**Данная документация описывает предоставленные фрагменты исходного кода информационной системы «Один», иллюстрирующие структуру веб-приложения на базе фреймворка Flask с использованием HTML, CSS и JavaScript для пользовательского интерфейса.**

***1. app.py (Серверная часть, Flask)***

Назначение: Основной файл серверного приложения. Реализует веб-сервер с использованием микрофреймворка Flask, обрабатывает входящие HTTP-запросы, управляет пользовательскими сессиями и взаимодействует с шаблонами страниц.

Flask(\_\_name\_\_, template\_folder='site'): Создание экземпляра Flask-приложения. Указывает Flask искать HTML-шаблоны в папке site.

app.secret\_key = '...': Настройка секретного ключа для обеспечения безопасности сессий пользователя.

login\_required: Функция-декоратор. Используется для защиты маршрутов (@app.route). Перед выполнением целевой функции он проверяет наличие ключа 'user' в пользовательской сессии. Если пользователь не авторизован, он перенаправляет на маршрут 'login', сохраняя запрошенный URL в параметре next.

@app.route('/...'): Декораторы Flask, связывающие URL-адреса с определенными функциями-обработчиками.

/login (login()): Обрабатывает запросы к странице авторизации. GET-запросы отображают шаблон login.html. POST-запросы считывают данные из формы (request.form.get()), выполняют проверку учетных данных (сравнение с '[admin@example.com](https://www.google.com/url?sa=E&q=mailto%3Aadmin%40example.com)' и '123'). При успехе устанавливает ключ 'user' в сессии и перенаправляет пользователя (либо на страницу, куда он хотел попасть (request.args.get('next')), либо на /prod). При ошибке отображает шаблон логина с сообщением об ошибке.

/logout (logout()): Обрабатывает запросы выхода из системы. Удаляет ключ 'user' из сессии и перенаправляет на страницу логина.

/ (home() ): Обрабатывает запросы к корневому URL. Перенаправляет либо на /prod (если залогинен), либо на /login.

/prod (prod() ): Обрабатывает запросы к основной странице продукта. Защищен декоратором @login\_required. Если пользователь авторизован, отображает шаблон prod.html, передавая в него информацию о залогиненном пользователе из сессии.

if \_\_name\_\_ == '\_\_main\_\_': app.run(...): Стандартный блок запуска приложения. Запускает встроенный в Flask веб-сервер для разработки на указанном хосте ('0.0.0.0') и порту (5000), делая его доступным в локальной сети. debug=True включает режим отладки.

**2. login.html (Шаблон страницы авторизации)**

HTML-шаблон для отображения формы входа в систему. Используется функцией login() в app.py.

Базовая структура HTML5 документа (<!DOCTYPE html>, <html>, <head>, <body>).

<head>: Содержит мета-информацию и ссылку (<link rel="stylesheet" href="../static/css/style.css">) на файл стилей CSS.

<body>: Содержит обертки (div с классами wrapper, title) для центрирования и стилизации формы.

<form action="{{ url\_for('login') }}" method="POST">: HTML-форма для отправки данных авторизации. Атрибут action использует Jinja2-синтаксис {{ url\_for('login') }}, который Flask заменит на корректный URL маршрута /login. Метод POST указывает способ отправки данных.

<div class="field">: Обертки для полей ввода.

<input type="text" name="email" required>: Поле ввода для логина/email. Атрибут name="email" важен для Flask (request.form.get('email')). required делает поле обязательным для заполнения браузером.

<input type="password" name="password" required>: Поле ввода для пароля. Атрибут name="password" важен для Flask. required делает поле обязательным.

<div class="content">: Контейнер для дополнительных опций.

<div class="pass-link">: Содержит ссылку "Forgot password?".

<input type="submit" value="Login">: Кнопка отправки формы.

**3. prod.html (Шаблон основной страницы приложения)**

HTML-шаблон для отображения основной рабочей области системы после успешной авторизации. Используется функцией prod() в app.py.

<head>: Содержит мета-информацию, заголовок страницы, определение пользовательского шрифта (@font-face) и встроенный блок стилей CSS (<style>).

Встроенный CSS (<style>): Определяет визуальное оформление и макет страницы (header, sidebar, main content), стили для меню, кнопок, форм фильтрации, таблиц и их элементов. Включает правила для скрытия/отображения боковой панели и секций контента.

<body>:

<div class="header">: Верхняя панель с логотипом и действиями (Info, Log out).

<div class="sidebar">: Левая боковая панель навигации с заголовком и ссылками на разделы (Заказы, Клиенты, Товары, Менеджеры). Ссылки имеют классы active и onclick события для JavaScript.

<div class="main">: Основная область контента. Содержит top-bar (с кнопкой для переключения видимости сайдбара и главной кнопкой действия) и контейнеры для различных секций (orders-section, clients-section), которые по умолчанию скрыты.

<div class="orders-section">: Содержит кнопку для переключения видимости формы фильтров и саму форму (<form class="filter-form">) с различными полями ввода и выпадающими списками. Также содержит HTML-структуру таблицы (<table>) для отображения списка заказов с заголовками столбцов и пустым телом (<tbody>).

<div class="clients-section">: Содержит HTML-структуру таблицы (<table>) для отображения списка клиентов с заголовками столбцов и примером одной строки данных.

Встроенный блок скриптов (<script>): Содержит JavaScript-функции:

addOrder(): Демонстрационная функция для динамического добавления строки в таблицу заказов.

addClient(): Демонстрационная функция для динамического добавления строки в таблицу клиентов.

toggleSidebar(): Переключает CSS-классы (hidden, sidebar-hidden) для боковой панели и основного контента, управляя их видимостью и отступами.

switchSection(section): Скрывает или отображает соответствующий раздел контента (ordersSection или clientsSection), меняет текст и onclick действие на главной кнопке в top-bar, и активирует соответствующий пункт в боковом меню.

toggleFilters(): Переключает CSS-класс active для формы фильтров, управляя ее видимостью.

**4. ../static/css/style.css (для login.html)**

Внешний файл стилей CSS, используемый шаблоном login.html.