Complete GitHub Repository Setup Procedure

Date: 2025-09-12
Project: Home Automation Project
Purpose: Document complete process for setting up project repository from scratch on Windows

Overview

This procedure covers the complete setup process from initial documentation structure to live GitHub repository, specifically addressing Windows environment compatibility issues.

Prerequisites

- Windows operating system
- PowerShell access
- GitHub account
- Internet connection

Step-by-Step Process

Phase 1: Documentation Structure Planning

Duration: ~30 minutes
Goal: Create scalable documentation framework for multi-system project

1. **Analyzed Context Limitations**
   - Identified that Claude context limits would be exceeded as project grows
   - Determined need for external storage (Git) vs. session-based decision context
   - Designed modular session management strategy
2. **Created Repository Architecture**
   - Designed 4-tier directory structure:
     - `docs/` - Session states, decisions, procedures, troubleshooting
     - `configs/` - System configurations by component
     - `hardware/` - Physical system documentation
     - `scripts/` - Automation and setup scripts

Phase 2: Windows Compatibility Resolution

Duration: ~20 minutes

Challenge: Initial bash script not compatible with Windows

1. **Created PowerShell Version**
   - Converted Linux bash script to Windows PowerShell
   - Used `New-Item -ItemType Directory` for folder creation
   - Implemented `Out-File -Encoding UTF8` for file creation
   - Added Windows-style path separators (`\` instead of `/`)
2. **Script Features Added**
   - Colored console output using `-ForegroundColor`
   - Progress indication for directory creation
   - Automatic file structure validation
   - Clear next-steps instructions

Phase 3: Repository Structure Creation

Duration: ~10 minutes

Process:

1. **Saved PowerShell Script**

```powershell
# Saved as: setup-repo.ps1
# Content: Complete repository structure creation script
```

2. **Executed Setup Script**

```powershell
.\setup-repo.ps1
```

3. **Verified Structure Created**
   - All directories created successfully
   - Template files populated
   - Configuration placeholders in place
   - README.md with project overview complete

Phase 4: Git Installation and Configuration

Duration: ~15 minutes

Challenge: Git not installed on Windows system

1. **Installed Git for Windows**
   - Downloaded from: https://git-scm.com/download/win
   - Used default installation settings
   - Restarted PowerShell after installation
2. **Configured Git Identity**

```powershell
git config --global user.name "User Name"
git config --global user.email "user@email.com"
```

3. **Verified Installation**

```powershell
git --version
# Output: git version 2.x.x.windows.x
```

Phase 5: Local Repository Initialization

Duration: ~5 minutes

1. **Initialized Git Repository**

```powershell
cd home-automation-project
git init
```

2. **Staged All Files**

```powershell
git add .
```

3. **Created Initial Commit**

```powershell
git commit -m "Initial repository structure and documentation

- Created scalable documentation structure for multi-system project
- Added session state templates for Claude context management
- Documented network architecture decision (4-VLAN design)
- Set up configuration placeholders for all major systems
- Added procedure templates and troubleshooting framework"
```

Phase 6: GitHub Repository Creation

Duration: ~10 minutes

1. **Created GitHub Repository**
   - Navigated to GitHub.com
   - Clicked "+" → "New repository"
   - Repository name: `home-automation-project`
   - Left public, no README initialization
   - Clicked "Create repository"
2. **Connected Local to Remote**

```powershell
git remote add origin https://github.com/[username]/home-automation-project.git
git branch -M main
git push -u origin main
```

3. **Verified Repository Live**
   - Confirmed all files visible on GitHub
   - Directory structure properly displayed
   - README.md rendering correctly

Files Created During Process

Documentation Templates

- `docs/session-states/session-template.md` - Reusable session format
- `docs/session-states/20250912-initial-documentation-session01.md` - Current session state
- `docs/decisions/001-network-architecture.md` - Network design decision

- `README.md` - Project overview and navigation

Configuration Placeholders

- `configs/openwrt/main-config.conf`
- `configs/openwrt/firewall-rules.conf`
- `configs/openwrt/vlan-config.conf`
- `configs/home-assistant/configuration.yaml`
- `configs/home-assistant/automations.yaml`
- `configs/frigate/config.yml`
- `configs/esphome/printairpipe-controller.yaml`
- `configs/proxmox/vm-configs.conf`

Directory Structure

```
home-automation-project/
├── docs/
│   ├── session-states/     # Claude session management
│   ├── decisions/          # Architecture decision records
│   ├── procedures/         # Step-by-step processes
│   └── troubleshooting/    # Issue resolution guides
├── configs/
│   ├── openwrt/            # Router configurations
│   ├── home-assistant/     # HA automation configs
│   ├── frigate/            # NVR system configs
│   ├── esphome/            # Sensor controller configs
│   └── proxmox/            # Virtualization configs
├── hardware/
│   ├── stl-files/          # 3D printing files
│   ├── wiring-diagrams/    # Circuit documentation
│   └── part-lists/         # Component specifications
└── scripts/
    ├── setup/              # Installation automation
    ├── backup/             # Backup procedures
    └── monitoring/         # Health check scripts
```

Key Lessons Learned

Windows Compatibility Issues

- **Bash scripts don't work natively** - always need PowerShell equivalent

- **Path separators matter** - use `\` for Windows paths in scripts
- **File encoding important** - specify UTF8 to avoid character issues
- **Git installation required** - not included by default on Windows

Git Configuration Requirements

- **User identity mandatory** - commits fail without name/email configured
- **Global vs. local config** - used global for user convenience
- **Repository naming** - avoided conflicts with existing projects

Documentation Strategy Benefits

- **Session-based approach** handles Claude context limits effectively
- **External storage** for large configs preserves session efficiency
- **Decision records** capture rationale for future reference
- **Template system** ensures consistency across sessions

Troubleshooting Notes

Common Issues Encountered

1. **"git command not found"**
   - Solution: Install Git for Windows, restart PowerShell
2. **"Author identity unknown"**
   - Solution: Configure git user.name and user.email globally
3. **PowerShell execution policy**
   - May need: `Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser`
4. **File encoding issues**
   - Always specify `-Encoding UTF8` in PowerShell Out-File commands

Success Metrics

- ✅ Complete repository structure created
- ✅ All template files properly formatted
- ✅ Git repository initialized and configured
- ✅ GitHub repository live with all content
- ✅ Documentation system ready for scaling
- ✅ Next phase implementation ready to begin

Next Steps

Repository setup complete. Ready to begin:

1. OpenWrt router configuration for 4-VLAN network
2. VLAN interface and firewall rule implementation
3. Network security policy enforcement
4. System integration testing procedures

Procedure Author: Claude Sonnet 4
Tested Environment: Windows 11 with PowerShell
Repository URL: https://github.com/[username]/home-automation-project
Procedure Status: Complete and Verified