# 🤖 بوت إدارة المجموعات المتقدم

بوت تليجرام احترافي لإدارة المجموعات مع ميزات متقدمة، مطور باستخدام Python و aiogram 3.x مع قاعدة بيانات Supabase.

## ✨ الميزات الرئيسية

### 🛡️ الحماية والأمان

* **مكافحة السبام التلقائي** - كتم المستخدمين عند تجاوز حد الرسائل
* **منع الروابط** - حذف الروابط غير المرغوب فيها مع إمكانية إضافة استثناءات
* **منع الكلمات المحظورة** - فلترة الكلمات غير المناسبة
* **نظام التحذيرات** - تحذيرات تراكمية مع عقوبات تلقائية

### 👮‍♂️ الأوامر الإدارية

* **حظر/إلغاء حظر** - إدارة شاملة للعضويات
* **كتم/إلغاء كتم** - مع دعم المدة الزمنية
* **طرد الأعضاء** - إزالة مؤقتة من المجموعة
* **تثبيت الرسائل** - إدارة الرسائل المثبتة
* **نظام الرتب** - إدارة صلاحيات مخصصة

### 👋 الترحيب والتفاعل

* **رسائل ترحيب قابلة للتخصيص** - مع متغيرات ديناميكية
* **أزرار تفاعلية** - واجهة سهلة الاستخدام
* **معاينة الرسائل** - اختبار الرسائل قبل التطبيق

### 📊 الإحصائيات والتقارير

* **إحصائيات شاملة** - أعضاء، أنشطة، إجراءات
* **تتبع الأنشطة** - مراقبة تفاعل الأعضاء
* **تقارير دورية** - ملخصات الأداء

## 🚀 التثبيت والإعداد

### 1. متطلبات النظام

Python 3.8+  
PostgreSQL (من خلال Supabase)  
Redis (اختياري للكاش)

### 2. تحميل المشروع

git clone <repository-url>  
cd telegram\_bot

### 3. تثبيت المتطلبات

pip install -r requirements.txt

### 4. إعداد متغيرات البيئة

انسخ ملف .env.example إلى .env واملأ البيانات:

cp .env.example .env

### 5. إعداد قاعدة البيانات Supabase

#### أ. إنشاء مشروع Supabase

1. اذهب إلى [supabase.com](https://supabase.com)
2. أنشئ حساب جديد أو ادخل لحسابك
3. أنشئ مشروع جديد
4. احصل على URL و API Key من إعدادات المشروع

#### ب. إنشاء الجداول

انسخ والصق SQL التالي في SQL Editor في Supabase:

-- جدول المستخدمين  
CREATE TABLE IF NOT EXISTS users (  
 user\_id BIGINT PRIMARY KEY,  
 username TEXT,  
 first\_name TEXT,  
 last\_name TEXT,  
 language\_code TEXT DEFAULT 'ar',  
 is\_bot BOOLEAN DEFAULT FALSE,  
 is\_premium BOOLEAN DEFAULT FALSE,  
 created\_at TIMESTAMP DEFAULT NOW(),  
 updated\_at TIMESTAMP DEFAULT NOW()  
);  
  
-- جدول المجموعات  
CREATE TABLE IF NOT EXISTS chats (  
 chat\_id BIGINT PRIMARY KEY,  
 title TEXT,  
 chat\_type TEXT DEFAULT 'group',  
 username TEXT,  
 description TEXT,  
 welcome\_enabled BOOLEAN DEFAULT TRUE,  
 welcome\_message TEXT,  
 antiflood\_enabled BOOLEAN DEFAULT TRUE,  
 antilink\_enabled BOOLEAN DEFAULT FALSE,  
 antiword\_enabled BOOLEAN DEFAULT FALSE,  
 warns\_enabled BOOLEAN DEFAULT TRUE,  
 max\_warns INTEGER DEFAULT 3,  
 banned\_words JSONB DEFAULT '[]',  
 allowed\_links JSONB DEFAULT '[]',  
 created\_at TIMESTAMP DEFAULT NOW(),  
 updated\_at TIMESTAMP DEFAULT NOW()  
);  
  
-- جدول المشرفين  
CREATE TABLE IF NOT EXISTS admins (  
 id SERIAL PRIMARY KEY,  
 user\_id BIGINT NOT NULL,  
 chat\_id BIGINT NOT NULL,  
 rank TEXT DEFAULT 'admin',  
 title TEXT,  
 can\_delete\_messages BOOLEAN DEFAULT TRUE,  
 can\_restrict\_members BOOLEAN DEFAULT TRUE,  
 can\_promote\_members BOOLEAN DEFAULT FALSE,  
 can\_change\_info BOOLEAN DEFAULT FALSE,  
 can\_invite\_users BOOLEAN DEFAULT TRUE,  
 can\_pin\_messages BOOLEAN DEFAULT TRUE,  
 created\_at TIMESTAMP DEFAULT NOW(),  
 UNIQUE(user\_id, chat\_id)  
);  
  
-- جدول التحذيرات  
CREATE TABLE IF NOT EXISTS warnings (  
 id SERIAL PRIMARY KEY,  
 user\_id BIGINT NOT NULL,  
 chat\_id BIGINT NOT NULL,  
 admin\_id BIGINT NOT NULL,  
 reason TEXT,  
 warn\_count INTEGER DEFAULT 1,  
 created\_at TIMESTAMP DEFAULT NOW()  
);  
  
-- جدول الحظر  
CREATE TABLE IF NOT EXISTS bans (  
 id SERIAL PRIMARY KEY,  
 user\_id BIGINT NOT NULL,  
 chat\_id BIGINT NOT NULL,  
 admin\_id BIGINT NOT NULL,  
 reason TEXT,  
 ban\_type TEXT DEFAULT 'ban',  
 duration INTEGER,  
 expires\_at TIMESTAMP,  
 is\_active BOOLEAN DEFAULT TRUE,  
 created\_at TIMESTAMP DEFAULT NOW()  
);  
  
-- جدول الفلاتر  
CREATE TABLE IF NOT EXISTS filters (  
 id SERIAL PRIMARY KEY,  
 chat\_id BIGINT NOT NULL,  
 keyword TEXT NOT NULL,  
 response TEXT NOT NULL,  
 filter\_type TEXT DEFAULT 'text',  
 media\_file\_id TEXT,  
 created\_by BIGINT NOT NULL,  
 created\_at TIMESTAMP DEFAULT NOW()  
);  
  
-- جدول الملاحظات  
CREATE TABLE IF NOT EXISTS notes (  
 id SERIAL PRIMARY KEY,  
 chat\_id BIGINT NOT NULL,  
 name TEXT NOT NULL,  
 content TEXT NOT NULL,  
 note\_type TEXT DEFAULT 'text',  
 media\_file\_id TEXT,  
 created\_by BIGINT NOT NULL,  
 created\_at TIMESTAMP DEFAULT NOW(),  
 UNIQUE(chat\_id, name)  
);  
  
-- جدول الإعدادات  
CREATE TABLE IF NOT EXISTS settings (  
 chat\_id BIGINT PRIMARY KEY,  
 settings\_data JSONB DEFAULT '{}',  
 created\_at TIMESTAMP DEFAULT NOW(),  
 updated\_at TIMESTAMP DEFAULT NOW()  
);

### 6. إعداد ملف .env

# إعدادات البوت  
BOT\_TOKEN=your\_bot\_token\_from\_botfather  
BOT\_USERNAME=your\_bot\_username  
  
# إعدادات Supabase  
SUPABASE\_URL=your\_supabase\_project\_url  
SUPABASE\_KEY=your\_supabase\_anon\_key  
  
# إعدادات المطورين (ضع معرف تليجرام الخاص بك)  
SUDO\_USERS=123456789  
  
# إعدادات اختيارية  
REDIS\_URL=redis://localhost:6379/0  
WEBHOOK\_URL=https://yourdomain.com

### 7. تشغيل البوت

python main.py

## 🌐 النشر على PythonAnywhere

### 1. رفع الملفات

# رفع الملفات إلى PythonAnywhere  
scp -r telegram\_bot/ username@ssh.pythonanywhere.com:/home/username/

### 2. تثبيت المتطلبات

pip3.10 install --user -r requirements.txt

### 3. إعداد متغيرات البيئة

# إنشاء ملف .env في المجلد الرئيسي  
nano .env

### 4. إعداد Task المجدولة

في لوحة تحكم PythonAnywhere، اذهب إلى “Tasks” وأنشئ task جديدة:

cd /home/username/telegram\_bot && python3.10 main.py

### 5. إعداد الويب هوك (اختياري)

لاستخدام webhooks بدلاً من polling:

# في web tab، أنشئ Flask app  
from main import bot, dp  
from aiohttp import web  
from aiohttp.web import Application  
  
app = Application()  
# إعداد webhook

## 📋 دليل الاستخدام

### الأوامر الأساسية

#### أوامر عامة

* /start - بدء البوت
* /help - عرض المساعدة
* /admin - لوحة الإدارة الرئيسية
* /info - معلومات المجموعة
* /ping - اختبار البوت

#### أوامر الإدارة

/ban @username سبب الحظر # حظر مستخدم  
/unban @username # إلغاء حظر  
/kick @username سبب الطرد # طرد مستخدم  
/mute @username 30m سبب الكتم # كتم لمدة 30 دقيقة  
/unmute @username # إلغاء كتم  
/warn @username سبب التحذير # تحذير مستخدم  
/unwarn @username # مسح التحذيرات  
/pin # تثبيت رسالة (رد على الرسالة)  
/unpin # إلغاء تثبيت

#### أوامر الحماية

/protection # لوحة إعدادات الحماية  
/antiflood # تشغيل/إيقاف مكافحة السبام  
/antilink # تشغيل/إيقاف منع الروابط  
/antiword # تشغيل/إيقاف منع الكلمات  
/addword كلمة\_محظورة # إضافة كلمة محظورة  
/removeword كلمة\_محظورة # إزالة كلمة محظورة  
/addlink example.com # إضافة رابط مسموح

#### أوامر الترحيب

/welcome # إعدادات الترحيب  
/setwelcome مرحباً {first\_name}! # تعيين رسالة ترحيب  
/resetwelcome # إعادة تعيين للافتراضي  
/togglewelcome # تشغيل/إيقاف الترحيب  
/testwelcome # اختبار رسالة الترحيب

### تنسيقات الوقت

* s = ثواني
* m = دقائق
* h = ساعات
* d = أيام
* w = أسابيع

**أمثلة:** - 30s = 30 ثانية - 5m = 5 دقائق - 2h = ساعتان - 1d = يوم واحد

### متغيرات رسائل الترحيب

* {first\_name} - الاسم الأول
* {last\_name} - الاسم الأخير
* {full\_name} - الاسم الكامل
* {username} - اسم المستخدم
* {user\_id} - معرف المستخدم
* {chat\_title} - اسم المجموعة
* {chat\_id} - معرف المجموعة

**مثال على رسالة ترحيب:**

🌟 أهلاً وسهلاً {first\_name}!  
  
مرحباً بك في {chat\_title}  
نتمنى لك وقتاً ممتعاً معنا 🎉  
  
معرفك: {user\_id}

## 🔧 التخصيص والتطوير

### هيكل المشروع

telegram\_bot/  
├── main.py # ملف البوت الرئيسي  
├── config/ # إعدادات البوت  
│ ├── \_\_init\_\_.py  
│ └── config.py  
├── database/ # قاعدة البيانات  
│ ├── \_\_init\_\_.py  
│ ├── database.py  
│ └── models.py  
├── handlers/ # معالجات الأوامر  
│ ├── \_\_init\_\_.py  
│ ├── admin\_handlers.py  
│ ├── protection\_handlers.py  
│ ├── welcome\_handlers.py  
│ ├── general\_handlers.py  
│ └── callback\_handlers.py  
├── filters/ # فلاتر مخصصة  
│ ├── \_\_init\_\_.py  
│ └── admin\_filter.py  
├── keyboards/ # لوحات المفاتيح  
│ ├── \_\_init\_\_.py  
│ └── admin\_keyboards.py  
├── utils/ # وظائف مساعدة  
│ ├── \_\_init\_\_.py  
│ └── helpers.py  
├── requirements.txt # متطلبات Python  
├── .env.example # مثال متغيرات البيئة  
└── README.md # هذا الملف

### إضافة ميزات جديدة

#### 1. إنشاء معالج جديد

# في handlers/new\_feature.py  
from aiogram import Router  
from aiogram.types import Message  
from aiogram.filters import Command  
  
new\_feature\_router = Router()  
  
@new\_feature\_router.message(Command("newcommand"))  
async def new\_command(message: Message):  
 await message.reply("ميزة جديدة!")

#### 2. تسجيل المعالج في main.py

from handlers.new\_feature import new\_feature\_router  
  
# في main.py  
dp.include\_router(new\_feature\_router)

### تخصيص الرسائل

يمكنك تعديل الرسائل في config/config.py:

EMOJI = {  
 "check": "✅",  
 "cross": "❌",  
 # ... إضافة رموز جديدة  
}

## 🐛 استكشاف الأخطاء

### مشاكل شائعة وحلولها

#### 1. خطأ في الاتصال بقاعدة البيانات

خطأ: connection refused

**الحل:** تأكد من صحة SUPABASE\_URL و SUPABASE\_KEY

#### 2. البوت لا يرد على الأوامر

البوت غير نشط

**الحل:** - تأكد من صحة BOT\_TOKEN - تأكد من إضافة البوت للمجموعة كمشرف - تحقق من السجلات (logs)

#### 3. صلاحيات غير كافية

ليس لديك صلاحية

**الحل:** - أضف معرفك إلى SUDO\_USERS - تأكد من كونك مشرف في المجموعة

### تفعيل السجلات التفصيلية

import logging  
logging.basicConfig(level=logging.DEBUG)

## 📞 الدعم والمساعدة

### طرق التواصل

* **GitHub Issues**: لتقارير الأخطاء والاقتراحات
* **Telegram**: [@your\_support\_username]
* **Email**: support@yourproject.com

### المساهمة في المشروع

1. Fork المشروع
2. أنشئ فرع جديد للميزة
3. اكتب الكود واختبره
4. أرسل Pull Request

### ترخيص المشروع

هذا المشروع مرخص تحت [MIT License](LICENSE)

## 🔄 التحديثات المستقبلية

### ميزات قادمة

* ☐ نظام النقاط والمكافآت
* ☐ تكامل مع APIs خارجية
* ☐ لوحة تحكم ويب
* ☐ نظام الإشعارات المتقدم
* ☐ دعم قواعد بيانات إضافية
* ☐ تحليلات متقدمة
* ☐ دعم متعدد اللغات
* ☐ نظام البلاغات

### سجل التغييرات

#### v1.0.0 (2024-01-XX)

* إطلاق النسخة الأولى
* ميزات الإدارة الأساسية
* نظام الحماية
* رسائل الترحيب
* واجهة إدارية تفاعلية

**تم التطوير بواسطة MiniMax Agent** 🤖

للحصول على أحدث التحديثات، تابع المشروع على GitHub.