# **HW3 Cursor - Телеграм-бот заметок + напоминаний (SQLite + MCP)**

### Бот готов к использованию!

Теперь вы можете:

* Создавать заметки через Telegram
* Устанавливать напоминания
* Искать заметки по содержимому
* Удалять заметки
* Просматривать статистику
* Анализировать работу через мониторинг

Попробуйте отправить /start вашему боту в Telegram! 🎉

4. 🧪 Протестируйте команды:

Создайте заметку:

/add Купить молоко

Посмотрите список заметок:

/list

Найдите заметки:

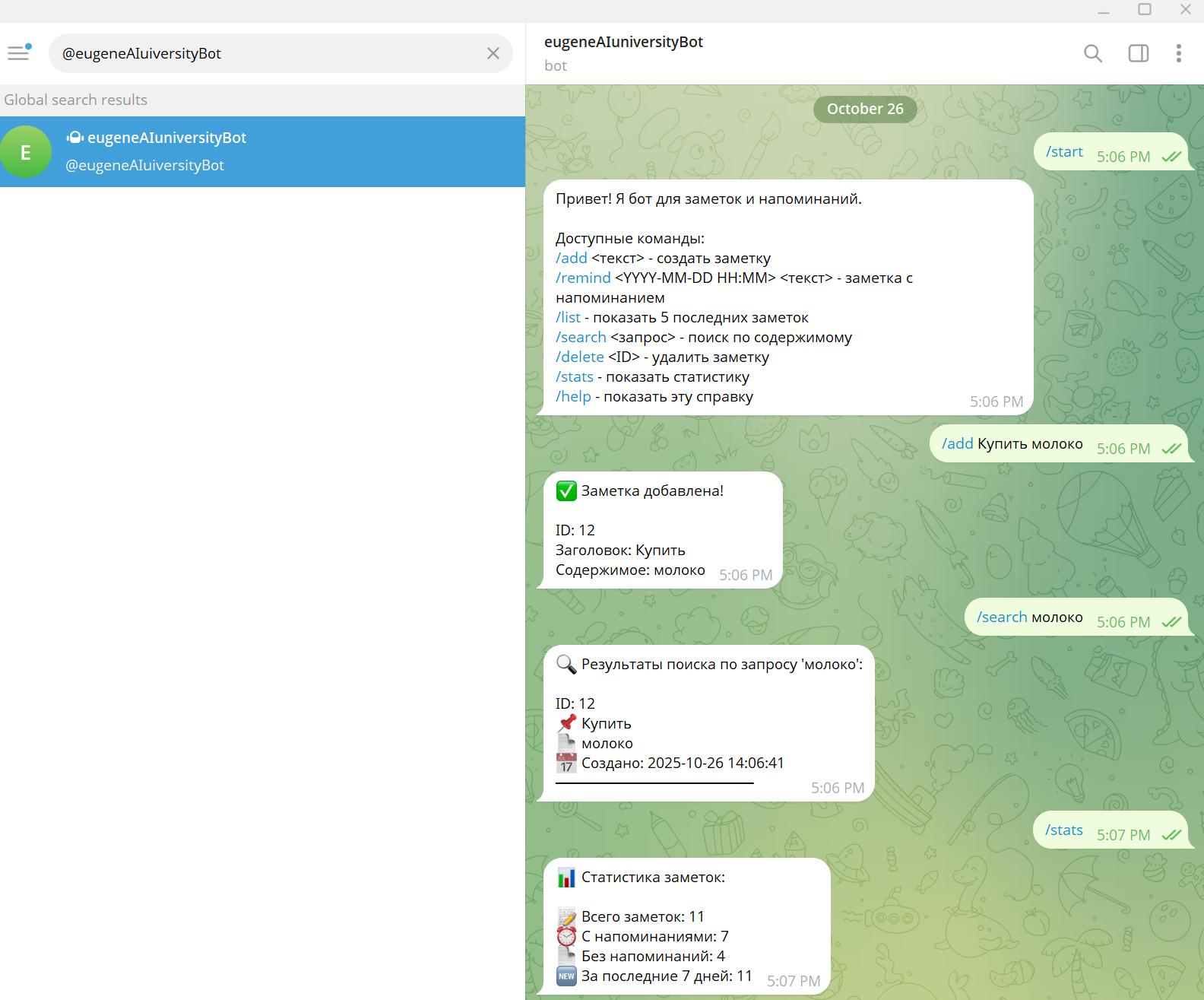
/search молоко

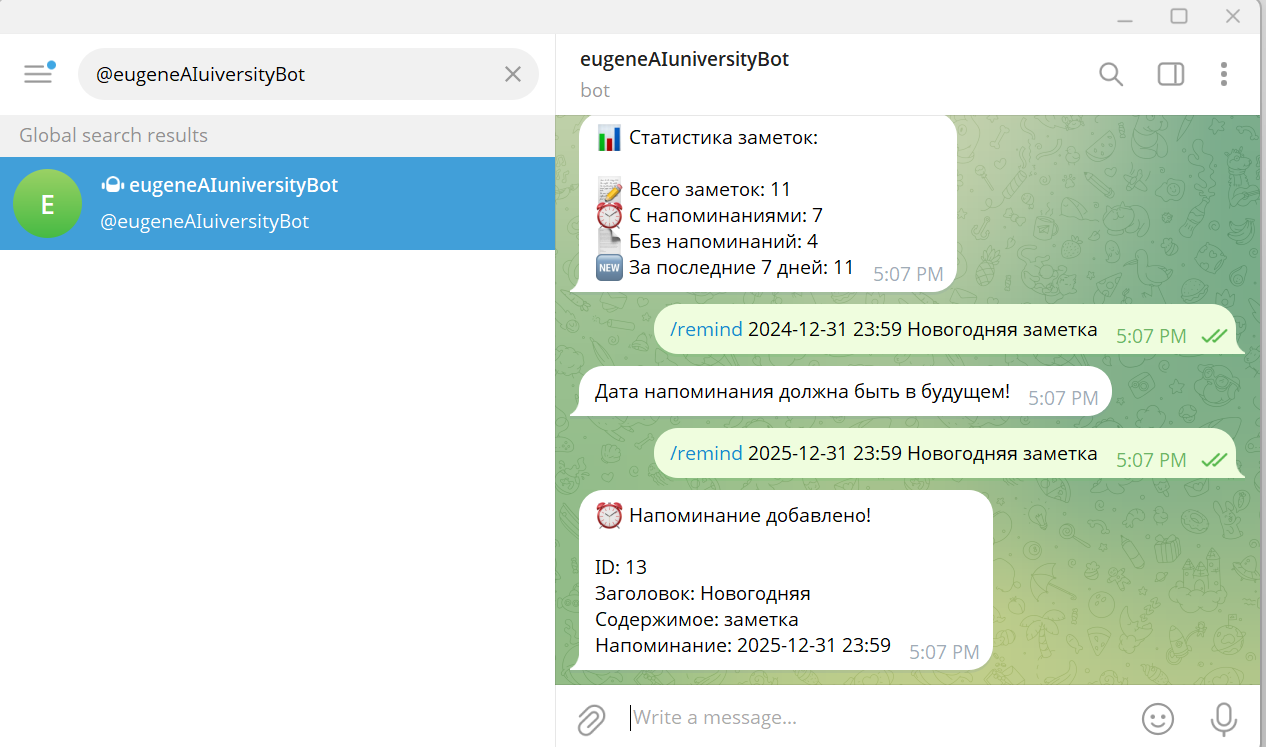
Посмотрите статистику:

/stats

Создайте напоминание:

/remind 2025-12-31 23:59 Новогодняя заметка





5. 🔧 Если бот не отвечает:

Проверьте, что бот запущен:

cd notes-mcp-sqlite

python monitor.py

Перезапустите бота:

cd notes-mcp-sqlite

$env:BOT\_TOKEN="8268378989:AAF1ETbzngLCBqmdf\_C8G-Ma0PCmYGGjkqo"

python bot.py

6. 📊 Мониторинг работы:

После тестирования команд запустите мониторинг:

cd notes-mcp-sqlite

python monitor.py

Вы увидите статистику действий пользователей и работу бота.

7. 🎯 результат:

Анализ работы телеграм-бота...

============================================================

ОТЧЕТ О РАБОТЕ ТЕЛЕГРАМ-БОТА ЗАМЕТОК

============================================================

Время генерации: 2025-10-26 17:13:53

📊 СТАТИСТИКА БАЗЫ ДАННЫХ:

📝 Всего заметок: 12

⏰ С напоминаниями: 8

📄 Без напоминаний: 4

📅 АКТИВНОСТЬ ПО ДНЯМ (последние 7 дней):

2025-10-26: 12 заметок

🏆 ТОП ЗАГОЛОВКОВ:

'Позвонить маме': 1 раз

'Подготовить презентацию': 1 раз

'Новогодняя': 1 раз

'Купить продукты': 1 раз

'Купить подарок': 1 раз

📋 АНАЛИЗ ЛОГОВ:

📁 Файл лога: logs\bot\_20251026.log

📏 Размер файла: 1425 байт

📝 Всего строк: 13

👥 Уникальных пользователей: 1

🎯 СТАТИСТИКА ДЕЙСТВИЙ:

start: 1 раз

add\_note: 1 раз

search: 1 раз

stats: 1 раз

add\_reminder: 1 раз

============================================================

Отчет сохранен: logs\bot\_report\_20251026\_171353.txt

Статус системы: healthy (DB), healthy (Logs)

# **TESTS**

**1. demo\_queries**  
PS E:\Python\Cursor\3\neuro-bot-telegram-remainder-mcp\notes-mcp-sqlite> python .\demo\_queries.py

SQLite Notes Database Demo

This demonstrates queries that can be run through MCP in Cursor

==================================================

Query: Show table structure

SQL: SELECT sql FROM sqlite\_master WHERE type='table' AND name='notes'

==================================================

Columns: sql

Results (1 rows):

1. {'sql': 'CREATE TABLE notes (\n id INTEGER PRIMARY KEY AUTOINCREMENT,\n title TEXT NOT NULL,\n content TEXT NOT NULL,\n due\_at DATETIME,\n created\_at DATETIME DEFAULT CURRENT\_TIMESTAMP\n )'}

==================================================

Query: Show all notes (newest first)

SQL: SELECT id, title, content, due\_at, created\_at FROM notes ORDER BY created\_at DESC

==================================================

Columns: id, title, content, due\_at, created\_at

Results (12 rows):

1. {'id': 13, 'title': 'Новогодняя', 'content': 'заметка', 'due\_at': '2025-12-31 23:59', 'created\_at': '2025-10-26 14:07:51'}

2. {'id': 12, 'title': 'Купить', 'content': 'молоко', 'due\_at': None, 'created\_at': '2025-10-26 14:06:41'}

3. {'id': 1, 'title': 'Важная встреча', 'content': 'Встреча с клиентом в 15:00', 'due\_at': '2024-01-15 15:00:00', 'created\_at': '2025-10-26 13:23:29'}

4. {'id': 2, 'title': 'Купить продукты', 'content': 'Молоко, хлеб, яйца', 'due\_at': None, 'created\_at': '2025-10-26 13:23:29'}

5. {'id': 3, 'title': 'Позвонить маме', 'content': 'Позвонить маме на выходных', 'due\_at': '2024-01-14 18:00:00', 'created\_at': '2025-10-26 13:23:29'}

6. {'id': 4, 'title': 'Подготовить презентацию', 'content': 'Слайды для проекта X', 'due\_at': '2024-01-16 10:00:00', 'created\_at': '2025-10-26 13:23:29'}

7. {'id': 5, 'title': 'Записаться к врачу', 'content': 'Проверить здоровье', 'due\_at': None, 'created\_at': '2025-10-26 13:23:29'}

8. {'id': 6, 'title': 'Важно: оплатить счета', 'content': 'Коммунальные услуги до 20 числа', 'due\_at': '2024-01-20 23:59:00', 'created\_at': '2025-10-26 13:23:29'}

9. {'id': 7, 'title': 'Изучить новый фреймворк', 'content': 'React для фронтенда', 'due\_at': None, 'created\_at': '2025-10-26 13:23:29'}

10. {'id': 8, 'title': 'Важно: дедлайн проекта', 'content': 'Сдать проект до конца месяца', 'due\_at': '2024-01-31 23:59:00', 'created\_at': '2025-10-26 13:23:29'}

11. {'id': 9, 'title': 'Встреча с командой', 'content': 'Еженедельный стендап', 'due\_at': '2024-01-17 09:00:00', 'created\_at': '2025-10-26 13:23:29'}

12. {'id': 10, 'title': 'Купить подарок', 'content': 'День рождения друга', 'due\_at': '2024-01-18 12:00:00', 'created\_at': '2025-10-26 13:23:29'}

==================================================

Query: Show notes from last 3 days

SQL: SELECT id, title, content, due\_at, created\_at FROM notes WHERE created\_at >= '2025-10-23 17:51:12' ORDER BY created\_at DESC

==================================================

Columns: id, title, content, due\_at, created\_at

Results (12 rows):

1. {'id': 13, 'title': 'Новогодняя', 'content': 'заметка', 'due\_at': '2025-12-31 23:59', 'created\_at': '2025-10-26 14:07:51'}

2. {'id': 12, 'title': 'Купить', 'content': 'молоко', 'due\_at': None, 'created\_at': '2025-10-26 14:06:41'}

3. {'id': 1, 'title': 'Важная встреча', 'content': 'Встреча с клиентом в 15:00', 'due\_at': '2024-01-15 15:00:00', 'created\_at': '2025-10-26 13:23:29'}

4. {'id': 2, 'title': 'Купить продукты', 'content': 'Молоко, хлеб, яйца', 'due\_at': None, 'created\_at': '2025-10-26 13:23:29'}

5. {'id': 3, 'title': 'Позвонить маме', 'content': 'Позвонить маме на выходных', 'due\_at': '2024-01-14 18:00:00', 'created\_at': '2025-10-26 13:23:29'}

6. {'id': 4, 'title': 'Подготовить презентацию', 'content': 'Слайды для проекта X', 'due\_at': '2024-01-16 10:00:00', 'created\_at': '2025-10-26 13:23:29'}

7. {'id': 5, 'title': 'Записаться к врачу', 'content': 'Проверить здоровье', 'due\_at': None, 'created\_at': '2025-10-26 13:23:29'}

8. {'id': 6, 'title': 'Важно: оплатить счета', 'content': 'Коммунальные услуги до 20 числа', 'due\_at': '2024-01-20 23:59:00', 'created\_at': '2025-10-26 13:23:29'}

9. {'id': 7, 'title': 'Изучить новый фреймворк', 'content': 'React для фронтенда', 'due\_at': None, 'created\_at': '2025-10-26 13:23:29'}

10. {'id': 8, 'title': 'Важно: дедлайн проекта', 'content': 'Сдать проект до конца месяца', 'due\_at': '2024-01-31 23:59:00', 'created\_at': '2025-10-26 13:23:29'}

11. {'id': 9, 'title': 'Встреча с командой', 'content': 'Еженедельный стендап', 'due\_at': '2024-01-17 09:00:00', 'created\_at': '2025-10-26 13:23:29'}

12. {'id': 10, 'title': 'Купить подарок', 'content': 'День рождения друга', 'due\_at': '2024-01-18 12:00:00', 'created\_at': '2025-10-26 13:23:29'}

==================================================

Query: Show reminders for next 24 hours

SQL: SELECT id, title, content, due\_at FROM notes WHERE due\_at IS NOT NULL AND due\_at BETWEEN '2025-10-26 17:51:12' AND '2025-10-27 17:51:12' ORDER BY due\_at ASC

==================================================

No results found.

==================================================

Query: Find notes with 'важно' in title

SQL: SELECT id, title, content, due\_at FROM notes WHERE title LIKE '%важно%' OR title LIKE '%Важно%'

==================================================

Columns: id, title, content, due\_at

Results (2 rows):

1. {'id': 6, 'title': 'Важно: оплатить счета', 'content': 'Коммунальные услуги до 20 числа', 'due\_at': '2024-01-20 23:59:00'}

2. {'id': 8, 'title': 'Важно: дедлайн проекта', 'content': 'Сдать проект до конца месяца', 'due\_at': '2024-01-31 23:59:00'}

==================================================

Query: Show statistics

SQL: SELECT COUNT(\*) as total\_notes, COUNT(due\_at) as notes\_with\_reminders FROM notes

==================================================

Columns: total\_notes, notes\_with\_reminders

Results (1 rows):

1. {'total\_notes': 12, 'notes\_with\_reminders': 8}

==================================================

Query: Show notes without reminders

SQL: SELECT id, title, content FROM notes WHERE due\_at IS NULL ORDER BY created\_at DESC

==================================================

Columns: id, title, content

Results (4 rows):

1. {'id': 12, 'title': 'Купить', 'content': 'молоко'}

2. {'id': 2, 'title': 'Купить продукты', 'content': 'Молоко, хлеб, яйца'}

3. {'id': 5, 'title': 'Записаться к врачу', 'content': 'Проверить здоровье'}

4. {'id': 7, 'title': 'Изучить новый фреймворк', 'content': 'React для фронтенда'}

==================================================

Demo completed! These queries can be run through MCP in Cursor

using natural language commands.

==================================================

**2. test\_db**

/neuro-bot-telegram-remainder-mcp/.venv/Scripts/python.exe e:/Python/Cursor/3/neuro-bot-telegram-remainder-mcp/notes-mcp-sqlite/test\_db.py

Total notes: 10

Sample notes:

ID: 1, Title: Важная встреча, Content: Встреча с клиентом в 15:00, Due: 2024-01-15 15:00:00

ID: 2, Title: Купить продукты, Content: Молоко, хлеб, яйца, Due: None

ID: 3, Title: Позвонить маме, Content: Позвонить маме на выходных, Due: 2024-01-14 18:00:00

Notes with reminders: 7

Notes without reminders: 3

Database test completed successfully!

**3. create\_test\_logs**

/neuro-bot-telegram-remainder-mcp/.venv/Scripts/python.exe e:/Python/Cursor/3/neuro-bot-telegram-remainder-mcp/notes-mcp-sqlite/create\_test\_logs.py

🔧 Создание тестовых логов для демонстрации мониторинга...

✅ Создан тестовый лог-файл: logs\bot\_20251026.log

📝 Добавлено 15 записей

✅ Создан дополнительный лог-файл: logs\bot\_20251025.log

📝 Добавлено 4 записей

📊 Статистика созданных логов:

📁 bot\_20251026.log: 15 строк, 1555 байт

📁 bot\_20251025.log: 4 строк, 476 байт

✅ Тестовые логи созданы успешно!

💡 Теперь можно запустить 'python monitor.py' для анализа логов

**4. monitor**

PS E:\Python\Cursor\3\neuro-bot-telegram-remainder-mcp> cd .\notes-mcp-sqlite\

PS E:\Python\Cursor\3\neuro-bot-telegram-remainder-mcp\notes-mcp-sqlite> python monitor.py

Анализ работы телеграм-бота...

============================================================

ОТЧЕТ О РАБОТЕ ТЕЛЕГРАМ-БОТА ЗАМЕТОК

============================================================

Время генерации: 2025-10-26 17:23:32

📊 СТАТИСТИКА БАЗЫ ДАННЫХ:

📝 Всего заметок: 12

⏰ С напоминаниями: 8

📄 Без напоминаний: 4

📅 АКТИВНОСТЬ ПО ДНЯМ (последние 7 дней):

2025-10-26: 12 заметок

🏆 ТОП ЗАГОЛОВКОВ:

'Позвонить маме': 1 раз

'Подготовить презентацию': 1 раз

'Новогодняя': 1 раз

'Купить продукты': 1 раз

'Купить подарок': 1 раз

📋 АНАЛИЗ ЛОГОВ:

📁 Файл лога: logs\bot\_20251026.log

📏 Размер файла: 1425 байт

📝 Всего строк: 13

👥 Уникальных пользователей: 1

🎯 СТАТИСТИКА ДЕЙСТВИЙ:

start: 1 раз

add\_note: 1 раз

search: 1 раз

stats: 1 раз

add\_reminder: 1 раз

============================================================

Отчет сохранен: logs\bot\_report\_20251026\_172332.txt

Статус системы: healthy (DB), healthy (Logs)

**5. clean\_test\_logs**

PS E:\Python\Cursor\3\neuro-bot-telegram-remainder-mcp\notes-mcp-sqlite> python .\clean\_test\_logs.py

🧹 Очистка тестовых логов...

❌ Ошибка при удалении bot\_20251026.log: [WinError 32] The process cannot access the file because it is being used by another process: 'logs\\bot\_20251026.log'

🗑️ Удален: bot\_report\_20251026\_164008.txt

🗑️ Удален: bot\_report\_20251026\_164927.txt

🗑️ Удален: bot\_report\_20251026\_165711.txt

🗑️ Удален: bot\_report\_20251026\_170316.txt

🗑️ Удален: bot\_report\_20251026\_171353.txt

🗑️ Удален: bot\_report\_20251026\_172332.txt

✅ Удалено файлов: 6

📁 Осталось файлов в logs/: 1

- bot\_20251026.log

✅ Очистка завершена!

**6. test\_bot\_functions**

PS E:\Python\Cursor\3\neuro-bot-telegram-remainder-mcp\notes-mcp-sqlite> python .\test\_bot\_functions.py

2025-10-26 17:26:46,565 - notes\_bot - INFO - Starting bot functions test

2025-10-26 17:26:46,601 - notes\_bot - INFO - Bot functions test completed

Тестирование функций бота...

Database initialized successfully

База данных инициализирована

Тест добавления заметки:

Note added: ID=14, Title='Тестовая заметка', Due=None

Добавлена заметка с ID: 14

Тест поиска:

Search performed: query='тест', results=1

Найдено заметок с 'тест': 1

- ID: 14, Заголовок: Тестовая заметка

Тест получения заметки по ID:

Найдена заметка: Тестовая заметка - Это тестовая заметка для проверки функций

Тест статистики:

Всего заметок: 13

С напоминаниями: 8

Без напоминаний: 5

За последние 7 дней: 13

Тест удаления:

Note deleted: ID=14

Заметка с ID 14 успешно удалена

Заметка действительно удалена

Все тесты функций бота завершены успешно!

Для полного тестирования установите BOT\_TOKEN и запустите:

python bot.py

**7. test\_db**

PS E:\Python\Cursor\3\neuro-bot-telegram-remainder-mcp\notes-mcp-sqlite> python .\test\_db.py

Total notes: 12

Sample notes:

ID: 1, Title: Важная встреча, Content: Встреча с клиентом в 15:00, Due: 2024-01-15 15:00:00

ID: 2, Title: Купить продукты, Content: Молоко, хлеб, яйца, Due: None

ID: 3, Title: Позвонить маме, Content: Позвонить маме на выходных, Due: 2024-01-14 18:00:00

Notes with reminders: 8

Notes without reminders: 4

Database test completed successfully!

**8. test\_new\_features**

PS E:\Python\Cursor\3\neuro-bot-telegram-remainder-mcp\notes-mcp-sqlite> python .\test\_new\_features.py

🚀 ТЕСТИРОВАНИЕ НОВЫХ ФУНКЦИЙ ТЕЛЕГРАМ-БОТА

==================================================

🧪 Тестирование новых функций базы данных...

1. Тест поиска заметок:

Найдено заметок с 'важно': 0

2. Тест получения заметки по ID:

Заметка ID 1: Важная встреча

3. Тест статистики:

Всего заметок: 12

С напоминаниями: 8

Без напоминаний: 4

За последние 7 дней: 12

✅ Тесты базы данных завершены!

🔍 Тестирование поиска...

Поиск 'важно': найдено 0 заметок

Поиск 'встреча': найдено 1 заметок

Поиск 'купить': найдено 0 заметок

Поиск 'проект': найдено 2 заметок

Поиск 'врач': найдено 1 заметок

✅ Тесты поиска завершены!

🗑️ Тестирование функции удаления...

Доступные заметки для тестирования удаления:

- ID: 1, Заголовок: Важная встреча

- ID: 2, Заголовок: Купить продукты

- ID: 3, Заголовок: Позвонить маме

- ID: 4, Заголовок: Подготовить презентацию

- ID: 5, Заголовок: Записаться к врачу

Проверка существования заметки ID 1: ✅ Существует

✅ Тесты удаления завершены!

==================================================

🎉 ВСЕ ТЕСТЫ ЗАВЕРШЕНЫ УСПЕШНО!

Новые функции готовы к использованию:

✅ Команда /search - поиск по содержимому

✅ Команда /delete - удаление заметок

✅ Команда /stats - статистика

✅ Логирование всех действий пользователей

✅ Мониторинг и анализ работы бота

# **README:**

# Телеграм-бот заметок + напоминаний (SQLite + MCP)

Проект включает телеграм-бота для создания заметок и напоминаний с интеграцией SQLite через MCP (Model Context Protocol) для работы в Cursor.

## Структура проекта

```

notes-mcp-sqlite/

├── .cursor/

│ └── mcp.json # Конфигурация MCP для Cursor

├── logs/ # Папка с логами (создается автоматически)

├── notes.db # База данных SQLite

├── bot.py # Основной код телеграм-бота

├── create\_db.py # Скрипт создания БД с тестовыми данными

├── monitor.py # Скрипт мониторинга и анализа

├── test\_new\_features.py # Тестирование новых функций

├── create\_test\_logs.py # Создание тестовых логов

├── clean\_test\_logs.py # Очистка тестовых логов

├── requirements.txt # Python зависимости

├── README.md # Этот файл

├── MCP\_SETUP.md # Инструкции по настройке MCP

└── CHANGELOG.md # Список изменений

```

## База данных

База данных `notes.db` содержит таблицу `notes` со следующими полями:

- `id` - уникальный идентификатор (INTEGER PRIMARY KEY)

- `title` - заголовок заметки (TEXT)

- `content` - содержимое заметки (TEXT)

- `due\_at` - дата и время напоминания (DATETIME, может быть NULL)

- `created\_at` - дата создания (DATETIME, по умолчанию CURRENT\_TIMESTAMP)

## Установка и запуск

### 1. Установка зависимостей

```bash

pip install -r requirements.txt

```

### 2. Создание базы данных

База данных уже создана с тестовыми данными. Если нужно пересоздать:

```bash

python create\_db.py

```

### 3. Настройка бота

1. Создайте бота через [@BotFather](https://t.me/BotFather) в Telegram

2. Получите токен бота

3. Установите переменную окружения:

```bash

# Windows PowerShell

$env:BOT\_TOKEN="YOUR\_BOT\_TOKEN\_HERE"

# Windows CMD

set BOT\_TOKEN=YOUR\_BOT\_TOKEN\_HERE

# Linux/macOS

export BOT\_TOKEN="YOUR\_BOT\_TOKEN\_HERE"

```

### 4. Запуск бота

```bash

python bot.py

```

## Команды бота

- `/start` или `/help` - показать справку

- `/add <текст>` - создать заметку

- `/remind <YYYY-MM-DD HH:MM> <текст>` - создать заметку с напоминанием

- `/list` - показать 5 последних заметок

- `/search <запрос>` - поиск по содержимому заметок

- `/delete <ID>` - удалить заметку по ID

- `/stats` - показать статистику заметок

### Примеры использования

```

/add Купить молоко

/add Важная встреча Встреча с клиентом в 15:00

/remind 2024-01-15 15:00 Встреча с командой

/search важно

/search встреча

/delete 5

/stats

/list

```

## MCP интеграция с Cursor

### Настройка MCP

Конфигурация MCP уже настроена в файле `.cursor/mcp.json`. Путь к базе данных указан абсолютный:

```json

{

"mcpServers": {

"SQLite": {

"command": "npx",

"args": ["-y", "mcp-sqlite", "E:/Python/Cursor/3/neuro-bot-telegram-remainder-mcp/notes-mcp-sqlite/notes.db"]

}

}

}

```

### Активация MCP в Cursor

1. Откройте Cursor

2. Перейдите в настройки (Ctrl+,)

3. Найдите раздел "MCP"

4. Скопируйте содержимое `.cursor/mcp.json` в настройки MCP

5. Перезапустите Cursor

### Примеры запросов через MCP

После настройки MCP вы можете использовать естественный язык для работы с базой данных в Cursor Composer:

#### Просмотр структуры базы данных

```

Покажи все таблицы и их поля

```

#### Получение последних заметок

```

Выведи заметки за последние 3 дня (новые сверху)

```

#### Поиск напоминаний

```

Покажи напоминания на ближайшие 24 часа

```

#### Поиск по содержимому

```

Найди заметки, где в заголовке есть "важно"

```

#### Дополнительные примеры запросов

```

Покажи все заметки без напоминаний

Сколько всего заметок в базе?

Покажи заметки, созданные сегодня

Найди все напоминания на завтра

Покажи заметки с напоминаниями, отсортированные по дате

```

## Тестовые данные

В базе данных уже есть 10 тестовых записей, включая:

- Обычные заметки без напоминаний

- Заметки с напоминаниями на разные даты

- Заметки с ключевым словом "важно" в заголовке

## Технические детали

- \*\*Фреймворк\*\*: aiogram 3.4.1

- \*\*База данных\*\*: SQLite

- \*\*MCP сервер\*\*: mcp-sqlite

- \*\*Язык\*\*: Python 3.8+

- \*\*Кодировка\*\*: UTF-8

## Новые функции (v2.0)

### 🔍 Поиск по содержимому

Команда `/search` позволяет искать заметки по заголовку или содержимому:

- `/search важно` - найдет все заметки со словом "важно"

- `/search встреча` - найдет заметки о встречах

- Поиск работает по частичному совпадению (LIKE %query%)

### 🗑️ Удаление заметок

Команда `/delete` позволяет удалять заметки по ID:

- `/delete 5` - удалит заметку с ID 5

- Перед удалением показывается информация о заметке

- Удаление логируется для безопасности

### 📊 Статистика

Команда `/stats` показывает подробную статистику:

- Общее количество заметок

- Количество заметок с напоминаниями

- Количество заметок без напоминаний

- Количество заметок за последние 7 дней

### 📝 Логирование и мониторинг

- Все действия пользователей логируются в файлы `logs/bot\_YYYYMMDD.log`

- Логи содержат информацию о пользователях, действиях и времени

- Скрипт `monitor.py` для анализа работы бота

- Скрипт `test\_new\_features.py` для тестирования функций

## Мониторинг и анализ

### Просмотр логов

```bash

# Запуск мониторинга

python monitor.py

# Тестирование новых функций

python test\_new\_features.py

# Создание тестовых логов для демонстрации

python create\_test\_logs.py

# Очистка тестовых логов

python clean\_test\_logs.py

```

### Структура логов

Логи сохраняются в папке `logs/` с именем `bot\_YYYYMMDD.log` и содержат:

- Время действия

- ID и имя пользователя

- Тип действия (add\_note, delete\_note, search, etc.)

- Детали действия

## Возможные улучшения

- Реализовать систему категорий

- Интегрировать с календарем

- Добавить уведомления о приближающихся напоминаниях

- Реализовать экспорт/импорт заметок

- Добавить команду редактирования заметок

- Реализовать систему тегов