Тестовое задание (ОС в течение 7 дней)

Задание: Разработка системы управления заказами в кафе

Описание Необходимо разработать полнофункциональное веб-приложение на Django для управления заказами в кафе. Приложение должно позволять добавлять, удалять, искать, изменять и отображать заказы. Каждый заказ должен содержать следующие поля:

- **id** (уникальный идентификатор, генерируется автоматически)
- **table_number** (номер стола)
- **items** (список заказанных блюд с ценами)
- **total_price** (общая стоимость заказа, вычисляется автоматически)
- **status** (статус заказа: “в ожидании”, “готово”, “оплачено”)

Стек технологий:

- **Python 3.8+**
- **Django 4+** (включая Django ORM для работы с базой данных)
- **HTML/CSS** (для базового пользовательского интерфейса)
- **SQLite/PostgreSQL** (для хранения данных)
- **Markdown** для написания README.md

Функциональные требования:

1. **Добавление заказа:**
    - Через веб-интерфейс пользователь вводит номер стола и список блюд с ценами. Система автоматически добавляет заказ с уникальным ID, рассчитанной стоимостью и статусом “в ожидании”.
2. **Удаление заказа:**
    - Пользователь через веб-интерфейс выбирает заказ по ID и удаляет его из системы.
3. **Поиск заказа:**
    - Возможность поиска заказов по номеру стола или статусу через поисковую строку.
4. **Отображение всех заказов:**
    - Веб-страница с таблицей всех заказов, отображающая их ID, номер стола, список блюд, общую стоимость и статус.
5. **Изменение статуса заказа:**
    - Пользователь через интерфейс выбирает заказ по ID и изменяет его статус (“в ожидании”, “готово”, “оплачено”).
6. **Расчет выручки за смену:**

    - Отдельная страница или модуль для расчета общего объема выручки за заказы со статусом “оплачено”.

Дополнительные требования:

- **Хранение данных:** Использование базы данных SQLite/PostgreSQL для хранения информации о заказах.
- **Обработка ошибок:** Обеспечение корректной обработки ошибок (например, при попытке удаления несуществующего заказа или ввода некорректных данных).
- **CRUD операции:** Реализация операций создания, чтения, обновления и удаления заказов через веб-интерфейс.
- **REST API:** Дополнительно, предоставить API для работы с заказами (добавление, удаление, поиск и т. д.).
- **ООП:** Использовать принципы ООП для построения модели Order и сопутствующих бизнес-логик.

Требования к проекту:

1. **Корректность и полнота функционала.**
2. **Чистота и читаемость кода.**
3. **Обработка ошибок и исключений.**
4. **Удобство использования веб-интерфейса.**
5. **Структура проекта:** Логическое разделение на приложения, модели, представления, шаблоны и маршруты.

Будет плюсом:

1. **Документация:**
    - Аннотирование функций и переменных (например, с использованием typing).
    - Подробное описание функций и основных блоков кода.
    - README файл с инструкцией по установке и использованию приложения.
2. **Тестирование:**
    - Покрытие ключевых функций тестами с использованием unittest или Pytest.
3. **Дополнительные возможности:**
    - Возможность редактирования заказа (добавление или удаление блюд).
    - Фильтрация списка заказов по статусу.

Формат сдачи: Проект должен быть предоставлен в виде ссылки на репозиторий GitHub или GitLab с открытым доступом.

Инструкция по выполнению:

1. Создать проект Django с разделением на приложение для управления заказами.
2. Использовать Django ORM для работы с базой данных.
3. Реализовать HTML-страницы с использованием шаблонов Django.

4. Подготовить README.md с инструкцией по развертыванию проекта.