Comprehensive Code Repository

11 Months of Development (September 2024 - August 2025)

📋 Repository Overview

This repository contains all scripts, configurations, and code developed across 11 months of conversations, organized by category and functionality. Each section includes version history, features, and usage instructions.

🔗 1. BLOCKCHAIN & CRYPTOCURRENCY

1.1 Ethereum Node Setup Scripts

Location: Multiple conversations (2024-08-14 to 2025-05-27) **Versions**: v2.3.0, v2.4.0, v12.0.0

Core Features:

- Automated deployment to remote servers
- Execution client (Geth) and consensus client (Lighthouse) setup
- Systemd service configuration
- Firewall and security configuration
- Configuration file management (node_config.env)

Latest Version (v12.0.0) Features:

```bash
# Key capabilities
- Multi-client support (Geth, Lighthouse, Besu, Teku)
- Validator setup with MEV-Boost integration
- Automated backup and monitoring
- Hardware optimization
- Security hardening with disk encryption
- Fallback node configuration
- Client rotation capabilities
- Gas price alerting
- Benchmarking tools
```

Configuration Structure:

```bash
# node_config.env template
ETH_NETWORK="mainnet"
EXECUTION_CLIENT="geth"
CONSENSUS_CLIENT="lighthouse"
ENABLE_VALIDATOR=false
ENABLE_MEV_BOOST=false
ENABLE_MONITORING=true
CPU_CORES=4
RAM_GB=16
DISK_SIZE_GB=1000
```

Deployment Methods:

1. **Remote Deployment** (v2.3.0/v2.4.0)
2. **Local Installation** (v12.0.0)
3. **Container Deployment** (Docker support)
4. **Multi-node Federation** (Enterprise scaling)

1.2 Chainlink Node Setup

Location: 2024-08-14 conversations **Integration**: Compatible with Ethereum node on same system

Features:

- Oracle network participation
- Data feeding capabilities
- Smart contract integration
- Monitoring and alerting

🧹 2. SYSTEM ADMINISTRATION & CLEANUP

2.1 TidyTux Linux Cleanup Utility

Location: Multiple conversations (2025-03-03 to 2025-06-09) **Versions**: v1.2.0, v1.2.1, v2.0.0, v2.1.0

Core Functionality:

```bash
# Main cleanup categories
- Disk space management and emergency cleanup
- File organization (Downloads, Documents, Media)
- Browser cache cleanup (Chrome, Firefox, Brave, Chromium)
- System maintenance (package updates, snap cleanup)
- Docker resource cleanup
- Duplicate file detection and management
- Installation file cleanup (.deb files)
- Development cache cleanup (node_modules, Python cache)
```

Safety Features:

```bash
# Built-in protections
- Backup creation before operations
- Confirmation prompts (bypassable with -y flag)
- Rollback capabilities
- Comprehensive logging
- Removable storage detection
- Root execution prevention
```

Command Line Options:

```bash
tidytux [OPTIONS]
-y, --yes       # Skip confirmations
-e, --emergency # Aggressive cleanup mode
-q, --quiet     # Silent operation
-v, --verbose   # Detailed output
-r, --report-only # Analysis without changes
-s, --schedule  # Setup scheduled cleanup
```

Emergency Mode Triggers:

- Disk space below 1GB
- Critical system space shortage
- User-initiated emergency cleanup

2.2 Enhanced System Cleanup (v2.1.0)

New Features:

```bash
# Advanced capabilities
- JSON logging output
- Memory-efficient large file operations
- Parallel processing with job control
- Cross-platform compatibility improvements
- Enhanced backup functionality
- Better dependency management
- Structured configuration management
- Resource monitoring during cleanup
```

☁️ 3. CLOUD STORAGE & BACKUP

3.1 Google Drive Manager

Location: 2025-07-15 conversations **Technology**: rclone integration with Python GUI

Core Components:

1. **Bash Script Backend** (gdrive-manager.sh)
2. **Python GUI Frontend** (tkinter-based)
3. **Launcher Script** (system integration)

Features:

```bash
# Functionality
- Automatic rclone installation and configuration
- Google Drive mounting at ~/GoogleDrive
- Intelligent backup suggestions based on file structure
- Smart file categorization
- Batch upload/download operations
- Sync monitoring and conflict resolution
- Bandwidth limiting and transfer optimization
- Scheduled backup operations
```

Smart Backup Categories:

```bash
# Auto-detected backup categories
Financial Documents    -> Financial/
Technical Guides       -> Documentation/Guides/
Scripts and Code       -> Scripts/Personal/
Media Files            -> Media/[Type]/
Project Files          -> Projects/[Name]/
```

Configuration Options:

```bash
# Enhanced settings
SYNC_ENABLED=true
AUTO_CATEGORIZE=true
SMART_BACKUP=true
BACKUP_COMPRESSION=true
VERIFY_TRANSFERS=true
BANDWIDTH_LIMIT=""
MAX_TRANSFERS=4
BACKUP_RETENTION_DAYS=90
```

3.2 Unified System Manager

Location: 2025-07-15 Software Development Program **Technology**: PySide6 GUI application

Architecture:

```python
# Combines multiple tools into unified interface
- Google Drive backup and sync
- TidyTux system cleanup
- Real-time progress monitoring
- Configuration management
- Scheduled operations
- Professional GUI with tabs and progress bars
```

🏢 4. ENTERPRISE INFRASTRUCTURE

4.1 NexusController Platform

Location: Multiple conversations (2025-08-04 to 2025-08-05) **Version**: v2.0 **Type**: Enterprise infrastructure management platform

Architecture Overview:

```
python

# Core Systems
- Event-driven architecture with pub/sub messaging
- Modular plugin system for extensibility
- Multi-cloud provider abstraction layer
- Real-time WebSocket communication
- RESTful API with FastAPI
- Federation support for horizontal scaling (5000+ resources)
```

Enterprise Features:

```
python

# Management capabilities
- State management with drift detection
- Automated remediation workflows
- Advanced monitoring and alerting
- Secure SSH implementation with key verification
- Docker containerization
- Comprehensive logging and audit trails
- RBAC (Role-Based Access Control)
- Encrypted configuration management
```

Security Implementation:

```
python

# Security hardening
- Fixed SSH vulnerabilities (strict host key verification)
- Encrypted configuration (Fernet encryption)
- Input validation for IPs, networks, ports, file paths
- Secure permissions (700 on sensitive directories)
- No hardcoded credentials
- JWT authentication
- Rate limiting
- Security headers
```

Deployment Options:

```
bash

# Multiple deployment methods
1. Development: ./nexus_launcher.sh
2. Production: Systemd service
3. Container: Docker Compose with monitoring
4. Enterprise: Kubernetes with auto-scaling
```

Technology Stack:

```
python

# Components
- FastAPI (REST API with OpenAPI docs)
- Docker/Kubernetes ready
- Prometheus/Grafana integration
- Thread-safe GUI
- Comprehensive testing suite
- Multi-cloud provider support (AWS, Azure, GCP)
```

🎬 5. MEDIA SERVER & AUTOMATION

5.1 Personal Cloud Network

Location: 2024-11-13 conversations **Purpose**: Media streaming and storage with remote access

Infrastructure Components:

```
yaml

# Core services
- Plex Media Server (primary streaming)
- NextCloud (personal cloud storage)
- Docker containerization
- Traefik reverse proxy
- Automatic media organization
- Remote access via DuckDNS
```

Media Organization Structure:

```
bash
```

```
/tank/media/
├── movies/
│   ├── Movie (Year)/
│   │   ├── Movie (Year).mkv
│   │   ├── Movie (Year).en.srt
│   │   └── poster.jpg
├── tv/
│   ├── Show Name/
│   │   ├── Season 01/
│   │   │   ├── Show Name - S01E01.mkv
├── music/
│   ├── Artist/
│   │   ├── Album/
│   │   │   ├── 01 - Track.flac
```

Automation Stack:

```yaml
# Arr suite for automation
services:
  sonarr:   # TV show management
  radarr:   # Movie management
  prowlarr: # Indexer management
  bazarr:   # Subtitle management
  watchtower: # Container updates
```

Storage Architecture:

```bash
# Recommended setup
- RAID configuration for reliability
- SSD cache for performance
- Automated backups with Duplicati
- Cloud sync for critical data
```

5.2 Media Organization Scripts

```python
```

```
# Automated file organization
def organize_media(source_dir, movies_dir, tv_dir):
    # Movie pattern: Title (Year) [Quality].mkv
    movie_pattern = r"(.+?)\s*\((\d{4})\).*\.(mkv|mp4|avi)$"
    # TV pattern: Show.Name.S01E02.mkv
    tv_pattern = r"(.+?)[\.\s][Ss](\d{2})[Ee](\d{2}).*\."
    # Automated categorization and file movement
```

🖥️ 6. SERVER CONFIGURATION & DEPLOYMENT

6.1 Multi-Service Server Setup

Location: 2025-05-27, 2025-08-10 conversations **Target**: HP Z4 G4 workstation configuration

Service Distribution:

```
bash

# Planned services on HP Z4 G4
- Ethereum full node server
- Chainlink node server
- Personal cloud storage (NextCloud)
- Media server (Plex/Jellyfin)
- Home automation platform
- Future expansion capacity
```

Hardware Recommendations:

```
bash

# Server specifications research
- CPU: Multi-core for parallel processing
- RAM: 32GB+ for multiple services
- Storage: NVMe SSD + large HDD array
- Network: Gigabit+ with VLAN support
- UPS: Clean power for 24/7 operation
```

Network Infrastructure:

```
bash
```

```
# Networking components
- Managed switch for VLAN segmentation
- pfSense/OPNsense firewall
- VPN for remote access
- Dynamic DNS configuration
- Port forwarding and security rules
```

6.2 Docker Orchestration

```yaml
# Example multi-service composition
version: '3.8'
networks:
  frontend:
    driver: bridge
  backend:
    driver: bridge
    internal: true

services:
  traefik:     # Reverse proxy
  ethereum:    # Blockchain node
  chainlink:   # Oracle node
  nextcloud:   # Cloud storage
  plex:        # Media server
  monitoring:  # Prometheus/Grafana
```

💰 7. FINANCIAL & BUDGET ANALYSIS

7.1 Louisville Metro Budget Analysis

Location: Multiple conversations (2025-02-28 to 2025-05-25) **Type**: Municipal government budget analysis and proposals

Budget Documents:

```bash
```

```
# Document types
- Executive Budget FY 2024-2025 ($874.8M)
- Hypothetical Budget Proposals
- Departmental budget breakdowns
- Comparative analysis tools
- Performance metrics frameworks
```

Key Budget Categories:

```bash
# Major allocations
Police Department:        $172.3M
Employee Benefits:        $213M investment
Capital Improvements:     $144.7M
Corrections:              $41.3M
Public Works:             Various allocations
Community Development:    Expanded funding
```

Analysis Tools:

```bash
# Comparative features
- Revenue vs expenditure analysis
- 5-year financial forecasts
- Performance metric tracking
- Cross-departmental integration
- Community impact assessment
```

🏭 8. INDUSTRIAL & TECHNICAL DOCUMENTATION

8.1 PECO Conveyor Systems

Location: 2024-11-24, 2024-12-05 conversations **Type**: Technical maintenance documentation

Equipment Documentation:

```bash
```

```
# PECO Ultimate Conveyor v.2
- Installation and Operations Manual
- Maintenance procedures
- Troubleshooting guides
- Parts lists and specifications
- Safety procedures and lockout protocols
```

Maintenance Procedures:

```bash
# Take-up assembly bearing replacement
1. Safety lockout (electrical, hydraulic, pneumatic)
2. Chain tension release
3. Bearing access and replacement
4. Proper reassembly and testing
# Specific torque specifications and part numbers
```

Technical Specifications:

```bash
# System components
- Three-tier conveyor design
- Roller-On-Call feature
- UHMW plastic guide systems
- Hydraulic take-up assist
- Safety systems and emergency stops
```

🔧 9. SYSTEM INTEGRATION & UTILITIES

9.1 Tool Integration Architecture

Location: 2025-07-15 Tool Integration Strategy **Purpose**: Unified system management

Integration Patterns:

```python
```

```
# Architectural approaches
- Microservices with shared API
- Plugin-based architecture
- Event-driven communication
- Unified GUI with modular backends
- Configuration management
- Cross-tool data sharing
```

Technology Stack:

```python
# Implementation technologies
- Python for core logic
- PySide6 for professional GUI
- FastAPI for REST interfaces
- Docker for containerization
- Configuration management
- Real-time progress monitoring
```

📊 10. MONITORING & MAINTENANCE

10.1 System Monitoring Stack

```yaml
# Monitoring infrastructure
prometheus:    # Metrics collection
grafana:       # Visualization dashboards
graylog:       # Log management
watchtower:    # Container updates
duplicati:     # Backup management
```

10.2 Automated Maintenance

```bash
```

```
# Scheduled operations
- Daily system cleanup
- Weekly backup verification
- Monthly security updates
- Quarterly system optimization
- Log rotation and archival
```

📝 11. DOCUMENTATION & GUIDES

11.1 Setup Guides

- Ethereum node deployment procedures
- Media server configuration
- Google Drive integration setup
- System cleanup automation
- Network security configuration

11.2 Troubleshooting Guides

- Common deployment issues
- Performance optimization
- Security hardening procedures
- Backup and recovery processes
- Hardware maintenance procedures

🔄 12. VERSION CONTROL & UPDATES

12.1 Release History

```bash
# Major versions across projects
TidyTux:          v1.2.0 → v2.1.0
Ethereum Scripts:  v2.3.0 → v12.0.0
NexusController:   v2.0 (current)
Google Drive Mgr:  v2.0 (enhanced)
```

12.2 Upcoming Enhancements

- Enhanced container orchestration
- Improved monitoring dashboards
- Advanced automation features
- Security compliance improvements
- Performance optimization
- Cross-platform compatibility

🎯 13. DEPLOYMENT RECOMMENDATIONS

13.1 Production Deployment Order

1. **Infrastructure Base**: NexusController platform
2. **Core Services**: Ethereum node, monitoring
3. **Storage Layer**: Google Drive integration, backup systems
4. **Media Services**: Plex server, automation stack
5. **Maintenance**: TidyTux integration, scheduled operations

13.2 Security Considerations

- Network segmentation with VLANs
- Encrypted configuration storage
- Regular security updates
- Backup verification procedures
- Access control and monitoring

This repository represents 11 months of development from September 2024 to August 2025, with over 100 conversations contributing to this comprehensive collection of tools, scripts, and configurations.