```
# FastAPI-сервис с CRUD для интеграции с расширением AI IDE BAS
*(Тестовое задание)*

Данный репозиторий содержит минимальный веб-сервис на FastAPI, реализующий полноценный

## ✅ Требования

- Python ≥ 3.13
- [uv](https://docs.astral.sh/uv/) (рекомендуется) или стандартные `venv`/`pip`
- Node.js ≥ 18 (для запуска расширения в режиме разработки)
- Visual Studio Code (для отладки расширения)

## 🚀 Реализованные компоненты

- RESTful API с 4 стандартными CRUD-эндпоинтами для сущности `Item`:
  - `POST /items` — создание новой записи
  - `GET /items` — получение списка всех записей
  - `GET /items/{id}` — получение записи по идентификатору
  - `PUT /items/{id}` — обновление существующей записи
  - `DELETE /items/{id}` — удаление записи
- Постоянное хранилище на основе SQLite (файл: `test.db`)
- Middleware CORS, разрешающий запросы с любых источников (требуется для взаимодействи
- Самодостаточный проект с управлением зависимостями через `uv`

## 📦 Установка и запуск

### 1. Инициализация FastAPI-сервиса

```bash
git clone https://github.com/ваш-логин/fastapi-crud-test.git
cd fastapi-crud-test
uv venv
source .venv/bin/activate  # Linux/macOS
uv add "fastapi>=0.127.0" "uvicorn[standard]>=0.34.0"
```

2. Запуск сервера

```
make run
# или напрямую:
uvicorn main:app --reload
```

Сервис будет доступен по адресу `http://localhost:8000`.
Интерактивная документация: `http://localhost:8000/docs`.

3. Интеграция с расширением AI IDE BAS
```

> **Примечание**: данный этап выполняется в репозитории AI IDE BAS.

1. Клонируйте расширение:

```
git clone https://github.com/dradns/AI-IDE-BAS.git
cd AI-IDE-BAS/ai_ide_bas_main
```

2. Установите зависимости:

```
pnpm install
```

3. В файле `src/core/webview/ClineProvider.ts` были внесены следующие изменения:
   - В тело HTML (в методах `getHtmlContent()` и `getHMRHtmlContent()`) добавлена фиксированная кнопка.
   - Встроен скрипт (с корректным CSP-`nonce`), выполняющий `POST`-запрос на `http://localhost:8000/items` при нажатии.
   - Расширена политика CSP (`connect-src`) для разрешения соединений с `http://localhost:8000`.
4. Запустите Vite-сервер для Webview-интерфейса:

```
pnpm --filter webview-ui dev
```

5. Запустите расширение в режиме отладки из VS Code (`F5`).
6. Откройте панель AI IDE BAS → нажмите кнопку **«🚀 Отправить в FastAPI»**.

🔍 Верификация работы

При нажатии кнопки:

- Расширение отправляет JSON-объект на эндпоинт `POST /items`.
- FastAPI-сервер сохраняет запись в файл `test.db` и возвращает HTTP 200.
- Успешность подтверждается:
  - Логом запроса в терминале (`INFO: ... "POST /items HTTP/1.1" 200 OK`)
  - Новой записью в таблице `items`
  - Визуальной обратной связью (эффект нажатия кнопки через `onmousedown` / `onmouseup`)

Пример отправляемых данных:

```
{
  "name": "Test from AI IDE BAS",
  "description": "Sent via custom button in VS Code extension"
}
```

🗂️ Структура проекта

```
.
├── main.py             # Основной модуль FastAPI с CRUD-логикой и интеграцией SQLite
├── Makefile            # Удобная команда запуска: `make run`
├── pyproject.toml      # Метаданные проекта и зависимости
└── .gitignore          # Исключает виртуальное окружение, БД и файлы IDE
```

📝 Примечания

- Файл базы данных (`test.db`) создаётся автоматически при первом запуске в корне проекта.
- CORS настроен в режиме `allow_origins=["*"]` для обеспечения совместимости с локальным Webview.
- Кнопка и скрипт внедрены напрямую в HTML-обёртку Webview, что исключает необходимость модификации React-приложения.

🧪 Ручное тестирование

API можно протестировать независимо, например, через `curl` :

```
curl -X POST http://localhost:8000/items \
  -H "Content-Type: application/json" \
  -d '{"name":"Интеграционный тест","description":"Отправлено через curl"}'
```

✉️ Автор

Реализация тестового задания по интеграции FastAPI-сервиса с расширением AI IDE BAS.

```
---
и это работает 🚀
```