

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА»
(СПбГУТ)

АРХАНГЕЛЬСКИЙ КОЛЛЕДЖ ТЕЛЕКОММУНИКАЦИЙ
ИМ. Б.Л. РОЗИНГА (ФИЛИАЛ) СПбГУТ
(АКТ (ф) СПбГУТ)

КУРСОВОЙ ПРОЕКТ

НА ТЕМУ

РАЗРАБОТКА ПОДСИСТЕМЫ «БЕЛОЗОРИЕ.

РАСПИСАНИЕ СОТРУДНИКОВ СКЛАДА

Л109.25КП01.019 ПЗ

(Обозначение документа)

МДК.02.01 Технология разработки

программного обеспечения

Студент	ИСПП-21	08.12.2025	А.В. Мошников
	(Группа)	(Подпись)	(И.О. Фамилия)
Преподаватель		09.12.2025	Ю.С. Маломан
		(Подпись)	(И.О. Фамилия)

Архангельск 2025

СОДЕРЖАНИЕ

Перечень сокращений и обозначений.....	3
Введение	4
1 Сбор и анализ требований.....	6
1.1 Назначение и область применения	6
1.2 Постановка задачи	6
1.3 Выбор состава программных и технических средств	7
2 Проектирование программного обеспечения	9
2.1 Проектирование интерфейса пользователя.....	9
2.2 Разработка архитектуры программного обеспечения	10
2.3 Проектирование базы данных.....	11
3 Разработка и интеграция модулей программного обеспечения.....	12
3.1 Разработка программных модулей.....	12
3.2 Реализация интерфейса пользователя.....	12
3.3 Разграничение прав доступа пользователей	14
3.4 Экспорт данных	15
4 Тестирование и отладка программного обеспечения	18
4.1 Структурное тестирование	18
4.2 Функциональное тестирование.....	20
5 Инструкция по эксплуатации программного обеспечения.....	21
5.1 Установка программного обеспечения	21
5.2 Инструкция по работе.....	21
Заключение	29
Список использованных источников	30

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

В настоящем курсовом проекте применяются следующие сокращения и обозначения:

АО – акционерное общество

БД – база данных

ОС – операционная система

ПО – программное обеспечение

СУБД – система управления базами данных

ASP – активные серверные страницы

CSS – каскадные таблицы стилей

EFCore – фреймворк для доступа к данным

ERD – диаграмма «сущность-связь»

HTML – язык гипертекстовой разметки

IDE – интегрированная среда разработки

MVC – модель-представление-контроллер

SDK – комплект для разработки программного обеспечения

ВВЕДЕНИЕ

Веб-приложения или же веб-сайты стали неотъемлемой частью современной жизни и используются почти во всех областях промышленности, образования и показали себя как крайне эффективный способ передачи информации и создания каких-либо площадок, для взаимодействия с этой информацией.

Одним из главных преимуществ веб-приложений является их доступность. Также, веб-приложения, как правило, требуют меньших затрат на развертывание и поддержку, чем традиционные настольные приложения. Кроме того, веб-приложения позволяют интегрироваться с другими системами и сервисами, что расширяет функциональные возможности и упрощает рабочие процессы. К примеру, интеграция с системами учета товаров может автоматизировать процесс загрузки и выгрузки, значительно минимизируя количество ошибок при вводе данных.

Во многих компаниях необходимо решать проблему взаимодействия сотрудников и организации рабочего времени и веб-сайт с расписанием задач предприятия является очень хорошей альтернативой, стандартному бумажному расписанию, ведь предоставляют упрощённый процесс составления, представления и простоту в использовании. Кроме того, они позволяют компаниям оптимизировать свои внутренние процессы и улучшать взаимодействие между сотрудниками и руководством. Веб-приложение не только сокращает время, потраченное на выполнение задач, но и снижает вероятность ошибок, возникающих из-за ручного ввода данных.

Актуальность разрабатываемого курсового проекта заключается в ускорении и упрощении процесса составления и предоставления расписания загрузки и выгрузки товаров на складе. Веб-приложение обеспечит возможность быстрого внесения изменений в расписание, что особенно важно в динамичной рабочей среде.

Целью курсового проектирования является разработка веб-приложения «Белозорие. Расписание сотрудников склада», которое упростит и ускорит процесс составления расписания загрузки и выгрузки товаров на складе, а также позволит просматривать это расписание в удобном электронном формате и экспортировать это расписание в формате `xlsx`.

Для достижения поставленной цели требуется решить следующие задачи:

- проанализировать предметную область;
- определить требования к разрабатываемому программному средству;
- спроектировать структуру БД для хранения информации, необходимой для функционирования веб-приложения;
- разработать интуитивно понятный пользовательский интерфейс;
- реализовать серверную часть, обеспечивающую выполнение бизнес-логики приложения и разграничение прав доступа;
- провести тестирование разработанного программного продукта.

1 Сбор и анализ требований

1.1 Назначение и область применения

Веб-приложение предназначено для АО «Молоко». Приложение упростит создание расписания загрузки и выгрузки товаров на складе, доступ к этой информации, а также позволит экспортировать расписание. Разрабатываемое приложение будет актуально для работников и менеджеров склада, администраторов.

1.2 Постановка задачи

Необходимо разработать веб-приложение для составления и просмотра расписания загрузки и выгрузки. Разрабатываемое приложение предоставит доступ к следующей функциональности:

- авторизации;
- регистрации пользователей администратором;
- просмотру расписаний загрузки и выгрузки товаров;
- созданию расписаний загрузки и выгрузки товаров;
- экспорту расписаний загрузки и выгрузки товаров в формате `xlsx`;
- поиску сотрудников администратором по имени сотрудника;
- фильтрации списка сотрудников по должности;
- сортировки списка сотрудников по имени.

Интерфейс должен быть интуитивно понятен для пользователя с базовыми компьютерными навыками. В целях безопасности в приложении необходимо проходить обязательную авторизацию перед использованием приложения. При открытии веб-приложения отобразится страница авторизации, после авторизации определится роль пользователя.

Пользователь сможет просматривать расписания загрузки и выгрузки товаров на 7 дней.

Менеджер сможет просматривать расписания загрузки и выгрузки, создавать, удалять, изменять расписание.

Администратор сможет добавлять, удалять пользователей, изменять информацию о пользователе и сможет выполнять все действия менеджера.

На рисунке 1 представлена диаграмма вариантов использования приложения различными категориями пользователей.

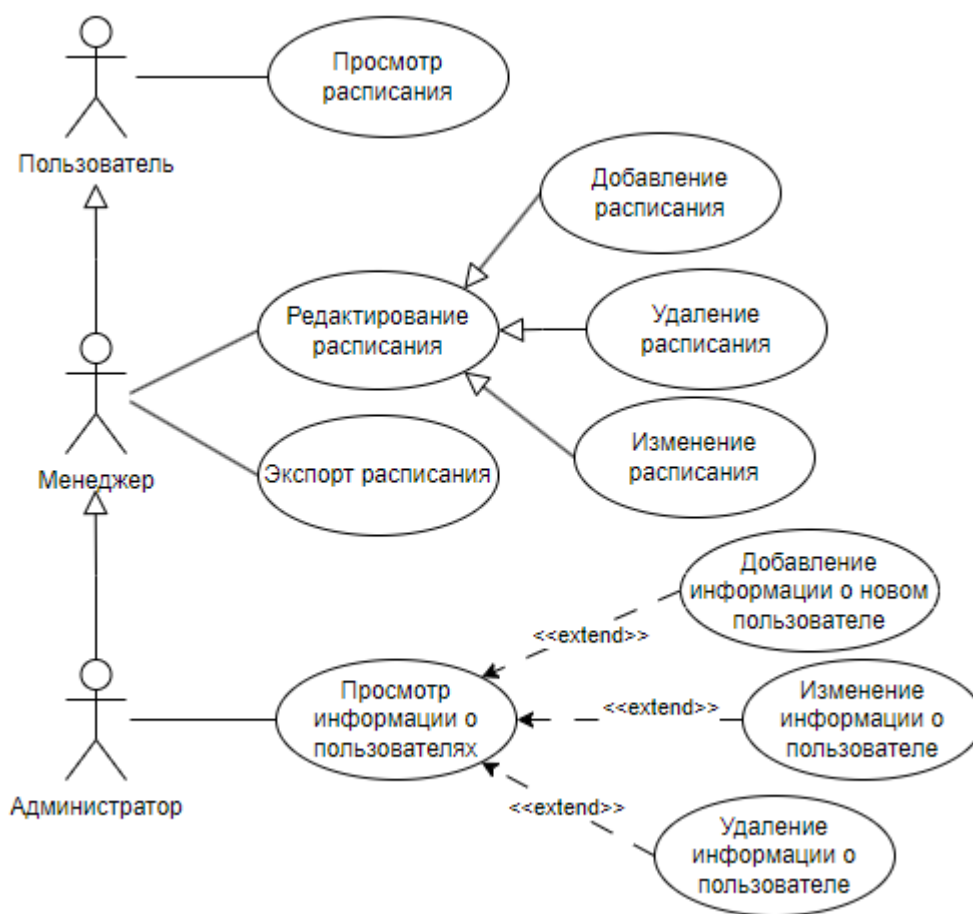


Рисунок 1 – Диаграмма вариантов использования

1.3 Выбор состава программных и технических средств

Согласно цели проекта требуется разработать веб-приложение для просмотра и редактирования информации и БД для хранения информации.

В качестве СУБД выбрана MySQL 8.0, так как она имеет высокую производительность, обладает кроссплатформенностью, проста в освоении, обладает широким набором данных и легко масштабируется.

Веб-приложение будет написано на языке программирования C# с использованием фреймворка ASP.NET MVC, так как он обеспечивает высокую производительность, кроссплатформенность, гибкую архитектуру и простоту тестирования.

Для разработки веб-приложения будет использоваться IDE Visual Studio 2022, так как эта IDE обеспечивает быструю и комфортную разработку благодаря поддержке Git и наличию функции горячей перезагрузки.

Для функционирования системы на стороне сервера необходимы следующие программные и технические средства:

- ОС Windows x86 64-бит или Linux x86 64-бит;
- MySQL Server не ниже 8.0;
- ПО для работы веб-приложения: .NET SDK не ниже 8.0 и ASP.NET MVC 5;
- доступная оперативная память 3 ГБ;
- процессор с частотой не менее 1 ГГц и не менее 2 ядер;
- минимальный объем дискового пространства 10 ГБ.

Для функционирования приложения на стороне клиента необходимы следующие программные и технические средства:

- браузер, поддерживающий HTML 5 и CSS 3;
- процессор с частотой 1 ГГц;
- доступная оперативная память 2 ГБ.

2 Проектирование программного обеспечения

2.1 Проектирование интерфейса пользователя

В рамках разработки веб-приложения «Белозорие. Расписание сотрудников склада» спроектирован интерфейс пользователя в виде wireframe при помощи сервиса draw.io. Эти визуальные представления позволяют наглядно увидеть структуру приложения, его основные элементы и функциональность.

Интерфейс страниц «Просмотра расписания», «Списка сотрудников» и «Регистрации сотрудника» представлены на рисунке 2.

The wireframe consists of three panels. The top-left panel, titled 'К работникам Расписание', contains a 'Список сотрудников' section with a search bar and a table with columns: 'Фамилия', 'Имя', 'Отчество', 'Номер телефона', 'Пароль', and 'Должность'. A red arrow points from the 'Зарегистрировать нового сотрудника' link to the top-right panel. The top-right panel, also titled 'К работникам Расписание', is for 'Регистрация нового сотрудника' and includes input fields for 'Фамилия', 'Имя', 'Отчество', 'Номер телефона', 'Пароль', and 'Должность', along with 'Зарегистрировать' and 'К списку сотрудников' buttons. A red arrow points from the 'Список сотрудников' table to the bottom panel. The bottom panel, titled 'К работникам Расписание', shows 'Расписание на неделю' with a table of orders. Each table header row contains 'Дата', a download icon, and a delete icon. The table body has columns for 'Время' and 'Название заказа'. Navigation links 'Предыдущая' and 'Следующая' are at the bottom.

К работникам Расписание

Список сотрудников [Зарегистрировать нового сотрудника](#)

Найти сотрудника по имени:

Фамилия	Имя	Отчество	Номер телефона	Пароль	Должность
---------	-----	----------	----------------	--------	-----------

К работникам Расписание

Регистрация нового сотрудника

Фамилия

Имя

Отчество

Номер телефона

Пароль

Должность

[К списку сотрудников](#)

К работникам Расписание

Расписание на неделю

Дата	Дата	Дата	Дата	Дата	Дата	Дата
Время Название заказа	Время Название заказа	Время Название заказа	Время Название заказа	Время Название заказа	Время Название заказа	Время Название заказа
Время Название заказа	Время Название заказа	Время Название заказа	Время Название заказа	Время Название заказа	Время Название заказа	Время Название заказа

[Предыдущая](#) [Следующая](#)

Рисунок 2 – draw.io.Wireframe страниц просмотра расписания, списка сотрудников и регистрации сотрудника

2.2 Разработка архитектуры программного обеспечения

Приложение предназначено для составления расписания для работы с ним. Архитектура приложения построена на основе клиент-серверной модели и включает в себя несколько ключевых компонентов: серверная часть приложения, клиентская часть приложения, БД. Диаграмма разворачивания компонентов представлена на рисунке 3.

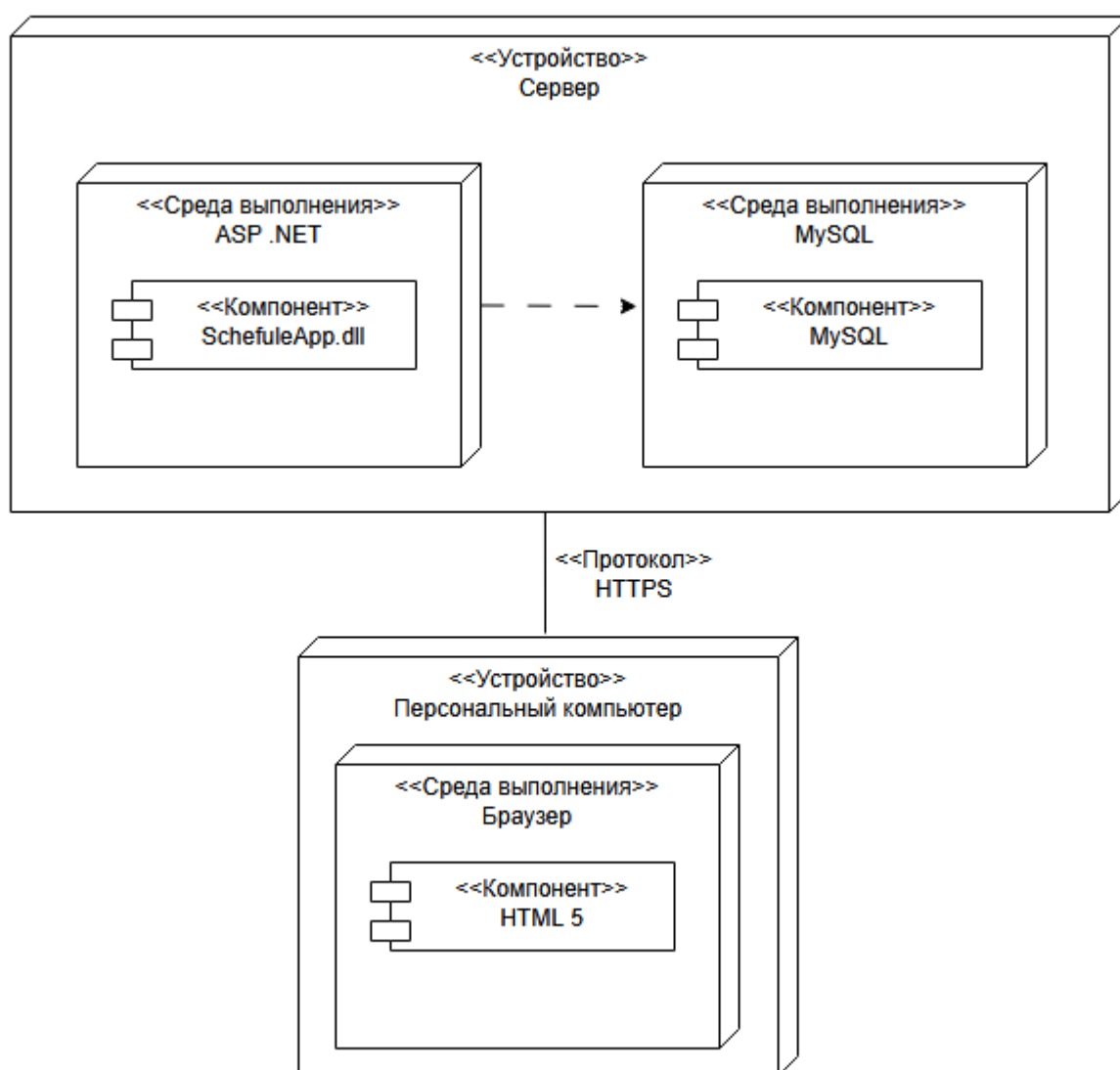


Рисунок 3 – Диаграмма разворачивания компонентов

2.3 Проектирование базы данных

Требуется разработать БД для хранения информации о расписании, продуктах и пользователях [3]. Модели БД созданы при помощи MySQL Workbench. На рисунке 4 в виде ERD показана физическая модель БД приложения.

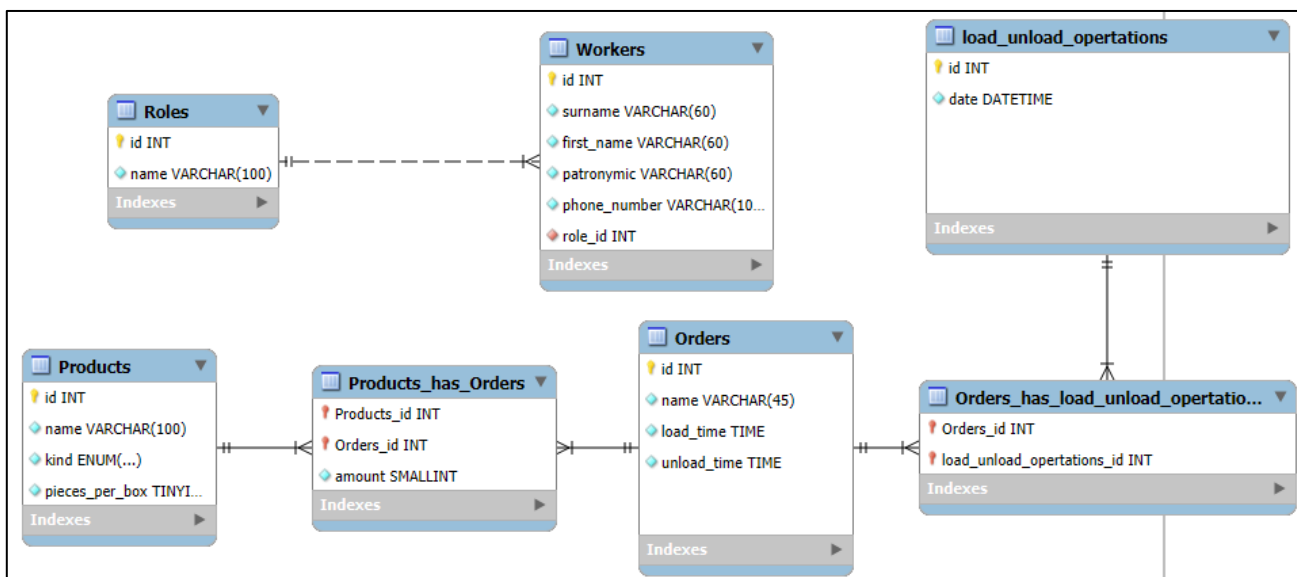


Рисунок 4 – MySQL Workbench. Физическая модель БД

3 Разработка и интеграция модулей программного обеспечения

3.1 Разработка программных модулей

Веб-приложение разработано на С# с использованием фреймворка ASP.NET MVC в Visual Studio 2022 [2] [4].

Взаимодействие веб-приложения с БД происходит через EFCore.

В веб-приложении созданы следующие контроллеры: AuthController (для управления авторизацией), ScheduleController (для работы с расписанием) и WorkerController (для работы с пользователями). Код метода получения информации о пользователе представлен листингом 1.

Листинг 1 – Код метода Details

```
public async Task<IActionResult> Details(int? id)
{
    if (id == null) //Проверка существования id
    {
        return NotFound();
    }

    var worker = await _context.Workers
        .Include(w => w.Role) //Поиск пользователя
        .FirstOrDefaultAsync(m => m.Id == id);
    if (worker == null)
    {
        return NotFound();
    }

    return View(worker);
}
```

3.2 Реализация интерфейса пользователя

Интерфейс приложения реализован с использованием Bootstrap для оформления страниц. Для вставки кода на С# на страницах использована

технология Razor Pages. Для отображения подробной информации расписания разработана страница Details, код верстки которой представлен листингом 2.

Листинг 2 – Код верстки страницы подробной информации

```
@model ScheduleApp.Models.Order <!--Модель данных-->

<h1>Подробности о заказе</h1>
<div>
    <dl class="row">
        <dt class="col-sm-2">@Html.DisplayFor(model =>
@model.Name)</dt>
        <dt class="col-sm-2">Начало в:
@model.LoadTime.ToString(@"hh\:mm")</dt> <!--Форматирование
времени-->
        <dt class="col-sm-2">Конец в:
@model.UnloadTime.ToString(@"hh\:mm")</dt>
    </dl>
    <hr />
    <dl class="row">
        <dt class="col-sm-2">
            Название
        </dt>
        <dt class="col-sm-2">
            Тип
        </dt>
        <dt class="col-sm-2">
            Количество
        </dt>
    </dl>
    @foreach(var product in Model.ProductsHasOrders)
    { <!--Отображение списка продуктов в заказе -->
        <dl class="row">
            <dd class="col-sm-2">
                @product.Products.Name
            </dd>
            <dd class="col-sm-2">
                @product.Products.Kind
            </dd>
            <dd class="col-sm-2">
                @product.Amount
            </dd>
        </dl>
    }
</div>
```

3.3 Разграничение прав доступа пользователей

В веб-приложении реализовано разграничение прав доступа пользователей посредством cookie. Данные пользователей хранятся в БД (таблица Users), роли пользователей хранятся в таблице Roles.

Код метода авторизации представлен листингом 3.

Листинг 3 – Код метода Login

```
[HttpPost]
public async Task<IActionResult> Login(LoginViewModel model,
string? returnUrl = null)
{
    var user = await _context.Workers.FirstOrDefaultAsync(emp =>
emp.PhoneNumber == model.Login); // Поиск пользователя по номеру
телефона
    if (user == null)
        return View();
    if (user.Password != model.Password) // Проверка пароля
        return View();
    if (user != null)
    {
        var claims = new List<Claim>
        {
            new Claim(ClaimTypes.Name, model.Login),
            new Claim("UserId", user.Id.ToString(),
ClaimValueTypes.Integer32)
        };
        var claimsIdentity = new ClaimsIdentity(claims,
CookieAuthenticationDefaults.AuthenticationScheme);
        var claimsPrincipal = new
ClaimsPrincipal(claimsIdentity);
        await
HttpContext.SignInAsync(CookieAuthenticationDefaults.Authenticat
ionScheme, claimsPrincipal);
        if (user.RoleId == 3) // Переход на страницу
пользователей, если администратор
            return Redirect(returnUrl ?? "/Workers/Index");
        return Redirect(returnUrl ?? "/Schedule/Index");
    }
    ModelState.AddModelError(string.Empty, "Неверные учетные
данные.");
    return View(model);
}
```

При переходе к редактированию заказа в контроллере `ScheduleController` выполняется получение необходимого заказа в методе `Edit`, код метода представлен листингом 4.

Листинг 4 – Код метода `Edit`

```
[Authorize(Roles = "Менеджер, Администратор")]
[HttpGet]
public ActionResult Edit(int id)
{
    var viewModel = new EditOrderViewModel();

    var order = _context.Orders.Include(o =>
o.ProductsHasOrders).ThenInclude(o =>
o.Products).FirstOrDefault(o => o.Id == id); // Поиск заказа

    if(order == null)// Проверка существования заказа
        return NotFound();

    //Формирование модели отображения для представления
    viewModel.Order = order;
    viewModel.Products = _context.Products.ToList();
    return View(viewModel);
}
```

3.4 Экспорт данных

В веб-приложении реализован экспорт расписания при помощи библиотеки `CloseXML`.

Код генерации документа с экспортируемым расписанием представлен листингом 5.

Листинг 5 – Код метода `ExportAsync`

```
[HttpPost, ActionName("ExportAsync")]
[ValidateAntiForgeryToken]
public async Task<ActionResult> ExportAsync(int id)
{
    var operation = await
    _context.LoadUnloadOperations.Include(l =>
l.Orders).ThenInclude(o => o.ProductsHasOrders)
```

```

        .ThenInclude(p => p.Products).FirstOrDefaultAsync(l =>
l.Id == id); // Получение данных о расписании

using (var workbook = new XLWorkbook())
{
    var worksheet = workbook.Worksheets.Add("Лист1");
    int row = 1;

    foreach (var order in operation.Orders)
    { // Заполнение ячеек заказами
        var headerStyle = worksheet.Cell(1, 1).Style;
        headerStyle.Font.Bold = true;
        headerStyle.Fill.BackgroundColor = XLColor.AshGrey;
        headerStyle.Alignment.Horizontal =
XLAAlignmentHorizontalValues.Center; // Настройка стиля

        worksheet.Cell(row, 1).Style = headerStyle;
        worksheet.Cell(row, 2).Style = headerStyle;
        worksheet.Cell(row, 3).Style = headerStyle;
        worksheet.Columns().AdjustToContents();

        worksheet.Cell(row, 1).Value = order.Name;
        worksheet.Cell(row, 2).Value = order.LoadTime;
        worksheet.Cell(row, 3).Value = order.UnloadTime;
        foreach (var product in order.ProductsHasOrders)
        { // Заполнение ячеек продуктами
            worksheet.Cell(row + 1, 1).Value =
product.Products.Name;
            worksheet.Cell(row + 1, 2).Value =
product.Products.Kind;
            worksheet.Cell(row + 1, 3).Value =
product.Amount;
            row++;
        }
        row++;
    }

    using (var stream = new MemoryStream())
    {
        workbook.SaveAs(stream);
        var content = stream.ToArray();
        var fileName = $"{operation.Date}.xlsx";
        return File(content,
"application/vnd.openxmlformats-
officedocument.spreadsheetml.sheet", fileName); // Сохранение
        файла xlsx
    }
}
}

```


На рисунке 5 представлен шаблон экспортированного файла в формате xlsx.

	A	B	C	D
1	Заказ 1	Время начала	Время конца	
2	Товар 1	Тип	кол-во	
3	Товар ...	Тип	кол-во	
4	Товар N	Тип	кол-во	
5	Заказ ...	Время начала	Время конца	
6	Товар 1	Тип	кол-во	
7	Товар ...	Тип	кол-во	
8	Товар N	Тип	кол-во	
9	Заказ N	Время начала	Время конца	
10	Товар 1	Тип	кол-во	
11	Товар ...	Тип	кол-во	
12	Товар N	Тип	кол-во	
13				

Рисунок 5 – Excel. Шаблон экспорта расписания в формате xlsx

4 Тестирование и отладка программного обеспечения

4.1 Структурное тестирование

Для проверки корректности работы поиска пользователей реализовано модульное тестирование [1] с использованием фреймворка xUnit.

Для выборки, сортировки и фильтрации данных о пользователях в `WorkersController` реализован метод `Index`. Для его проверки разработан тестовый сценарий `Index_ReturnsWithFilteredSearchedAndSortedWorkers` (представлен листингом 6).

Листинг 6 – Код метода `Index_ReturnsWithFilteredSearchedAndSortedWorkers`

```
[Fact]
public async Task
Index_ReturnsWithFilteredSearchedAndSortedWorkers()
{
    // Arrange
    // Создаем экземпляр контроллера WorkersController,
    // передавая контекст базы данных
    var controller = new WorkersController(_context);

    // Устанавливаем начальные параметры для фильтрации, поиска
    // и сортировки
    string filter = "Администратор";           // Фильтр для поиска
    string searchString = "Алиса";             // Строка для поиска
    string sortOrder = "name_desc";            // Параметр
    // сортировки по убыванию имени

    // Act
    // Получаем результат метода Index с фильтром по роли
    var filteredResult = await controller.Index(sortOrder, null,
    filter) as ViewResult;
    var filteredModel = filteredResult.Model as List<Worker>;

    // Получаем результат метода Index с поиском по имени
    var searchedResult = await controller.Index(null,
    searchString, null) as ViewResult;
    var searchedModel = searchedResult.Model as List<Worker>;
```

```

// Получаем результат метода Index с сортировкой по имени
var sortedResult = await controller.Index(sortOrder, null,
null) as ViewResult;
var sortedModel = sortedResult.Model as List<Worker>;

// Assert
// Проверяем, что результат фильтрации корректен
Assert.NotNull(filteredResult);
Assert.Equal(2, filteredModel.Count); // Ожидаем 2
работников с ролью "Администратор"
Assert.All(filteredModel, w => Assert.Equal("Администратор",
w.Role.Name)); // Проверяем, что все работники имеют роль
"Администратор"

// Проверяем, что результат поиска корректен
Assert.NotNull(searchedResult);
Assert.Single(searchedModel); // Ожидаем, что найдётся
только один работник по имени "Алиса"
Assert.Equal("Алиса", searchedModel[0].Name); // Проверяем,
что найденный работник – именно "Алиса"

// Проверяем, что результат сортировки корректен
Assert.NotNull(sortedResult);
Assert.Equal("Чарли", sortedModel[0].Name); // Ожидаем, что
первым будет "Чарли" после сортировки по убыванию
Assert.Equal("Дима", sortedModel[1].Name);
Assert.Equal("Витя", sortedModel[2].Name);
Assert.Equal("Алиса", sortedModel[3].Name); // Ожидаем, что
последним будет "Алиса"
}

```

Результат тестирования метода Index представлен на рисунке 6

Тестирование	Длительность	Признаки	Сообщение об ошибке
✓ ScheduleApp.Test (1)	5,6 с		
✓ ScheduleApp.Test (1)	5,6 с		
✓ WorkersControllerTests (1)	5,6 с		
✓ Index_ReturnsWithFilteredSearchedAndSorted...	5,6 с		

Рисунок 6 - Visual Studio 2022. Вид вкладки обозревателя тестов

4.2 Функциональное тестирование

В ходе функционального тестирования [5] веб-приложения проверена корректность работы ключевых пользовательских сценариев для авторизованного пользователя. Тестирование блока работы с информацией о пользователях на странице пользователей выполнено методом «чёрного ящика». Результаты тестирования представлены в таблице 1.

Т а б л и ц а 1 – Набор тестов страницы пользователей

Действие	Ожидаемый результат	Полученный результат
Ввод фамилии в поле поиска «Мошников» и нажатие кнопки «Поиск»	Отображение списка пользователей с фамилией «Мошников»	Совпадает с ожидаемым
Нажатие на иконку фильтра, выбор роль «Работник»	Отображение списка пользователей с ролью «Работник»	Совпадает с ожидаемым
Нажатие на «Имя»	Отображение отсортированного по убыванию списка пользователей по их имени	Совпадает с ожидаемым
Нажатие на ссылку «Редактировать»	Открытие окна редактирования информации о пользователе	Совпадает с ожидаемым
Нажатие на ссылку «Удалить»	Отображение всплывающего окна с подтверждением удаления пользователя	Совпадает с ожидаемым
Нажатие на «Да» в сплывающем окне при удалении	Пользователь удален и больше не отображается в списке	Совпадает с ожидаемым
Нажатие на «Да» в сплывающем окне при удалении	Пользователь отображается в списке пользователей	Совпадает с ожидаемым

По результатам тестирования можно сделать вывод, что разработанное приложение функционирует корректно и соответствует ожидаемому поведению.

5 Инструкция по эксплуатации программного обеспечения

5.1 Установка программного обеспечения

Для функционирования системы на стороне сервера необходимы следующие программные и технические средства:

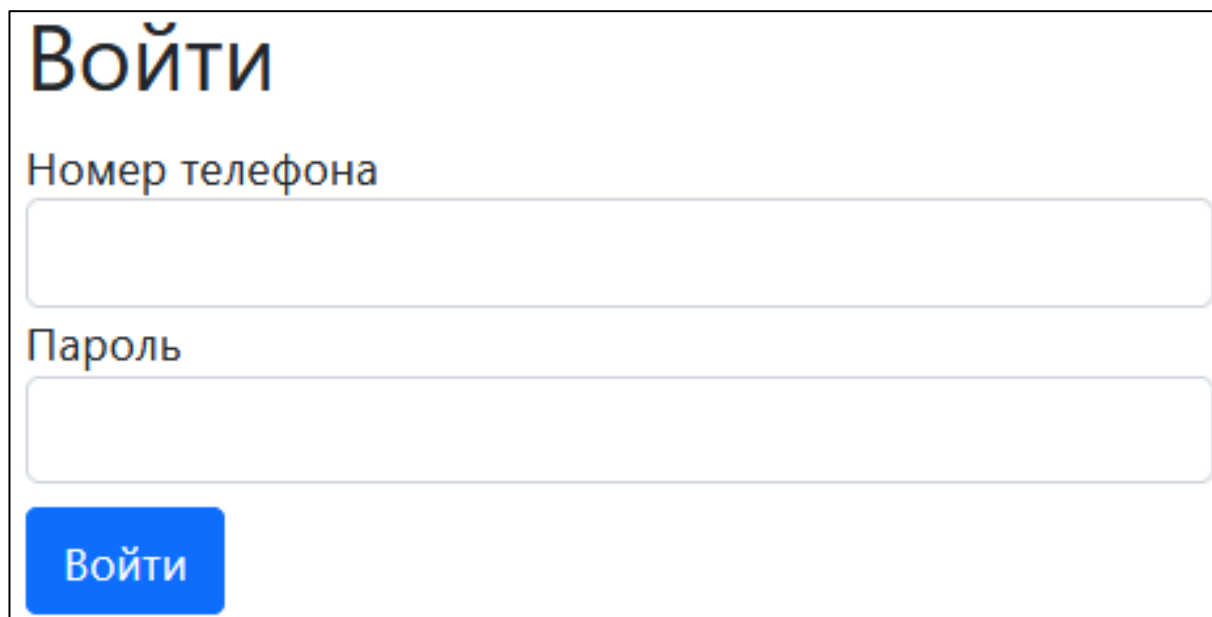
- ОС Windows x86 64-бит или Linux x86 64-бит;
- MySQL Server не ниже 8.0;
- ПО для работы сайта: .NET SDK не ниже 8.0 и ASP.NET Core 8.0;
- доступная оперативная память 3 ГБ;
- процессор с частотой не менее 1 ГГц и не менее 2 ядер;
- минимальный объем дискового пространства 10 ГБ.

Доступ к веб-приложению осуществляется с помощью браузеров, поддерживающих HTML5 и CSS3, необходимых для функционирования системы на стороне клиента. Также необходимы следующие технические средства:

- процессор с частотой 1 ГГц;
- доступная оперативная память 2 ГБ.

5.2 Инструкция по работе

При открытии веб-приложения пользователя встречает страница авторизации. На этой странице расположены поля ввода номера телефона и пароля пользователя. Для авторизации требуется ввести учетные данные пользователя и нажать кнопку «Войти». Вид формы авторизации представлен на рисунке 7. Регистрация в приложении не предусмотрена. Для регистрации необходимо обратиться к сотруднику, который назначен администратором приложения.



Войти

Номер телефона

Пароль

Войти

Рисунок 7 – ScheduleApp. Вид формы авторизации



























После успешной авторизации осуществляется перенаправление на страницу просмотра расписания. Пользователю с ролью «Администратор» доступна кнопка «К работникам». Интерфейс обновляется для авторизованного пользователя: становятся доступна кнопка «Выйти». Пример вида панели навигации представлен на рисунке 8.



ScheduleApp К работникам Расписание Выйти

Рисунок 8 – ScheduleApp. Вид панели навигации

На странице просмотра расписания присутствуют различные элементы управления: кнопки с пиктограммами для удаления, создания расписания и редактирования заказа, данные пиктограммы будут видны только пользователям с ролью «Администратор» и «Менеджер». Страница просмотра расписания представлена на рисунке 9.

Расписание на неделю							
01.11.2025  	02.11.2025  	03.11.2025  	04.11.2025  	10.12.2025  	11.12.2025  	12.12.2025  	
10:22 - 11:00 Заказ 1 	01:00 - 02:00 Заказ 1 	02:00 - 04:00 Заказ 1 	01:00 - 02:00 Заказ 1 	12:00 - 15:00 Заказ 1 	01:00 - 12:00 Заказ 1 	02:00 - 03:00 Заказ 1 	
12:30 - 13:30 Заказ 2 		04:00 - 05:00 Заказ 1 		14:05 - 17:00 Заказ 2 		03:00 - 06:00 Заказ 2 	
14:00 - 15:00 Заказ 3 				17:00 - 23:00 Заказ 3 			

[Следующая неделя](#)

Рисунок 9 – ScheduleApp. Вид страницы просмотра расписания

Для создания нового расписания необходимо нажать на кнопку с пиктограммой «Создание нового документа», после чего пользователь перенаправляется на страницу создания расписания, где есть поля ввода названия, даты заказа, времени начала и конца погрузки и есть функциональность для добавления товаров в заказ. Слева от полей ввода отображается предпросмотр создаваемого расписания. Страница создания представлена на рисунке 10.

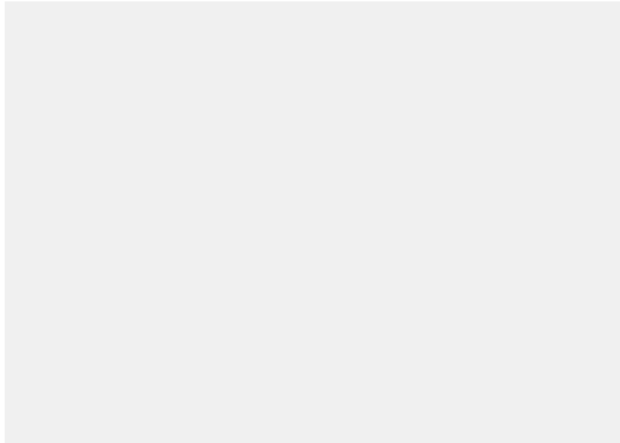




Создание заказа		Вернуться к расписанию		
			Дата заказа:	<input type="text" value="11.12.2025"/> 
			Название заказа:	<input type="text"/>
			Время начала погрузки:	<input type="text" value="00:00:00"/>
			Время конца погрузки:	<input type="text" value="00:00:00"/>
			Выберете продукт:	<input type="text" value="Молоко 2.5"/> 
Количество упаковок:	<input type="text" value="1"/>  			
			<input type="button" value="Добавить товар"/>	
			<input type="button" value="Сбросить заказ"/>	<input type="button" value="Создать заказ"/>

Рисунок 10 - ScheduleApp. Вид страницы создания заказа

Для редактирования заказа нужно нажать на кнопку с пиктограммой «Редактирование», после чего пользователь перенаправляется на соответствующую страницу, где можно изменить название заказа, время начала и конца погрузки и отредактировать список товаров в заказе. Страница редактирования заказа представлена на рисунке 11.

Редактирование заказа [Вернуться к расписанию](#)

Название заказа:
ваа

Начало погрузки:
12:00:00

Конец погрузки:
13:00:00

Сохранить

Список продуктов в заказе:
Кефир 2.5% - 4 шт. ✕

Выберете продукт: Молоко 2.5 Количество упаковок: 1

Добавить товар

Рисунок 11 – ScheduleApp. Вид страницы редактирования заказа

Для удаления расписания на день необходимо нажать на кнопку с пиктограммой «Мусорка», после чего пользователь увидит всплывающее окно, запрашивающее подтверждение этого действия, после нажатия «Да» расписание на день будет удалено. Также на странице просмотра расписания предусмотрена пагинация для просмотра расписания на 7 дней. Эта пагинация представлена в виде ссылок «Следующая неделя» и «Предыдущая неделя», при нажатии на которые будет происходить переход на следующую и предыдущую неделю соответственно.

Для просмотра подробной информации о заказе пользователю необходимо нажать на заказ, после чего пользователя перенаправит на нужную страницу. У пользователей с ролью «Администратор» и «Менеджер» на странице присутствует кнопка с пиктограммой «Мусорка» для удаления заказа. При нажатии на неё пользователь увидит всплывающее окно для

подтверждения его выбора. При нажатии на «Да» заказ будет удален, а пользователь будет перенаправлен на страницу просмотра расписания. Страница подробной информации представлена рисунком 12.


Подробности о заказе			Вернуться к расписанию 
Заказ 1	Начало в: 10:22	Конец в: 11:00	
Название	Тип	Количество	
Молоко 2.5	Термоупаковка	12	
Кефир 2.5%	Ящик тетра-крейт	5	
У-Кефир ОТБОРНЫЙ	Село Устьяны	2	

Рисунок 12 – ScheduleApp. Вид страницы дополнительной информации о заказе

Пользователь с ролью «Администратор» может выполнять администрирование пользователей. При нажатии на кнопку «К работникам» пользователь будет перенаправлен на страницу просмотра информации о пользователях. Страница просмотра информации о пользователях представлена рисунком 13.





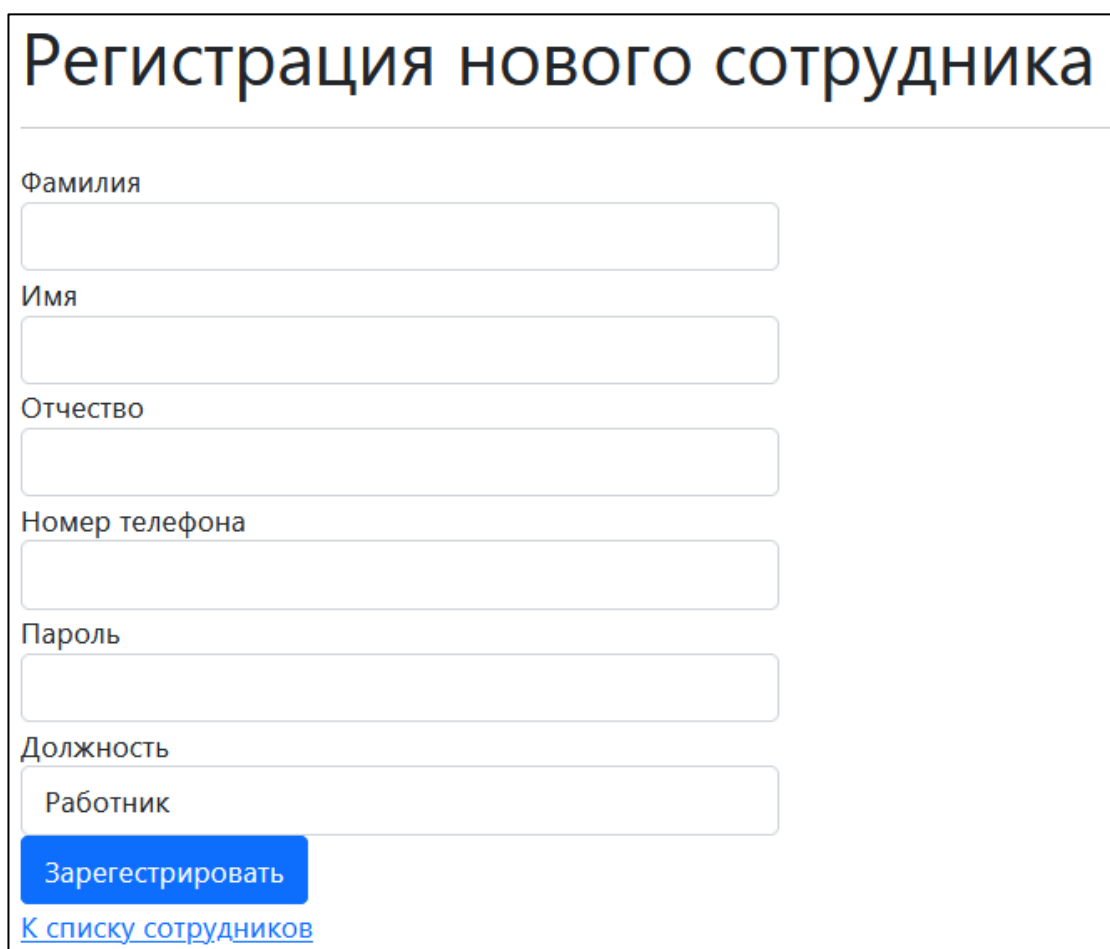
Список сотрудников						Зарегистрировать нового сотрудника
Найти сотрудника по фамилии:			<input type="text"/>	Поиск		
Фамилия	Имя	Отчество	Номер телефона	Пароль	Должность	
Мошников	Артём	Вячеславович	89600043011	*****	Менеджер	Редактировать 
Мошников	Вадим	Александрович	89600075566	*****	Работник	Редактировать 
Андерсон	Томас	Вадимович	89600075544	*****	Администратор	Редактировать 

Рисунок 13 – ScheduleApp. Вид страницы просмотра информации о пользователях

Для регистрации сотрудника необходимо нажать на ссылку «Зарегистрировать нового пользователя». При нажатии на ссылку пользователя перенаправит на соответствующую страницу, где есть поля ввода фамилии, имени, отчества, номера телефона и пароля. Должность сотрудника выбирается при помощи выпадающего списка. Страница регистрации сотрудника представлена рисунком 14.



Регистрация нового сотрудника

Фамилия

Имя

Отчество

Номер телефона

Пароль

Должность

Работник

Зарегистрировать

[К списку сотрудников](#)

Рисунок 14 – ScheduleApp. Вид страницы регистрации сотрудника

Для удаления сотрудника на странице присутствует кнопка с пиктограммой «Мусорка». При нажатии на неё пользователь увидит всплывающее окно для подтверждения его выбора. При нажатии на «Да» сотрудник будет удален.

Для редактирования информации о сотруднике присутствует ссылка «Редактировать». При нажатии на ссылку пользователя перенаправит на страницу редактирования информации о сотруднике. На странице присутствуют поля фамилии, имени, отчества, номера телефона и пароля. Смена должности сотрудника производится с помощью выбора необходимой роли в выпадающем списке. При нажатии на кнопку «Сохранить изменения» данные о сотруднике будут обновлены, а пользователя перенаправит на страницу списка сотрудников. Страница редактирования информации о сотруднике представлена рисунком 15.

Редактирование данных о сотруднике

Фамилия
Мошников

Имя
Артём

Отчество
Вячеславович

Номер телефона
89600043011

Пароль
12345

Должность
Менеджер

[Сохранить изменения](#)

[К списку сотрудников](#)

Рисунок 15 – ScheduleApp. Вид страницы редактирования информации о сотруднике

Помимо вышеперечисленных функций на странице просмотра сотрудников также присутствует фильтрация сотрудников по должности, поиск по фамилии, сортировка по имени.

Для фильтрации необходимо нажать на кнопку с пиктограммой «Фильтрация», после чего в боковом списке выбрать интересующую должность. После этого список сотрудников обновится и в списке будут только те сотрудники, у которых выбрана должность.

Для сортировки по имени необходимо нажать на ссылку «Имя», после чего список сотрудников обновится и будет отсортирован в обратном алфавитном порядке по имени сотрудника.

Для регистронезависимого поиска по фамилии сотрудника необходимо ввести фамилию сотрудника в поле поиска, далее необходимо нажать на кнопку «Поиск» и после выполненных действий в списке сотрудников будут находиться только те сотрудники, у которых есть набранная фамилия.

Так же на страницах редактирования и создания сотрудников присутствуют ссылки «К списку сотрудников», после нажатия на них пользователь будет перенаправлен на страницу со списком сотрудников.

ЗАКЛЮЧЕНИЕ

Целью курсового проектирования являлась разработка веб-приложения, предназначенного для упрощения составления, изменения и удаления расписания загрузки и выгрузки товаров на складе, доступа к этой информации и поддерживающего возможность экспорта этого расписания в формате `xlsx`.

Разработанное программное средство полностью соответствует поставленной цели. Созданное программное средство решает задачу по составлению, изменению и удалению расписания, а возможность просмотреть это расписание в электронном виде позволяет обеспечить высокую доступность информации.

Цель курсового проектирования достигнута, в процессе ее достижения решены следующие задачи:

- проанализирована предметная область;
- определены требования к разрабатываемому программному средству;
- спроектирована структура базы данных для хранения информации, необходимой для функционирования веб-приложения;
- разработан интуитивно понятный пользовательский интерфейс;
- реализована серверная часть, обеспечивающая выполнение бизнес-логики приложения и разграничение прав доступа;
- проведено тестирование разработанного программного продукта.

Таким образом, разработанное веб-приложение представляет собой эффективное и масштабируемое решение, способное существенно упростить создание и предоставление расписания загрузки и выгрузки товаров склада.

В ходе работы закрепились профессиональные навыки проектирования информационных систем и освоены современные методы разработки веб-приложений с использованием паттерна MVC, технологии Bootstrap и с использованием CSS. Также были освоены современные методы экспорта данных с использованием библиотеки `CloseXML`.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Бек, К. Экстремальное программирование: разработка через тестирование. – Санкт-Петербург : Питер, 2021. – 224 с. – URL: <https://ibooks.ru/bookshelf/376974/reading> (дата обращения: 27.11.2025). – Режим доступа: для зарегистрир. пользователей. – Текст : электронный.

2. Гагарина, Л. Г. Технология разработки программного обеспечения : учебное пособие / Л. Г. Гагарина, Е. В. Кокорева, Б. Д. Сидорова-Виснадул ; под ред. Л. Г. Гагариной. – Москва : ФОРУМ : ИНФРА-М, 2025. – 400 с. – URL: <https://znanium.ru/catalog/product/2178802> (дата обращения: 10.11.2025). – Режим доступа: по подписке. – Текст : электронный.

3. Мартишин, С. А. Базы данных. Практическое применение СУБД SQL и NoSQL-типа для проектирования информационных систем : учебное пособие / С. А. Мартишин, В. Л. Симонов, М. В. Храпченко. – Москва : ФОРУМ : ИНФРА-М, 2024. – 368 с. – URL: <https://znanium.ru/catalog/product/2096940> (дата обращения: 3.11.2025). – Режим доступа: по подписке. – Текст : электронный.

4. Тидвелл, Д. Разработка интерфейсов. Паттерны проектирования. 3-е изд. – Санкт-Петербург : Питер, 2022. – 560 с. – URL: <https://ibooks.ru/bookshelf/386796/reading> (дата обращения: 10.11.2025). – Режим доступа: для зарегистрир. пользователей. – Текст : электронный.

5. Федорова, Г. Н. Разработка, внедрение и адаптация программного обеспечения отраслевой направленности : учебное пособие. — Москва : КУРС : ИНФРА-М, 2024. — 336 с. – URL: <https://znanium.ru/catalog/product/2083407> (дата обращения: 15.11.2025). – Режим доступа: по подписке. – Текст : электронный.