

Министерство образования Иркутской области
Государственное бюджетное профессиональное
образовательное учреждение Иркутской области
«Иркутский авиационный техникум»
(ГБПОУИО «ИАТ»)

ДП.09.02.03.24.201.08.ПЗ

УТВЕРЖДАЮ

Зам. Директора по УР, к.т.н.

_____ Е.А. Коробкова

ИНФОРМАЦИОННАЯ СИСТЕМА
«СКЛАД»

Нормоконтролер:	_____	(Е.С. Кудрявцева)
	(подпись, дата)	
Руководитель:	_____	(А.С. Некипелова)
	(подпись, дата)	
Студент:	_____	(Д.В. Козлов)
	(подпись, дата)	

Иркутск 2024

СОДЕРЖАНИЕ

Введение.....	3
1 Предпроектное исследование	4
1.1 Описание предметной области ИС.....	4
1.2 Анализ инструментов, используемых в разработке программного продукта	6
1.3 Архитектура программного продукта.....	18
2 Техническое задание.....	20
3 Проектирование ИС	21
3.1 Структурная схема ИС	21
3.2 Функциональная схема ИС	23
3.3 Проектирование базы данных.....	26
3.4 Проектирование интерфейса.....	29
4 Разработка ИС	31
4.1 Разработка интерфейса ИС	31
4.2 Разработка базы данных ИС	36
4.3 Разработка ИС	38
4.4 Тестирование ИС.....	41
5 Обучающая документация для пользователей информационной системы в соответствии с ГОСТ Р 59795–2021	48
5.1 Подготовка к работе.....	48
5.2 Описание операций	48
Заключение	50
Список используемых материалов	51

Введение

Склады — это здания, сооружения и разнообразные устройства, предназначенные для приемки, размещения и хранения поступивших на них товаров, подготовки их к потреблению и отпуску потребителю.

Склады являются одним из важнейших элементов логистических систем. Объективная необходимость в специально обустроенных местах для содержания запасов существует на всех стадиях движения материального потока, начиная от первичного источника сырья и кончая конечным потребителем. Этим объясняется наличие большого количества разнообразных видов складов.

Для более удобного и быстрого поиска товаров, необходимо создать информационную систему, с помощью которой можно будет ускорить и облегчить работу на складе.

Целью дипломной работы является создание информационной системы «Склад».

Для достижения конечного результата, а именно создание информационной системы «Склад», необходимо решить следующие задачи:

- разработать техническое задание;
- проанализировать инструментальные средства разработки;
- спроектировать базу данных;
- спроектировать информационную систему;
- спроектировать интерфейс;
- разработать базу данных;
- разработать информационную систему;
- разработать интерфейс;
- продемонстрировать программный продукт;
- провести тестирование информационной системы;
- создать руководство пользователя по информационной системе.

1 Предпроектное исследование

1.1 Описание предметной области ИС

Выбранной областью является "Склад", который предназначен для хранения продукции от сторонних лиц. Склад включает помещения для хранения товаров, пути для подъезда, зоны погрузки–выгрузки и другие сооружения. Этот комплекс спроектирован для конкретных бизнес–потребностей, обеспечивая эффективное использование пространства, улучшая трудовой процесс и обеспечивая учет продукции. Он также может быть легко пересмотрен или расширен.

На складе проводится инвентаризация, включающая этапы, такие как пересмотр продукции, ответственным сотрудником, который вручную записывает данные о товарах и их количестве, сопровождая книгу переучета. Затем эти данные сверяются с книгой учета товаров сотрудниками, отвечающими за отчетную документацию на складе, и составляется соответствующий отчет по результатам переучета продукции.

Информационная система предназначена для сотрудников и облегчает работу с товарами, позволяя добавлять, изменять и удалять информацию о них.

Таким образом, в функционирование ИС входит:

- авторизация, регистрация;
- разграничение прав доступа;
- манипулирование заказами;
- манипулирование сотрудниками.

Основные понятия предметной области:

- сотрудник;
- товар;
- заказы.

Требования к информационной системе в предметной области «Склад»:

- автоматизация основных процессов работы склада;
- обеспечение безопасности конфиденциальных данных;
- удобный и интуитивно понятный интерфейс для участников предметной области.

Информационная система «Склад» должна позволять:

- вести учет товаров на складе;
- выводить списки всех имеющихся товаров на складе;
- давать полную информацию о товарах;
- фильтровать данные товаров по характеристикам.

Для обслуживания процессов склада создается база данных, которая содержит следующие данные:

1. сведения о товарах;
2. сведения о сотрудниках;
3. сведения о заказчиках.

База данных строится с учетом следующих особенностей:

1. сотрудники могут редактировать/добавлять/удалять информацию о товарах;
2. сотрудники и администратор могут просматривать информацию о заказчиках;
3. администратор может удалять/добавлять/редактировать данные о сотрудниках.

1.2 Анализ инструментов, используемых в разработке программного продукта

Инструменты, правильно выбранные для разработки программы, играют ключевую роль в ее итоговом качестве.

Структурировать информационную систему можно с помощью phpMyAdmin и Draw.io, тогда как для дизайна отлично подходит Figma. Планируемая система будет серверного типа, предназначенная для сотрудников. Для реализации будут использованы технологии HTML5, CSS3, JS, в том числе AJAX.

phpMyAdmin – это веб–приложение на PHP, предоставляющее пользовательский интерфейс для управления базами данных MySQL. С его помощью можно управлять сервером MySQL, выполнять SQL–запросы и просматривать содержание таблиц через веб–браузер. В данном проекте оно применяется для построения ER–диаграмм.

Draw.io – бесплатное приложение для создания диаграмм, основанное на HTML5 и JavaScript. Его можно использовать для разработки различных схем, включая блок–диаграммы, макеты, UML и другие. В этом проекте оно служит для разработки прототипов страниц.

Figma – это инструмент для дизайна и прототипирования веб–сайтов и приложений. Он позволяет нескольким пользователям работать над одним проектом одновременно, предоставляя доступ к редактированию или комментированию. В Figma создаются прототипы, иллюстрации, интерфейсы и даже макеты для платформы Tilda. В этом проекте он служит для визуального оформления.

HTML – это язык разметки, который предоставляет браузеру инструкции по отображению содержимого страницы. Браузеры интерпретируют HTML–код и отображают его в виде веб–страницы, предназначенной для восприятия пользователем. CSS – каскадные таблицы стилей, которые используются для определения стилей (правил) оформления

документов — включая дизайн, вёрстку и вариации макета для различных устройств и размеров экрана.

JavaScript представляет собой универсальный язык программирования, который часто используется для внедрения в приложения с целью управления различными их аспектами. В контексте веб-разработки, знание JavaScript является неотъемлемым для создания современных интерактивных веб-сайтов. Этот язык придает жизнь структуре HTML-страниц и обеспечивает функциональность пользовательского интерфейса веб-приложений. JavaScript позволяет страницам и их элементам реагировать на действия пользователей. В настоящее время JavaScript является основным языком программирования для веб-браузеров и полностью совместим с различными операционными системами, включая Windows, Linux, Mac OS, а также мобильными платформами.

Были рассмотрены следующие варианты реализации СУБД:

1. MySQL;
2. SQLite;
3. PostgreSQL.

MySQL – одна из наиболее используемых систем управления базами данных. MySQL управляет реляционными базами данных, то есть такими, в которых таблицы связаны между собой. MySQL работает по принципу клиент-сервер. Компьютер пользователя (клиент) отправляет запрос. Сервер баз данных его обрабатывает и предоставляет ответ. Именно поэтому часто можно услышать понятие MySQL-сервер. Это сервер, на котором хранится база данных. Система MySQL написана на языках программирования C и C++. Для работы MySQL используется язык структурированных запросов SQL.

SQLite — компактная встраиваемая СУБД с исходным кодом. Возможные типы значений: INTEGER, REAL, TEXT и BLOB. Также поддерживается специальное значение NULL. Размеры значений типа TEXT и BLOB не ограничены ничем, кроме константы SQLITE_MAX_LENGTH в исходниках sqlite, равной миллиарду.

SQLite напрямую хранит информацию в одном файле, что облегчает его копирование. Большая популярность в мобильной разработке и небольших автономных приложениях, поскольку она занимает меньше места на дисковом пространстве, имеет высокую скорость работы и не требует в отличие от MySQL наличие сервера для запуска. Минусы: ограничения на запись, всего 5 типов данных, отсутствие встроенного механизма аутентификации.

PostgreSQL – это объектно–реляционная система управления базами данных (ORDBMS), наиболее развитая из открытых СУБД в мире. Имеет открытый исходный код и является альтернативой коммерческим базам данных. СУБД позволяет гибко управлять базами данных. С ее помощью можно создавать, модифицировать или удалять записи, отправлять транзакцию – набор из нескольких последовательных запросов на особом языке запросов SQL.

Для наглядности сравнения вариантов реализации базы данных была составлена таблица 1.

Таблица 1 – Сравнение средств реализации базы данных

Название СУБД	MySQL	SQLite	PostgreSQL
Большое кол–во типов данных	+	-	+
Популярность	+	+	–
Отказоустойчивость	–	-	+
Не требует удаленного сервера	–	=	–
Простота использования	+	+	–
Портативность	–	+	–

Таким образом, в качестве базы данных для будущего продукта была выбрана MySQL, так как она предоставляет весь необходимый функционал для разработки продукта – скорость, возможность использовать в облаке, простота использования, ведь установка не требует особых навыков, а для еще

более легкой работы можно использовать дополнительное GUI. Большинство функций для настройки безопасности поддерживаются по умолчанию. MySQL имеет богатый функционал и предлагает бесплатную лицензию для работы с открытым кодом.

Для взаимосвязи баз данных и серверной части продукта необходимо использовать серверный язык. Для реализации этого были рассмотрены два языка программирования – Python и Php.

Python – это активно развивающийся скриптовый язык, который используют для решения большого объема самых разноплановых проблем и задач. Python пригодится в создании компьютерных и мобильных приложений, его применяют в работе с большим объемом информации, при разработке web-сайтов и других разнообразных проектов, используют в машинном обучении. Данный язык программирования используют крупные известные корпорации, такие как Spotify и Амазон (например, для анализа данных и создания алгоритма рекомендаций), YouTube и даже Walt Disney. Таким образом, Python нашел свое место в различных областях — с его помощью можно решить множество задач разной сложности. Синтаксис ядра языка минималистичен, за счёт чего на практике редко возникает необходимость обращаться к документации. Сам же язык известен как интерпретируемый и используется в том числе для написания скриптов. Недостатками языка являются зачастую более низкая скорость работы и более высокое потребление памяти написанных на нём программ по сравнению с аналогичным кодом, написанным на компилируемых языках, таких как C или C++. Недостатками языка являются использование языка в серверной разработке, только благодаря фреймворку, а также зачастую более низкая скорость работы и более высокое потребление памяти написанных на нём программ по сравнению с аналогичным кодом, написанным на компилируемых языках, таких как C или C++. Основные архитектурные черты – динамическая типизация, автоматическое управление памятью, полная интроспекция, механизм обработки исключений, поддержка многопоточных

вычислений, высокоуровневые структуры данных. Поддерживается разбиение программ на модули, которые, в свою очередь, могут объединяться в пакеты.

PHP – это распространённый язык программирования общего назначения с открытым исходным кодом. PHP специально сконструирован для веб-разработок и его код может внедряться непосредственно в HTML.

Язык PHP обладает рядом неоспоримых преимуществ:

- Высокая скорость работы и, соответственно, общая производительность ресурсов.
- Простота освоения, простой синтаксис.
- Отличная совместимость и переносимость – php-коды работают одинаково хорошо с разными платформами.
- Набор текста кода и его редактирование можно осуществлять в любом текстовом или html-редакторе.

Для наглядности сравнения языков программирования была составлена таблица 2.

Таблица 2 – Сравнение языков программирования для разработки программного продукта

Название языка программирования	Php	Python
Наличие библиотек	+	+
Инструменты для работы с БД	+	+
Объектно-ориентированные возможности	+	+
Лёгкий понятный синтаксис	+	–
Более активное сообщество	+	–
Более лёгкая простая модульность	+	–

Таким образом, php будет более лучшим вариантом, ведь он более компактный и простой в освоении язык. Он вобрал все лучшие особенности таких популярных языков, как C, Java и Perl, а также его сообщество более активное.

Для разработки программного продукта рассмотрены следующие инструментальные средства разработки программных продуктов:

- Visual Studio Code
- Netbeans.
- Atom.

Visual Studio Code – это продукт от компании Майкрософт. Инструмент, предназначенный для верстальщиков и разработчиков. Один из самых популярных в соответствующей нише.

Представляет собой редактор кода от Microsoft, выступающий «облегченной» интерпретацией VisualStudio. С помощью него можно не только заниматься написанием приложений. Visual Studio Code поддерживает большое количество плагинов, которые позволят «разогнать» редактор до полноценной среды программирования.

Подходит для работы на операционных системах:

- Windows;
- MacOS;
- Linux.

Распространяется на бесплатной основе, благодаря чему набрал огромную популярность. Работает одинаково хорошо как на стареньких устройствах, так и на современных компьютерах.

Visual Studio Code – это редактор, который больше подходит новичкам за счет своей первоначальной «облегченности». Данный продукт может использоваться и продвинутыми разработчиками при создании достаточно сложных кодов. К преимуществам Visual Studio Code относят следующие моменты:

- Простое освоение. Редактор оснащен интуитивно понятным интерфейсом, а также возможностью настройки.
- Небольшой вес.
- Кроссплатформенность.

- Наличие online версии. В случае ее использования для написания программного кода не придется вообще ничего устанавливать. Visual Studio Online работает через браузер.
- Поддержка разных языков. Опция активируется за счет установки плагинов.
- Гибкость.

NetBeans IDE — свободная интегрированная среда разработки приложений (IDE) на языках программирования Java, Python, PHP, JavaScript, C, C++, Ада и ряда других. Проект NetBeans IDE поддерживается и спонсируется компанией Oracle, однако разработка NetBeans ведётся независимым сообществом разработчиков–энтузиастов (NetBeans Community) и компанией NetBeans Org. Последние версии NetBeans IDE поддерживают рефакторинг, профилирование, выделение синтаксических конструкций цветом, автодополнение набираемых конструкций на лету и множество предопределённых шаблонов кода.

Atom (в прошлом Atomicity) — бесплатный текстовый редактор с открытым исходным кодом для Linux, macOS, Windows с поддержкой плагинов, написанных на JavaScript, и встраиваемых под управлением Git. Большинство плагинов имеют статус свободного программного обеспечения, разрабатываются и поддерживаются сообществом.

Сравнение IDE для разработки программного продукта наглядно представлено в таблице 3.

					КП.09.02.07-5.23.201.08. ПЗ	Лист 12
Изм.	Лист	№ докум.	Подпись	Дата		

Таблица 3 – Сравнение IDE для разработки программного продукта

Название IDE	Visual Studio Code	NetBeans	Atom
Распространяется бесплатно	+	+	+
Автоматическое сохранение	+	–	+
Подсказки по коду	+	+	+
Интеграция с системой контроля версия (GIT)	+	+	–
Возможность расширения функционала библиотеками	+	+	+
Заточен под PHP-разработку	+	+	+
Поддержка CSS/HTML/JS	+	+	+
Комфортное использование на слабых ПК	+	–	+

Таким образом, после рассмотрения вариантов средств разработок, было принято решение использовать Visual Studio Code. Visual Studio Code намного функциональнее, чем приведённые выше Atom и NetBeans. В Visual Studio Code имеется автоматическое сохранение, которое не позволит случайно потерять все наработки, а также он более удобен в работе.

Для упрощения и ускорения написания кода, а также обеспечения большей безопасности системы, необходимо использование фреймворков. Для дальнейшей разработки были рассмотрены следующие варианты:

1. Laravel.
2. Yii.
3. CodeIgniter.
4. Symfony.
5. CakePHP.

Laravel – это бесплатный PHP-фреймворк с открытым исходным кодом. Laravel разработали в качестве помощника при создании сложных веб-ресурсов и приложений. С его помощью специалисты упрощают процесс аутентификации, а также работу с БД, кэширование, сессии, структуру приложения, маршрутизацию и другие не менее важные процессы.

Возможностей у платформы Laravel немало. Одна из них – построение логичной архитектуры для проектов любой сложности и типа. Платформа характеризуется:

- высокой производительностью;
- возможностью интеграции с другими платформами, а также библиотеками;
- немалым количеством интересных возможностей для разработчиков сайтов и приложений.

Yii - высокопроизводительный компонентно-ориентированный PHP фреймворк наилучшим образом подходящий для масштабируемых веб-приложений. Yii появился с широким набором возможностей, включая MVC, DAO/ActiveRecord, I18N/L10N, поддержку AJAX на основе jQuery, управление доступом на основе ролей, генерация рутинного кода (scaffolding), проверку ввода, виджеты, события, темы оформления, веб-сервисы и еще много чего. Написанный на чистом ООП, Yii является простым в использовании, а также чрезвычайно гибким и расширяемым.

CodeIgniter — это платформа PHP MVC, используемая для быстрой разработки веб-приложений. CodeIgniter предоставляет box библиотеки для подключения к базе данных и выполнения различных операций, таких как отправка электронных писем mails, загрузка файлов, управление сеансами и т. д.

Весь исходный код платформы CodeIgniter занимает около 2 МБ. Это позволяет легко освоить CodeIgniter и понять, как он работает. Это также упрощает его развертывание и обновление.

Пользователи склонны отдавать предпочтение приложениям, которые загружаются очень быстро. Если вы работали с некоторыми современными фреймворками, то вы понимаете, что их загрузка сразу после установки занимает менее одной секунды.

Symfony – это фреймворк для бэкэнд-разработки, который работает на базе языка программирования PHP. Данный фреймворк используют в разработке серверной части сложных веб-сервисов с большим количеством посетителей и уникальным функционалом. Также Symfony может работать как микрофреймворк, что позволяет использовать его частично, не устанавливая и не настраивая полностью.

Разработчики фреймворка представляют Symfony как самый быстрый фреймворк, написанный на PHP. Чтобы эффективно оптимизировать Symfony, нужно обладать большим опытом в работе с этим фреймворком и разбираться в его многочисленных опциях. К примеру, PHP-фреймворк Phalcon предлагает лучшую производительность без дополнительной отладки, поскольку написан на языке C.

SakePHP – это бесплатный фреймворк для веб-разработки с открытым исходным кодом, написанный на языке программирования PHP. PHP-фреймворк, реализующий паттерн MVC. Унаследовал многие особенности Ruby on Rails, включая поддержку разных СУБД, плагинов, абстракцию данных и файловую структуру. У фреймворка большое международное комьюнити, которое занимается доработкой продукта. SakePHP регулярно обновляется, обрастая все новыми функциями.

					КП.09.02.07-5.23.201.08. ПЗ	Лист 15
Изм.	Лист	№ докум.	Подпись	Дата		

Сравнение фреймворков для разработки программного продукта наглядно представлено в таблице 4.

Таблица 4 – Сравнение фреймворков для разработки программного продукта

Название фреймворка	Laravel	Yii	CodeIgniter	Symfony	CakePHP
ORM	ActiveRecord	Doctrine 2, Propel (active record)	ELOQUENT ORM (active record)	Custom	Database Access Objects (DAO), Active Record (AR)
Библиотека тестирования	PHPUnit (In development)	PHPUnit	PHPUnit	PHPUnit	PHPUnit, Selenium
Веб2.0	jQuery HTML5boilerplate			Полный jQuery, пользовательский интерфейс jQuery, система Grid, собственный AJAX, RestFul	встроенный jQuery, расширяемый до любого javascript-фреймворка
Система шаблонов	PHP, Simple template parser "{ var_name }"	PHP, Twig	Blade, PHP, Custom	Custom but Smarty/Twig can be used	PHP and Prado's - Several others using Extensions (Razor, Smarty, Twig, etc)

Таким образом было решено использовать фреймворк Laravel.

Для создания программного продукта было решено использовать средства:

1) Для создания структурных схем, контекстной и диаграмм декомпозиции использовались CASE–средства – Draw.io.

2) Для наглядного составления структуры базы данных использовался инструмент для визуального проектирования баз данных, их редактирования и полного администрирования – MySQL Workbench.

3) Для разработки дизайна web–приложения использовался онлайн–сервис для разработки дизайна и прототипа сайта – Figma.

4) На этапе разработки программного продукта использовались инструменты и среды для разработки: PHP фреймворк Laravel, CSS, а также редактор кода Visual Studio Code.

Основные преимущества веб–технологий:

1) Распределенность (пользователь может работать с системой из любого места, связанного с WEB–сервером по сети, находясь в любой точке земного шара);

2) Переносимость (Web–клиенты (браузеры) существуют для любых платформ, от настольных компьютеров до сотовых телефонов. Web–сервера используются для большинства платформ, а Web–приложения обычно пишутся на переносимых языках);

3) Привычность интерфейса (почти каждый пользователь компьютера хотя бы раз запускал браузер и работал в нем);

4) Простота установки и обслуживания (новую версию web–приложения не надо устанавливать на все компьютеры – достаточно установить на сервер);

5) Простота интерфейса – Пользователи не любят гигантских окошек с сотнями полей ввода, а программисты и разработчики интерфейсов почему–то любят. Web как раз не поощряет сложный интерфейс, скорее поощряет простой.

1.3 Архитектура программного продукта

Архитектура программного обеспечения - описание структуры программной системы, включающее программные компоненты, их свойства и отношения между ними. Правильно выбранная архитектура помогает создать успешный программный продукт, который в дальнейшем может легко адаптироваться к изменяющимся требованиям заказчика.

Для дипломного проекта была выбрана клиент-серверная архитектура.

Клиент-серверная архитектура — это модель организации вычислительных систем, в которой задачи распределены между клиентами и серверами. В такой архитектуре клиент, обычно являющийся пользователем или программой, запрашивает услуги или ресурсы у сервера, который отвечает на запросы, предоставляя необходимые данные или функциональность.

На рисунке 1 представлена клиент-серверная архитектура

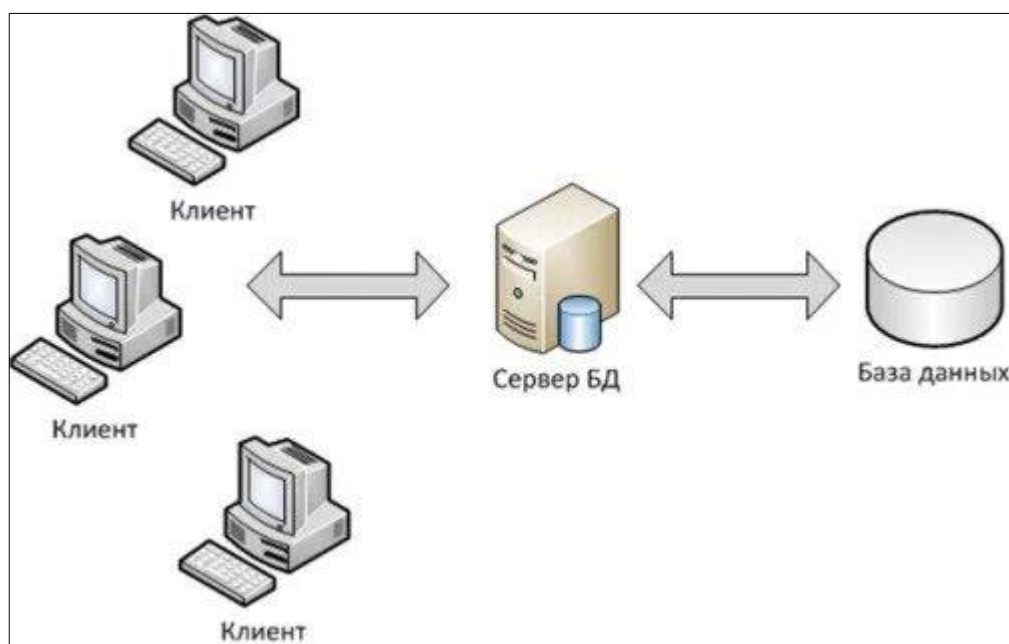


Рисунок 1 – Схема клиент-серверной архитектуры

Клиент-серверная архитектура является распространенной моделью для построения сетевых приложений. Она состоит из двух основных компонентов: клиента и сервера, которые взаимодействуют друг с другом посредством сетевого соединения.

Основные особенности клиент-серверной архитектуры:

Разделение функций: как уже упоминалось выше, в клиент-серверной модели клиент выполняет запросы к серверу, а сервер осуществляет обработку этих запросов и предоставляет клиенту необходимые ресурсы или услуги. Разделение функций позволяет распределить нагрузку между клиентом и сервером, улучшить масштабируемость и обеспечить более эффективную обработку запросов.

Сервер как «черный ящик»: клиент не знает, каким образом сервер выполняет его запросы и какие конкретно ресурсы используются. Для клиента сервер выглядит как единая сущность, с которой он взаимодействует, без необходимости знания о его внутренней работе.

Надежность: клиент-серверная архитектура позволяет повысить надежность системы за счет распределения нагрузки между серверами. Например, в случае отказа одного сервера, клиенты могут переключиться на другой без прерывания обслуживания.

Масштабируемость: клиент-серверная архитектура позволяет добавлять новых клиентов и сервера, что обеспечивает горизонтальную и вертикальную масштабируемость. Это позволяет системе эффективно обрабатывать растущую нагрузку и адаптироваться к изменениям в требованиях пользователей.

Централизованное управление: сервер выполняет управление и контроль за ресурсами, данные и услуги которых предоставляются клиентам. Это упрощает управление системой и обеспечивает централизованные политики безопасности и доступа к данным.

Клиент-серверная архитектура позволяет взаимодействовать с различными платформами и технологиями, используя открытые стандарты для обмена данными и коммуникаций.

2 Техническое задание

Техническое задание, или ТЗ — это документ, в котором фиксируются требования к проекту. Условно ТЗ можно назвать любое поручение исполнителю, главное, чтобы в нем были ясно прописаны характеристики итогового продукта.

Для создания технического задания использовался стандарт ГОСТ 19.

Согласно ГОСТ 19 техническое задание должно включать следующие разделы:

Введение.

1. Общие сведения.
2. Цели и назначение создания автоматизированной системы.
3. Характеристика объекта автоматизации.
4. Требования к системе в целом.
 - 4.1 Требования к структуре и функционированию системы
 - 4.2 Требования к надежности
 - 4.3 Требования к безопасности
 - 4.4 Требования к эксплуатации, техническому обслуживанию, ремонту и хранению компонентов системы
 - 4.5 Требования к документированию
5. Состав и содержание работ по созданию системы.

ТЗ на разработку информационной системы представлено в отдельном документе.

					КП.09.02.07-5.23.201.08. ПЗ	Лист 20
Изм.	Лист	№ докум.	Подпись	Дата		

3 Проектирование ИС

3.1 Структурная схема ИС

Проектирование информационной системы началось с построения диаграммы вариантов использования. На рисунке 2 представлена диаграмма прецедентов Uses CASE. Она содержит 2 актёра, которые могут выполнять суммарно 9 функций, часть из которых может делать только один актёр, а часть – только другой актёр.

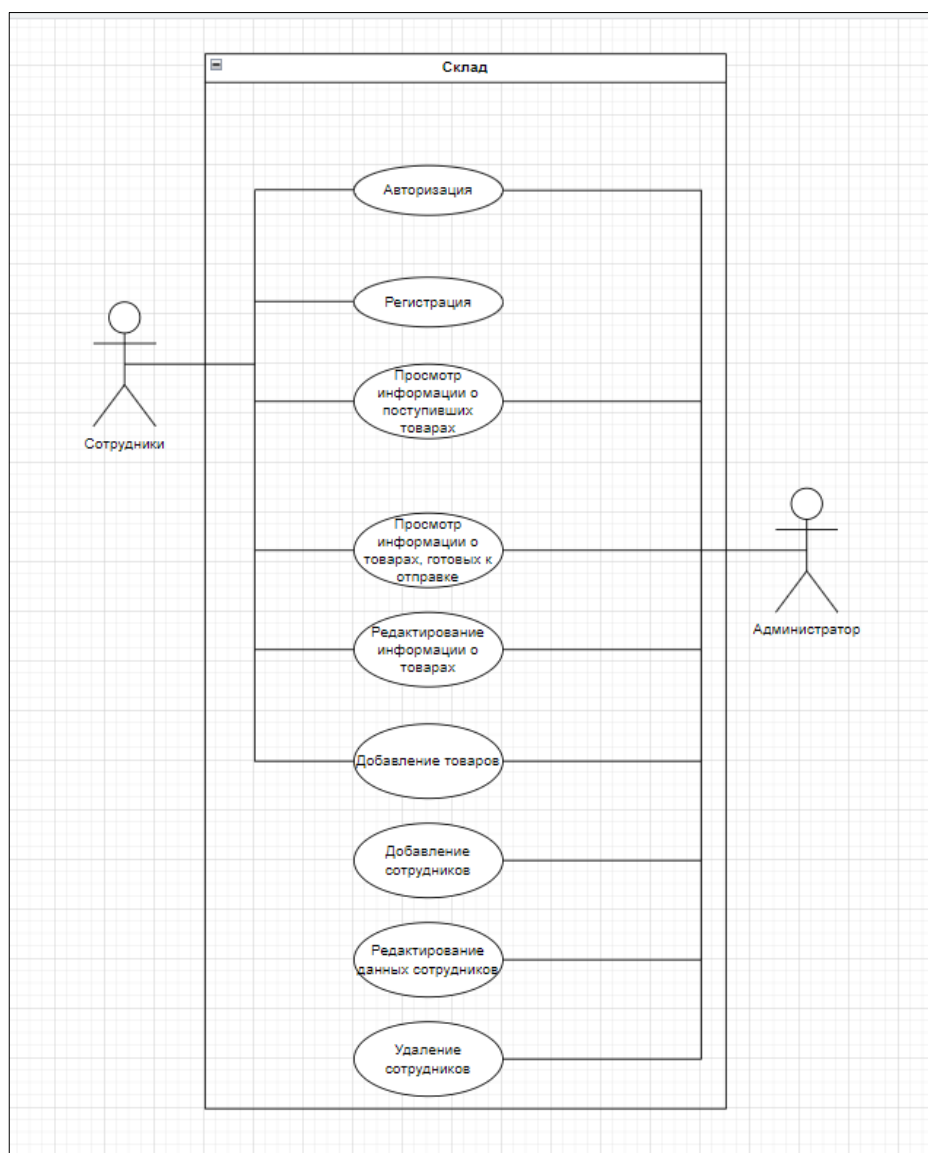


Рисунок 2 – Диаграмма прецедентов

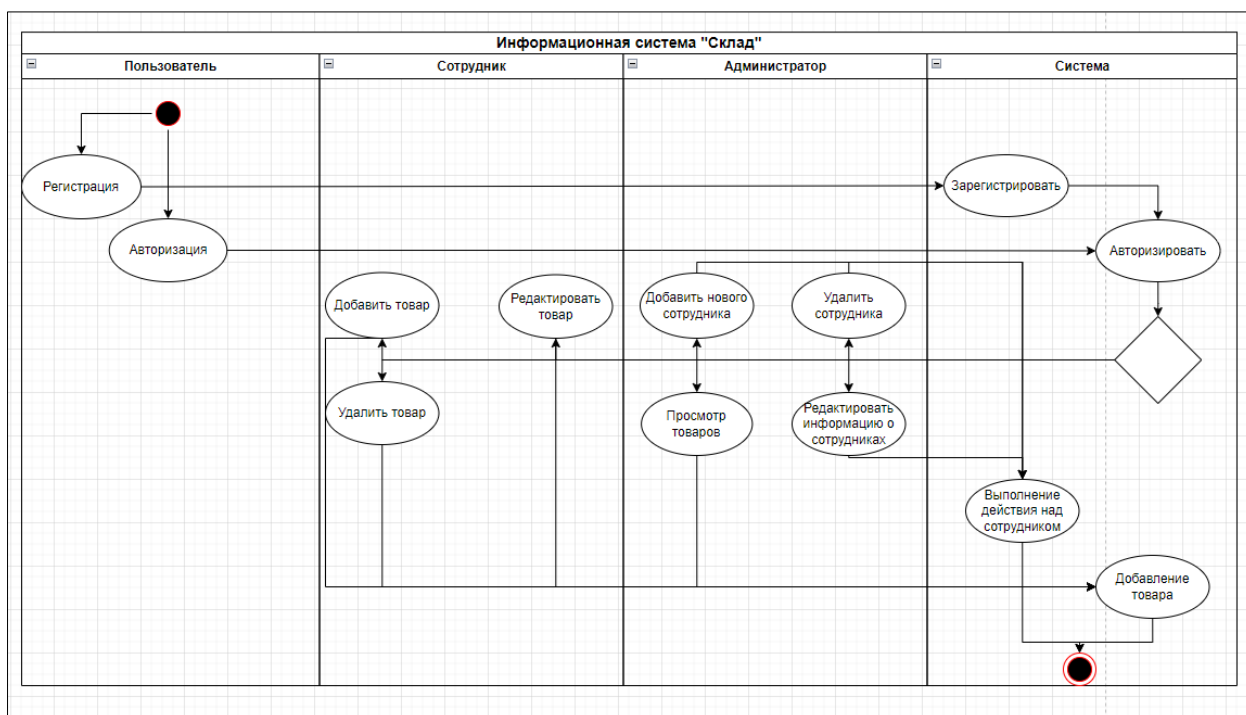


Рисунок 3 – Диаграмма деятельности

На рисунке 3 представлена диаграмма деятельности. Она содержит 4 роли. Процесс начинается с пользователя, который должен зарегистрироваться в системе, чтобы в дальнейшем использовать её или же просто авторизоваться в ней. Права доступа роли «Сотрудник» позволяют добавлять, изменять или же удалять товары, находящиеся на складе. Права доступа роли «Администратор» позволяют добавлять, удалять и редактировать информацию о сотрудниках, которые пользуются данной системой. Также «Администратор» может просматривать товары, находящиеся на складе.

Информационная система обрабатывает данные, а затем сохраняет их в БД.

На рисунке 4 представлена диаграмма развёртывания. Она показывает, что, чтобы пользоваться программным продуктом, необходим web-сервер, на котором размещаются ИС и БД, а также для администратора и сотрудников необходимо каждому устройство с браузером и выходом в интернет.

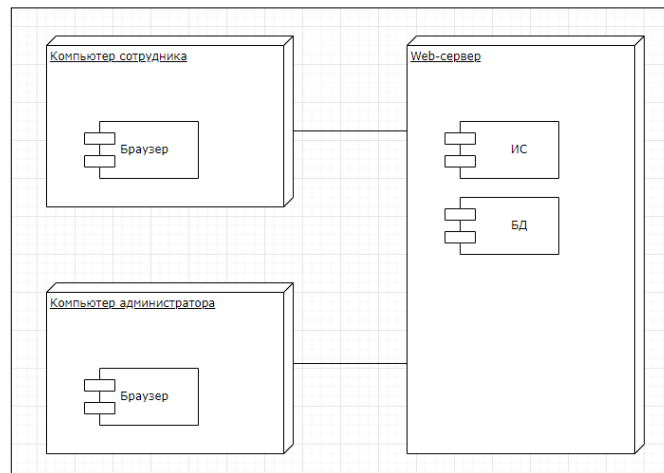


Рисунок 4 – Диаграмма развёртывания

3.2 Функциональная схема ИС

На рисунке 5 представлена контекстная диаграмма, отображающая деятельность ИС.

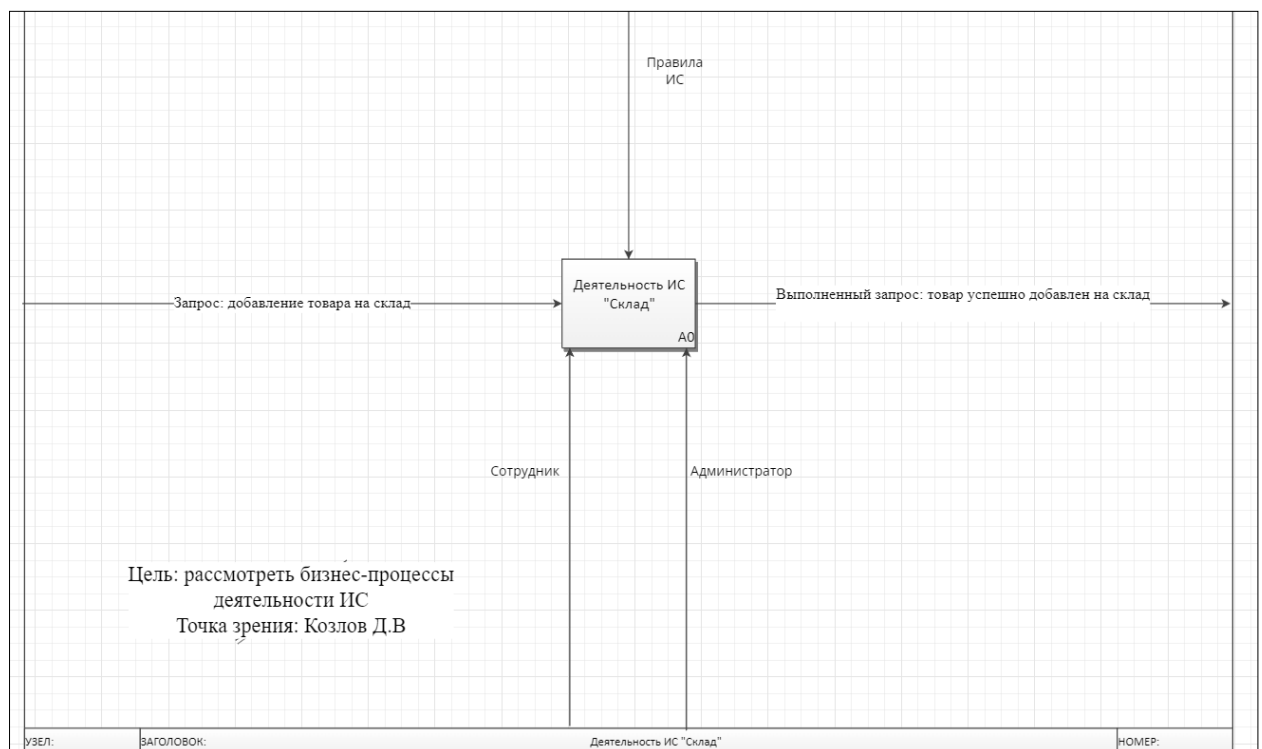


Рисунок 5 – Контекстная диаграмма IDEF0

```
graph LR
    I1[Сотрудник, добавляющий информацию о товарах] --> A1
    I2[Данные о сотруднике] --> A1
    A1 --> A2
    A2 --> A3
    A3 --> O1[Сотрудник, добавил информацию о товарах]
    A3 -- P3 --> A1
    A1 --> P3
    A2 --> P3
    A3 --> P3
```

На рисунке 7 представлена диаграмма классов. Она содержит 6 классов, среди них пользователь, сотрудник, администратор, прибывший товар, товар готовый к отгрузке, товар.

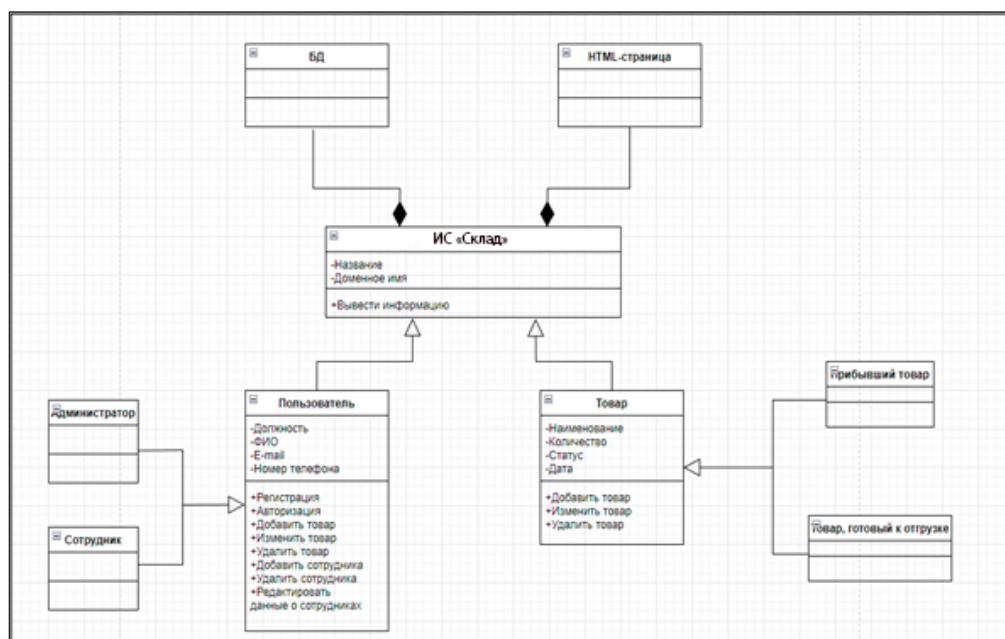


Рисунок 7 – Диаграмма классов

На рисунке 8 представлена диаграмма потоков данных. В центре всего сотрудник, который может взаимодействовать с товарами, для соответствующих действий в ИС предусмотрены соответствующие БД.

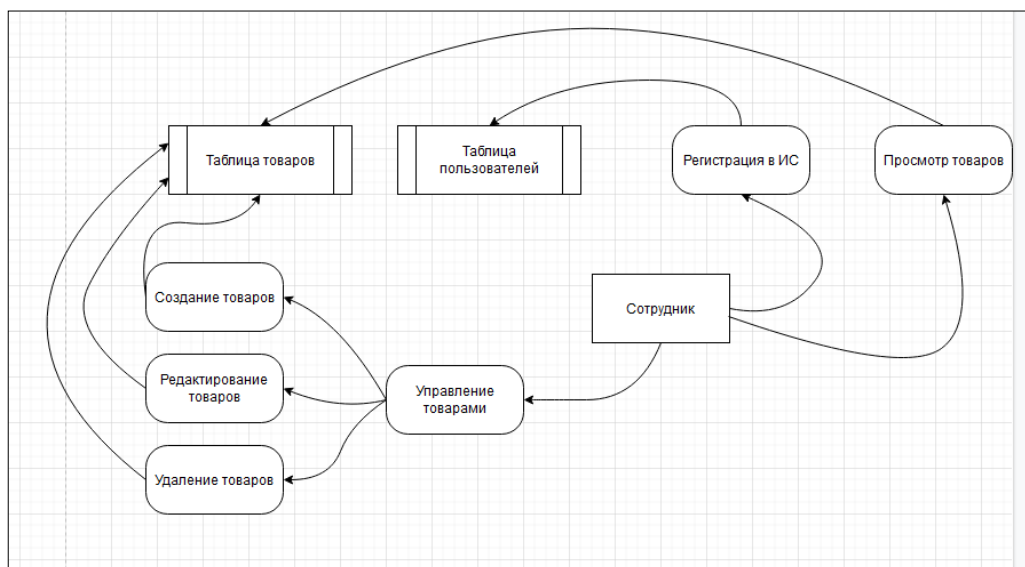


Рисунок 8 – Диаграмма потоков данных

3.3 Проектирование базы данных

Проектирование базы данных начинается с концептуального проектирование базы данных.

Концептуальное проектирование – построение семантической модели предметной области, то есть информационной модели наиболее высокого уровня абстракции. Такая модель создаётся без ориентации на какую-либо конкретную СУБД и модель данных.

На рисунке 9 представлена ER–диаграмма базы данных. Она содержит 6 таблиц для полного функционирования и качественной сортировки информации.

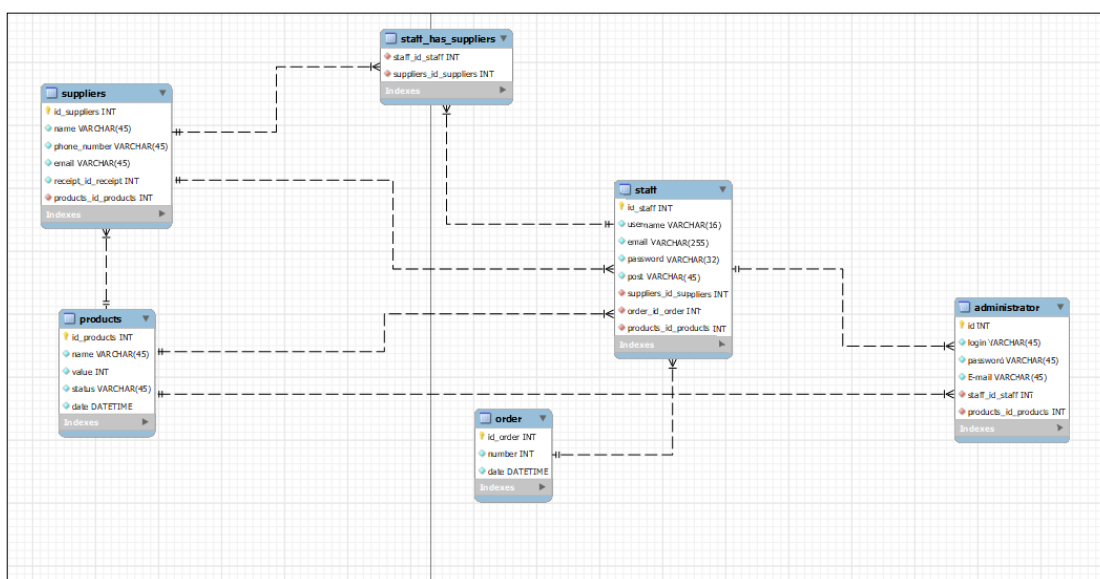


Рисунок 9 – ER–модель базы данных

Была выбрана третья нормальная форма нормализации базы данных. Вот основные преимущества данной формы:

– Устранение избыточности данных. ЗНФ помогает избежать повторения информации в таблицах, что позволяет сократить объём хранимых данных и уменьшить вероятность ошибок при обновлении данных.

– Улучшение структуры базы данных. 3НФ требует разделения таблиц на более мелкие и логически связанные части, что делает структуру базы данных более понятной и удобной для работы.

– Упрощение обновления данных. Благодаря разделению данных на более мелкие таблицы обновление информации становится более простым и эффективным процессом.

– Улучшение производительности. 3НФ может помочь улучшить производительность базы данных, так как более мелкие таблицы обеспечивают более быстрый доступ к данным и уменьшают нагрузку на систему.

Таблица 1 – Таблица staff

Таблица	Тип данных	Описание
Id	INT	ID (PK)
Login	VARCHAR(16)	Логин
Email	VARCHAR(255)	Электронная почта
Password	VARCHAR(32)	Пароль
Post	VARCHAR(45)	Должность
suppliers_id_suppliers	INT	Вторичный ключ
order_id_order	INT	Вторичный ключ
products_id_products	INT	Вторичный ключ

Таблица 2 – Таблица administrator

Таблица	Тип данных	Описание
Id	INT	ID (PK)
Login	VARCHAR(45)	Логин
Email	VARCHAR(255)	Электронная почта
Password	VARCHAR(45)	Пароль
staff_id_staff	INT	Вторичный ключ
products_id_products	INT	Вторичный ключ

Таблица 3 – Таблица order

Таблица	Тип данных	Описание
Id	INT	ID (PK)
Number	INT	Номер заказа
Date	DATETIME	Дата
staff_id_staff	INT	Вторичный ключ
products_id_products	INT	Вторичный ключ

Таблица 4 – Таблица products

Таблица	Тип данных	Описание
Id	INT	ID (PK)
Name	VARCHAR(45)	Наименование
Value	INT	Количество
Status	VARCHAR(45)	Статус товара
Date	DATETIME	Дата

Таблица 5 – Таблица suppliers

Таблица	Тип данных	Описание
Id	INT	ID (PK)
Name	VARCHAR(45)	Наименование
phone_number	INT	Телефонный номер
Email	VARCHAR(45)	Электронная почта
receipt_id_receipt	INT	Вторичный ключ
products_id_products	INT	Вторичный ключ

Таблица 6 – Таблица staff_has_suppliers

Таблица	Тип данных	Описание
staff_id_staff	INT	Вторичный ключ
suppliers_id_suppliers	INT	Вторичный ключ

3.4 Проектирование интерфейса

Для разработки пользовательского интерфейса был выбран инструмент Figma — онлайн-сервис для разработки интерфейсов и прототипирования с возможностью организации совместной работы в режиме реального времени.

В результате проектирование интерфейса будущей информационной системы были спроектированы прототипы трёх страниц: главная страница (рисунок 10), страница авторизации (рисунок 11), страница просмотра списка товаров (рисунок 12). Благодаря созданию прототипов, разработка непосредственно программного продукта будет значительно облегчена, за счёт наглядных примеров будущих страниц ИС.

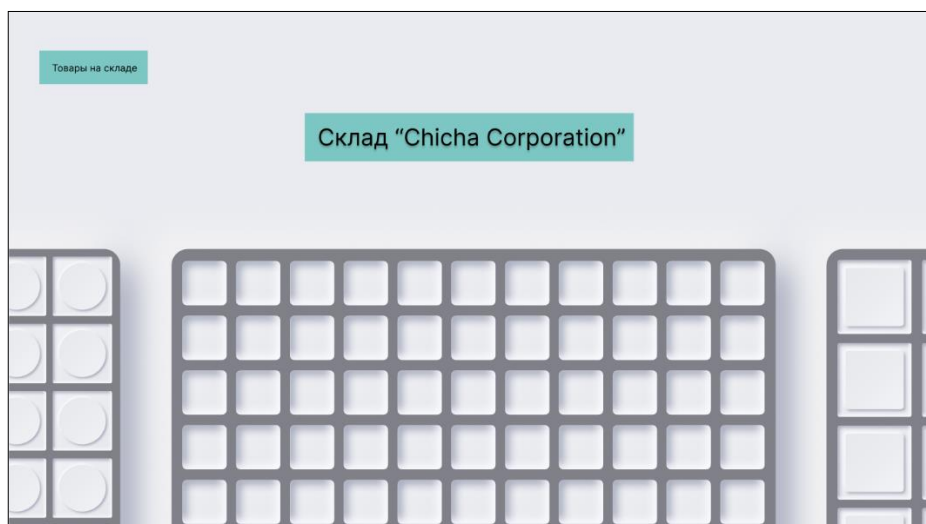


Рисунок 10 – Главная страница

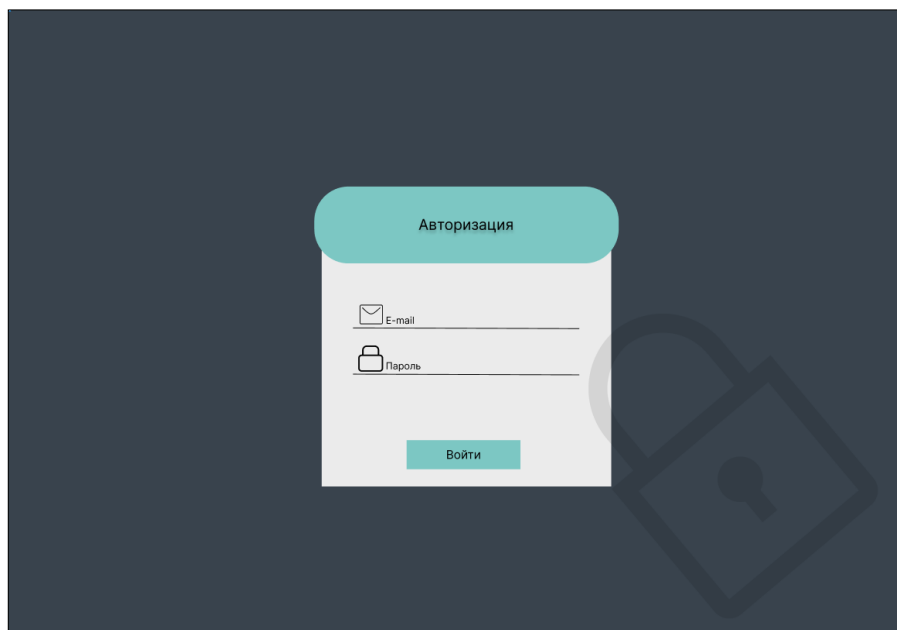


Рисунок 11 – Страница авторизации

Список		
	А	В
1	Товар	Количество
2	Покрышки	150 шт
3	Сапоги	32 шт

Рисунок 12 – Страница со списком товаров

4 Разработка ИС

4.1 Разработка интерфейса ИС

В информационной системе «Склад» были поставлены и выполнены следующие задачи:

- авторизация;
- регистрация;
- создание товаров;
- редактирование товаров;
- удаление товаров;
- создание таблицы с сотрудниками;
- редактирование таблицы сотрудников;
- удаление таблицы сотрудников;
- фильтрация списка товаров;

Программный продукт прост в освоении, т.к. имеет интуитивный интерфейс, в котором разобраться не составит труда даже начинающему пользователю ИС.

Общий размер дисковой памяти, занимаемой информационной системой, составляет 2,60 МБ (рисунок 13).

Размер:	2,60 МБ (2 728 426 байт)
На диске:	2,67 МБ (2 801 664 байт)

Рисунок 13 – Объем дисковой памяти ИС

Объем потребляемой ОЗУ составляет 30 668К (что примерно равно 30Мб) на одну вкладку в браузере Яндекс (рисунок 14).

Вкладка: Склад "ChiCha Corp"	30 668К
------------------------------	---------

Рисунок 14 – Потребляемая ОЗУ в Яндекс браузере

На рисунках 15, 16 показан html код для основной страницы, которую видят все пользователи не зависимо от того, зарегистрированы ли они или нет, а также прошли ли авторизацию и результат кода.

```
<?php
require_once ('views/layout/header.php');
require('controllers/Products.php');
?>
<div class="container mb-2 d-flex justify-content-between align-items-center p-2 bg-dark">
  <div>
    <a class="batan" href="/index.php">Главная</a>
  </div>
  <div>
    <a class="batan" href="views/auth/auth.php" >Вход</a>
    <a class="batan" href="views/auth/registration.php" >Регистрация</a>
  </div>
</div>
<table class="table table-hover table-dark">
  <thead>
    <tr>
      <th></th>
      <th>Название товара</th>
      <th>Количество</th>
      <th>Статус</th>
      <th>Дата</th>
    </tr>
  </thead>
  <tbody>
    <?php
    $db = new Products();
    $data = $db->get();
    foreach ($data as $key=>$row){
      ?>
      <tr>
        <td><?php echo ++$key;?></td>
        <td><?php echo $row['name'];?></td>
```

Рисунок 15 – Часть HTML кода основной страницы

<div>Главная</div>		<div>Вход</div> <div>Регистрация</div>		
Название товара		Количество	Статус	Дата
1	Куклы "Funky Monkey"	500	ждет отгрузки	2023-11-13 00:00:00
2	ChiCha12	453	ждет отгрузки	2023-11-16 00:00:00
3	ChiCha	453	отправлен	2023-11-16 03:15:00

Рисунок 16 – Результат HTML кода основной страницы

На рисунках 17, 18 показан html код для регистрации новых сотрудников, для последующего внесения в базу данных и дальнейшего прохождения авторизации, проверяется: зарегистрированы ли они или нет, а также прошли ли авторизацию и результат кода.


```

<?php
require_once ("../../views/layout/header.php");
require_once ('../../controllers/roles.php');
$db= new roles();
?>
<div>
    <a class="batan" href="/index.php">Главная страница</a>
</div>
<div class="d-flex flex-column justify-content-center align-items-center">
</div>
<form action="../../middleware/auth/registration.php" method="post" class="d-flex flex-column justify-content-center align-items-center">
    <!DOCTYPE html>
    <html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>Регистрация</title>
        <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css">
    </head>
    <body>
        <div class="bg-white container">
            <div class="row">
                <div class="col-md-12">
                    <h2>Регистрация</h2>
                    <p>Пожалуйста, заполните эту форму, чтобы зарегистрироваться</p>
                    <form action="" method="post">
                        <div class="form-group">
                            <label>Фамилия</label>
                            <input type="text" name="last_name" class="form-control" required>
                        </div>
                        <div class="form-group">
                            <label>Имя</label>
                            <input type="text" name="first_name" class="form-control" required />

```

Рисунок 17 – Часть HTML кода страницы регистрации

Главная страница

Регистрация

Пожалуйста, заполните эту форму, чтобы зарегистрироваться

Фамилия

Имя

Отчество

Пароль

[Зарегистрироваться](#)

Уже есть аккаунт ? [Авторизироваться здесь.](#)

Рисунок 18 – Результат html кода страницы регистрации

На рисунках 19, 20 изображен html код страницы авторизации пользователя, для последующего использования информационной системы, и результат кода.

```

<?php
require_once ("../../views/layout/header.php");
require_once ('../../controllers/roles.php');
$db= new roles();
?>
<div>
  <a class="batan" href="/index.php">Главная страница</a>
</div>
<div class="d-flex flex-column justify-content-center align-items-center">
</div>
<form action="../../middleware/auth/auth.php" method="post" class="d-flex flex-column justify-content-center align-items-center">
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Авторизация</title>
  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css">
</head>
<body>
  <div class="bg-white container">
    <div class="row">
      <div class="col-md-12">
        <h2>Авторизация</h2>
        <p>Пожалуйста введите данные для входа в систему</p>
        <form action="" method="post">
          <div class="form-group">
            <label>Фамилия</label>
            <input type="text" name="last_name" class="form-control" required />
          </div>
          <div class="form-group">
            <label>Имя</label>
            <input type="text" name="first_name" class="form-control" required />

```

Рисунок 19 – Часть HTML кода страницы авторизации

Главная страница

Авторизация

Пожалуйста введите данные для входа в систему

Фамилия

Имя

Отчество

Пароль

Нет аккаунта ? [Зарегистрироваться здесь.](#)

Рисунок 20 – Результат html кода страницы авторизации

Далее было создано 2 личных кабинета общего вида: для администратора и сотрудника. На рисунках 21, 22 изображен html код личного кабинета администратора, с помощью которого администратор может выполнять все действия в ИС.

```

<?php
require_once('../layout/header.php');
require ('../../controllers/Products.php');
$db= new Products();
?>
<!doctype html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
        content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="/public/css/bootstrap.css">
</head>
<body>
<?php
if(isset($_GET['message'])){
    echo $_GET['message'];
}
?>
<div class="container d-flex justify-content-between align-items-center p-2 mb-2 border_width">
    <div>
        <a class="btn btn-primary" href="/views/admin/menu.php">Действия</a>
    </div>
    <form action="../../middleware/auth/logout.php" method="post">
        <button class="btn btn-primary" type="submit"
            onclick="document.location.replace('../../middleware/auth/logout.php');">Выход</button>
    </form>
</div>
<table class="table table-hover table-dark">
    <thead>
    <tr>

```

Рисунок 21 – Часть html код личного кабинета администратора

Действия		Выход		
Название	Количество	Статус	Дата	
1 Куклы "Funky Monkey"	500	ждет отгрузки	2023-11-13 00:00:00	
2 ChiCha12	453	ждет отгрузки	2023-11-16 00:00:00	
3 ChiCha	453	отправлен	2023-11-16 03:15:00	

Рисунок 22 – Итоговой вид личного кабинета администратора

На рисунках 23, 24 изображен html код личного кабинета сотрудника, с помощью которого сотрудник может выполнять действия над товарами, прибывшими на склад.

```

<?php
require ('../layout/header.php');
require ('../controllers/Products.php');
$db = new Products();
?>
<!doctype html>
<html lang="ru">
<head>
<meta charset="UTF-8">
<meta name="viewport"
content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
<meta http-equiv="X-UA-Compatible" content="ie=edge">
<link rel="stylesheet" href="/public/css/bootstrap.min.css">
</head>
<body>
<?php
if(isset($_GET['message'])){
echo $_GET['message'];
}
}
<div class="container d-flex justify-content-between align-items-center p-2 mb-2">
<div>
<a class="knopka" href="menu.php">Действия с товарами</a>
</div>
<form action="../middleware/auth/logout.php" method="post">
<button class="btn btn-primary" type="submit" onclick="document.location.replace('../middleware/auth/logout.php');">Выход</button>
</form>
</div>
<table class="table table-hover table-dark">
<thead>
<tr>
<th> </th>

```

Рисунок 23 – Часть html кода личного кабинета сотрудника

Действия с товарами				Выход
Дата	Название	Количество	Статус	
1 2023-11-13 00:00:00	Куклы "Funky Monkey"	500	ждет отгрузки	
2 2023-11-16 00:00:00	ChiCha12	453	ждет отгрузки	
3 2023-11-16 03:15:00	ChiCha	453	отправлен	

Рисунок 24 – Результат html кода личного кабинета сотрудника

4.2 Разработка базы данных ИС

После того, как разработан интерфейс, необходимо, где-то хранить данные, которые будут заноситься пользователями ИС. Для этого необходимо разработать базу данных (далее – БД).

БД состоит из 6 таблиц: users, suppliers, staff, products, order, administrator

Таблица users на рисунке 25.

← T →						id	last_name	first_name	father_name	password	role	
<input type="checkbox"/>		Изменить		Копировать		Удалить	1	Козлов	Дмитрий	Витальевич	0	1
<input type="checkbox"/>		Изменить		Копировать		Удалить	2	Иванов	Иван	Иванович	123123	2

Рисунок 25 – Таблица users

Таблица staff на рисунке 26.

<div><div>←T→</div><div>▼</div></div>				id	last_name	first_name	father_name	post	
<input type="checkbox"/>		Изменить	Копировать	Удалить	1	Шаинов	Руслан	Хуршедович	Грузчик

Рисунок 26 – Таблица staff

Таблица products на рисунке 27.







			id	name	value	status	date				
<input type="checkbox"/>		Изменить		Копировать		Удалить	6	Автомобильные диски	200	Прибыл	2022-12-18 06:04:00
<input type="checkbox"/>		Изменить		Копировать		Удалить	7	Автомобильные коврики	50	Готов к отгрузке	2022-12-15 00:00:00

Рисунок 27 – Таблица products

Таблица order на рисунке 28.




				id	number	date
<input type="checkbox"/>		Изменить	 Копировать	 Удалить	1	24 2022-12-15 23:51:25

Рисунок 28 – Таблица order

Таблица suppliers на рисунке 29.

id	name	phone_number	email
1	ООО:"ChichTORG"	0	dasd23@mail.ru

Рисунок 29 – Таблица suppliers

Таблица administrator на рисунке 30.

				id	login	email	password
<input type="checkbox"/>		Изменить		Копировать		Удалить	1 DimaDead dimadead55@gmail.com camio019tk

Рисунок 30 – Таблица administrator

4.3 Разработка ИС

Для того, чтобы ИС работала, был создан файл db.php (рисунок 31), в котором осуществляется подключение к БД через PDO (PHP Data Object). В подключении указывается адрес хостинга, а также название БД, кодировка текста, имя пользователя и пароль (который по умолчанию является пустым).

```
<?php

class DB
{
    protected function connect(){
        return new PDO( dsn: 'mysql:host=localhost;dbname=20085_sklad;charset=utf8', username: 'root', password: '');
    }

    protected function DBAll($query){
        $connect=$this->connect();
        $sql = $connect->query($query);
        $sql->execute();
        return $sql->fetchAll();
    }

    protected function transaction($query,$message){
        $connect=$this->connect();
        try{
            $connect->beginTransaction();
            $connect->exec($query);
            $connect->commit();
            return json_encode([
                'message'=>$message
            ]);
        }catch (PDOException $e){
            $connect->rollBack();
            return json_encode([
                'message'=>$e->getMessage()
            ]);
        }
    }
}
```

Рисунок 31 – Подключение к базе данных

Затем были реализованы функции авторизации, регистрации и выхода из учётной записи с помощью моделей auth.php, registration.php и logout.php (рисунки 32, 33, 34). Также были созданы представления авторизации (auth.php) и регистрации (registration.php). (рисунок 35, 36) Полная версия программного кода контроллера находится в приложении Б.

```

<?php
require(' ../../controllers/roles.php');
$db = new roles();
$last_name = $_POST['last_name'];
$first_name = $_POST['first_name'];
$father_name = $_POST['father_name'];
$password = $_POST['password'];

$db->login(json_encode([
    'last_name'=>$last_name,
    'first_name'=>$first_name,
    'father_name'=>$father_name,
    'password'=>$password,
]));
session_start();
if($_SESSION['user']->role==2) {
    header( header: 'Location: ../../views/staff/index.php');
}
if($_SESSION['user']->role==1) {
    header( header: 'Location: ../../views/admin/index.php');
}

```

Рисунок 32 – Модель auth.php

```

<?php
require(' ../../controllers/roles.php');
$db = new roles();
$last_name = $_POST['last_name'];
$first_name = $_POST['first_name'];
$father_name = $_POST['father_name'];
$password = $_POST['password'];

$response = $db->registration(json_encode([
    'last_name'=>$last_name,
    'first_name'=>$first_name,
    'father_name'=>$father_name,
    'password'=>$password,
]));

header( header: 'Location: ../../views/auth/auth.php');

```

Рисунок 33 – Модель registration.php

```

<?php
session_start();
unset($_SESSION['user']);
session_write_close();

header( header: "Location: ../../index.php");
exit;
?>

```

Рисунок 34 – Модель logout.php

```

<?php
require_once ("../../views/layout/header.php");
require_once ("../../controllers/roles.php");
$db= new roles();
?>
<div>
<a class="batan" href="/index.php">Главная страница</a>
</div>
<div class="d-flex flex-column justify-content-center align-items-center">
<h3 style="background-color: rgb(112, 128, 144); padding: .3em .4em; border-radius: 5px; ">Авторизация</h3>
</div>
<form action="../../middleware/auth/auth.php" method="post" class="d-flex flex-column justify-content-center align-items-center">
<div class="col-2">
<label style="background-color: rgb(112, 128, 144); padding: .3em .4em; border-radius: 5px; " for="last_name">Фамилия</label>
<input id="last_name" name="last_name" type="text" class="form-control" aria-describedby="emailHelp" placeholder="Фамилия" required>
</div>
<div class="col-2">
<label style="background-color: rgb(112, 128, 144); padding: .3em .4em; border-radius: 5px; " for="name">Имя</label>
<input id="first_name" name="first_name" type="text" class="form-control" aria-describedby="emailHelp" placeholder="Имя" required>
</div>
<div class="col-2">
<label style="background-color: rgb(112, 128, 144); padding: .3em .4em; border-radius: 5px; " for="father_name">Отчество</label>
<input id="father_name" name="father_name" type="text" class="form-control" aria-describedby="emailHelp" placeholder="Отчество" required>
</div>
<div class="password">
<label style="background-color: rgb(112, 128, 144); padding: .1em .2em; border-radius: 5px; " for="password">Пароль</label>
<input id="password-input" name="password" type="password" class="form-control" placeholder="Введите пароль" required>
<a href="#" class="password-control"></a>
</div>
<script src="https://snipp.ru/cdn/jquery/2.1.1/jquery.min.js"></script>
<script>
$( 'body' ).on( 'click', '.password-control', function() {
if ( $( '#password-input' ).attr( 'type' ) == 'password' ) {
$( this ).addClass( 'view' );
$( '#password-input' ).attr( 'type', 'text' );
} else {
$( this ).removeClass( 'view' );
$( '#password-input' ).attr( 'type', 'password' );
}
return false;
}
});
</script>
<button type="submit" class="btn btn-primary">Вход</button>
</form>

```

Рисунок 35 – Представление auth.php

```

<?php
require_once ("../../views/layout/header.php");
require_once ("../../controllers/roles.php");
$db= new roles();
?>
<div>
<a class="batan" href="/index.php">Главная страница</a>
</div>
<div class="d-flex flex-column justify-content-center align-items-center">
<h3 style="background-color: rgb(112, 128, 144); padding: .1em .2em; border-radius: 5px; ">Регистрация</h3>
</div>
<form action="../../middleware/auth/registration.php" method="post" class="d-flex flex-column justify-content-center align-items-center">
<div class="col-3">
<label style="background-color: rgb(112, 128, 144); padding: .1em .2em; border-radius: 5px; " for="last_name">Фамилия</label>
<input id="last_name" name="last_name" type="text" class="form-control" aria-describedby="emailHelp" placeholder="Фамилия" required>
</div>
<div class="col-3">
<label style="background-color: rgb(112, 128, 144); padding: .1em .2em; border-radius: 5px; " for="first_name">Имя</label>
<input id="first_name" name="first_name" type="text" class="form-control" aria-describedby="emailHelp" placeholder="Имя" required>
</div>
<div class="col-3">
<label style="background-color: rgb(112, 128, 144); padding: .1em .2em; border-radius: 5px; " for="father_name">Отчество</label>
<input id="father_name" name="father_name" type="text" class="form-control" aria-describedby="emailHelp" placeholder="Отчество" required>
</div>
<div class="password">
<label style="background-color: rgb(112, 128, 144); padding: .1em .2em; border-radius: 5px; " for="password">Пароль</label>
<input id="password-input" name="password" type="password" class="form-control" placeholder="Введите пароль" required>
<a href="#" class="password-control"></a>
</div>
<script src="https://snipp.ru/cdn/jquery/2.1.1/jquery.min.js"></script>
<script>
$( 'body' ).on( 'click', '.password-control', function() {
if ( $( '#password-input' ).attr( 'type' ) == 'password' ) {
$( this ).addClass( 'view' );
$( '#password-input' ).attr( 'type', 'text' );
} else {
$( this ).removeClass( 'view' );
$( '#password-input' ).attr( 'type', 'password' );
}
return false;
}
});
</script>
<button type="submit" class="btn btn-primary">Зарегистрироваться</button>

```

Рисунок 36 – Представление registration.php

4.4 Тестирование ИС

Тестирование и отладка являются двумя важными процессами в разработке программного продукта.

Тестирование — это процесс проверки программного продукта на соответствие требованиям и ожиданиям пользователя. Оно включает в себя различные виды тестов, такие как модульные тесты, интеграционные тесты, системные тесты и пользовательские тесты. Цель тестирования заключается в выявлении ошибок, дефектов и несоответствий, а также в проверке функциональности, производительности и безопасности продукта.

Отладка — это процесс идентификации и исправления ошибок в программном продукте. Она включает в себя анализ кода, выполнение шагов программы для идентификации проблемного участка, использование инструментов для отслеживания ошибок и исправления дефектов. Основная цель отладки состоит в обнаружении и исправлении ошибок, которые могут привести к неправильной работе программы или сбоям.

Тестирование и отладка являются неотъемлемой частью разработки программного продукта и имеют важное значение. Они позволяют обнаруживать и устранять ошибки и дефекты еще на ранних этапах разработки, что ведет к повышению качества и надежности программного продукта. Эти процессы также помогают улучшить производительность и безопасность программы, а также повысить удовлетворенность пользователей.

Таблица 7 – Сценарий тестирования для сотрудника 1

Поле	Описание
Дата(ы) теста	5 октября 2023
Приоритет тестирования (Низкий/Средний/Высокий)	Высокий
Заголовок/название теста	Тестовый сценарий 1: переход во вкладку со всеми товарами
Этапы теста	<ol style="list-style-type: none"> 1. Сотрудник входит в систему, используя свои учетные данные 2. Сотрудник попадает в свой личный кабинет, где переходит в интерфейс просмотра всех товаров 3. Сотрудник просматривает список всех товаров, которые отображаются из базы данных
Тестовые данные	Данные о товарах (ID, наименование товара, время прибытия, статус)
Ожидаемый результат	Сотрудник успешно просматривает список всех товаров, отображаемых в интерфейсе, включая информацию о наименовании товара, его количестве, времени прибытия, а также статусах товаров
Фактический результат	Сотрудник без проблем просматривает список всех товаров, которые выводятся напрямую из базы данных.

Таблица 8 – Сценарий тестирования для сотрудника 2

Поле	Описание
Дата(ы) теста	5 октября 2023
Приоритет тестирования (Низкий/Средний/Высокий)	Высокий
Заголовок/название теста	Тестовый сценарий 2: добавление новых товаров, используя внутренний интерфейс информационной системы

Изм.	Лист	№ докум.	Подпись	Дата

Продолжение таблицы 8

Этапы теста	<ol style="list-style-type: none"> 1. Сотрудник входит в систему, используя свои учетные данные 2. Сотрудник попадает в свой личный кабинет, где переходит в интерфейс просмотра всех товаров 3. Сотрудник нажимает на кнопку добавления нового товара 4. Сотрудник переходит в интерфейс добавления нового товара, где вводит необходимые данные в поля 5. Сотрудник добавляет новый товар
Тестовые данные	Данные о товарах (ID, наименование товара, время прибытия, статус)
Ожидаемый результат	Сотрудник успешно просматривает список всех товаров, отображаемых в интерфейсе, включая информацию о наименовании товара, его количестве, времени прибытия, а также статусах товаров
Фактический результат	Сотрудник без проблем просматривает список всех товаров, которые выводятся напрямую из базы данных.

Таблица 9 – Сценарий тестирования для администратора 1

Поле	Описание
Дата(ы) теста	6 октября 2023
Приоритет тестирования (Низкий/Средний/Высокий)	Высокий
Заголовок/название теста	Тестовый сценарий 1: Регистрация нового пользователя администратором
Этапы теста	<ol style="list-style-type: none"> 1. Администратор входит в систему, используя свои учетные данные. 2. Администратор переходит в окно добавления сотрудника. 3. Администратор нажимает кнопку "Добавить нового сотрудника". 4. Администратор вводит данные нового пользователя: имя, фамилия, логин, пароль 5. Администратор подтверждает создание пользователя. 6. Система сохраняет нового пользователя в базе данных.

Продолжение таблицы 9

Тестовые данные	Имя: Руслан Фамилия: Шаинов Отчество: Хуршедович Пароль: 112233
Ожидаемый результат	Система успешно создает нового пользователя, пользователь отображается в списке существующих сотрудников.
Фактический результат	Новый пользователь добавлен и отображается в системе

Таблица 10 – Сценарий тестирования для администратора 2

Поле	Описание
Дата(ы) теста	6 октября 2023
Приоритет тестирования (Низкий/Средний/Высокий)	Средний
Заголовок/название теста	Тестовый сценарий 2: Просмотр всех товаров администратором
Этапы теста	<ol style="list-style-type: none"> 1. Администратор входит в систему, используя свои учетные данные. 2. Администратор переходит в интерфейс просмотра всех товаров. 3. Администратор просматривает список всех товаров.
Тестовые данные	Данные о товарах (ID, наименование товара, время прибытия, статус)
Ожидаемый результат	Администратор успешно просматривает список всех товаров, включая всю информацию, указанную выше
Фактический результат	Администратор успешно просматривает список всех товаров.

Был создан чек–лист, представленный в таблице 11.

Таблица 11– Чек лист тестирования

Тест	Вводные данные	Ожидаемый результат	Фактический результат	Результат тестирования	Комментарий
Авторизация	Данные пользователя: логин, пароль	Пользователь вошел в систему	Пользователь авторизован	Неуспешно	–
Регистрация	Данные пользователя: логин, пароль	Данные добавляются в систему	Данные добавлены	Неуспешно	–
Добавление товаров	Данные товаров: id, статус, наименование, количество, время	Данные о новых товарах добавляются в базу данных	Данные сохранены	Неуспешно	–
Удаление товаров	Данные товаров: id, статус, наименование, количество, время	Данные о товарах удаляются из системы	Данные удалены	Неуспешно	–
Редактирование товаров	Данные товаров: id, статус, наименование, количество, время	Данные товаров успешно изменяются	Изменения в данных товарах успешно внеслись	Неуспешно	–

```

<?php
use PHPUnit\Framework\TestCase;

class RegistrationTest extends TestCase {
    public function testRegistrationSuccess() {
        // Устанавливаем необходимые значения для теста
        $first_name = 'john';
        $last_name = 'johnsson'
        $father_name = 'Johnson'
        $password = 'password123';

        // Создаем экземпляр класса, ответственного за регистрацию
        $registration = new Registration();

        // Выполняем регистрацию
        $result = $registration->register($first_name,$last_name,$father_name $password);

        // Проверяем, что результат успешен
        $this->assertTrue($result);
    }

    public function testRegistrationFailure() {
        // Устанавливаем необходимые значения для теста
        $first_name = 'john';
        $last_name = 'johnsson'
        $father_name = 'Johnson'
        $password = 'password123'

        // Создаем экземпляр класса, ответственного за регистрацию
        $registration = new Registration();

        // Регистрируем пользователя с тем же именем
        // (предполагаем, что это должно приводить к ошибке)
        $registration->register($first_name,$last_name,$father_name $password);

        // Попытка зарегистрировать пользователя с тем же именем
        $result = $registration->register($first_name,$last_name,$father_name $password);

        // Проверяем, что результат неуспешен
        $this->assertFalse($result);
    }
}

```

Рисунок 37 – Модульное тестирование регистрации

Также, для тестирования функциональности базы данных использовался подход CRUD.

CRUD-тестирование — это метод тестирования черного ящика для проверки функциональности программного продукта.

Были созданы запросы, которые покрывают технологию CRUD.

1. SELECT * FROM `users` ORDER BY id DESC;

Вывод записей из таблицы users, сортированных по возрастанию ID.

2. UPDATE `users` SET `role` = '1' WHERE `users`.`id` = 3;

Обновление столбца role и добавление роли админ для пользователя с id = 3.

3. DELETE FROM `users` WHERE id = 3

Удаление из таблицы users пользователя с id = 3.

4. UPDATE `users` SET `last_name` = 'Шайнов', `first_name` = 'Руслан', `father_name` = 'Хурshedович' WHERE `users`.`id` = 2

Обновление данных пользователя с id = 2

Фамилия: Шаинов,

Имя: Руслан,

Отчество: Хуршедович.

5. INSERT INTO `products` (`name`, `value`) VALUES (Тормозные колодки`, `300`,

Добавление данных заказа:

Название: Тормозные колодки

Количество: 300

6. SELECT * FROM `products` where value is not null;

Выбор из таблицы products товаров с заполненным полем value (количество)

7. DELETE FROM `products` WHERE date is not null

Удаление из таблицы products заказов, где поле date не заполнено

					КП.09.02.07-5.23.201.08. ПЗ	Лист 47
Изм.	Лист	№ докум.	Подпись	Дата		

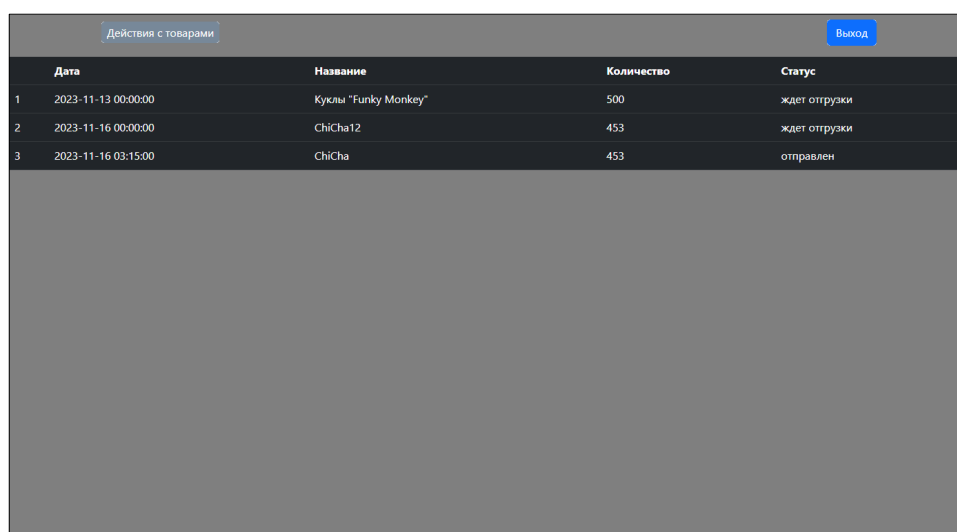
5 Обучающая документация для пользователей информационной системы в соответствии с ГОСТ Р 59795–2021

5.1 Подготовка к работе

Для того, чтобы войти в веб–сервер нужно запустить файл start.bat, скопировать ссылку <http://localhost:8080>, и вставить её в поисковую строку браузера. После этого откроется главная страница системы.

5.2 Описание операций

В данном руководстве пользователя рассмотрен пример личного кабинета администратора. На рисунке 38 можно видеть список товаров, поступивших на склад, а также 2 кнопки: «Выход» и «Действия с товарами» для взаимодействия с товарами.



Действия с товарами		Выход	
Дата	Название	Количество	Статус
1 2023-11-13 00:00:00	Куклы "Funky Monkey"	500	ждет отгрузки
2 2023-11-16 00:00:00	ChiCha12	453	ждет отгрузки
3 2023-11-16 03:15:00	ChiCha	453	отправлен

Рисунок 38 – Личный кабинет пользователя

При нажатии на кнопку «Действия с товарами» открывается страница, на которой можно выбрать либо «Удаление товара», либо «Добавление товара», при нажатии на «Добавление товара» открывается соответствующее меню. Это можно увидеть на рисунке 39.

Назад

Добавление товара

Дата

ДД.ММ.ГГГГ --:--

Название товара

Название

Количество

Количество

Статус

Отправить

Рисунок 39 – Создание товара

При нажатии на кнопку «Удаление товара», открывается соответствующая вкладка. Представлено на рисунке 40.

Назад

Дата: 2023-11-13 00:00:00
Название товара: Куклы "Funky Monkey"
Количество: 500
Статус: ждет отгрузки
Удалить

Дата: 2023-11-16 00:00:00
Название товара: ChiCha12
Количество: 453
Статус: ждет отгрузки
Удалить

Дата: 2023-11-16 03:15:00
Название товара: ChiCha
Количество: 453
Статус: отправлен
Удалить

Рисунок 40 – Удаление товара

Заключение

В ходе выполнения, данного дипломного проекта, была разработана информационная система «Склад» – система, в которой сотрудники могут легко взаимодействовать с товарами, что заметно облегчает их работу. Был определен и реализован следующий функционал информационной системы:

для сотрудников:

- авторизация;
- добавление товаров;
- удаление товаров;
- редактирование товаров.

для администратора:

- создание таблиц с сотрудниками;
- удаление таблиц сотрудников;
- редактирование таблиц сотрудников;
- добавление новых товаров;
- удаление товаров;
- редактирование товаров.

Были рассмотрены возможные реализации ИС с использованием разных технологий и языков программирования, но в результате анализа инструментальных средств разработки выбор остановился на языке программирования PHP.

Был разработан браузерный программный продукт, с широким функционалом, а также понятным и лёгким интерфейсом.

В дальнейшем ИС может развиваться путём расширения функционала и совершенствования интерфейса, тем самым набирая большее сообщество пользователей, а также и актуальность программного продукта.

Все поставленные цели и задачи дипломного проекта были успешно выполнены.

Список используемых материалов

- 1 Colorscheme – Цвета HTML. Таблица из 147 имён цветов для HTML и CSS – URL: <https://colorscheme.ru/html-colors.html> (дата обращения: 15.10.2023). – Текст: электронный.
- 2 dev.to - Laravel Breeze Tutorial: The Definitive Guide (2021) – URL: <https://dev.to/kaperskyguru/laravel-breeze-tutorial-the-definitive-guide-2021-gdp> (дата обращения: 21.09.2023). – Текст: электронный.
- 3 Dmosk – Примеры SQL-запросов в MariaDB (MySQL) – URL: <https://www.dmosk.ru/miniinstruktions.php?mini=sql-mysql> (дата обращения: 11.10.2023). – Текст: электронный.
- 4 FrontEnd Resource – 107 Beautiful CSS Cards examples to improve your UI – URL: <https://frontendresource.com/css-cards/> (дата обращения: 11.10.2023). – Текст: электронный.
- 5 Htmlbook – Самоучитель HTML4 – URL: <http://htmlbook.ru/samhtml> (дата обращения: 15.10.2023). – Текст: электронный.
- 6 HTML5 BOOK – Основы CSS – URL: <https://html5book.ru/osnovy-css/> (дата обращения: 20.10.2023). – Текст: электронный.
- 7 Itnan – MariaDB в сравнении с MySQL в 2022 году – URL: <https://itnan.ru/post.php?c=1&p=662870> (дата обращения: 24.10.2023). – Текст: электронный.
- 8 JetBrains – Возможности PhpStorm – URL: <https://www.jetbrains.com/ru-ru/phpstorm/features/> (дата обращения: 25.10.2023). – Текст: электронный.
- 9 Laravel – Blade Templates – URL: <https://laravel.com/docs/10.x/blade> (дата обращения: 16.09.2023). – Текст: электронный.
- 10 Laravel – Documentation – URL: <https://laravel.com/docs/10.x> (дата обращения: 16.09.2023). – Текст: электронный.

11 MDN – Основы HTML – URL:
https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/HTML_basic (дата обращения: 22.09.2023). – Текст: электронный.

12 Oracleplsql – URL: <https://oracleplsql.ru/mariadb-manual.html> (дата обращения: 24.10.2023). – Текст: электронный.

13 Php.net – Что такое PHP? – URL:
<https://www.php.net/manual/ru/intro-what-is.php> (дата обращения: 24.09.2023). – Текст: электронный.

14 StudFiles – Инфологическое моделирование. Ег-модель. – URL:
<https://studfile.net/preview/6862142/page:11/> (дата обращения: 28.09.2023). – Текст: электронный.

15 zaLinux – Изучение MySQL/ MariaDB – URL:
<https://zlinux.ru/?p=760> (дата обращения: 24.09.2023). – Текст: электронный.

					КП.09.02.07-5.23.201.08. ПЗ	Лист 52
Изм.	Лист	№ докум.	Подпись	Дата		

Приложение А – Техническое задание
Министерство образования Иркутской области
Государственное бюджетное профессиональное
образовательное учреждение Иркутской области
«Иркутский авиационный техникум»
(ГБПОУИО «ИАТ»)

ДП.09.02.07–5.24.201.08 ПЗ

ТЕХНИЧЕСКОЕ ЗАДАНИЕ
ИНФОРМАЦИОННАЯ СИСТЕМА
«СКЛАД»

Руководитель: _____ (А.С. Некипелова)
(подпись, дата)

Студент: _____ (Д.В. Козлов)
(подпись, дата)

Иркутск 2024

					КП.09.02.07-5.23.201.08. ПЗ	Лист 53
Изм.	Лист	№ докум.	Подпись	Дата		

1 Общие сведения

Наименование работы: информационная система «Склад».

Исполнитель: студент иркутского авиационного техникума, группа ИС 20–1, Козлов Д.В

Разработка информационной системы проходит в рамках дипломного проекта.

Сроки разработки информационной системы с 22.02.2024 по 18.05.2024 года.

2 Цели и назначение создания автоматизированной системы

Дипломная работа нацелена на создание информационной системы "Склад", которая предоставляет функциональность для добавления новых заказов, редактирования существующих и отслеживания состояния товаров.

В рамках этой информационной системы будут реализованы следующие возможности:

- Аутентификация пользователей.
- Регистрация новых пользователей.
- Создание записей о заказах.
- Редактирование существующих записей.
- Удаление записей.

3 Характеристика объекта автоматизации

Данная информационная система разрабатывается с целью упростить процессы работы сотрудников склада, облегчая создание и отслеживание заказов и их статусов.

4 Требования к системе в целом

4.1 Требования к структуре и функционированию приложения

Функции веб–приложения:

1. Окно «Авторизации»:
 - 1.1. авторизация пользователей.
2. Окно «Регистрация»:
 - 2.1. регистрация пользователей.
3. Окно «Главная»:
 - 3.1. Информация о заказах;
 - 3.1.1. добавление наименований;
 - 3.1.2. удаление наименований;
 - 3.1.3. редактирование данных.
 - 3.2. Окно «Сотрудники»:
 - 3.2.1. добавление сотрудников;
 - 3.2.2. удаление сотрудников;
 - 3.2.3. редактирование данных.

4.2 Требования к надежности

Для гарантированно надежной работы, необходимо осуществлять проверку корректности входных данных и обеспечивать валидность полей. Входные данные получаются от пользователя через клавиатуру и отображаются в отдельных ячейках таблицы.

4.3 Требования к безопасности

Для обеспечения безопасности в информационной системе, необходимо реализовать разграничение прав доступа.

4.4 Требования к эксплуатации, техническому обслуживанию, ремонту и хранению компонентов системы

Минимальные системные требования для сервера:

- 1) Процессор: Intel Pentium 4 2.0Ghz / AMD XP 2200+;
- 2) Оперативная память: 512 Мб;
- 3) Жёсткий диск: 150мб;
- 4) Операционная система: Windows 7/8/10.
- 5) Версия MySQL 5.0 и выше;

Минимальные системные требования для рабочей станции:

- 1) Процессор: Intel Pentium 4 2.0Ghz / AMD XP 2200+;
- 2) Оперативная память: 512 Мб;
- 3) Жёсткий диск: 150мб;
- 4) Операционная система: Windows 7/8/10.

4.5 Требования к документированию

Основным документом, регламентирующими использование информационной системы является руководство пользователя.

Основным документом, регламентирующими разработку информационной системы является техническое задание.