установленные пакеты
public	Хранит иконку приложения, логотип, html файл
scripts	Хранит скрипты для работы с приложением
src	Хранит все содержимое приложения
assets	Хранит все статические ресурсы приложения
components	Хранит компоненты приложения, используемые на разных страницах
css	Хранит все файлы с типом CSS
fonts	Хранит все файлы с шрифтами
hooks	Хранит все пользовательские хуки
pages	Хранит папки с страницами и компонентами, которые привязаны к страницам
redux	Хранит actions и reducers
scss	Хранит все файлы с типом SCSS

В процессе разработки приложения, макеты были изменены в целях удобства для пользователей и расширения функционала. При первом открытии приложения открывается страница авторизации, представленный на рисунке 15. На данной странице пользователь вводит логин и пароль и нажимает на кнопку «Продолжить» для авторизации. На рисунке Б.1 представлен код обработчика кнопки «Продолжить».



Рисунок 15 – Страница авторизации

При успешной авторизации, открывается страница с объектами. Если учетные данные неверные, то появляется всплывающее уведомление о том, что необходимо ввести правильные данные. На рисунке 16 изображена страница, на которой есть возможность просмотра, поиска и создания новых объектов и при необходимости удалить или изменить объекты. Поиск работает по имени и коду объекта. На рисунке Б.2 представлен код обработчика кнопки «Создать объект».

FixPhoto

Объекты

Создать объект

Поиск

Q

Имя объекта	Код объекта	Адрес	
Ель СКАНЬ	АХП тип Е60	г.Москва, СВАО, ВДНХ на главной ал лее	...
Деревья световые в кадешках	АХП тип М10	г.Москва, ЦАО, Делегатский парк	...
Ремонт СФ2	СФ2	Краснодар	...
Ремонт СФ2,5	СФ2,5	Краснодар	...

Рисунок 16 – Страница с объектами

При нажатии на кнопку «Создать объект» появится окно с возможностью ввода информации в поля ввода для добавления объекта в БД, представленное на рисунке 17. На рисунке Б.3 представлен код обработчика кнопки «Сохранить» для создания объекта [11].

Новый объект

Имя

Адрес

Код объекта

Договор

Количество

1

Сохранить

Рисунок 17 – Окно добавления объекта



При нажатии на ссылку «Задачи» в верхнем меню открывается новая страница, представленная на рисунке 18. На ней можно создать новую задачу, а так же посмотреть текущие по выбранному объекту и при необходимости удалить или изменить их. На рисунке Б.4 представлен код обработчика «Создать задачу».

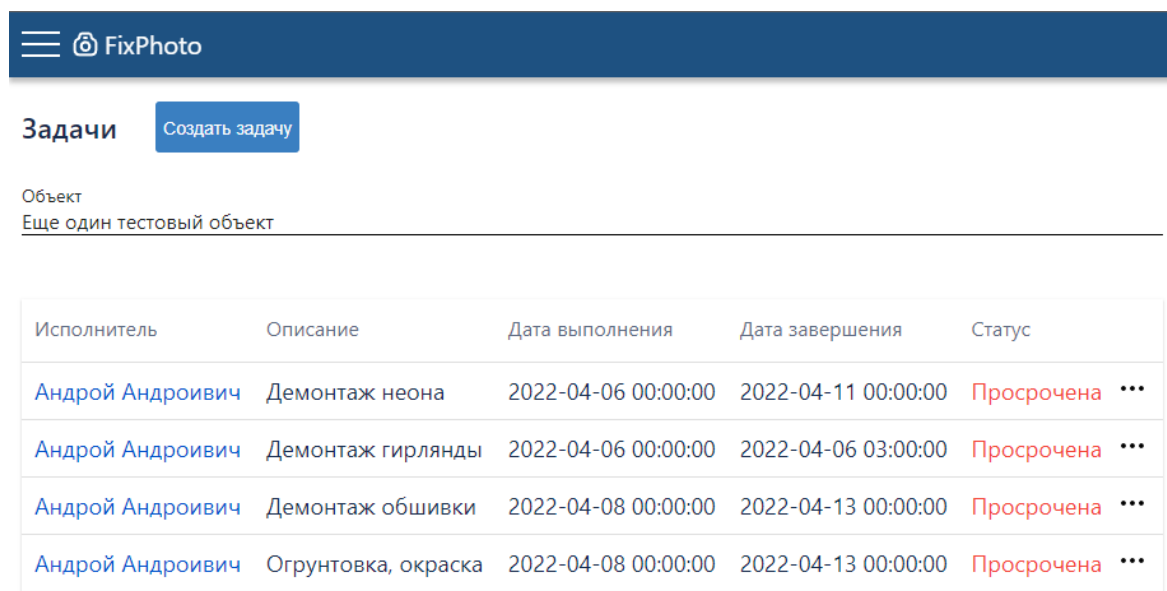


Рисунок 18 – Страница с задачами

При нажатии на кнопку «Создать задачу» появится окно с возможностью ввода информации в поля ввода для добавления задачи в БД, представленное на рисунке 19. На рисунке Б.5 представлен код обработчика кнопки «Сохранить» для создания задачи [9].

Новая задача

Описание

Количество фотографий  
1

Время на исполнение  
12

Объект

Исполнитель

Дата

Время

Сохранить

Рисунок 19 – Окно добавления задачи

При нажатии на ссылку «Фотографии» в верхнем меню открывается новая страница, представленная на рисунке 20. На ней можно посмотреть фотографии по выбранному объекту и задаче, изменить ориентацию картинки и время создания.

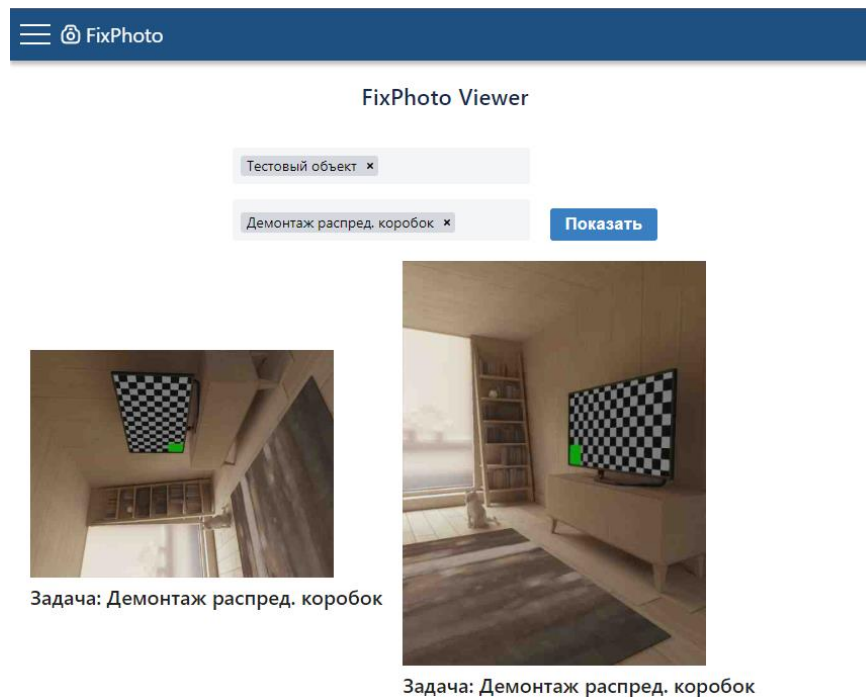


Рисунок 20 – Страница с фотографиями

При нажатии на картинку появится окно с возможностью редактирования даты или времени и ориентации для сохранения в БД представленное на рисунке 21. На рисунке Б.6 представлен код обработчика при нажатии на фотографию.

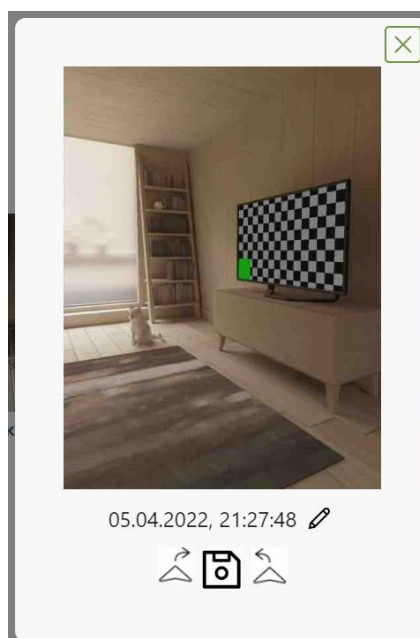


Рисунок 21 – Окно изменения картинки

Изм.	Лист	№ докум.	Подпись	Дата

ККЭП 09.02.07 0104 ПЗ

Лист

28

При нажатии на ссылку «Отчеты» в верхнем меню открывается новая страница, представленная на рисунке 22. На ней можно посмотреть отчеты, скачать или удалить выбранный отчет. Поиск работает по названию отчета. На рисунке Б.7 представлен код обработчика при нажатии на кнопку скачивания отчета.

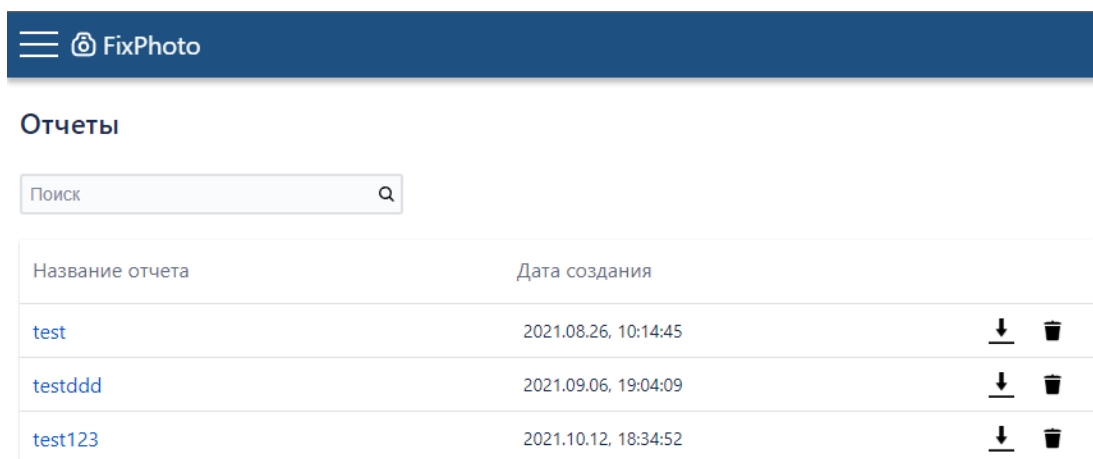


Рисунок 22 – Страница с отчетами

При нажатии на ссылку «Исполнители» в верхнем меню открывается новая страница, представленная на рисунке 23. На ней можно посмотреть отчеты, скачать или удалить выбранный отчет. Поиск работает по фамилии и имени. На рисунке Б.8 представлен код обработчика при нажатии на кнопку «Создать исполнителя».

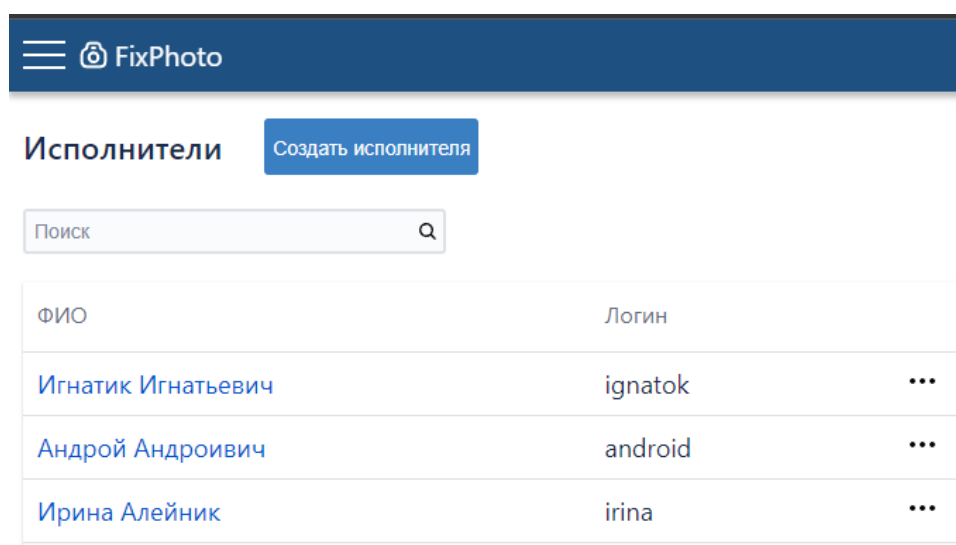


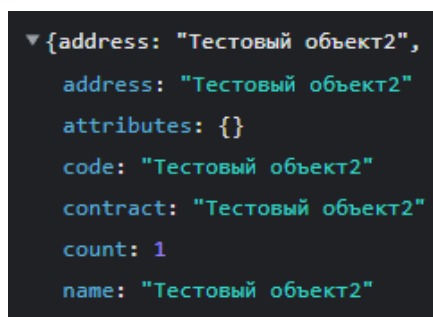
Рисунок 23 – Страница с исполнителями

## 4 Тестирование программных модулей

### 4.1 Интеграционное тестирование

Необходимо протестировать способность веб-приложения FixPhoto обрабатывать разные ответы, полученные с сервера [4]. Задачей тестирования является проверка запроса `postObject()`, который добавляет новый объект.

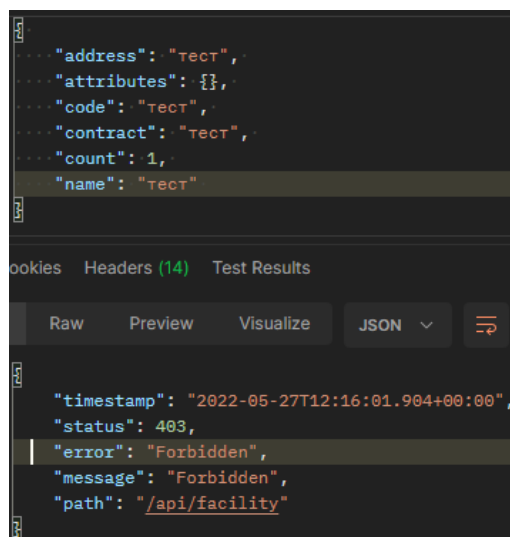
На рисунке 24 представлен результат позитивного теста.



```
▼ {address: "Тестовый объект2",  
  address: "Тестовый объект2"  
  attributes: {}  
  code: "Тестовый объект2"  
  contract: "Тестовый объект2"  
  count: 1  
  name: "Тестовый объект2"}
```

Рисунок 24 – Результат позитивного теста

На рисунке 25 представлен результат негативного теста.



```
... "address": "тест",  
... "attributes": {},  
... "code": "тест",  
... "contract": "тест",  
... "count": 1,  
... "name": "тест"  
...  
"timestamp": "2022-05-27T12:16:01.904+00:00",  
"status": 403,  
"error": "Forbidden",  
"message": "Forbidden",  
"path": "/api/facility"
```

Рисунок 25 – Результат негативного теста

Негативный тест заключался в том, чтобы проверить, как отработает запрос `postObject()`, если токен устареет.

Тест-кейсы представлены в приложении В.

## 5 Эксплуатационная документация

### 5.1 Руководство пользователя

Разработанное приложения используется в компании ООО «Зодиак-Электро». Оно предоставляет следующие возможности:

- быстрое и удобное добавление новых объектов, задач, исполнителей;
- формирование отчетов;
- назначение задачи на исполнителя и объект.

Приложение максимально позволит автоматизировать процесс «Управления задачами» и позволит формировать отчеты по выбранному периоду. Работа с приложением возможна только при наличии интернет-соединения. Для работы необходимо иметь пароль и логин, полученный от администратора компании [2].

Для работы с приложением необходимо иметь любой браузер.

Перед началом работы необходимо зайти на сайт <http://u129434.test-handyhost.ru>.

Для проверки доступности приложения с рабочего места необходимо выполнить следующие условия:

- открыть браузер;
- зайти на сайт <http://u129434.test-handyhost.ru>;
- в форме аутентификации ввести пользовательский логин и пароль. Нажать кнопку «Продолжить»;
- убедиться, что в окне открылось приложение.

В случае возникновения ошибок при работе с приложением необходимо обратиться к сотруднику технической поддержки.

## 5.2 Руководство программиста

Специальное веб-приложение «FixPhoto» входит в состав системы «FixPhoto» и обеспечивает полноценную работу менеджера, имеет 6 страниц, которые описаны в таблице 8.

Таблица 8 – Страницы приложения

Название	Описание
AuthorizationPage	Страница авторизации, на которой располагаются поля ввода, а так же кнопка «Войти»
ObjectsPage	Страница с объектами, на которой расположен список объектов, кнопка «Создать объект», строка поиска, а так же кнопки в виде изображений «Редактировать» и «Удалить».
TasksPage	Страница с задачами, на которой расположен список задач, кнопка «Создать задачу», а так же кнопки в виде изображений «Редактировать» и «Удалить».
ObjectsPage	Страница с фотографиями, на которой расположен выпадающий список с объектами, выпадающий список с задачами, кнопка «Показать», список изображений с возможность отредактировать ориентацию и время создания фотографии.
ReportsPage	Страница с отчетами, на которой расположен список с отчетами, строка поиска, кнопки в виде изображений «Удалить» или «Установить» выбранный отчет.

Продолжение таблицы 8

Название	Описание
PerformersPage	Страница с исполнителями, на которой расположен список исполнителей, кнопка «Создать исполнителя», строка поиска, а так же кнопки в виде изображений «Редактировать» и «Удалить».

Для локальной разработки необходимо установить NodeJS и npm, потому что ReactJS работает с помощью них.

В ходе разработки приложения вы можете получать различные ошибки в консоли редакторе кода, которые подробно будут описаны. Помимо них, можно получить ошибки, связанные с запросами к API в приложении. Чтобы посмотреть ошибку нужно нажать правую кнопку мыши по любой части страницы и выбрать пункт «Посмотреть код», после чего откроется консоль разработчика с возможностью открыть пункт «Консоль» или «Сеть». В пункте «Консоль» можно посмотреть все ошибки, которые будет выдавать программа в ходе работы. В пункте «Сеть» отображены все запросы с их параметрами и результат возвращаемого ответа.



## Заключение

В результате выполнения выпускной квалификационной работы было разработана подсистема «Управления задачами», представленная в виде веб-приложения «FixPhoto» с помощью средств разработки языка программирования JavaScript и фреймворка ReactJS.

Веб-приложение соответствует требованиям заказчика и обладает следующими функциональными возможностями:

- создание и редактирование объектов;
- создание и редактирование задач;
- просмотр фотографий по выбранному объекту и задаче с возможностью отредактировать ориентацию;
- формирование отчетов;
- удаление отчета;
- добавление новых исполнителей или редактирование их.

Разработанную подсистема «Управления задачами», представленную в виде веб-приложения можно рассмотреть как удобное средство для автоматизации работы менеджера.

## Список использованных источников

1. ГОСТ 19.201-78. Единая система программной документации (ЕСПД). Техническое задание. Требования к содержанию и оформлению (с Изменением N 1) = Unified system for program documentation. Technical specifications for development. Requirements to contents and form of presentation: межгосударственный стандарт: издание официальное: утвержден и введен в действие Постановлением Государственного комитета СССР по стандартам от 18 декабря 1978 г. № 3351: введен впервые: дата введения 1980-01-01. – Москва: Стандартинформ, 2010. – 4 с. – Текст непосредственный.

2. РД 50-34.698-90. Методические указания. Информационная технология. Комплекс стандартов и руководящих документов на автоматизированные системы. Автоматизированные системы. Требования к содержанию документов: руководящий документ по стандартизации: издание официальное: утверждены и введены в действие Постановлением Государственного комитета СССР по управлению качеством продукции и стандартами от 27 декабря 1990 г. № 3380: дата введения 1992-01-01 / Разработан Министерством электротехнической промышленности и приборостроения СССР. – Москва.: ИПК Издательство стандартов, 2002 г. – 27 с. – Текст непосредственный.

3. Онлайн-школа IT профессий и сообщество программистов. [сайт] – Текст: электронный. – URL: <https://itproger.com/> (дата обращения 19.05.2022).

4. Сообщество IT-специалистов [сайт] – Текст: электронный. – URL: <https://habr.com/> (дата обращения 07.04.2022).

5. UML-диаграммы классов. Программирование. [сайт] – Текст: электронный. – URL: <https://prog-cpp.ru/uml-classes> (дата обращения 17.04.2022).

6. JavaScript-библиотека для создания пользовательских интерфейсов (ReactJS). [сайт] – Текст: электронный. – URL: <https://ru.reactjs.org/> (дата обращения 10.04.2022).

7. JavaScript-библиотека для создания пользовательских модальных окон (react-popup). [сайт] – Текст: электронный. – URL: <https://www.npmjs.com/package/reactjs-popup> (дата обращения 19.05.2022).

8. JavaScript-библиотека для форматирования даты и времени (dateformat). [сайт] – Текст: электронный. – URL: <https://www.npmjs.com/package/dateformat> (дата обращения 22.05.2022).

9. JavaScript-библиотека для создания TimePicker и DatePicker (material-ui/pickers). [сайт] – Текст: электронный. – URL: <https://material-ui-pickers.dev/demo/datetime-picker> (дата обращения 22.05.2022).

10. Простая утилита JavaScript для условного соединения имен классов (classnames). [сайт] – Текст: электронный. – URL: <https://www.npmjs.com/package/classnames> (дата обращения 13.05.2022).

11. Официальные привязки React для Redux (react-redux). [сайт] – Текст: электронный. – URL: <https://react-redux.js.org/> (дата обращения 14.05.2022).

Приложение А  
(обязательное)

Техническое задание. Требования к программным модулям

1 Введение

Настоящее техническое задание распространяется на разработку веб-приложения для системы «FixPhoto», предназначенной для автоматизации работы менеджеров. Использовать данное веб-приложение будут менеджеры организации ООО «Зодиак-Электро».

Веб-приложение FixPhoto предоставляет графический интерфейс для сбора, хранения, обработки и отправки информации на сервер об объектах, задачах, фотографий, исполнителях и позволяет формировать отчеты.

Подобная автоматизация повышает эффективность управленческой деятельности и позволяет создать единую систему, состоящую из веб-приложения и мобильного приложения, работающую как слаженный механизм, своевременно анализировать процессы работы организации и повысить качество и скорость выполнения работ по монтажу.

					ККЭП 09.02.07 0104 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		37

## 2 Основания для разработки

Основанием для разработки является заказ системы FixPhoto для автоматизации работы сбора фотографий, их обработка и формирование отчетов для компании ООО «Зодиак-Электро» и согласовано с директором Блиновым Е.А.

Наименование темы разработки – «Разработка модулей подсистемы «Управление задачами» информационной системы «FixPhoto» ООО «Зодиак-Электро»».

					<i>ККЭП 09.02.07 0104 ПЗ</i>	Лист
Изм.	Лист	№ докум.	Подпись	Дата		38

### 3 Назначение разработки

Система должна автоматизировать следующие задачи:

- вести учет объектов;
- вести учет задач, назначенных на монтажников;
- вести учет хода выполнения монтажных работ;
- формировать отчеты о выполненной работе;
- вести учет исполнителей.

					<i>ККЭП 09.02.07 0104 ПЗ</i>	Лист
Изм.	Лист	№ докум.	Подпись	Дата		39

## 4 Требования к программе или программным модулям

### 4.1 Требования к функциональным характеристикам

После запуска программы пользователю отображается окно авторизации с формой для ввода логина и пароля. При авторизации приложение получает токен, действующий две недели, пока действует токен, приложение не должно запрашивать логин и пароль у пользователя.

Для менеджера программа предоставляет следующие возможности:

- просмотр всех объектов, редактирование, добавление и удаление;
- просмотр задач на выбранный объект, а также создание новой задачи, редактирование и удаление;
- просмотр фотографий по выполненной задаче;
- просмотр исполнителей, возможность добавления новых, а так же редактирование и удаление текущих;
- формирование отчетов.

### 4.2 Требования к надежности и безопасности

Приложение должно соответствовать современному уровню требований к надежности программного обеспечения:

- предусматривать контроль вводимой информации и блокировку некорректных действий пользователя при работе с веб-приложением;
- обеспечивать корректную работу при отправке больших объемов информации;

- токен пользователя необходимо хранить в зашифрованном виде, для обеспечения безопасности всей системы;
- обрабатывать все ошибки, которые были получены с сервера и сообщать о них пользователю;
- обрабатывать все исключения для корректной работы приложения.

#### 4.3 Условия эксплуатации

Приложение запускается на любом цифровом устройстве. Устройство должно обладать устойчивым интернет-соединением для оперативной работы.

Специальные климатические условия не требуются.

Приложение не требует проведения каких-либо видов обслуживания.

#### 4.4 Требования к составу и параметрам технических средств

Требования к компьютеру выполняющего роль сервера представлены в таблице А.1.

Таблица А.1 – Требования к персональному компьютеру

Компонент	Требование
Жесткий диск	Минимум 10 ГБ свободного места
Монитор	Требуется монитор с разрешением 1280x720 или более высоким



Продолжение таблицы А.1

Оперативная память	Не менее 8 ГБ
Быстродействие процессора	Минимум: процессор x64 с тактовой частотой 1,4 ГГц
Операционная система	Windows 10

## Приложение Б

(обязательное)

### Программный код

Программный код представлен в контексте текстового редактора Visual Studio Code.

```
const authorizationHandler = () => {  
  request('auth', 'POST', { login, password }).then((result) => {  
    if (result.token !== undefined) {  
      window.location.assign('/projects');  
      localStorage.setItem('token', result.token);  
    }  
  });  
};
```

Рисунок Б.1 – Код обработчика кнопки «Продолжить»

```
<button className='btn create-post' onClick={() => setOpen((e) => !e)}>  
  Создать объект  
</button>
```

Рисунок Б.2 – Код обработчика кнопки «Создать объект»

```

const saveHandler = async () => {
  if (
    name.length > 0 &&
    address.length > 0 &&
    code.length > 0 &&
    contract.length > 0 &&
    count > 0
  ) {
    const attributes = {};
    request('facility', 'POST', { address, attributes, code, contract, count, name }).then(() =>
      dispatch(dataFiltersObjects()),
    );
    setName('');
    setAddress('');
    setCode('');
    setContract('');
    setCount(1);
    setOpen(false);

    setModalText('Объект сохранен');
    setError('');
    setModalActive(true);
    setTimeout(() => {
      setModalActive(false);
    }, 1500);
  } else {
    setModalText('Ошибка, заполнены не все поля');
    setError('error');
    setModalActive(true);
    setTimeout(() => {
      setModalActive(false);
    }, 1500);
  }
};

```

Рисунок Б.3 – Код обработчика кнопки «Сохранить» для создания объекта

```

<div className="tasks__top-right">
  <Popup
    trigger={<button className="btn create-post">Создать задачу</button>}
    modal
    nested
  >

```

Рисунок Б.4 – Код обработчика «Создать задачу»

```

const saveHandler = (close) => {
  const currentDate = dateFormat(executionDate, 'yyyy-mm-dd');
  const currentTime = dateFormat(executionTime, 'HH:MM');
  const editDate = currentDate + 'T' + currentTime + ':00.000Z';
  if (!/[a-za-яё]/i.test(photoCount) && !/[a-za-яё]/i.test(executionHours)) {
    const endTime = null;
    const gallery = true;
    const repeatUnit = null;
    const repeatable = false;
    const watermarks = ['FACILITY', 'DATE', 'TIME'];
    request('tasks', 'POST', {
      description,
      endTime,
      executionDate: currentDate,
      executionHours,
      executionTime: editDate,
      facilityId,
      gallery,
      performerId,
      photoCount,
      repeatUnit,
      repeatable,
      watermarks,
    }).then(() => {
      if (currentObject !== undefined)
        request(`tasks/facility/${currentObject.id}`).then((data) => setTasks(data));
    });
    setDescription('');
    setExecutionDate(null);
    setExecutionHours(12);
    setExecutionTime(null);
    setFacilityId();
    setPerformerId();
    setPhotoCount(1);
    setObject('');
    setPerformer('');

    setModalText('Объект сохранен');
    setError('');
    setModalActive(true);
    setTimeout(() => {
      setModalActive(false);
    }, 1500);
    close();
  } else {
    setModalText('Ошибка, введены не корректные значения');
    setError('error');
  }
}

```

Рисунок Б.5 – Код обработчика кнопки «Сохранить» для создания задачи

```

{currentPhotos.map((item) =>
  item.photos.map((src, index) => (
    <div className='images__list-item' key={index}>
      <CustomPopup
        trigger={
          <div>
            <LazyLoad height={200}>
              <img
                className='images__list-img'
                src={`http://fixphoto.zodiak-elektro.ru/api/photos/${src}`}
                alt='task'
              ></img>
            </LazyLoad>
          </div>
        }
        openHandle={openPopupHandler}
      >

```

Рисунок Б.6 – Код обработчика при нажатии на фотографию

```

const downloadHandler = (id, name) => {
  const url = `http://fixphoto.zodiak-elektro.ru/api/reports/${id}`;
  const token = localStorage.getItem('token');
  fetch(url, {
    method: 'GET',
    headers: {
      Authorization: `Bearer ${token}`,
    },
  })
    .then(function (response) {
      return response.blob();
    })
    .then(function (blob) {
      saveAs(blob, name);
    })
    .catch((error) => {
      console.log(error);
    });
};

```

Рисунок Б.7 – Код обработчика при нажатии на кнопку скачивания отчета

```

<div className='projects__top-right'>
  <button className='btn create-post' onClick={() => setOpen((e) => !e)}>
    Создать исполнителя
  </button>

```

Рисунок Б.8 – Код обработчика при нажатии на кнопку «Создать исполнителя»

Приложение В  
(обязательное)

Тест-кейсы

Название проекта	FixPhoto
Номер версии	1
Имя тестировщика	Арьков Александр
Дата тестирования	13.05.2022

Рисунок В.1 – Общая информация о тестировании

Test Case #	TC_PostObject_1.
Приоритет теста	Средний.
Название тестирования/Имя	Проверка работоспособности запроса создания нового объекта
Резюме испытания	При отправлении запроса сервер должен проверить токен на актуальность
Шаги тестирования	Отправить запрос создания объекта
Данные тестирования	Адрес, код, контракт, количество, имя, действующий токен
Ожидаемый результат	Статус ответа: 200
Фактический результат	Статус ответа: 200
Предпосылки	Отсутствуют.
Постусловия	Сервер после выполнения запроса должен вернуть статус: 200
Статус (Pass/Fail)	Pass.
Комментарии	-

Рисунок В.2 – Проверка запроса создания объекта. Блок 1

<b>Test Case #</b>	TC_PostObject_2.
<b>Приоритет теста</b>	Средний.
<b>Название тестирования/Имя</b>	Проверка работоспособности запроса создания нового объекта
<b>Резюме испытания</b>	При отправлении запроса сервер должен проверить токен на актуальность
<b>Шаги тестирования</b>	Отправить запрос создания объекта
<b>Данные тестирования</b>	Адрес, код, контракт, количество, имя, устаревший токен
<b>Ожидаемый результат</b>	Статус ответа: 403 Сообщение: «Forbidden»
<b>Фактический результат</b>	Статус ответа: 403 Сообщение: «Forbidden»
<b>Предпосылки</b>	Отсутствуют.
<b>Постусловия</b>	Сервер после выполнения запроса должен вернуть статус: 200 и сообщение: «Forbidden»
<b>Статус (Pass/Fail)</b>	Pass.
<b>Комментарии</b>	-

Рисунок В.3 – Проверка запроса создания объекта. Блок 2

## Приложение Г (обязательное)

### Руководство пользователя

Для того, чтобы войти в приложение администратор компании вам должен выдать логин и пароль для входа.

После входа открывается страница с объектами, где есть возможность добавить новый объект, изменить или удалить существующий. В верхней части каждой страницы, кроме авторизации находится навигация. Если вы открыли приложение на мобильном устройстве, то навигация представлена в виде меню, представленное на рисунке Г.1. Мобильное меню навигации открывается и закрывается при нажатии на него.



Рисунок Г.1 – Мобильное меню навигации

При нажатии на любую ссылку открывается страница, содержащая контент, который соответствует ее названию.



Приложение Д  
(обязательное)

Руководство программиста

Для разработки необходимо установить NodeJS, а так же установить npm глобально. Для того, чтобы запустить сборку или начать локальную работу приложения необходимо запустить консоль. В консоли нужно перейти в корень проекта и для запуска написать команду «npm start». Чтобы собрать приложение нужно написать команду «npm run build». Для того, чтобы изменить конфигурацию сборки нужно перейти в директорию config и изменить конфигурационные файлы. Менять конфигурационные файлы нужно очень осторожно, потому что при одной ошибке может собраться не правильно приложение или можно повредить команды запуска приложения. После сборки приложения появится папка build с обновленными минимизированными файлами проекта. Для запуска сборки приложения необходимо написать команду «npm start build».

					ККЭП 09.02.07 0104 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		50