🍌 ⭐ Star Platanos Bot: Crypto Market Tracker

Proyecto de Administración de Proyectos de Software

Este proyecto es el MVP (Producto Mínimo Viable) de un bot de Telegram financiero implementado en Python, orquestado con Docker y gestionado con Git/GitHub.

El proyecto demuestra un ciclo de vida de desarrollo ágil, incluyendo un **Pivote de Alcance (Scope Pivot)**: evolucionó de un bot de entretenimiento (JoJo's Bizarre Adventure) a una herramienta financiera de consulta de precios en tiempo real mediante la integración de la API pública de Binance.

1. ⚙️ Herramientas y Configuración Inicial

El proyecto fue levantado en un entorno Windows 10/11 utilizando las siguientes versiones, asegurando la **Paridad de Entornos** (local y contenedor):

Herramienta	Versión Utilizada	Propósito
Python	3.12.4	Lenguaje de programación principal.
Docker	28.5.2	Contenerización del ambiente de ejecución.
Git	2.51.2	Control de versiones y trazabilidad de cambios.
Librerías	`python-telegram-bot` , `requests`	Comunicación con Telegram y Binance API.

Flujo de Levantamiento del Ambiente (PowerShell Log)

Registro de comandos utilizados para la inicialización del entorno:

```
# 1. Creación e ingreso al directorio del proyecto
PS C:\Users\osval> mkdir Star_Platanos_Bot
PS C:\Users\osval> cd Star_Platanos_Bot

# 2. Verificación de versiones (Criterios de Aceptación)
python --version
docker --version
git --version

# 3. Creación de archivos de código fuente
code .
```

2. 🚀 Despliegue del Bot (Docker)

El bot se ejecuta en un contenedor Docker aislado.

Requisitos Previos

- Token de Telegram configurado en `star_platanos.py`.
- Archivo `requirements.txt` guardado correctamente.

Pasos de Ejecución

1. **Construir la Imagen de Docker:** Instala las dependencias críticas (`requests` para la API de Binance).

```
docker build -t star-platanos .
```

2. **Ejecutar el Contenedor:** El bot inicia en modo *polling* y conecta con los mercados.

```
docker run star-platanos
```

3. 📝 Reporte de Gestión: Pivote y Solución de Errores

Este proyecto destaca por la adaptación a nuevos requerimientos y la resolución de problemas técnicos.

A. Gestión del Alcance: El Gran Pivote

El proyecto sufrió una transformación radical en su propuesta de valor:

- **Alcance Inicial (Legacy):** Generador de Stands aleatorios de anime.
- **Cambio de Requerimiento:** El cliente solicitó funcionalidad financiera real.
- **Alcance Final (Actual):** Monitor de precios de Criptomonedas (Bitcoin, Ethereum, Pepe) conectado a la API de Binance en tiempo real.
- **Acción Técnica:** Se refactorizó el 100% del código en `star_platanos.py` y se añadieron nuevas dependencias.

B. Solución de Errores Críticos (Troubleshooting)

1. **Error de Dependencias (`ModuleNotFoundError`):**
   - **Problema:** Al agregar la funcionalidad de Binance, Docker fallaba al no encontrar `requests`.
   - **Causa Raíz:** El archivo `requirements.txt` se modificó pero **no se guardó** en el editor antes de construir la imagen.
   - **Solución:** Guardado del archivo y reconstrucción forzada (`docker build --no-cache`).
2. **Identidad de Git:**
   - **Problema:** Fallo al realizar el primer commit.

- **Solución:** Configuración global de `user.name` y `user.email`.

4. 💡 Funcionalidad Actual

El bot ofrece un menú interactivo con las siguientes opciones:

- ₿ **Bitcoin (BTC):** Consulta precio en tiempo real (USDT).
- Ξ **Ethereum (ETH):** Consulta precio en tiempo real (USDT).
- 🐸 **Pepe (PEPE):** Consulta precio con alta precisión decimal.
- **Botón Actualizar:** Refresca los datos llamando nuevamente a la API.

📁 Estructura del Proyecto

Archivo	Propósito
`star_platanos.py`	Lógica del bot financiero y conexión con API Binance.
`Dockerfile`	Define el entorno y ejecuta el script principal.
`requirements.txt`	Lista: `python-telegram-bot==20.7` y `requests`.
`.gitignore`	Exclusión de archivos temporales.