

Telegram Shop Bot — Full Explanation

TELEGRAM SHOP BOT — CODE EXPLANATION

This document explains how the Telegram shop bot works and how its features connect together.

1. **OVERVIEW** The bot is an interactive Telegram shop system. It allows users to browse products, select quantities, provide delivery addresses, and receive a payment QR code (PayNow mock). It also notifies the admin about new orders. The system uses the python-telegram-bot framework and features a clean inline-button interface.

2. **CONFIGURATION & SECURITY** The .env file holds sensitive information like BOT_TOKEN and ADMIN_ID. The load_dotenv() function loads them at runtime. This prevents exposing secrets in code and enables easy deployment.

3. **STATE MANAGEMENT** Two dictionaries are used for temporary in-memory storage: • user_order_state: tracks each user's progress through the order flow (e.g., step = "address"). • last_message_time: tracks timestamps to prevent spamming.

4. **PRODUCT CATALOG** The PRODUCTS dictionary defines available items with names, prices, and emojis. This acts as a simple database for the bot.

5. **RATE LIMITING** The is_spamming() function limits user messages to once every 1.5 seconds to prevent spam and accidental double-taps.

6. **PAYMENT QR GENERATION** generate_paynow_qr() creates a mock QR code using the qrcode library. Later, this can be replaced with a real PayNow, HitPay, or Stripe integration.

7. **SHOP UI CONSTRUCTION** build_shop_text_and_keyboard() dynamically builds the product list and "Buy" buttons. build_qty_keyboard() builds the "+ / - / Checkout" interface for quantity selection.

8. **ERROR HANDLING** The error_handler() function logs exceptions and sends friendly messages to users instead of crashing the bot.

9. **COMMANDS** • /start: Greets new users. • /shop: Displays products with Buy buttons. • /order: Optional, same as /shop but used for backward compatibility.

10. **CALLBACK HANDLERS** These handle user clicks on inline buttons: • on_buy(): Displays selected product and total. • on_qty(): Adjusts quantity up or down. • on_checkout(): Prompts the user for a delivery address. • on_back_to_shop(): Returns to the product catalog. callback_router() interprets button data (e.g., buy:hoodie:1) and routes to the right handler.

11. **MESSAGE HANDLER** Once the user is in the "address" step, handle_message() captures their reply, validates it, generates the QR, sends confirmation, and notifies the admin. It then clears the user's order state.

12. **APPLICATION BOOTSTRAP** In main(), the ApplicationBuilder loads handlers for all commands and events, attaches the error handler, and starts the bot in polling mode.

13. **SECURITY FEATURES** ✓ Environment variables for credentials ✓ Rate-limiting anti-spam ✓ Safe error handling ✓ Admin notification isolation (only ADMIN_ID receives order alerts)

14. LIMITATIONS • State stored in RAM only (resets on restart) • No real payment processing • No persistent product database • No user authentication beyond Telegram

15. NEXT STEPS • Integrate Stripe or HitPay APIs for real payments • Store orders in SQLite or Firebase • Add product images & captions • Deploy on Railway/Render with webhooks

This architecture ensures a stable, secure, and extensible foundation for an e-commerce Telegram bot.