ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САРАТОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ ГАГАРИНА Ю.А.»

Кафедра «Естественные и математические науки»

Специальность ПИНЖ — Программная инженерия

Текст программы Информационной системы «Магазин программных продуктов»

Выполнил: студент X курса
учебной группы XXXX
очной формы обучения
XXXXXXX

Проверил: преподаватель кафедры
ЕМН XXXXXXX

Энгельс 2025

СОДЕРЖАНИЕ

Код – MySQL

```
-- 1. Создание базы данных
DROP DATABASE IF EXISTS SoftKeyDB;
CREATE DATABASE SoftKeyDB;
USE SoftKeyDB;

-- 2. Таблица: Роли (для разделения Админа и Клиента)
CREATE TABLE Роли (
    ID_Роли INT PRIMARY KEY AUTO_INCREMENT,
    Название VARCHAR(50) NOT NULL
);

-- 3. Таблица: Пользователи
CREATE TABLE Пользователи (
    ID_Пользователя INT PRIMARY KEY AUTO_INCREMENT,

    Фамилия VARCHAR(50) NOT NULL,
    Имя VARCHAR(50),
    Отчество VARCHAR(50),

    Телефон VARCHAR(20) UNIQUE,
    Дата_рождения DATE,
    Логин VARCHAR(50) NOT NULL UNIQUE,
    Пароль VARCHAR(255) NOT NULL,
    ID_Роли INT,

    Дата_регистрации TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (ID_Роли) REFERENCES Роли(ID_Роли)
);

-- 4. Таблица: Категории товаров
CREATE TABLE Категории (
    ID_Категории INT PRIMARY KEY AUTO_INCREMENT,
    Название_категории VARCHAR(100) NOT NULL
```

```
);

-- 5. Таблица: Товары (Витрина)
CREATE TABLE Товары (
    ID_Товара INT PRIMARY KEY AUTO_INCREMENT,
    Название VARCHAR(255) NOT NULL,
    Описание TEXT,
    Цена DECIMAL(10, 2) NOT NULL,
    ID_Категории INT,
    Изображение VARCHAR(255) DEFAULT 'default.jpg',
    FOREIGN KEY (ID_Категории) REFERENCES Категории(ID_Категории)
);

-- 6. Таблица: Корзина (Временные данные)
CREATE TABLE Корзина (
    ID_Корзины INT PRIMARY KEY AUTO_INCREMENT,
    ID_Пользователя INT,
    ID_Товара INT,
    Количество INT DEFAULT 1,
    FOREIGN KEY (ID_Пользователя) REFERENCES Пользователи(ID_Пользователя),
    FOREIGN KEY (ID_Товара) REFERENCES Товары(ID_Товара)
);

-- 7. Таблица: Заказы (Общая информация о покупке)
CREATE TABLE Заказы (
    ID_Заказа INT PRIMARY KEY AUTO_INCREMENT,
    ID_Пользователя INT,
    Дата_заказа DATETIME DEFAULT CURRENT_TIMESTAMP,
    Статус VARCHAR(50) DEFAULT 'Новый',
    Итоговая_сумма DECIMAL(10, 2),
    FOREIGN KEY (ID_Пользователя) REFERENCES Пользователи(ID_Пользователя)
);

-- 8. Таблица: Состав_заказа (Здесь фиксируется цена и срок на момент покупки)
CREATE TABLE Состав_заказа (
```

```
    ID_Позиции INT PRIMARY KEY AUTO_INCREMENT,
    ID_Заказа INT,
    ID_Товара INT,
    Цена_продажи DECIMAL(10, 2) NOT NULL, -- Цена может измениться в магазине, но в заказе она останется прежней
    Срок_лицензии_дни INT NOT NULL,       -- Например: 30, 365 или 9999 (бессрочно)
    Количество INT NOT NULL,
    FOREIGN KEY (ID_Заказа) REFERENCES Заказы(ID_Заказа),
    FOREIGN KEY (ID_Товара) REFERENCES Товары(ID_Товара)
);

-- 9. Таблица: Лицензии (Результат покупки — ключи)
CREATE TABLE Лицензии (
    ID_Лицензии INT PRIMARY KEY AUTO_INCREMENT,
    ID_Позиции_заказа INT,
    Лицензионный_ключ VARCHAR(100) NOT NULL,
    Дата_активации DATETIME,
    Дата_истечения DATETIME,
    FOREIGN KEY (ID_Позиции_заказа) REFERENCES Состав_заказа (ID_Позиции)
);

-- =========================================================
-- (Триггеры и Процедуры)
-- =========================================================

-- Триггер для расчета итоговой суммы заказа
DELIMITER //

CREATE TRIGGER Расчет_итоговой_суммы
BEFORE INSERT ON Заказы
FOR EACH ROW
BEGIN
    DECLARE v_сумма DECIMAL(10,2);

    -- Если сумма не указана
```

```
    IF NEW.Итоговая_сумма IS NULL THEN
        SET NEW.Итоговая_сумма = 0.00;
    END IF;

    -- Проверяем, что сумма не отрицательная
    IF NEW.Итоговая_сумма < 0 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Итоговая сумма не может быть отрицательной';
    END IF;
END //

DELIMITER ;

-- ПРОЦЕДУРА: Оформление заказа из корзины
-- Эта процедура переносит всё из корзины в заказ и очищает корзину
DELIMITER //
CREATE PROCEDURE ОформитьЗаказ(IN p_ID_Пользователя INT, IN p_Срок INT)
BEGIN
    DECLARE v_ID_Заказа INT;
    DECLARE v_Total DECIMAL(10,2);

    -- Считаем общую сумму в корзине
    SELECT SUM(Т.Цена * К.Количество) INTO v_Total
    FROM Корзина К JOIN Товары Т ON К.ID_Товара = Т.ID_Товара
    WHERE К.ID_Пользователя = p_ID_Пользователя;

    -- Создаем запись в Заказах
    INSERT INTO Заказы (ID_Пользователя, Итоговая_сумма, Статус)
    VALUES (p_ID_Пользователя, v_Total, 'Оплачен');

    SET v_ID_Заказа = LAST_INSERT_ID();

    -- Переносим товары из корзины в Состав_заказа
    INSERT INTO Состав_заказа (ID_Заказа, ID_Товара, Цена_продажи,
Срок_лицензии_дни, Количество)
```

```
    SELECT v_ID_Заказа, ID_Товара,
        (SELECT Цена FROM Товары WHERE ID_Товара = Корзина.ID_Товара),
        p_Срок, Количество
    FROM Корзина WHERE ID_Пользователя = p_ID_Пользователя;

    -- Очищаем корзину
    DELETE FROM Корзина WHERE ID_Пользователя = p_ID_Пользователя;
END //
DELIMITER ;

-- ============================================================
-- Тестовые данные
-- ============================================================

-- 1. Заполняем Роли
INSERT INTO Роли (Название) VALUES ('Администратор'), ('Клиент');

-- 2. Заполняем Пользователей (с учетом твоих исправлений)
-- Пароли в учебных проектах обычно хранят в открытом виде или простом хеше
INSERT INTO Пользователи (Фамилия, Имя, Отчество, Логин, Пароль, ID_Роли) VALUES
('Иванов', 'Иван', 'Иванович', 'admin@key.ru', 'admin123', 1),
('Петров', 'Алексей', 'Сергеевич', 'petrov_as@key.ru', 'password55', 2),
('Сидорова', 'Мария', 'Павловна', 'masha_99@key.ru', '123456', 2);

-- 3. Заполняем Категории
INSERT INTO Категории (Название_категории) VALUES
('Операционные системы'),
('Офисные программы'),
('Антивирусы'),
('Графические редакторы');

-- 4. Заполняем Товары
INSERT INTO Товары (Название, Описание, Цена, ID_Категории) VALUES
('Windows 11 Pro', 'Windows 11 Pro — это современная операционная система для профессионального использования с улучшенным интерфейсом и расширенными
```

функциями безопасности. Система предлагает переработанное меню "Пуск", центрированную панель задач, виджеты для быстрого доступа к информации и улучшенную поддержку многозадачности с функциями Snap Layouts и Snap Groups.

Ключевые особенности:
• Повышенная безопасность с аппаратной изоляцией, безопасной загрузкой и Microsoft Defender
• DirectStorage для ускорения загрузки игр
• Auto HDR для улучшения качества изображения в играх
• Виртуальные рабочие столы для организации различных рабочих процессов
• Нативная поддержка приложений Android через Amazon Appstore
• Улучшенная интеграция с Microsoft Teams

Требования к системе:
Процессор: 1 ГГц или быстрее с 2+ ядрами, 64-битный
ОЗУ: 4 ГБ и выше
Хранилище: 64 ГБ и выше
Видеокарта: DirectX 12 совместимая с WDDM 2.0 драйвером
Безопасная загрузка и TPM 2.0', 15000.00, 1),

('Microsoft Office 2021', 'Microsoft Office 2021 Professional помогает создать единую среду для бизнеса и коммуникации, повысить производительность и эффективность работы за счет облачных технологий. Пользователи могут совместно создавать, редактировать, просматривать документы и вносить правки с любого устройства.

Офис профессиональный включает приложения версии для дома и бизнеса и дополнен новыми программами. В комплектацию входят:
• Word – текстовый редактор с улучшенными функциями совместной работы
• Excel – программа для работы с таблицами с новыми функциями анализа данных
• PowerPoint для подготовки интерактивных презентаций
• OneNote для создания цифровых заметок и организации информации
• Outlook – менеджер-органайзер с функциями почтового клиента и календаря
• Publisher – программа для создания качественных публикаций и маркетинговых материалов
• Access – приложение для работы с базами данных

Новые функции Office 2021:

• Динамические массивы в Excel

• XLOOKUP функция в Excel

• Улучшенная темизация в PowerPoint

• Функция "Диктор" в Word

• Интеграция с Microsoft Teams

• Локальное сохранение данных (без обязательной облачной подписки)

Требования к системе:

Процессор: 2 ГГц и выше

Операционная система: Windows 10/11

Места на диске: 4 ГБ

Оперативная память: 4 ГБ и выше

Видеопамять: 4 ГБ и выше

DirectX: 10', 12000.00, 2),

('Kaspersky Total Security', 'Kaspersky Total Security — комплексное решение для защиты на 1 год для 2 устройств, обеспечивающее безопасность от всех типов угроз в интернете и офлайн. Продукт сочетает в себе антивирусную защиту, файервол, защиту от фишинга и дополнительные функции для обеспечения приватности.

Основные функции:

• Защита от вирусов, троянов, шпионского ПО и ransomware

• Файервол для контроля сетевой активности

• Защита онлайн-платежей и банковских операций

• Менеджер паролей для безопасного хранения учетных данных

• Родительский контроль с настройкой времени использования

• Защита веб-камеры от несанкционированного доступа

• VPN с ограниченным трафиком (200 МБ/день)

• Резервное копирование важных файлов в облако

Преимущества:

• Многоуровневая защита без замедления работы системы

• Проактивная защита от новых угроз

• Автоматическое обновление баз данных
• Круглосуточная техническая поддержка
• Совместимость с Windows, macOS, Android и iOS

Системные требования для Windows:
ОС: Windows 7/8/8.1/10/11 (32/64-bit)
ОЗУ: 1 ГБ (32-bit) или 2 ГБ (64-bit)
Свободное место: 1.5 ГБ
Интернет: для активации и обновлений', 1800.00, 3),

('Adobe Photoshop', 'Adobe Photoshop — профессиональный графический редактор и подписка на облачный сервис для работы с растровой графикой. Индустриальный стандарт для фотографов, дизайнеров, иллюстраторов и специалистов по обработке изображений.

Возможности:
• Профессиональная ретушь фотографий и цветокоррекция
• Работа со слоями, масками и каналами
• Интеллектуальные инструменты выделения (Select Subject, Object Selection)
• Искусственный интеллект Adobe Sensei для автоматизации задач
• 3D-моделирование и текстурирование
• Анимация и создание GIF
• Поддержка RAW-форматов фотокамер
• Интеграция с другими приложениями Adobe Creative Cloud

Новые функции последней версии:
• Neural Filters для AI-обработки фотографий
• Sky Replacement для автоматической замены неба
• Enhance Portrait для улучшения портретов
• Pattern Preview для создания бесшовных текстур
• Облачные документы для работы на разных устройствах
• Прямая интеграция с Adobe Stock

Требования к системе:
Процессор: Intel или AMD с поддержкой 64-bit
ОС: Windows 10 (версия 1909 или новее)

ОЗУ: 8 ГБ (рекомендуется 16 ГБ)
Видеокарта: DirectX 12, 2 ГБ видеопамяти
Свободное место: 4 ГБ для установки
Разрешение монитора: 1280x800 (рекомендуется 1920x1080)', 35000.00, 4),

('Dr.Web Desktop Security', 'Dr.Web Desktop Security — антивирус для дома на 6 месяцев, обеспечивающий комплексную защиту от вредоносного ПО, шпионских программ, рекламного ПО и сетевых угроз. Российская разработка с более чем 25-летним опытом в области кибербезопасности.

Основные компоненты:
• Антивирусный сканер с ежедневными обновлениями
• Проактивная защита от неизвестных угроз
• Файервол для контроля сетевых подключений
• Антиспам фильтр для почтовых клиентов
• Родительский контроль с ограничением времени
• Защита от фишинговых сайтов
• Предотвращение краж данных
• Мониторинг процессов в реальном времени

Уникальные технологии Dr.Web:
• Origins Tracing™ для лечения зараженных файлов
• Fly-Code для распаковки сложных упаковщиков
• SelfProtect для защиты от выключения антивируса
• Нативная поддержка русского языка
• Минимальное потребление системных ресурсов

Системные требования:
ОС: Windows 7/8/8.1/10/11 (32/64-bit)
Процессор: 1 ГГц или выше
ОЗУ: 512 МБ (32-bit) или 1 ГБ (64-bit)
Свободное место: 650 МБ
Интернет-соединение: для активации и обновлений
Дополнительно: мышь, клавиатура, CD/DVD привод (для коробочной версии)', 900.00, 3);

```
-- 5. Тестовая Корзина (Петров положил товар, но еще не купил)
INSERT INTO Корзина (ID_Пользователя, ID_Товара, Количество) VALUES
(2, 3, 1), -- Петров хочет Касперский
(2, 1, 1); -- Петров хочет Windows

-- 6. Оформленный Заказ (Сидорова уже совершила покупку)
-- Сначала создаем "шапку" заказа
INSERT INTO Заказы (ID_Пользователя, Дата_заказа, Статус, Итоговая_сумма) VALUES
(3, '2024-05-20 14:30:00', 'Оплачен', 12900.00);

-- 7. Состав заказа для Сидоровой (купила Office и Dr.Webзаказы)
-- Предположим, ID заказа получился = 1
INSERT INTO Состав_заказа (ID_Заказа, ID_Товара, Цена_продажи, Срок_лицензии_дни,
Количество) VALUES
(1, 2, 12000.00, 9999, 1), -- Office (бессрочно)
(1, 5, 900.00, 180, 1);    -- Dr.Web (на 6 месяцев)

-- 8. Генерация ключей для купленных лицензий
-- Привязываем к позициям из Состава_заказа (ID 1 и 2)
INSERT INTO Лицензии (ID_Позиции_заказа, Лицензионный_ключ, Дата_активации,
Дата_истечения) VALUES
(1, 'OFF-2021-XXXX-YYYY-ZZZZ', '2024-05-20 15:00:00', '2099-12-31 23:59:59'),
(2, 'DRWEB-6MO-AAAA-BBBB-CCCC', '2024-05-20 15:00:00', '2024-11-20 15:00:00');
```

Код – base.html

```
<!DOCTYPE html>
<html lang="ru">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>{% block title %}SoftKey{% endblock %}</title>
```

```
    <script src="https://cdn.tailwindcss.com"></script>
    <script>
        tailwind.config = {
            theme: {
                extend: {
                    colors: {
                        primary: '#3B82F6', // Синий
                        secondary: '#10B981', // Зеленый
                    },
                    fontSize: {
                        'heading': '34px',
                        'subheading': '21px',
                        'base': '14px',
                        'description': '10px'
                    }
                }
            }
        }
    </script>

    <link  href="https://fonts.googleapis.com/css2?family=Pacifico&display=swap" rel="stylesheet">
    <link href="https://cdnjs.cloudflare.com/ajax/libs/remixicon/4.6.0/remixicon.min.css" rel="stylesheet">

    <style>
        body {
            font-family: ui-sans-serif, system-ui, -apple-system, BlinkMacSystemFont, "Segoe UI", Roboto, "Helvetica Neue", Arial, sans-serif;
```

```
        }

        /* Логотип в левом верхнем углу */
        .logo-box {
            width: 192px;
            height: 64px;
        }

        /* Кнопки высотой 32px и центрирование (96-32)/2 = 32px сверху */
        .nav-btn {
            height: 32px;
            margin-top: 16px;
            display: flex;
            align-items: center;
            justify-content: center;
            padding: 0 16px;
            border-radius: 6px;
            transition: all 0.2s;
        }
    </style>
    {% block head_extra %}{% endblock %}
</head>

<body class="bg-white text-base min-h-screen flex flex-col">

    <header class="w-full h-[64px] bg-white border-b border-gray-100 shadow-sm z-50">
        <div class="mx-auto h-full flex justify-between relative">
```

```
<a href="{{ url_for('index') }}" class="logo-box flex items-center px-5 bg-primary text-white">
    <i class="ri-key-2-line text-heading mr-2"></i>
    <span style="font-family: 'Pacifico', cursive;" class="text-subheading">SoftKey</span>
</a>

<nav class="flex items-start pr-[20px] gap-[20px]">
    <a href="{{ url_for('index') }}" class="nav-btn text-gray-600 hover:bg-gray-100">
        <i class="ri-home-4-line mr-1 text-[18px]"></i> Главная
    </a>

    {% if session.get('user_id') %}
    <a href="/orders" class="nav-btn text-gray-600 hover:bg-gray-100">
        <i class="ri-history-line mr-1"></i> Заказы
    </a>

    <a href="/cart" class="nav-btn text-gray-600 hover:bg-gray-100">
        <i class="ri-shopping-cart-2-line mr-1"></i> Корзина
    </a>

    {% if session.get('role_id') == 1 %}
    <a href="{{ url_for('admin_dashboard') }}"
        class="nav-btn text-gray-600 hover:bg-gray-100">
        <i class="ri-admin-line"></i> Панель управления
    </a>
    {% endif %}
```

```
            <a href="/profile" class="nav-btn bg-primary text-white hover:bg-blue-600">
                <i class="ri-user-3-line mr-1"></i> {{ session.get('user_name', 'Профиль') }}
            </a>
            {% else %}
            <a href="/login" class="nav-btn border border-primary text-primary hover:bg-blue-50">
                <i class="ri-login-box-line mr-1"></i> Войти
            </a>
            {% endif %}
        </nav>
    </div>
</header>

<main class="flex-grow w-full max-w-[1440px] mx-auto bg-white shadow-sm">
    {% with messages = get_flashed_messages(with_categories=true) %}
    {% if messages %}
    <div class="fixed top-[106px] right-5 z-[100] space-y-2">
        {% for category, message in messages %}
        <div
            class="p-4 rounded-lg shadow-lg border-l-4 flex items-center gap-3 animate-slide-in
                    {% if category == 'error' %} bg-red-50 border-red-500 text-red-700 {% else %} bg-green-50 border-green-500 text-green-700 {% endif %}">
            <i
                class="{% if category == 'error' %} ri-error-warning-line {% else %} ri-checkbox-circle-line {% endif %} text-subheading"></i>
            <span class="text-base font-medium">{{ message }}</span>
        </div>
```

```
        {% endfor %}
      </div>
      {% endif %}
      {% endwith %}

      {% block content %}{% endblock %}
    </main>

    <footer class="w-full h-[64px] bg-gray-900 text-white border-t border-gray-800">
      <div class="mx-auto h-full flex items-center justify-between px-5">
        <p class="text-description opacity-60">© 2025 SoftKey Store. Курсовой проект БД.</p>
        <div class="flex gap-4 text-description uppercase tracking-wider font-bold">
          <a href="#" class="hover:text-primary transition">Поддержка</a>
          <a href="#" class="hover:text-primary transition">Контакты</a>
        </div>
      </div>
    </footer>

    <style>
      @keyframes slide-in {
        from {
          transform: translateX(100%);
          opacity: 0;
        }

        to {
          transform: translateX(0);
```

```
            opacity: 1;
        }
    }

    .animate-slide-in {
        animation: slide-in 0.3s ease-out forwards;
    }
</style>

{% block scripts %}{% endblock %}
</body>

</html>
```

Код – admin.html

```
{% extends "base.html" %}
{% block title %}Админ-панель | SoftKey{% endblock %}

{% block content %}
<div class="flex justify-center bg-gray-0 min-h-screen py-10">
    <div class="w-[1196px]">

        <div class="flex justify-between items-center mb-10">
            <div>
                <h1 class="text-heading font-bold text-gray-900">Управление магазином</h1>
                <p class="text-base text-gray-500">Добро пожаловать в панель администратора</p>
            </div>
            <button onclick="openModal('productModal')"
```

```
            class="bg-primary text-white px-6 py-3 rounded-xl font-bold flex items-center gap-2 hover:bg-blue-600 transition">
                <i class="ri-add-circle-line"></i> ДОБАВИТЬ ТОВАР
            </button>
        </div>

        <div class="grid grid-cols-4 gap-6 mb-10">
            <div class="bg-white p-6 rounded-2xl border border-gray-100 shadow-sm">
                <p class="text-description font-bold text-gray-400 uppercase">Общая выручка</p>
                <h3 class="text-subheading font-bold text-gray-900">{{ stats.revenue|round|int }} ₽</h3>
            </div>
            <div class="bg-white p-6 rounded-2xl border border-gray-100 shadow-sm">
                <p class="text-description font-bold text-gray-400 uppercase">Всего заказов</p>
                <h3 class="text-subheading font-bold text-gray-900">{{ stats.orders_count }}</h3>
            </div>
            <div class="bg-white p-6 rounded-2xl border border-gray-100 shadow-sm">
                <p class="text-description font-bold text-gray-400 uppercase">Товаров в базе</p>
                <h3 class="text-subheading font-bold text-gray-900">{{ stats.products_count }}</h3>
            </div>
            <div class="bg-white p-6 rounded-2xl border border-gray-100 shadow-sm">
                <p class="text-description font-bold text-gray-400 uppercase">Клиентов</p>
                <h3 class="text-subheading font-bold text-gray-900">{{ stats.users_count }}</h3>
```

```
        </div>
    </div>

    <div class="bg-white rounded-3xl border border-gray-100 shadow-sm overflow-hidden">
        <div class="flex border-b border-gray-100">
            <button onclick="switchTab('products')" id="tab-products-btn" class="admin-tab active">Товары</button>
            <button onclick="switchTab('orders')" id="tab-orders-btn" class="admin-tab">Заказы</button>
            <button onclick="switchTab('reports')" id="tab-reports-btn" class="admin-tab text-secondary">Аналитика и
                отчеты</button>
        </div>

        <div id="tab-reports" class="p-8 hidden bg-gray-50/30">
            <div class="grid grid-cols-2 gap-8">

                <div class="bg-white p-6 rounded-2xl border border-gray-100 shadow-sm">
                    <h3 class="text-base font-bold text-gray-900 mb-6 flex items-center gap-2">
                        <i class="ri-fire-line text-orange-500"></i> Популярные товары (ТОП-5)
                    </h3>
                    <div class="space-y-4">
                        {% for item in reports.top_products %}
                        <div class="flex justify-between items-center">
                            <span class="text-base text-gray-700 truncate w-48">{{ item.Название }}</span>
```

```
                    <div class="flex items-center gap-3">
                        <div class="w-32 h-2 bg-gray-100 rounded-full overflow-hidden">
                            <div class="bg-primary h-full"
                                style="width: {{ (item.total_qty / reports.top_products[0].total_qty * 100)|int }}%">
                            </div>
                        </div>
                        <span class="font-bold text-gray-900">{{ item.total_qty }} шт.</span>
                    </div>
                </div>
                {% endfor %}
            </div>
        </div>

        <div class="bg-white p-6 rounded-2xl border border-gray-100 shadow-sm">
            <h3 class="text-base font-bold text-gray-900 mb-6 flex items-center gap-2">
                <i class="ri-pie-chart-line text-primary"></i> Выручка по категориям
            </h3>
            <div class="space-y-4">
                {% for cat in reports.category_revenue %}
                <div class="flex justify-between items-center">
                    <span class="text-base font-medium text-gray-600">{{ cat.Название_категории }}</span>
                    <span class="text-base font-bold text-secondary">{{ cat.total_revenue|round|int }}
```

```
                    ₽</span>
                </div>
                {% endfor %}
            </div>
        </div>

        <div class="col-span-2 bg-white p-6 rounded-2xl border border-gray-100 shadow-sm">
            <h3 class="text-base font-bold text-gray-900 mb-6">Динамика продаж</h3>
            <div class="overflow-x-auto">
                <table class="w-full">
                    <thead>
                        <tr class="text-description text-gray-400 uppercase border-b border-gray-50">
                            <th class="pb-3 text-left">Дата</th>
                            <th class="pb-3 text-center">Заказов</th>
                            <th class="pb-3 text-right">Сумма за день</th>
                        </tr>
                    </thead>
                    <tbody class="divide-y divide-gray-50">
                        {% for day in reports.daily_sales %}
                        <tr class="hover:bg-gray-50">
                            <td class="py-3 text-base text-gray-600">{{ day.day }}</td>
                            <td class="py-3 text-center font-bold">{{ day.order_count }}</td>
                            <td class="py-3 text-right font-bold text-gray-900">{{ day.daily_sum|round|int
                                }} ₽</td>
```

```
                    </tr>
                    {% endfor %}
                </tbody>
            </table>
        </div>
    </div>

    <div
        class="col-span-2 bg-gradient-to-r from-blue-600 to-indigo-700 p-6 rounded-2xl shadow-lg text-white">
        <h3 class="text-base font-bold mb-6 flex items-center gap-2">
            <i class="ri-vip-crown-line text-yellow-400"></i> Наши лучшие покупатели
        </h3>
        <div class="grid grid-cols-5 gap-4">
            {% for user in reports.vip_customers %}
            <div class="bg-white/10 p-4 rounded-xl backdrop-blur-sm border border-white/10">
                <p class="text-description uppercase font-bold opacity-60">{{ user.Логин.split('@')[0]
                }}</p>
                <p class="text-base font-bold truncate">{{ user.Имя }} {{ user.Фамилия }}</p>
                <p class="text-subheading font-bold mt-2">{{ user.total_spent|round|int }} ₽</p>
            </div>
            {% endfor %}
        </div>
    </div>
```

```
            </div>
        </div>

        <div id="tab-products" class="p-6">
            <table class="w-full text-left">
                <thead>
                    <tr class="text-description font-bold text-gray-400 uppercase border-b border-gray-50">
                        <th class="pb-4 px-2">ID</th>
                        <th class="pb-4">Название</th>
                        <th class="pb-4">Категория</th>
                        <th class="pb-4">Цена</th>
                        <th class="pb-4 text-right">Действия</th>
                    </tr>
                </thead>
                <tbody class="divide-y divide-gray-50">
                    {% for p in products %}
                    <tr class="hover:bg-gray-50 transition">
                        <td class="py-4 px-2 text-gray-400">#{{ p.ID_Товара }}</td>
                        <td class="py-4 font-bold text-gray-900">{{ p.Название }}</td>
                        <td class="py-4"><span
                                class="px-3 py-1 bg-blue-50 text-primary rounded-full text-[12px] font-bold">{{
                                p.Название_категории }}</span></td>
                        <td class="py-4 font-bold">{{ p.Цена|round|int }} ₽</td>
                        <td class="py-4 text-right">
                            <div class="flex justify-end gap-2">
                                <button onclick="openEditModal({{ p | tojson | forceescape }})"
                                    class="p-2 text-gray-400 hover:text-primary transition">
```

```
                        <i class="ri-pencil-line text-lg"></i>
                    </button>

                    <a href="{{ url_for('delete_product', product_id=p.ID_Товара) }}"
                        onclick="return confirm('Вы уверены, что хотите удалить этот товар?')"
                        class="p-2 text-gray-400 hover:text-red-500 transition">
                        <i class="ri-delete-bin-line text-lg"></i>
                    </a>
                </div>
            </td>
        </tr>
        {% endfor %}
    </tbody>
</table>
</div>

<div id="tab-orders" class="p-6 hidden">
    <table class="w-full text-left">
        <thead>
            <tr class="text-description font-bold text-gray-400 uppercase border-b border-gray-50">
                <th class="pb-4 px-2">ID</th>
                <th class="pb-4">Клиент</th>
                <th class="pb-4">Дата</th>
                <th class="pb-4">Сумма</th>
                <th class="pb-4">Статус</th>
            </tr>
        </thead>
```

```
                <tbody class="divide-y divide-gray-50">
                    {% for o in orders %}
                    <tr>
                        <td class="py-4 px-2">#{{ o.ID_Заказа }}</td>
                        <td class="py-4 font-bold">{{ o.Имя }} {{ o.Фамилия }}</td>
                        <td class="py-4 text-gray-500">{{ o.Дата_заказа.strftime('%d.%m.%Y %H:%M') }}</td>
                        <td class="py-4 font-bold">{{ o.Итоговая_сумма|round|int }} ₽</td>
                        <td class="py-4">
                            <span
                                class="px-3 py-1 {% if o.Статус == 'Оплачен' %}bg-green-50 text-secondary{% else %}bg-orange-50 text-orange-500{% endif %} rounded-lg font-bold text-[12px]">
                                {{ o.Статус }}
                            </span>
                        </td>
                    </tr>
                    {% endfor %}
                </tbody>
            </table>
        </div>
    </div>
</div>


<div id="editModal" class="fixed inset-0 bg-black/50 hidden items-center justify-center z-50 backdrop-blur-sm">
    <div class="bg-white w-[500px] rounded-2xl shadow-2xl p-8">
```

```
<div class="flex justify-between items-center mb-6">
    <h2 class="text-subheading font-bold text-gray-900">Редактировать товар</h2>
    <button onclick="closeModal('editModal')" class="text-gray-400 hover:text-gray-600">
        <i class="ri-close-line text-2xl"></i>
    </button>
</div>

<form action="{{ url_for('edit_product') }}" enctype="multipart/form-data" method="POST" class="space-y-4">
    <input type="hidden" name="id" id="edit-id">

    <div>
        <label class="block text-description font-bold text-gray-400 uppercase mb-1">
            Изображение (Выберите новое для замены)
        </label>
        <input type="file" name="image" id="edit-image" accept="image/png, image/jpeg, image/gif"
            class="w-full px-4 py-2 border border-gray-200 rounded-lg outline-none focus:border-primary bg-gray-50 text-sm">
        <p class="text-xs text-gray-400 mt-1">Поддерживаются JPG, PNG, GIF. Если не выбрать, останется старое.
        </p>
    </div>

    <div>
        <label class="block text-description font-bold text-gray-400 uppercase mb-1">Описание</label>
```

```
            <textarea name="description" id="edit-desc" rows="4"
              class="w-full px-4 py-2 border border-gray-200 rounded-lg outline-none focus:border-primary"></textarea>
          </div>


          <input type="hidden" name="id" id="edit-id">

          <div>
            <label class="block text-description font-bold text-gray-400 uppercase mb-1">Название</label>
            <input type="text" name="name" id="edit-name" required
              class="w-full px-4 py-2 border border-gray-200 rounded-lg outline-none focus:border-primary">
          </div>

          <div class="grid grid-cols-2 gap-4">
            <div>
              <label class="block text-description font-bold text-gray-400 uppercase mb-1">Цена (₽)</label>
              <input type="number" name="price" id="edit-price" required
                class="w-full px-4 py-2 border border-gray-200 rounded-lg outline-none focus:border-primary">
            </div>
            <div>
              <label class="block text-description font-bold text-gray-400 uppercase mb-1">Категория</label>
              <select name="category_id" id="edit-category"
                class="w-full px-4 py-2 border border-gray-200 rounded-lg outline-none focus:border-primary">
```

```
                {% for cat in categories %}
                <option value="{{ cat.ID_Категории }}">{{ cat.Название_категории }}</option>
                {% endfor %}
              </select>
            </div>
          </div>

          <button type="submit"
            class="w-full bg-primary text-white py-3 rounded-xl font-bold hover:bg-blue-600 transition shadow-lg shadow-blue-100 mt-4">
            СОХРАНИТЬ ИЗМЕНЕНИЯ
          </button>
        </form>
    </div>
</div>

<div id="productModal" class="fixed inset-0 bg-black/50 hidden items-center justify-center z-50">
    <div class="bg-white rounded-2xl p-8 w-[500px] shadow-2xl">
        <div class="flex justify-between items-center mb-6">
            <h2 class="text-subheading font-bold text-gray-900">Новый товар</h2>
            <button onclick="closeModal('productModal')" class="text-gray-400 hover:text-gray-600">
                <i class="ri-close-line text-2xl"></i>
            </button>
        </div>

        <form action="{{ url_for('add_product') }}" method="POST" enctype="multipart/form-data" class="space-y-4">
```

```
        <div>
            <label class="block text-description font-bold text-gray-400 uppercase mb-1">Название</label>
            <input type="text" name="name" required
                class="w-full px-4 py-2 border border-gray-200 rounded-lg outline-none focus:border-primary">
        </div>

        <div class="grid grid-cols-2 gap-4">
            <div>
                <label class="block text-description font-bold text-gray-400 uppercase mb-1">Цена (₽)</label>
                <input type="number" name="price" required
                    class="w-full px-4 py-2 border border-gray-200 rounded-lg outline-none focus:border-primary">
            </div>
            <div>
                <label class="block text-description font-bold text-gray-400 uppercase mb-1">Категория</label>
                <select name="category" required
                    class="w-full px-4 py-2 border border-gray-200 rounded-lg outline-none focus:border-primary bg-white">
                    {% for cat in categories %}
                    <option value="{{ cat.ID_Категории }}">{{ cat.Название_категории }}</option>
                    {% endfor %}
                </select>
            </div>
        </div>
```

```
        <div>
            <label class="block text-description font-bold text-gray-400 uppercase mb-1">Описание</label>
            <textarea name="description" rows="3"
                class="w-full px-4 py-2 border border-gray-200 rounded-lg outline-none focus:border-primary"></textarea>
        </div>

        <div>
            <label class="block text-description font-bold text-gray-400 uppercase mb-1">Изображение</label>
            <input type="file" name="image" accept="image/*"
                class="w-full px-4 py-2 border border-gray-200 rounded-lg outline-none focus:border-primary bg-gray-50 text-sm">
        </div>

        <button type="submit"
            class="w-full bg-primary text-white py-3 rounded-xl font-bold hover:bg-blue-600 transition mt-4">
            СОЗДАТЬ ТОВАР
        </button>
    </form>
  </div>
</div>

<style>
  .admin-tab {
    padding: 20px 32px;
    font-weight: 700;
    color: #9CA3AF;
```

```
        border-bottom: 2px solid transparent;
        transition: all 0.2s;
    }

    .admin-tab.active {
        color: #3B82F6;
        border-bottom-color: #3B82F6;
    }
</style>



<script>
    function switchTab(tab) {
        // 1. Список всех ID контента вкладок
        const contents = ['tab-products', 'tab-orders', 'tab-reports'];

        // 2. Скрываем абсолютно все вкладки
        contents.forEach(id => {
            const element = document.getElementById(id);
            if (element) {
                element.classList.add('hidden');
            }
        });

        // 3. Убираем класс 'active' со всех кнопок
        document.querySelectorAll('.admin-tab').forEach(btn => {
            btn.classList.remove('active');
        });

        // 4. Показываем нужную вкладку
```

```
    const activeContent = document.getElementById('tab-' + tab);
    if (activeContent) {
        activeContent.classList.remove('hidden');
    }

    // 5. Делаем нажатую кнопку активной
    const activeBtn = document.getElementById('tab-' + tab + '-btn');
    if (activeBtn) {
        activeBtn.classList.add('active');
    }
}

function openEditModal(product) {
    console.log("Данные продукта:", product);

    document.getElementById('edit-id').value = product.ID_Товара;
    document.getElementById('edit-name').value = product.Название;
    document.getElementById('edit-price').value = Math.round(product.Цена);
    document.getElementById('edit-category').value = product.ID_Категории;
    document.getElementById('edit-desc').value = product.Описание;

    // НОВОЕ: Очищаем поле выбора файла при открытии окна
    document.getElementById('edit-image').value = '';

    const modal = document.getElementById('editModal');
    modal.classList.remove('hidden');
    modal.classList.add('flex');
}

// Функция закрытия (универсальная)
```

```
    function closeModal(id) {
        const modal = document.getElementById(id);
        modal.classList.add('hidden');
        modal.classList.remove('flex');
    }
    // Функция для открытия любого модального окна
    function openModal(id) {
        const modal = document.getElementById(id);
        if (modal) {
            modal.classList.remove('hidden');
            modal.classList.add('flex');
        }
    }
</script>
{% endblock %}
```

Код – cart.html

```
{% extends "base.html" %}
{% block title %}Корзина | SoftKey{% endblock %}

{% block content %}
<div class="bg-white min-h-screen">
    <div class="w-[1196px] mx-auto pt-[64px] pb-20">

        <div class="flex justify-between items-end mb-8">
            <div>
                <nav class="text-sm text-gray-400 mb-6">
                    <a href="/" class="hover:text-primary">Главная</a> /
                    <span class="text-gray-900 font-medium">Корзина</span>
                </nav>
```

```
            <div class="flex items-center gap-3">
                <i class="ri-shopping-cart-2-fill text-primary text-4xl"></i>
                <h1 class="text-[40px] font-bold text-gray-900 leading-none">Корзина покупок</h1>
            </div>
        </div>
        <a href="/" class="text-primary font-medium flex items-center gap-2 hover:underline mb-2">
            <i class="ri-arrow-left-line"></i> Продолжить покупки
        </a>
    </div>

    <div class="flex gap-[22px] items-start">

        <div class="w-[710px] flex flex-col gap-5">
            {% if items %}
            <div
                class="grid grid-cols-[1fr_100px_120px_100px_40px] px-6 text-description text-gray-400 font-bold uppercase">
                <span>Товар</span>
                <span class="text-center">Цена</span>
                <span class="text-center">Количество</span>
                <span class="text-right">Сумма</span>
                <span></span>
            </div>

            {% for item in items %}
            <div
                class="bg-white border border-gray-100 rounded-[12px] p-6 shadow-sm hover:shadow-md transition-shadow">
```

```
                <div class="flex items-center gap-6">
                    <div
                        class="w-20 h-20 bg-gray-50 rounded-lg flex items-center justify-center text-primary text-3xl">
                        <div
                            class="w-16 h-16 bg-gray-50 rounded-lg flex-shrink-0 overflow-hidden border border-gray-100">
                            {% if item.Изображение %}
                            <img src="{{ url_for('static', filename='images/' + item.Изображение) }}"
                                class="w-full h-full object-cover">
                            {% else %}
                            <div class="w-full h-full flex items-center justify-center text-gray-300">
                                <i class="ri-image-line"></i>
                            </div>
                            {% endif %}
                        </div>
                    </div>

                    <div class="flex-1 grid grid-cols-[1fr_100px_120px_100px_40px] items-center">
                        <div>
                            <h3 class="text-base font-bold text-gray-900 leading-tight mb-1">{{ item.Название }}
                            </h3>
                            <div class="flex flex-wrap gap-x-4 gap-y-1">
                                <span class="text-[12px] text-gray-400 flex items-center gap-1">
```

```
                        <i class="ri-download-cloud-line"></i> Мгновенная
доставка
                    </span>
                    <span class="text-[12px] text-gray-400 flex items-center
gap-1">
                        <i class="ri-shield-check-line"></i> Официальная
лицензия
                    </span>
                </div>
            </div>

            <div class="text-center font-bold text-gray-900">
                {{ item.Цена|round|int }} ₽
            </div>

            <div class="flex justify-center">
                <div class="flex items-center border border-gray-200 rounded-
lg bg-gray-50">
                    <a href="/cart/update/{{ item.ID_Товара }}/minus"
                        class="w-8 h-8 flex items-center justify-center hover:text-
primary">-</a>
                    <span class="w-8 h-8 flex items-center justify-center font-
bold text-sm">{{
                        item.Количество }}</span>
                    <a href="/cart/update/{{ item.ID_Товара }}/plus"
                        class="w-8 h-8 flex items-center justify-center hover:text-
primary">+</a>
                </div>
            </div>
```

```
                    <div class="text-right font-bold text-primary">
                        {{ (item.Цена * item.Количество)|round|int }} ₽
                    </div>

                    <div class="text-right">
                        <a href="/cart/remove/{{ item.ID_Товара }}"
                           class="text-gray-300 hover:text-red-500 transition ml-4">
                            <i class="ri-delete-bin-line text-xl"></i>
                        </a>
                    </div>
                </div>
            </div>
        </div>
        {% endfor %}
        {% else %}
        <div class="bg-white border border-gray-100 rounded-[12px] p-20 shadow-sm text-center">
            <i class="ri-shopping-cart-2-line text-[64px] text-gray-200 mb-4 block"></i>
            <h2 class="text-subheading font-bold text-gray-900">Ваша корзина пуста</h2>
            <a href="/"
               class="inline-block mt-8 bg-primary text-white px-10 py-3 rounded-md font-bold hover:bg-blue-600 transition">В
               КАТАЛОГ</a>
        </div>
        {% endif %}
    </div>

    {% if items %}
```

```
        <aside class="w-[464px] sticky top-[40px]">
          <div class="bg-white border border-gray-100 rounded-[12px] p-8 shadow-sm">
            <h3 class="text-2xl font-bold text-gray-900 mb-8">Сумма заказа</h3>

            <div class="space-y-4 mb-8">
              <div class="flex justify-between text-base text-gray-500">
                <span>Товары ({{ items|length }} шт.)</span>
                <span class="font-medium text-gray-900">{{ total_price|round|int }} ₽</span>
              </div>
              <div class="flex justify-between text-base text-gray-500">
                <span>Способ получения</span>
                <span class="font-medium text-gray-900">Email</span>
              </div>
            </div>

            <div class="border-t border-gray-100 pt-6 mb-8">
              <div class="flex justify-between items-center">
                <span class="text-xl font-bold text-primary">Итого к оплате</span>
                <span class="text-3xl font-bold text-primary">{{ total_price|round|int }} ₽</span>
              </div>
            </div>

            <form action="/checkout" method="POST">
              <button type="submit"
```

```
                    class="w-full bg-[#52b788] text-white py-4 rounded-lg font-bold text-lg hover:bg-[#40a076] transition shadow-lg shadow-green-100 flex items-center justify-center gap-2">
                        <i class="ri-lock-line"></i> ПЕРЕЙТИ К ОФОРМЛЕНИЮ
                    </button>
                </form>

                <div class="mt-6 space-y-3">
                    <div class="flex items-center gap-2 text-sm text-gray-500">
                        <i class="ri-checkbox-circle-fill text-green-500"></i> Безопасная оплата
                    </div>
                    <div class="flex items-center gap-2 text-sm text-gray-500">
                        <i class="ri-checkbox-circle-fill text-green-500"></i> Гарантия возврата 14 дней
                    </div>
                    <div class="flex items-center gap-2 text-sm text-gray-500">
                        <i class="ri-checkbox-circle-fill text-green-500"></i> Техподдержка 24/7
                    </div>
                </div>
            </div>
        </aside>
        {% endif %}

    </div>
</div>
{% endblock %}
```

Код – index.html

```
{% extends "base.html" %}
{% block title %}Главная | SoftKey Store{% endblock %}

{% block content %}
<div class="flex flex-col items-center pt-[48px] pb-20">

  <div class="mb-[20px] text-center">
    <h1 class="text-heading font-bold text-gray-900 mb-2">Найдите свое идеальное ПО</h1>

  </div>

  <form action="/" method="GET" class="w-[1196px] h-[32px] flex gap-0 shadow-sm">
    <input type="text" name="search" value="{{ request.args.get('search', '') }}"
      placeholder="Поиск по названию программы..."
      class="flex-1 px-4 border border-gray-300 rounded-l-md outline-none focus:border-primary text-base">
    <button type="submit"
      class="bg-primary text-white px-8 rounded-r-md hover:bg-blue-600 transition font-bold text-base">
      НАЙТИ
    </button>
  </form>

  <div class="h-[64px] flex items-center w-[1196px]">
    <p class="text-description uppercase tracking-widest font-bold text-gray-400">Настройте фильтры для точного
```

```
        выбора</p>
    </div>

    <form id="filterForm" action="/" method="GET" class="w-[1196px] flex gap-4">
        <select name="sort_price" onchange="this.form.submit()"
            class="px-4 py-2 border rounded-md text-base bg-white cursor-pointer">
            <option value="">Цена: любая</option>
            <option value="asc" {% if request.args.get('sort_price')=='asc' %}selected{% endif %}>Сначала дешевле
            </option>
            <option value="desc" {% if request.args.get('sort_price')=='desc' %}selected{% endif %}>Сначала дороже
            </option>
        </select>

        <select name="category" onchange="this.form.submit()"
            class="px-4 py-2 border rounded-md text-base bg-white cursor-pointer">
            <option value="all">Все категории</option>
            {% for cat in categories %}
            <option value="{{ cat.ID_Категории }}" {% if request.args.get('category')|string==cat.ID_Категории|string
                %}selected{% endif %}>
                {{ cat.Название_категории }}
            </option>
            {% endfor %}
        </select>

        <select name="sort_date" onchange="this.form.submit()"
            class="px-4 py-2 border rounded-md text-base bg-white cursor-pointer">
```

```
        <option value="">По дате добавления</option>
        <option value="new" {% if request.args.get('sort_date')=='new' %}selected{% endif %}>Сначала новые</option>
        <option value="old" {% if request.args.get('sort_date')=='old' %}selected{% endif %}>Сначала старые</option>
      </select>
    </form>

    <div class="h-[96px] flex items-center w-[1196px] mb-[-8px]">
      <h2 class="text-subheading font-bold text-gray-800">
        {% if products %} Найдено товаров: {{ products|length }} {% else %} Ничего не найдено {% endif %}
      </h2>
    </div>

    <div class="grid grid-cols-3 gap-[25px] w-[1196px]">
      {% for prod in products %}
      <div
        class="w-[382px] h-[256px] relative group bg-white rounded-[8px] overflow-hidden border border-gray-300 shadow-md hover:shadow-lg transition-shadow">

        <div class="w-full h-[110px] bg-gray-100 overflow-hidden">
          {% if prod.Изображение %}
          <img src="{{ url_for('static', filename='images/' + prod.Изображение) }}" alt="{{ prod.Название }}"
            class="w-full h-full object-cover">
          {% else %}
          <div class="w-full h-full flex items-center justify-center bg-blue-50 text-blue-200">
```

```
            <i class="ri-image-2-fill text-[40px]"></i>
          </div>
          {% endif %}
        </div>

        <div
          class="absolute top-[80px] left-0 w-full h-[176px] bg-white rounded-t-[12px] p-[20px] flex flex-col justify-between shadow-[0_-4px_10px_rgba(0,0,0,0.05)]">
          <div>
            <div class="flex justify-between items-center mb-1">
              <span class="text-description font-bold text-primary uppercase">{{ prod.Название_категории
                }}</span>
              <span class="text-subheading font-bold text-secondary">{{ prod.Цена|round|int }} ₽</span>
            </div>
            <h3 class="text-base font-bold text-gray-900 truncate">{{ prod.Название }}</h3>
            <p class="text-description text-gray-500 line-clamp-2 mt-1">
              {{ prod.Описание }}
            </p>
          </div>

          <a href="/product/{{ prod.ID_Товара }}"
            class="w-full h-[32px] bg-gray-100 text-gray-800 rounded flex items-center justify-center text-base font-bold hover:bg-primary hover:text-white transition uppercase tracking-wider">
            Подробнее
          </a>
```

```
        </div>
      </div>
      {% endfor %}
    </div>
</div>
{% endblock %}
```

Код – login.html

```
{% extends "base.html" %}
{% block title %}Вход | SoftKey{% endblock %}

{% block content %}
<div class="flex items-center justify-center min-h-[calc(100vh-96px)] bg-gray-50">
  <div class="flex w-[750px] h-[608px] bg-white rounded-[8px] shadow-2xl overflow-hidden border border-gray-100">

    <div class="w-[375px] bg-primary p-10 flex flex-col justify-between text-white">
      <div>
        <div class="flex items-center gap-2 mb-8">
          <i class="ri-key-2-line text-heading"></i>
          <span style="font-family: 'Pacifico', cursive;" class="text-subheading">SoftKey</span>
        </div>
        <h1 class="text-heading font-bold leading-tight mb-4">Добро пожаловать в магазин ПО</h1>
        <p class="text-base opacity-90">Получите мгновенный доступ к лицензионным ключам и программным продуктам мировых вендоров.</p>
      </div>
```

```
        <div class="space-y-4">
            <div class="flex items-center gap-3">
                <i class="ri-shield-check-line text-subheading"></i>
                <span class="text-base">100% Гарантия активации</span>
            </div>
            <div class="flex items-center gap-3">
                <i class="ri-customer-service-2-line text-subheading"></i>
                <span class="text-base">Поддержка 24/7</span>
            </div>
        </div>
    </div>

    <div class="w-[375px] p-10 flex flex-col justify-center bg-white">
        <h2 class="text-subheading font-bold text-gray-800 mb-2">Авторизация</h2>
        <p class="text-description text-gray-500 mb-8">Введите данные для доступа к покупкам</p>

        <form action="{{ url_for('login') }}" method="POST" class="space-y-4">
            <div>
                <label class="block text-description uppercase font-bold text-gray-400 mb-1">Email</label>
                <input type="email" name="login" required placeholder="example@mail.com"
                       class="w-full px-4 py-3 bg-gray-50 border border-gray-200 rounded-lg focus:ring-2 focus:ring-primary outline-none text-base transition">
            </div>
            <div>
```

```
            <label class="block text-description uppercase font-bold text-gray-400 mb-1">Пароль</label>
            <input type="password" name="password" required placeholder="••••••••"
                   class="w-full px-4 py-3 bg-gray-50 border border-gray-200 rounded-lg focus:ring-2 focus:ring-primary outline-none text-base transition">
        </div>

        <button type="submit" class="w-full bg-primary text-white py-3 rounded-lg font-bold hover:bg-blue-600 transition shadow-lg shadow-blue-200">
            Войти в систему
        </button>
    </form>

    <div class="mt-8 text-center">
        <p class="text-base text-gray-600">Нет аккаунта?
            <a href="{{ url_for('register') }}" class="text-primary font-bold hover:underline">Создать</a>
        </p>
    </div>
  </div>
 </div>
</div>
{% endblock %}
```

Код – orders.html

```
{% extends "base.html" %}
{% block title %}Мои заказы | SoftKey{% endblock %}

{% block content %}
```

```
<div class="bg-gray-0 min-h-screen">
  <div class="w-[1196px] mx-auto pt-8 pb-20">

    <nav class="flex items-center gap-2 text-sm text-gray-400 mb-8">
      <a href="/" class="hover:text-primary transition">Главная</a> /
      <span class="text-gray-900 font-medium">Мои заказы</span>
    </nav>

    <div class="space-y-6">
      {% if orders %}
      {% for order in orders %}
      <div
        class="bg-white border border-gray-100 rounded-[16px] shadow-sm overflow-hidden transition-all duration-300">
        <button onclick="toggleOrder('{{ order.ID_Заказа }}')"
          class="w-full p-8 flex items-center justify-between hover:bg-gray-50/50 transition text-left group">
          <div class="flex items-center gap-12">
            <div>
              <div class="flex items-center gap-3 mb-1">
                <h3 class="text-xl font-bold text-primary">Заказ #{{ order.ID_Заказа }}</h3>
                {% if order.Статус == 'Активен' %}
                <span
                  class="px-3 py-1 bg-blue-50 text-blue-500 rounded-full text-xs font-bold flex items-center gap-1">
                  <i class="ri-checkbox-circle-fill"></i> {{ order.Статус }}
                </span>
                {% elif order.Статус == 'Выполнен' %}
                <span
```

```
                        class="px-3 py-1 bg-green-50 text-green-500 rounded-full
text-xs font-bold flex items-center gap-1">
                        <i class="ri-checkbox-circle-fill"></i> {{ order.Статус }}
                    </span>
                    {% else %}
                    <span
                        class="px-3 py-1 bg-orange-50 text-orange-500 rounded-full
text-xs font-bold flex items-center gap-1">
                        <i class="ri-time-fill"></i> {{ order.Статус }}
                    </span>
                    {% endif %}
                </div>
                <p class="text-sm text-gray-400">
                    <i                class="ri-calendar-line"></i>                {{
order.Дата_заказа.strftime('%d марта %Y • %H:%M') }}
                </p>
            </div>
        </div>

        <div class="flex items-center gap-10">
            <div class="text-right">
                <span class="text-2xl font-black text-gray-900">{{
order.Итоговая_сумма|round|int }}
                    ₽</span>
            </div>
            <i id="icon-{{ order.ID_Заказа }}"
                class="ri-arrow-down-s-line text-2xl text-primary transition-
transform duration-300"></i>
        </div>
    </button>
```

```
<div id="content-{{ order.ID_Заказа }}" class="hidden border-t border-gray-50 bg-white p-8">
    <div class="mb-6 flex items-center gap-2 text-gray-900 font-bold">
        <i class="ri-key-2-line"></i>
        <h4>Лицензии в заказе</h4>
    </div>

    <div class="w-full">
        <div
            class="grid grid-cols-[1.5fr_1fr_100px_1fr_1.5fr] px-4 py-3 bg-gray-50 rounded-t-lg text-[11px] font-bold text-gray-400 uppercase tracking-wider">
            <span>Продукт</span>
            <span class="text-center">Лицензионный ключ</span>
            <span class="text-center">Статус</span>
            <span class="text-center">Срок действия</span>
            <span class="text-right">Действия</span>
        </div>

        <div class="divide-y divide-gray-50 border border-gray-50 border-t-0 rounded-b-lg">
            {% for item in order.products_list %}
            <div
                class="grid grid-cols-[1.5fr_1fr_100px_1fr_1.5fr] items-center px-4 py-6 hover:bg-gray-50/30 transition">
                <div class="flex items-center gap-4">
                    <div
                        class="w-12 h-12 bg-gray-100 rounded-lg flex items-center justify-center text-gray-300">
```

```
                    <i class="ri-image-2-line text-2xl"></i>
                </div>
                <div>
                    <p class="font-bold text-gray-900 leading-tight">{{ item.Название }}</p>
                    <p class="text-[11px] text-gray-400">Бизнес • 2 года обновлений</p>
                </div>
            </div>

            <div class="flex flex-col gap-2 items-center">
                {% if item['keys'] %}
                {% for key in item['keys'] %}
                <div
                    class="flex items-center gap-2 bg-white px-3 py-1.5 rounded border border-dashed border-blue-200 font-mono text-[11px] text-primary shadow-sm w-fit">
                    <span>{{ key }}</span>
                    <button onclick="copyToClipboard(this)"
                        class="copy-btn text-gray-300 hover:text-primary transition relative"
                        data-key="{{ key }}">
                        <i class="ri-file-copy-line"></i>
                        <span
                            class="copy-tooltip absolute -top-8 left-1/2 transform -translate-x-1/2 bg-gray-800 text-white text-[10px] px-2 py-1 rounded opacity-0 pointer-events-none transition-opacity whitespace-nowrap">
                            Скопировано!
                        </span>
                    </button>
```

```
                    </div>
                    {% endfor %}
                    {% else %}
                    <span class="text-[10px] text-gray-400 italic">Генерация ключа...</span>
                    {% endif %}
                </div>

                <div class="flex justify-center">
                    <span
                        class="px-2 py-1 bg-blue-50 text-blue-600 rounded text-[10px] font-bold flex items-center gap-1">
                        <i class="ri-checkbox-circle-fill"></i> АКТИВНА
                    </span>
                </div>

                <div class="text-center">
                    <p class="text-[10px] text-gray-400 uppercase font-bold">Бессрочно</p>
                    <p class="text-[10px] text-gray-300 leading-tight">Retail License</p>
                </div>

                <div class="flex justify-end gap-2">
                    <button
                        class="px-4 py-2 bg-blue-50 text-primary rounded-lg text-xs font-bold hover:bg-blue-100 transition flex items-center gap-1">
                        <i class="ri-download-cloud-line"></i> Скачать
                    </button>
                    <button
```

```
                        class="px-4 py-2 bg-green-50 text-green-600 rounded-lg text-xs font-bold hover:bg-green-100 transition flex items-center gap-1">
                            <i class="ri-flashlight-line"></i> Инструкция
                        </button>
                    </div>
                </div>
                {% endfor %}
            </div>
        </div>
    </div>
    {% endfor %}
    {% else %}
    <div class="bg-white border border-gray-100 rounded-[16px] p-24 text-center shadow-sm">
        <div class="w-20 h-20 bg-gray-50 rounded-full flex items-center justify-center mx-auto mb-6">
            <i class="ri-shopping-bag-line text-[40px] text-gray-200"></i>
        </div>
        <h3 class="text-xl font-bold text-gray-900 mb-2">У вас пока нет заказов</h3>
        <a href="/"
            class="inline-block bg-primary text-white px-10 py-4 rounded-xl font-bold hover:bg-blue-600 transition shadow-lg shadow-blue-100">
            В КАТАЛОГ
        </a>
    </div>
    {% endif %}
</div>
</div>
```

```
</div>

<script>
    function toggleOrder(id) {
        const content = document.getElementById('content-' + id);
        const icon = document.getElementById('icon-' + id);

        if (content.classList.contains('hidden')) {
            content.classList.remove('hidden');
            icon.style.transform = 'rotate(180deg)';
        } else {
            content.classList.add('hidden');
            icon.style.transform = 'rotate(0deg)';
        }
    }

    // Функция для копирования ключа в буфер обмена
    function copyToClipboard(button) {
        const key = button.getAttribute('data-key');
        const tooltip = button.querySelector('.copy-tooltip');

        // Используем современный Clipboard API
        navigator.clipboard.writeText(key).then(() => {
            // Показываем tooltip
            tooltip.style.opacity = '1';
            tooltip.textContent = 'Скопировано!';

            // Меняем иконку на успешное копирование
            const icon = button.querySelector('i');
            icon.className = 'ri-check-line';
```

```
            button.classList.remove('text-gray-300', 'hover:text-primary');
            button.classList.add('text-green-500');

            // Возвращаем обратно через 2 секунды
            setTimeout(() => {
                tooltip.style.opacity = '0';
                icon.className = 'ri-file-copy-line';
                button.classList.remove('text-green-500');
                button.classList.add('text-gray-300', 'hover:text-primary');
            }, 2000);

        }).catch(err => {
            // Fallback для старых браузеров
            console.error('Ошибка копирования: ', err);

            // Показываем ошибку
            tooltip.style.opacity = '1';
            tooltip.textContent = 'Ошибка!';
            tooltip.classList.add('bg-red-500');

            setTimeout(() => {
                tooltip.style.opacity = '0';
                tooltip.classList.remove('bg-red-500');
            }, 2000);
        });
    }

</script>
{% endblock %}
```

Код – product_detail.html

```
{% extends "base.html" %}
{% block title %}{{ product.Название }} | SoftKey{% endblock %}

{% block content %}
<div class="flex flex-col items-center">
    <div class="w-[1196px] mx-auto pt-[48px] pb-20">
        <nav class="text-sm text-gray-400 mb-8">
            <a href="/" class="hover:text-primary transition">Главная</a> /
            <span class="text-gray-900 font-medium">{{ product.Название }}</span>
        </nav>

        <div class="w-[1196px] flex gap-[20px] mb-[64px]">
            <div
                class="w-[345px] h-[345px] bg-white rounded-[8px] flex items-center justify-center border border-blue-100 overflow-hidden">
                {% if product.Изображение %}
                <img src="{{ url_for('static', filename='images/' + product.Изображение) }}"
                    alt="{{ product.Название }}" class="w-full h-full object-contain">
                {% else %}
                <i class="ri-shield-flash-line text-[120px] text-primary opacity-20"></i>
                {% endif %}
            </div>
            <div class="w-[831px] min-h-[345px] flex flex-col">
                <div class="mb-6">
                    <span class="text-description font-bold text-primary uppercase tracking-widest">{{
                        product.Название_категории }}</span>
```

```
            <h1 class="text-heading font-bold text-gray-900 leading-tight">{{ product.Название }}</h1>
          </div>

          <div class="flex-grow">
            <h3 class="text-base font-bold text-gray-400 uppercase mb-2">Описание продукта</h3>
            <p class="text-base text-gray-700 leading-relaxed" style="white-space: pre-line;">
              {{ product.Описание }}
            </p>
          </div>

          <div class="mt-6 flex gap-8">
            <div class="flex items-center gap-2 text-secondary">
              <i class="ri-flashlight-line text-subheading"></i>
              <span class="text-base font-bold uppercase">Мгновенная доставка</span>
            </div>
            <div class="flex items-center gap-2 text-gray-500">
              <i class="ri-shield-check-line text-subheading"></i>
              <span class="text-base font-bold uppercase">Официальная лицензия</span>
            </div>
          </div>
        </div>
      </div>

      <div class="flex gap-[22px] items-start">
```

```
<div class="w-[710px]">
  <h2 class="text-2xl font-bold text-gray-900 mb-6">Варианты лицензии</h2>

  <div class="grid grid-cols-1 gap-4">
    <div
      class="flex items-center justify-between p-6 border border-gray-100 rounded-[16px] bg-white shadow-sm hover:border-primary hover:shadow-md transition-all group">
      <div class="flex items-center gap-6">
        <div
          class="w-16 h-16 bg-primary text-white rounded-xl flex items-center justify-center text-3xl shadow-lg shadow-blue-100 group-hover:scale-105 transition-transform">
          <i class="ri-global-line"></i>
        </div>
        <div>
          <h4 class="text-xl font-bold text-gray-900">{{ product.Название }}</h4>
          <p class="text-gray-500">Global Retail • Бессрочная активация</p>
          <div class="flex items-center gap-4 mt-2">
            <span class="text-green-500 text-sm flex items-center gap-1 font-bold">
              <i class="ri-checkbox-circle-fill"></i> В НАЛИЧИИ
            </span>
          </div>
        </div>
      </div>
```

```
                <div class="flex items-center gap-8">
                    <div class="text-right">
                        <span class="text-3xl font-bold text-gray-900">{{ product.Цена|round|int }} ₽</span>
                    </div>
                    <form action="/add_to_cart/{{ product.ID_Товара }}" method="POST">
                        <button type="submit"
                            class="bg-gray-900 text-white px-8 py-4 rounded-xl font-bold hover:bg-black transition flex items-center gap-3 shadow-lg shadow-gray-200">
                            <i class="ri-shopping-cart-2-line"></i>
                            В КОРЗИНУ
                        </button>
                    </form>
                </div>
            </div>
        </div>

        <div class="mt-12">
            <h3 class="text-xl font-bold text-gray-900 mb-6 uppercase tracking-tight">Технические характеристики
            </h3>
            <div class="grid grid-cols-2 gap-x-12 gap-y-4 border-t border-gray-100 pt-6">
                <div class="flex justify-between border-b border-gray-50 pb-2">
                    <span class="text-gray-500">Тип лицензии:</span>
                    <span class="font-medium">Retail</span>
                </div>
```

```
                <div class="flex justify-between border-b border-gray-50 pb-2">
                    <span class="text-gray-500">Срок действия:</span>
                    <span class="font-medium">Бессрочно</span>
                </div>
                <div class="flex justify-between border-b border-gray-50 pb-2">
                    <span class="text-gray-500">Язык:</span>
                    <span class="font-medium">Мультиязычный</span>
                </div>
                <div class="flex justify-between border-b border-gray-50 pb-2">
                    <span class="text-gray-500">Регион:</span>
                    <span class="font-medium">Global (Весь мир)</span>
                </div>
            </div>
        </div>
    </div>

    <aside class="w-[464px] space-y-4">
        <div class="bg-blue-50 border border-blue-100 rounded-[20px] p-8">
            <h3 class="text-lg font-bold text-blue-900 mb-4">Почему покупают у нас?</h3>
            <ul class="space-y-4">
                <li class="flex gap-4">
                    <i class="ri-shield-check-fill text-primary text-xl"></i>
                    <p class="text-sm text-blue-800"><span class="font-bold">100% Гарантия.</span> Все ключи
                        проходят валидацию в официальных сервисах.</p>
                </li>
                <li class="flex gap-4">
                    <i class="ri-customer-service-2-fill text-primary text-xl"></i>
```

```
                    <p class="text-sm text-blue-800"><span class="font-bold">Поддержка 24/7.</span> Поможем с
                        активацией в любое время дня и ночи.</p>
                </li>
                <li class="flex gap-4">
                    <i class="ri-mail-send-fill text-primary text-xl"></i>
                    <p class="text-sm text-blue-800"><span class="font-bold">Доставка за 1 минуту.</span> Ключ
                        придет на почту сразу после оплаты.</p>
                </li>
            </ul>
        </div>

        <div class="border border-gray-100 rounded-[20px] p-8 flex items-center gap-6">
            <div class="text-4xl text-gray-300"><i class="ri-visa-line"></i></div>
            <div class="text-4xl text-gray-300"><i class="ri-mastercard-line"></i></div>
            <div class="text-4xl text-gray-300"><i class="ri-mir-line"></i></div>
            <p class="text-xs text-gray-400 leading-tight">Безопасная оплата через защищенные шлюзы</p>
        </div>
    </aside>

  </div>
 </div>
</div>
</div>
{% endblock %}
```

Код – profile.html

```
{% extends "base.html" %}
{% block title %}Личный кабинет | SoftKey{% endblock %}

{% block content %}
<div class="flex justify-center pt-[48px] pb-20">
   <div class="w-[1196px] flex gap-[22px]">

      <aside class="w-[466px]  flex flex-col gap-6">
         <div class="bg-white border border-gray-100 rounded-[12px] p-8 shadow-sm min-h-[578px]">
            <div class="flex items-center gap-4 mb-6">
               <div
                  class="w-16 h-16 bg-primary/10 text-primary rounded-full flex items-center justify-center text-[32px]">
                  <i class="ri-user-smile-line"></i>
               </div>
               <div>
                  <h2 class="text-subheading font-bold text-gray-900">{{ user.Имя }} {{ user.Фамилия }}</h2>
                  <p class="text-description text-gray-500 uppercase tracking-wider">ID: {{ user.ID_Пользователя
                     }}</p>
               </div>
            </div>

            <div class="space-y-4 border-t border-gray-50 pt-6 mb-8">
               <div class="flex justify-between text-base">
                  <span class="text-gray-500">Логин:</span>
```

```
                <span class="font-medium">{{ user.Логин }}</span>
            </div>
            <div class="flex justify-between text-base">
                <span class="text-gray-500">Заказов:</span>
                <span class="font-medium text-primary">{{ orders|length }}</span>
            </div>
        </div>

        <nav class="flex flex-col gap-2">
            <button onclick="switchTab('personal')" id="btn-personal"
                class="tab-btn active-tab flex items-center gap-3 px-4 py-3 rounded-lg text-left transition font-bold hover:bg-blue-400 hover:text-gray-50">
                <i class="ri-user-line"></i> Личные данные
            </button>
            <button onclick="switchTab('security')" id="btn-security"
                class="tab-btn flex items-center gap-3 px-4 py-3 rounded-lg text-left transition font-bold text-gray-500 hover:bg-blue-400 hover:text-gray-50">
                <i class="ri-shield-keyhole-line"></i> Безопасность
            </button>
        </nav>
    </div>
</aside>

<main class="w-[708px]">

    <section id="tab-personal" class="bg-white border border-gray-100 rounded-[12px] p-8 shadow-sm min-h-[578px]">
        <div class="flex justify-between items-center mb-8">
```

```
            <h3 class="text-subheading font-bold text-gray-900">Личные данные</h3>
            <a href="/logout" class="flex items-center gap-2 text-red-500 font-bold hover:underline">
                <i class="ri-logout-box-r-line"></i> ВЫЙТИ
            </a>
        </div>

        <form id="profile-form" action="/update_profile" method="POST" class="grid grid-cols-2 gap-6">
            <div class="flex flex-col gap-2">
                <label class="text-description font-bold text-gray-400 uppercase">Фамилия</label>
                <input type="text" name="last_name" value="{{ user.Фамилия }}" class="form-input"
                    placeholder="Иванов" oninput="validateFIO(this)">
                <div class="text-description text-gray-400">Только буквы и дефисы</div>
            </div>
            <div class="flex flex-col gap-2">
                <label class="text-description font-bold text-gray-400 uppercase">Имя</label>
                <input type="text" name="first_name" value="{{ user.Имя }}" class="form-input"
                    placeholder="Иван" oninput="validateFIO(this)">
                <div class="text-description text-gray-400">Только буквы и дефисы</div>
            </div>
            <div class="flex flex-col gap-2">
```

```
                <label class="text-description font-bold text-gray-400 uppercase">Отчество</label>
                <input type="text" name="middle_name" value="{{ user.Отчество or '' }}" placeholder="Иванович"
                    class="form-input" oninput="validateFIO(this)">
                <div class="text-description text-gray-400">Только буквы и дефисы</div>
            </div>
            <div class="flex flex-col gap-2">
                <label class="text-description font-bold text-gray-400 uppercase">Логин</label>
                <input type="text" name="login" value="{{ user.Логин }}" class="form-input bg-gray-50" readonly>
            </div>
            <div class="flex flex-col gap-2">
                <label class="text-description font-bold text-gray-400 uppercase">Телефон</label>
                <input type="tel" name="phone" value="{{ formatPhone(user.Телефон or '') }}"
                    placeholder="+7(900)-000-00-00" class="form-input" oninput="formatPhoneInput(this)"
                    onblur="validatePhone(this)">
                <div class="text-description text-gray-400">Формат: +7(XXX)-XXX-XX-XX</div>
            </div>
            <div class="flex flex-col gap-2">
                <label class="text-description font-bold text-gray-400 uppercase">Дата рождения</label>
                <input type="date" name="birth_date" value="{{ user.Дата_рождения }}" class="form-input"
```

```
                    onblur="validateBirthDate(this)" max="{{ max_date }}">
                </div>

                <div class="col-span-2 pt-4">
                    <button type="submit" onclick="return validateForm()"
                        class="border-2 border-gray-900 text-gray-900 px-6 py-2 rounded-md font-bold hover:bg-gray-900 hover:text-white">
                        СОХРАНИТЬ ИЗМЕНЕНИЯ
                    </button>
                </div>
            </form>
        </section>

        <!-- В секции безопасности замените форму смены пароля на эту: -->
        <section id="tab-security"
            class="hidden bg-white border border-gray-100 rounded-[12px] p-8 shadow-sm min-h-[578px]">
            <h3 class="text-subheading font-bold text-gray-900 pb-[20px]">Смена пароля</h3>

            <div class="pb-[20px] border-b border-gray-50">
                <form id="password-form" action="/change_password" method="POST"
                    class="max-w-[400px] flex flex-col gap-1">
                    <div class="flex flex-col gap-2">
                        <label class="text-description font-bold text-gray-400 uppercase">Текущий пароль</label>
                        <input type="password" name="current_password" placeholder="Введите текущий пароль" class="form-input"
                            onblur="validateCurrentPassword(this)" required>
```

```
                </div>

                <div class="flex flex-col gap-2">
                    <label class="text-description font-bold text-gray-400 uppercase">Новый пароль</label>
                    <input type="password" name="new_password" placeholder="Минимум 8 символов" class="form-input"
                        onblur="validatePasswordStrength(this)" required>
                </div>

                <div class="flex flex-col gap-2">
                    <label class="text-description font-bold text-gray-400 uppercase">Подтверждение
                        пароля</label>
                    <input type="password" name="confirm_password" placeholder="Повторите новый парль" class="form-input"
                        onblur="validatePasswordMatch(this)" required>
                </div>

                <button type="submit" onclick="return validatePasswordForm()"
                    class="border-2 border-gray-900 text-gray-900 px-6 py-2 rounded-md font-bold hover:bg-gray-900 hover:text-white transition w-fit mt-4">
                    ОБНОВИТЬ ПАРОЛЬ
                </button>
            </form>
        </div>

        <div class="flex justify-between items-center p-4 bg-red-50 rounded-xl border border-red-100">
            <div>
```

```
                <h4 class="text-base font-bold text-red-700">Удаление аккаунта</h4>
                <p class="text-description text-red-600/70">Это действие нельзя отменить. Все данные будут
                    стерты.</p>
            </div>
            <button onclick="confirmDelete()"
                class="bg-red-600 text-white px-6 py-2 rounded-md font-bold hover:bg-red-700 transition">
                УДАЛИТЬ
            </button>
        </div>
    </section>

    </main>
  </div>
</div>

<style>
  .form-input {
    height: 48px;
    padding: 0 16px;
    border: 1px solid #E5E7EB;
    border-radius: 8px;
    outline: none;
    font-size: 14px;
    transition: border-color 0.2s;
  }

  .form-input:focus {
```

```
        border-color: #3B82F6;
    }

    .form-input.error {
        border-color: #EF4444;
        background-color: #FEF2F2;
    }

    .active-tab {
        background-color: #3B82F6;
        color: white !important;
    }

    .error-message {
        color: #EF4444;
        font-size: 12px;
        margin-top: 4px;
        display: none;
    }
</style>

<script>
    // Установка максимальной даты (сегодня) для даты рождения
    const today = new Date();
    const maxDate = today.toISOString().split('T')[0];
    document.querySelector('input[name="birth_date"]').max = maxDate;

    // Функция для форматирования телефона из БД
    function formatPhone(phone) {
        if (!phone) return '';
```

```
        // Убираем все нецифровые символы
        const cleaned = phone.replace(/\D/g, '');
        // Форматируем в +7(XXX)-XXX-XX-XX
        if (cleaned.length === 11 && cleaned.startsWith('7')) {
            return `+7(${cleaned.substring(1, 4)})-${cleaned.substring(4, 7)}-${cleaned.substring(7, 9)}-${cleaned.substring(9, 11)}`;
        } else if (cleaned.length === 10) {
            return `+7(${cleaned.substring(0, 3)})-${cleaned.substring(3, 6)}-${cleaned.substring(6, 8)}-${cleaned.substring(8, 10)}`;
        }
        return phone;
    }

    // Автоматическое форматирование телефона при вводе
    function formatPhoneInput(input) {
        let value = input.value.replace(/\D/g, '');

        if (value.startsWith('8')) {
            value = '7' + value.substring(1);
        }

        if (value.length > 0) {
            // Если начинается не с +7 или 7, добавляем +7
            if (!value.startsWith('7')) {
                value = '7' + value;
            }

            // Ограничиваем длину до 11 цифр (7 + 10)
            value = value.substring(0, 11);
```

```
        // Форматируем
        let formatted = '+7';
        if (value.length > 1) {
            formatted += '(' + value.substring(1, 4);
        }
        if (value.length >= 4) {
            formatted += ')-' + value.substring(4, 7);
        }
        if (value.length >= 7) {
            formatted += '-' + value.substring(7, 9);
        }
        if (value.length >= 9) {
            formatted += '-' + value.substring(9, 11);
        }

        input.value = formatted;
    }

    clearError(input);
}

// Валидация телефона
function validatePhone(input) {
    const value = input.value.replace(/\D/g, '');
    const phonePattern = /^7\d{10}$/;

    if (!value) {
        showError(input, 'Телефон обязателен для заполнения');
        return false;
    }
```

```
        if (!phonePattern.test(value)) {
            showError(input, 'Введите корректный номер телефона (10 цифр после +7)');
            return false;
        }

        clearError(input);
        return true;
    }

    // Валидация ФИО (без цифр)
    function validateFIO(input) {
        const value = input.value.trim();
        const fioPattern = /^[А-Яа-яЁёA-Za-z\s\-]+$/;

        // Очищаем от цифр
        if (/\d/.test(value)) {
            input.value = value.replace(/\d/g, '');
        }

        if (!value) {
            clearError(input);
            return true;
        }

        if (!fioPattern.test(value)) {
            showError(input, 'Можно использовать только буквы, пробелы и дефисы');
            return false;
```

```
    }

    // Проверка на минимальную длину
    if (value.length < 2) {
        showError(input, 'Минимум 2 символа');
        return false;
    }

    // Проверка на первую заглавную букву
    const firstChar = value.charAt(0);
    if (!/[А-ЯA-ZЁ]/.test(firstChar)) {
        showError(input, 'Первая буква должна быть заглавной');
        return false;
    }

    clearError(input);
    return true;
}

// Валидация даты рождения
function validateBirthDate(input) {
    const value = input.value;
    if (!value) {
        clearError(input);
        return true;
    }

    const birthDate = new Date(value);
    const today = new Date();
```

```
    // Проверка что дата не в будущем
    if (birthDate > today) {
        showError(input, 'Дата рождения не может быть в будущем');
        return false;
    }

    // Проверка что возраст не менее 14 лет
    const minAgeDate = new Date();
    minAgeDate.setFullYear(today.getFullYear() - 14);
    if (birthDate > minAgeDate) {
        showError(input, 'Вам должно быть не менее 14 лет');
        return false;
    }

    // Проверка что возраст не более 120 лет
    const maxAgeDate = new Date();
    maxAgeDate.setFullYear(today.getFullYear() - 120);
    if (birthDate < maxAgeDate) {
        showError(input, 'Проверьте правильность даты рождения');
        return false;
    }

    clearError(input);
    return true;
}

// Валидация пароля
function validatePassword(input) {
    const value = input.value;
```

```
    if (!value) {
        showError(input, 'Пароль обязателен');
        return false;
    }

    if (value.length < 6) {
        showError(input, 'Пароль должен содержать минимум 6 символов');
        return false;
    }

    clearError(input);
    return true;
}

// Валидация всей формы профиля
function validateForm() {
    const form = document.getElementById('profile-form');
    let isValid = true;

    // Проверяем ФИО
    const lastName = form.querySelector('input[name="last_name"]');
    const firstName = form.querySelector('input[name="first_name"]');

    if (!validateFIO(lastName)) isValid = false;
    if (!validateFIO(firstName)) isValid = false;

    // Проверяем телефон
    const phone = form.querySelector('input[name="phone"]');
    if (!validatePhone(phone)) isValid = false;
```

```
    // Проверяем дату рождения
    const birthDate = form.querySelector('input[name="birth_date"]');
    if (birthDate.value && !validateBirthDate(birthDate)) isValid = false;

    return isValid;
}

// Валидация формы смены пароля
function validatePasswordForm() {
    const form = document.getElementById('password-form');
    const password = form.querySelector('input[name="new_password"]');
    return validatePassword(password);
}

// Функции для работы с ошибками
function showError(field, message) {
    clearError(field);
    field.classList.add('error');

    let errorDiv = document.createElement('div');
    errorDiv.className = 'error-message';
    errorDiv.textContent = message;
    errorDiv.id = field.name + '-error';

    field.parentNode.appendChild(errorDiv);
    errorDiv.style.display = 'block';
}

function clearError(field) {
    field.classList.remove('error');
```

```
        const errorDiv = document.getElementById(field.name + '-error');
        if (errorDiv) {
            errorDiv.remove();
        }
    }

    // Табы
    function switchTab(tabName) {
        // Скрываем все секции
        document.getElementById('tab-personal').classList.add('hidden');
        document.getElementById('tab-security').classList.add('hidden');

        // Снимаем активный класс со всех кнопок
        document.getElementById('btn-personal').classList.remove('active-tab', 'text-white');
        document.getElementById('btn-personal').classList.add('text-gray-500');
        document.getElementById('btn-security').classList.remove('active-tab', 'text-white');
        document.getElementById('btn-security').classList.add('text-gray-500');

        // Показываем нужную
        document.getElementById('tab-' + tabName).classList.remove('hidden');

        // Делаем кнопку активной
        const activeBtn = document.getElementById('btn-' + tabName);
        activeBtn.classList.add('active-tab', 'text-white');
        activeBtn.classList.remove('text-gray-500');
    }

    function confirmDelete() {
```

```
        if (confirm("Вы уверены? Все ваши лицензии будут аннулированы.")) {
            window.location.href = "/delete_account";
        }
    }

    // Автоматическая валидация при потере фокуса
    document.addEventListener('DOMContentLoaded', function () {
        const form = document.getElementById('profile-form');

        ['last_name', 'first_name', 'middle_name', 'phone',
'birth_date'].forEach(fieldName => {
            const field = form.querySelector(`input[name="${fieldName}"]`);
            if (field) {
                field.addEventListener('blur', function () {
                    if (fieldName === 'phone') validatePhone(this);
                    else if (fieldName === 'birth_date') validateBirthDate(this);
                    else if (['last_name', 'first_name', 'middle_name'].includes(fieldName))
validateFIO(this);
                });
            }
        });
    });

    // Валидация текущего пароля
    function validateCurrentPassword(input) {
        const value = input.value;
        if (!value) {
            showError(input, 'Введите текущий пароль');
            return false;
        }
```

```
        clearError(input);
        return true;
    }

    // Проверка сложности нового пароля
    function validatePasswordStrength(input) {
        const value = input.value;
        const passwordPattern = /^(?=.*[A-Za-z])(?=.*\d)[A-Za-z\d@$!%*#?&]{8,}$/;

        if (!value) {
            showError(input, 'Введите новый пароль');
            return false;
        }

        if (value.length < 8) {
            showError(input, 'Пароль должен содержать минимум 8 символов');
            return false;
        }

        if (!passwordPattern.test(value)) {
            showError(input, 'Пароль должен содержать буквы и цифры');
            return false;
        }

        // Проверка на совпадение с текущим паролем
        const currentPassword = document.querySelector('input[name="current_password"]');
        if (currentPassword && value === currentPassword.value) {
            showError(input, 'Новый пароль должен отличаться от текущего');
```

```
            return false;
        }

        clearError(input);
        return true;
    }

    // Проверка совпадения паролей
    function validatePasswordMatch(input) {
        const confirmValue = input.value;
        const newPassword =
document.querySelector('input[name="new_password"]').value;

        if (!confirmValue) {
            showError(input, 'Подтвердите новый пароль');
            return false;
        }

        if (confirmValue !== newPassword) {
            showError(input, 'Пароли не совпадают');
            return false;
        }

        clearError(input);
        return true;
    }

    // Валидация всей формы смены пароля
    function validatePasswordForm() {
        const form = document.getElementById('password-form');
```

```
        let isValid = true;

        // Проверяем все поля
        const currentPass = form.querySelector('input[name="current_password"]');
        const newPass = form.querySelector('input[name="new_password"]');
        const confirmPass = form.querySelector('input[name="confirm_password"]');

        if (!validateCurrentPassword(currentPass)) isValid = false;
        if (!validatePasswordStrength(newPass)) isValid = false;
        if (!validatePasswordMatch(confirmPass)) isValid = false;

        return isValid;
    }
</script>
{% endblock %}
```

Код – register.html

```
{% extends "base.html" %}
{% block title %}Регистрация | SoftKey{% endblock %}

{% block content %}
<div class="flex items-center justify-center min-h-[calc(100vh-96px)] bg-gray-50">
    <div class="flex w-[750px] h-[608px] bg-white rounded-[8px] shadow-2xl overflow-hidden">

        <div class="w-[375px] bg-primary p-10 flex flex-col justify-between text-white">
            <div>
                <div class="flex items-center gap-2 mb-8">
```

```
            <i class="ri-user-add-line text-heading"></i>
            <span style="font-family: 'Pacifico', cursive;" class="text-subheading">SoftKey</span>
        </div>
        <h1 class="text-heading font-bold leading-tight mb-4">Присоединяйтесь к нам</h1>
        <p class="text-base opacity-90">Создайте аккаунт, чтобы сохранять свои лицензии и получать персональные скидки.</p>
    </div>
    <div class="text-description opacity-75 italic">
      * Регистрация занимает не более 30 секунд
    </div>
  </div>

  <div class="w-[375px] p-10 flex flex-col justify-center">
    <h2 class="text-subheading font-bold text-gray-800 mb-2">Регистрация</h2>
    <p class="text-description text-gray-500 mb-6">Создайте новый профиль пользователя</p>

    <form action="{{ url_for('register') }}" method="POST" class="space-y-4">
        <div>
            <label class="block text-description uppercase font-bold text-gray-400 mb-1">Ваше Имя</label>
            <input type="text" name="name" required placeholder="Иван"
                class="w-full px-4 py-2 bg-gray-50 border border-gray-200 rounded-lg text-base outline-none focus:ring-2 focus:ring-primary">
        </div>
        <div>
```

```
            <label class="block text-description uppercase font-bold text-gray-400 mb-1">Email (Логин)</label>
            <input type="email" name="email" required placeholder="mail@example.com"
                class="w-full px-4 py-2 bg-gray-50 border border-gray-200 rounded-lg text-base outline-none focus:ring-2 focus:ring-primary">
        </div>
        <div>
            <label class="block text-description uppercase font-bold text-gray-400 mb-1">Пароль</label>
            <input type="password" name="password" required placeholder="••••••••"
                class="w-full px-4 py-2 bg-gray-50 border border-gray-200 rounded-lg text-base outline-none focus:ring-2 focus:ring-primary">
        </div>

        <button type="submit" class="w-full bg-secondary text-white py-3 rounded-lg font-bold hover:bg-green-600 transition mt-4 shadow-lg shadow-green-100">
            Зарегистрироваться
        </button>
    </form>

    <div class="mt-6 text-center">
        <a href="{{ url_for('login') }}" class="text-base text-primary hover:underline font-medium">
            <i class="ri-arrow-left-line"></i> Вернуться ко входу
        </a>
    </div>
</div>
```

```
    </div>
</div>
{% endblock %}
```

Код – app.py

```
from flask import Flask, render_template, session, request, redirect, url_for, flash, jsonify
import pymysql as db
from config import host, user, password, port, db_name
import uuid
from datetime import datetime, timedelta
import os
from werkzeug.utils import secure_filename

app = Flask(__name__)
app.secret_key = 'softkey_secret_key'

# Папка, куда будут сохраняться изображения товаров (должна существовать!)
UPLOAD_FOLDER = os.path.join('static', 'images')
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
# Разрешенные расширения файлов
ALLOWED_EXTENSIONS = {'png', 'jpg', 'jpeg', 'gif'}

def allowed_file(filename):
    return '.' in filename and \
           filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS

def get_db_connection():
    try:
        return db.connect(
```

```
            host=host, port=port, user=user, password=password,
            database=db_name, cursorclass=db.cursors.DictCursor, autocommit=True
        )
    except Exception as ex:
        print("Ошибка подключения:", ex)
        return None

@app.route('/index')
@app.route('/')
def index():
    category_id = request.args.get('category')
    search_query = request.args.get('search')
    sort_price = request.args.get('sort_price') # asc / desc
    sort_date = request.args.get('sort_date')   # new / old

    conn = get_db_connection()
    categories = []
    products = []

    if conn:
        with conn.cursor() as cursor:
            # Получаем все категории для выпадающего списка
            cursor.execute("SELECT * FROM Категории")
            categories = cursor.fetchall()

            # Строим запрос
            sql = "SELECT t.*, k.Название_категории FROM Товары t JOIN Категории k ON t.ID_Категории = k.ID_Категории WHERE 1=1"
            params = []
```

```
            if category_id and category_id != 'all':
                sql += " AND t.ID_Категории = %s"
                params.append(category_id)
            if search_query:
                sql += " AND t.Название LIKE %s"
                params.append(f"%{search_query}%")

            # Сортировка
            order_clauses = []
            if sort_price == 'asc': order_clauses.append("t.Цена ASC")
            elif sort_price == 'desc': order_clauses.append("t.Цена DESC")

            if sort_date == 'new': order_clauses.append("t.ID_Товара DESC")
            elif sort_date == 'old': order_clauses.append("t.ID_Товара ASC")

            if order_clauses:
                sql += " ORDER BY " + ", ".join(order_clauses)

            cursor.execute(sql, tuple(params))
            products = cursor.fetchall()
        conn.close()

    return render_template('index.html', products=products, categories=categories)

@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        login_input = request.form['login']
        password_input = request.form['password']
```

```
        conn = get_db_connection()
        if conn:
            with conn.cursor() as cursor:
                # Обязательно выбираем ID_Роли
                sql = "SELECT * FROM Пользователи WHERE Логин = %s AND
Пароль = %s"
                cursor.execute(sql, (login_input, password_input))
                user = cursor.fetchone()

                if user:
                    # Сохраняем данные в сессию
                    session['user_id'] = user['ID_Пользователя']
                    session['role_id'] = user['ID_Роли'] # Сохраняем ID роли
                    session['user_name'] = user['Имя']

                    flash(f"Добро пожаловать, {user['Имя']}!", "success")

                    # ПЕРЕНАПРАВЛЕНИЕ ПО РОЛИ
                    if user['ID_Роли'] == 1:
                        return redirect(url_for('admin_dashboard')) # В админку
                    else:
                        return redirect(url_for('index')) # На главную для клиентов
                else:
                    flash("Неверный логин или пароль", "error")
            conn.close()

    return render_template('login.html')


@app.route('/register', methods=['GET', 'POST'])
```

```
def register():
    if request.method == 'POST':
        email = request.form['email']
        password_val = request.form['password']
        name = request.form.get('name', 'Пользователь')

        conn = get_db_connection()
        if conn:
            try:
                with conn.cursor() as cursor:
                    # Проверяем, нет ли уже такого логина
                    cursor.execute("SELECT * FROM Пользователи WHERE Логин = %s", (email,))
                    if cursor.fetchone():
                        flash('Пользователь с такой почтой уже существует', 'error')
                    else:
                        # ID_Роли = 2 (Обычный клиент), Фамилия обязательна по БД
                        sql = "INSERT INTO Пользователи (Фамилия, Имя, Логин, Пароль, ID_Роли) VALUES (%s, %s, %s, %s, %s)"
                        cursor.execute(sql, ('Клиент', name, email, password_val, 2))
                        flash('Регистрация успешна! Теперь войдите.', 'success')
                        return redirect(url_for('login'))
            except Exception as e:
                flash(f'Ошибка: {str(e)}', 'error')
            finally:
                conn.close()
    return render_template('register.html')

# --- ОТОБРАЖЕНИЕ КОРЗИНЫ ---
@app.route('/cart')
```

```
def cart():
    if 'user_id' not in session:
        flash("Войдите, чтобы пользоваться корзиной", "error")
        return redirect(url_for('login'))

    conn = get_db_connection()
    items = []
    total_price = 0
    if conn:
        with conn.cursor() as cursor:
            sql = """
                SELECT k.*, t.Название, t.Цена, t.Описание, t.Изображение,
cat.Название_категории
                FROM Корзина k
                JOIN Товары t ON k.ID_Товара = t.ID_Товара
                JOIN Категории cat ON t.ID_Категории = cat.ID_Категории
                WHERE k.ID_Пользователя = %s
            """
            cursor.execute(sql, (session['user_id'],))
            items = cursor.fetchall()
            # Считаем сумму
            total_price = sum(item['Цена'] * item['Количество'] for item in items)
        conn.close()
    return render_template('cart.html', items=items, total_price=total_price)

# --- ИЗМЕНЕНИЕ КОЛИЧЕСТВА ---
@app.route('/cart/update/<int:product_id>/<action>')
def update_cart(product_id, action):
    if 'user_id' not in session: return redirect(url_for('login'))
```

```
    conn = get_db_connection()
    if conn:
        with conn.cursor() as cursor:
            if action == 'plus':
                cursor.execute("UPDATE Корзина SET Количество = Количество + 1 WHERE ID_Пользователя = %s AND ID_Товара = %s", (session['user_id'], product_id))
            elif action == 'minus':
                # Удаляем, если количество станет 0
                cursor.execute("UPDATE Корзина SET Количество = Количество - 1 WHERE ID_Пользователя = %s AND ID_Товара = %s AND Количество > 1", (session['user_id'], product_id))
        conn.close()
    return redirect(url_for('cart'))

# --- УДАЛЕНИЕ ИЗ КОРЗИНЫ ---
@app.route('/cart/remove/<int:product_id>')
def remove_from_cart(product_id):
    if 'user_id' not in session: return redirect(url_for('login'))
    conn = get_db_connection()
    if conn:
        with conn.cursor() as cursor:
            cursor.execute("DELETE FROM Корзина WHERE ID_Пользователя = %s AND ID_Товара = %s", (session['user_id'], product_id))
        conn.close()
    return redirect(url_for('cart'))



@app.route('/checkout', methods=['POST'])
def checkout():
```

```
    if 'user_id' not in session:
        return redirect(url_for('login'))

    conn = get_db_connection()
    if conn:
        try:
            with conn.cursor() as cursor:
                # 1. Получаем товары из корзины
                cursor.execute("""
                    SELECT k.ID_Товара, k.Количество, t.Цена
                    FROM Корзина k
                    JOIN Товары t ON k.ID_Товара = t.ID_Товара
                    WHERE k.ID_Пользователя = %s
                """, (session['user_id'],))
                cart_items = cursor.fetchall()

                if not cart_items:
                    flash("Корзина пуста", "error")
                    return redirect(url_for('cart'))

                # 2. Считаем общую сумму
                total_sum = sum(item['Цена'] * item['Количество'] for item in
cart_items)

                # 3. Создаем основной заказ
                cursor.execute("INSERT INTO Заказы (ID_Пользователя,
Дата_заказа, Статус, Итоговая_сумма) VALUES (%s, NOW(), 'Оплачен', %s)",
                               (session['user_id'], total_sum))
                order_id = cursor.lastrowid
```

```
            # 4. Для каждого товара в корзине...
            for item in cart_items:
                # Вставляем в Состав_заказа
                cursor.execute("""
                    INSERT INTO Состав_заказа (ID_Заказа, ID_Товара, Цена_продажи, Срок_лицензии_дни, Количество)
                    VALUES (%s, %s, %s, %s, %s)
                """, (order_id, item['ID_Товара'], item['Цена'], 365, item['Количество']))

                pos_id = cursor.lastrowid # ID только что созданной строки в составе заказа

                # 5. Генерируем лицензионные ключи (по одному на каждую единицу товара)
                for _ in range(item['Количество']):
                    new_key = str(uuid.uuid4()).upper()[:18] # Пример: 123E4567-E89B-12D3
                    # Рассчитываем дату истечения (например, + год)
                    expire_date = datetime.now() + timedelta(days=365)

                    cursor.execute("""
                        INSERT INTO Лицензии (ID_Позиции_заказа, Лицензионный_ключ, Дата_активации, Дата_истечения)
                        VALUES (%s, %s, NOW(), %s)
                    """, (pos_id, new_key, expire_date))

            # 6. Очищаем корзину
            cursor.execute("DELETE FROM Корзина WHERE ID_Пользователя = %s", (session['user_id'],))
```

```
            flash(f"Заказ #{order_id} успешно оформлен! Ключи созданы.", "success")
        except Exception as ex:
            print("Ошибка оформления:", ex)
            flash("Ошибка при создании заказа", "error")
        finally:
            conn.close()

    return redirect(url_for('orders'))

@app.route('/product/<int:product_id>')
def product_detail(product_id):
    conn = get_db_connection()
    product = None

    if conn:
        with conn.cursor() as cursor:
            sql = """
                SELECT t.*, k.Название_категории
                FROM Товары t
                JOIN Категории k ON t.ID_Категории = k.ID_Категории
                WHERE t.ID_Товара = %s
            """
            cursor.execute(sql, (product_id,))
            product = cursor.fetchone()
        conn.close()

    if not product:
        flash("Товар не найден", "error")
```

```
        return redirect(url_for('index'))

    return render_template('product_detail.html', product=product)


# --- ДОБАВЛЕНИЕ В КОРЗИНУ ---
@app.route('/add_to_cart/<int:product_id>', methods=['POST'])
def add_to_cart(product_id):
    if 'user_id' not in session:
        flash("Войдите в аккаунт, чтобы добавить товар в корзину", "error")
        return redirect(url_for('login'))

    conn = get_db_connection()
    if conn:
        try:
            with conn.cursor() as cursor:
                # Проверяем, есть ли уже такой товар в корзине у этого пользователя
                cursor.execute("SELECT * FROM Корзина WHERE ID_Пользователя
= %s AND ID_Товара = %s",
                               (session['user_id'], product_id))
                item = cursor.fetchone()

                if item:
                    # Если есть — увеличиваем количество
                    cursor.execute("UPDATE Корзина SET Количество = Количество +
1 WHERE ID_Пользователя = %s AND ID_Товара = %s",
                                   (session['user_id'], product_id))
                else:
                    # Если нет — добавляем новую запись
```

```
                cursor.execute("INSERT INTO Корзина (ID_Пользователя,
ID_Товара, Количество) VALUES (%s, %s, 1)",
                               (session['user_id'], product_id))

            flash("Товар добавлен в корзину!", "success")
        except Exception as ex:
            print("Ошибка добавления в корзину:", ex)
            flash("Не удалось добавить товар", "error")
        finally:
            conn.close()

    # Возвращаемся обратно в корзину или на ту же страницу
    return redirect(url_for('cart'))

@app.route('/profile')
def profile():
    if 'user_id' not in session:
        return redirect(url_for('login'))

    conn = get_db_connection()
    user_data = None
    orders = []

    if conn:
        with conn.cursor() as cursor:
            cursor.execute("SELECT * FROM Пользователи WHERE
ID_Пользователя = %s", (session['user_id'],))
            user_data = cursor.fetchone()
```

```
        cursor.execute("SELECT * FROM Заказы WHERE ID_Пользователя = %s ORDER BY Дата_заказа DESC", (session['user_id'],))
        orders = cursor.fetchall()
    conn.close()

    # Рассчитываем максимальную дату (сегодня) для ограничения выбора даты рождения
    from datetime import date
    max_date = date.today().isoformat()

    # Функция для форматирования телефона
    def format_phone(phone):
        if not phone:
            return ''
        cleaned = ''.join(filter(str.isdigit, str(phone)))
        if len(cleaned) == 11 and cleaned.startswith('7'):
            return f"+7({cleaned[1:4]})-{cleaned[4:7]}-{cleaned[7:9]}-{cleaned[9:11]}"
        return phone

    return render_template('profile.html',
                           user=user_data,
                           orders=orders,
                           formatPhone=format_phone,
                           max_date=max_date)

# --- ВЫХОД ИЗ СИСТЕМЫ ---
@app.route('/logout')
def logout():
    session.clear() # Полная очистка сессии
```

```
    flash("Вы вышли из системы", "success")
    return redirect(url_for('index'))

# --- ОБНОВЛЕНИЕ ЛИЧНЫХ ДАННЫХ ---
@app.route('/update_profile', methods=['POST'])
def update_profile():
    if 'user_id' not in session:
        return redirect(url_for('login'))

    # Получаем данные из формы
    last_name = request.form.get('last_name')
    first_name = request.form.get('first_name')
    middle_name = request.form.get('middle_name')
    phone = request.form.get('phone')
    birth_date = request.form.get('birth_date')

    # Очищаем телефон от форматирования (оставляем только цифры)
    if phone:
        phone = ''.join(filter(str.isdigit, phone))
        # Если номер начинается с +7, оставляем 7
        if phone.startswith('7'):
            phone = phone
        elif len(phone) == 10:
            phone = '7' + phone
        elif len(phone) == 11 and phone.startswith('8'):
            phone = '7' + phone[1:]

    conn = get_db_connection()
    if conn:
        try:
```

```
        with conn.cursor() as cursor:
            # Проверяем, не занят ли телефон другим пользователем
            if phone:
                cursor.execute("SELECT ID_Пользователя FROM Пользователи WHERE Телефон = %s AND ID_Пользователя != %s",
                               (phone, session['user_id']))
                if cursor.fetchone():
                    flash("Этот телефон уже используется другим пользователем", "error")
                    return redirect(url_for('profile'))

            # Обновляем только разрешенные поля (логин не трогаем)
            sql = """
                UPDATE Пользователи
                SET Фамилия = %s, Имя = %s, Отчество = %s, Телефон = %s, Дата_рождения = %s
                WHERE ID_Пользователя = %s
            """
            cursor.execute(sql, (last_name, first_name, middle_name, phone, birth_date, session['user_id']))
        flash("Данные успешно обновлены!", "success")
    except Exception as ex:
        print("Ошибка при обновлении профиля:", ex)
        flash("Ошибка при обновлении данных", "error")
    finally:
        conn.close()

    return redirect(url_for('profile'))

# --- СМЕНА ПАРОЛЯ ---
```

```
@app.route('/change_password', methods=['POST'])
def change_password():
    if 'user_id' not in session:
        return redirect(url_for('login'))

    current_password = request.form.get('current_password')
    new_password = request.form.get('new_password')
    confirm_password = request.form.get('confirm_password')

    # Валидация на стороне сервера
    if not current_password or not new_password or not confirm_password:
        flash("Все поля обязательны для заполнения", "error")
        return redirect(url_for('profile'))

    if len(new_password) < 8:
        flash("Пароль должен содержать минимум 8 символов", "error")
        return redirect(url_for('profile'))

    # Проверка сложности пароля (хотя бы одна буква и одна цифра)
    if not any(c.isalpha() for c in new_password) or not any(c.isdigit() for c in new_password):
        flash("Пароль должен содержать буквы и цифры", "error")
        return redirect(url_for('profile'))

    if new_password != confirm_password:
        flash("Пароли не совпадают", "error")
        return redirect(url_for('profile'))

    conn = get_db_connection()
    if conn:
```

```
        try:
            with conn.cursor() as cursor:
                # Проверяем текущий пароль
                cursor.execute("SELECT Пароль FROM Пользователи WHERE
ID_Пользователя = %s",
                               (session['user_id'],))
                user_data = cursor.fetchone()

                if not user_data:
                    flash("Пользователь не найден", "error")
                    return redirect(url_for('profile'))

                # Проверяем, совпадает ли текущий пароль
                if user_data['Пароль'] != current_password:
                    flash("Неверный текущий пароль", "error")
                    return redirect(url_for('profile'))

                # Проверяем, не совпадает ли новый пароль с текущим
                if new_password == current_password:
                    flash("Новый пароль должен отличаться от текущего", "error")
                    return redirect(url_for('profile'))

                # Обновляем пароль
                sql = "UPDATE Пользователи SET Пароль = %s WHERE
ID_Пользователя = %s"
                cursor.execute(sql, (new_password, session['user_id']))

                flash("Пароль успешно изменен!", "success")

        except Exception as ex:
```

```
            print("Ошибка при смене пароля:", ex)
            flash("Ошибка при смене пароля", "error")
        finally:
            conn.close()

    return redirect(url_for('profile'))

# --- УДАЛЕНИЕ АККАУНТА ---
@app.route('/delete_account')
def delete_account():
    if 'user_id' not in session:
        return redirect(url_for('login'))

    user_id = session['user_id']
    conn = get_db_connection()
    if conn:
        try:
            with conn.cursor() as cursor:
                # Важно: если в БД нет каскадного удаления,
                # сначала нужно удалить товары из корзины этого пользователя
                cursor.execute("DELETE FROM Корзина WHERE ID_Пользователя =
%s", (user_id,))
                # Затем удаляем самого пользователя
                cursor.execute("DELETE FROM Пользователи WHERE
ID_Пользователя = %s", (user_id,))

            session.clear()
            flash("Ваш аккаунт был полностью удален", "success")
            return redirect(url_for('index'))
        except Exception as ex:
```

```
            print("Ошибка при удалении:", ex)
            flash("Не удалось удалить аккаунт (возможно, у вас есть активные заказы)", "error")
        finally:
            conn.close()

    return redirect(url_for('profile'))

@app.route('/orders')
def orders():
    if 'user_id' not in session:
        return redirect(url_for('login'))

    conn = get_db_connection()
    orders_list = []

    if conn:
        with conn.cursor() as cursor:
            # Получаем заказы, товары и их ключи одним запросом
            cursor.execute("""
                SELECT
                    z.ID_Заказа, z.Дата_заказа, z.Статус, z.Итоговая_сумма,
                    sz.ID_Позиции, sz.Количество, sz.Цена_продажи,
                    t.Название,
                    l.Лицензионный_ключ
                FROM Заказы z
                JOIN Состав_заказа sz ON z.ID_Заказа = sz.ID_Заказа
                JOIN Товары t ON sz.ID_Товара = t.ID_Товара
                LEFT JOIN Лицензии l ON sz.ID_Позиции = l.ID_Позиции_заказа
                WHERE z.ID_Пользователя = %s
```

```
            ORDER BY z.Дата_заказа DESC
        """, (session['user_id'],))
        rows = cursor.fetchall()

        # Группируем данные: Заказ -> Позиции -> Ключи
        orders_dict = {}
        for row in rows:
            o_id = row['ID_Заказа']
            if o_id not in orders_dict:
                orders_dict[o_id] = {
                    'ID_Заказа': o_id,
                    'Дата_заказа': row['Дата_заказа'],
                    'Статус': row['Статус'],
                    'Итоговая_сумма': row['Итоговая_сумма'],
                    'products_list': {} # Используем словарь для группировки по ID позиции
                }

            pos_id = row['ID_Позиции']
            if pos_id not in orders_dict[o_id]['products_list']:
                orders_dict[o_id]['products_list'][pos_id] = {
                    'Название': row['Название'],
                    'Количество': row['Количество'],
                    'Цена_продажи': row['Цена_продажи'],
                    'keys': []
                }

            # Добавляем ключ, если он есть
            if row['Лицензионный_ключ']:
```

```
orders_dict[o_id]['products_list'][pos_id]['keys'].append(row['Лицензионный_ключ'])

            # Превращаем в список для шаблона
            for o_id in orders_dict:
                order = orders_dict[o_id]
                # Превращаем словарь продуктов обратно в список
                order['products_list'] = list(order['products_list'].values())
                orders_list.append(order)

        conn.close()

    # Получаем данные пользователя для шапки
    user_data = None
    conn = get_db_connection()
    if conn:
        with conn.cursor() as cursor:
            cursor.execute("SELECT * FROM Пользователи WHERE ID_Пользователя = %s", (session['user_id'],))
            user_data = cursor.fetchone()
        conn.close()

    return render_template('orders.html', user=user_data, orders=orders_list)


@app.route('/admin')
def admin_dashboard():
    # Проверка на права админа (ID_Роли = 1)
    if 'user_id' not in session or session.get('role_id') != 1:
```

```
        flash("Доступ запрещен", "error")
        return redirect(url_for('index'))

    conn = get_db_connection()
    stats = {}
    products = []
    orders = []
    categories = []
    reports = {}
    if conn:
        with conn.cursor() as cursor:
            # 1. Считаем статистику
            cursor.execute("SELECT SUM(Итоговая_сумма) as total FROM Заказы
WHERE Статус = 'Оплачен'")
            stats['revenue'] = cursor.fetchone()['total'] or 0

            cursor.execute("SELECT COUNT(*) as count FROM Заказы")
            stats['orders_count'] = cursor.fetchone()['count']

            cursor.execute("SELECT COUNT(*) as count FROM Товары")
            stats['products_count'] = cursor.fetchone()['count']

            cursor.execute("SELECT COUNT(*) as count FROM Пользователи")
            stats['users_count'] = cursor.fetchone()['count']

            # 2. Список товаров с категориями
            cursor.execute("""
                SELECT t.*, k.Название_категории
                FROM Товары t
                LEFT JOIN Категории k ON t.ID_Категории = k.ID_Категории
```

```
        """)
        products = cursor.fetchall()

        cursor.execute("SELECT * FROM Категории")
        categories = cursor.fetchall()

        # 3. Список заказов с именами пользователей
        cursor.execute("""
            SELECT z.*, p.Имя, p.Фамилия
            FROM Заказы z
            JOIN Пользователи p ON z.ID_Пользователя = p.ID_Пользователя
            ORDER BY z.Дата_заказа DESC
        """)
        orders = cursor.fetchall()

        # 1. ТОП-5 популярных товаров
        cursor.execute("""
            SELECT t.Название, SUM(sz.Количество) as total_qty
            FROM Состав_заказа sz
            JOIN Товары t ON sz.ID_Товара = t.ID_Товара
            GROUP BY t.ID_Товара
            ORDER BY total_qty DESC LIMIT 5
        """)
        reports['top_products'] = cursor.fetchall()

        # 2. Выручка по категориям
        cursor.execute("""
            SELECT k.Название_категории, SUM(sz.Цена_продажи *
sz.Количество) as total_revenue
            FROM Состав_заказа sz
```

```
            JOIN Товары t ON sz.ID_Товара = t.ID_Товара
            JOIN Категории k ON t.ID_Категории = k.ID_Категории
            GROUP BY k.ID_Категории
            ORDER BY total_revenue DESC
        """)
        reports['category_revenue'] = cursor.fetchall()

        # 3. Активность продаж по дням (последние 14)
        cursor.execute("""
            SELECT DATE(Дата_заказа) as day, COUNT(ID_Заказа) as order_count, SUM(Итоговая_сумма) as daily_sum
            FROM Заказы
            WHERE Статус = 'Оплачен'
            GROUP BY day
            ORDER BY day DESC LIMIT 14
        """)
        reports['daily_sales'] = cursor.fetchall()

        # 4. ТОП-5 Покупателей (Самые ценные клиенты)
        cursor.execute("""
            SELECT p.Имя, p.Фамилия, p.Логин, SUM(z.Итоговая_сумма) as total_spent
            FROM Заказы z
            JOIN Пользователи p ON z.ID_Пользователя = p.ID_Пользователя
            WHERE z.Статус = 'Оплачен'
            GROUP BY p.ID_Пользователя
            ORDER BY total_spent DESC LIMIT 5
        """)
        reports['vip_customers'] = cursor.fetchall()
```

```
        conn.close()

    return render_template('admin.html', stats=stats, products=products, orders=orders, reports=reports, categories=categories)

# Маршрут для удаления товара
@app.route('/admin/delete_product/<int:product_id>')
def delete_product(product_id):
    if 'user_id' not in session or session.get('role_id') != 1:
        return redirect(url_for('login'))

    conn = get_db_connection()
    if conn:
        with conn.cursor() as cursor:
            # Проверяем, нет ли товара в заказах, чтобы не нарушить целостность (опционально)
            cursor.execute("DELETE FROM Товары WHERE ID_Товара = %s", (product_id,))
        conn.close()
        flash("Товар успешно удален", "success")
    return redirect(url_for('admin_dashboard'))

# Маршрут для редактирования товара
@app.route('/admin/edit_product', methods=['POST'])
def edit_product():
    if 'user_id' not in session or session.get('role_id') != 1:
        return redirect(url_for('login'))

    # Получаем текстовые данные из формы
    p_id = request.form.get('id')
```

```
    name = request.form.get('name')
    price = request.form.get('price')
    cat_id = request.form.get('category_id')
    desc = request.form.get('description')

    # Проверка цены на отрицательное значение
    try:
        price_float = float(price)
        if price_float < 0:
            flash("Цена не может быть отрицательной", "error")
            return redirect(url_for('admin_dashboard'))
    except ValueError:
        flash("Неверный формат цены", "error")
        return redirect(url_for('admin_dashboard'))

    # Получаем файл изображения
    file = request.files.get('image')
    print(f"DEBUG: Получен файл: {file}")
    if file:
        print(f"DEBUG: Имя файла: {file.filename}")

    filename_to_save = None

    # Если файл был загружен и у него разрешенное расширение
    if file and file.filename != '' and allowed_file(file.filename):
        # Безопасное имя файла (защита от хакеров)
        filename = secure_filename(file.filename)
        # Генерируем уникальное имя, чтобы не затереть старые (добавляем ID
товара)
        filename_to_save = f"product_{p_id}_{filename}"
```

```
            # Сохраняем файл на диск
            file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename_to_save))

    conn = get_db_connection()
    if conn:
        with conn.cursor() as cursor:
            # Формируем SQL запрос динамически
            sql = "UPDATE Товары SET Название = %s, Цена = %s, ID_Категории
= %s, Описание = %s"
            params = [name, price, cat_id, desc]

            # Если было загружено НОВОЕ изображение, добавляем его в запрос
            if filename_to_save:
                sql += ", Изображение = %s"
                params.append(filename_to_save)

            # Завершаем запрос условием WHERE
            sql += " WHERE ID_Товара = %s"
            params.append(p_id)

            cursor.execute(sql, tuple(params))
        conn.close()
        flash("Данные товара обновлены", "success")

    return redirect(url_for('admin_dashboard'))

# --- ДОБАВЛЕНИЕ ТОВАРА (АДМИН) ---
@app.route('/add_product', methods=['POST'])
def add_product():
    if 'role_id' not in session or session['role_id'] != 1:
```

```
    return redirect(url_for('login'))

name = request.form.get('name')
price = request.form.get('price')
cat_id = request.form.get('category')
desc = request.form.get('description')

# Проверка цены на отрицательное значение
try:
    price_float = float(price)
    if price_float < 0:
        flash("Цена не может быть отрицательной", "error")
        return redirect(url_for('admin_dashboard'))
except ValueError:
    flash("Неверный формат цены", "error")
    return redirect(url_for('admin_dashboard'))

# Обработка изображения
file = request.files.get('image')
filename_to_save = "default.jpg"

if file and file.filename != '' and allowed_file(file.filename):
    filename = secure_filename(file.filename)
    # Генерируем уникальное имя
    filename_to_save = f"new_{uuid.uuid4().hex}_{filename}"
    file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename_to_save))



conn = get_db_connection()
if conn:
```

```
        with conn.cursor() as cursor:
            sql = "INSERT INTO Товары (Название, Цена, ID_Категории, Описание, Изображение) VALUES (%s, %s, %s, %s, %s)"
            cursor.execute(sql, (name, price, cat_id, desc, filename_to_save))
        conn.close()
        flash("Товар успешно добавлен!", "success")

    return redirect(url_for('admin_dashboard'))

if __name__ == '__main__':
    app.run(debug=True)
```