Crypto Trading Bot Project Guide

Prepared for: Mychael Gray (GitHub: graymychael)

PHASE 1: Local Bot Setup (macOS)

1. Install Requirements:

   pip install python-binance pandas ta requests streamlit flask

2. Clone the Repo:

   git clone https://github.com/graymychael/crypto-trading-bot-suite.git

   cd crypto-trading-bot-suite

Or upload files from: crypto-trading-bot-suite.zip

PHASE 2: Run Desktop GUI App

cd gui

python crypto_bot_gui.py

To create a macOS app:

pip install pyinstaller

pyinstaller --onefile --windowed crypto_bot_gui.py

.app file will appear in /dist

PHASE 3: Streamlit iPhone Dashboard

cd web_dashboard

streamlit run app.py

To deploy:

- Visit https://share.streamlit.io
- Connect GitHub and deploy app.py

- Open on iPhone and 'Add to Home Screen'

PHASE 4: Flask API Backend (Replit)

1. Go to https://replit.com and create a new Flask Repl.
2. Replace main.py with:

```
from flask import Flask, jsonify, request
app = Flask(__name__)
... (see full code in project)
```

3. Run and get your live API link.

PHASE 5: Telegram Alerts Setup

1. Create bot via @BotFather on Telegram.
2. Use /newbot and get your token.
3. Get your chat ID:

```
https://api.telegram.org/bot<YOUR_TOKEN>/getUpdates
```

4. Add your token and chat ID to your code.

PHASE 6: Final Touches

- Store secrets in a .env file
- Add README and LICENSE
- Deploy and maintain repo at github.com/graymychael/crypto-trading-bot-suite

Optional:

- Add auth, analytics, or convert to native iOS app.

Crypto Trading Bot Project Guide

Prepared for: Mychael Gray (GitHub: graymychael)

PHASE 1: Local Bot Setup (macOS)

1. Install Requirements:

   pip install python-binance pandas ta requests streamlit flask

2. Clone the Repo:

   git clone https://github.com/graymychael/crypto-trading-bot-suite.git

   cd crypto-trading-bot-suite

Or upload files from: crypto-trading-bot-suite.zip

PHASE 2: Run Desktop GUI App

cd gui

python crypto_bot_gui.py

To create a macOS app:

pip install pyinstaller

pyinstaller --onefile --windowed crypto_bot_gui.py

.app file will appear in /dist

PHASE 3: Streamlit iPhone Dashboard

cd web_dashboard

streamlit run app.py

To deploy:

- Visit https://share.streamlit.io
- Connect GitHub and deploy app.py

- Open on iPhone and 'Add to Home Screen'

PHASE 4: Flask API Backend (Replit)

1. Go to https://replit.com and create a new Flask Repl.
2. Replace main.py with:

```
from flask import Flask, jsonify, request
app = Flask(__name__)
... (see full code in project)
```

3. Run and get your live API link.

PHASE 5: Telegram Alerts Setup

1. Create bot via @BotFather on Telegram.
2. Use /newbot and get your token.
3. Get your chat ID:

```
https://api.telegram.org/bot<YOUR_TOKEN>/getUpdates
```

4. Add your token and chat ID to your code.

PHASE 6: Final Touches

- Store secrets in a .env file
- Add README and LICENSE
- Deploy and maintain repo at github.com/graymychael/crypto-trading-bot-suite

Optional:

- Add auth, analytics, or convert to native iOS app.

Set Up GitHub Actions for Your Crypto Bot

Automate testing, deployment, or updates using GitHub CI/CD

Step 1: Create a Workflow Folder

1. In your GitHub project directory, create a `.github/workflows/` folder.
2. Inside, add a new YAML file, e.g., `bot-deploy.yml`.

Step 2: Sample Workflow File (bot-deploy.yml)

```
name: Run Crypto Bot

on:
  push:
    branches: [ main ]
  schedule:
    - cron: '0 * * * *'  # Every hour

jobs:
  run-bot:
    runs-on: ubuntu-latest
    steps:
    - uses: actions/checkout@v3

    - name: Set up Python
      uses: actions/setup-python@v4
      with:
        python-version: 3.10

    - name: Install dependencies
      run: |
        pip install -r requirements.txt
```

```
    - name: Run the bot
      run: |
        python gui/crypto_bot_gui.py
```

Step 3: Push to GitHub

Once the file is created locally:

1. Save it to `.github/workflows/bot-deploy.yml`
2. Commit and push:

```
git add .github/workflows/bot-deploy.yml
git commit -m "Add GitHub Actions workflow"
git push
```

Step 4: Check Actions Tab

- Go to your GitHub repo
- Click the 'Actions' tab
- You'll see the job run on every push or hourly as scheduled

Notes & Tips

- You can add environment variables in the Secrets tab (for API keys).
- Modify the `cron` expression to change the frequency.
- Use Streamlit/Flask workflows to deploy to Render or other platforms.