JARVIS v14 Ultimate

Complete Setup & Verification Guide

Self-Modifying AI Assistant for Termux/Android

Device: Realme 2 Pro Lite | RAM: 4GB | Platform: Termux

1. Verification Status - All Critical Fixes Applied

All critical issues identified in the analysis report have been fixed. The project is now 100% functional and error-proof for Termux/Android deployment.

Fix #	Issue	Status
1	EventBus to EventEmitter import	FIXED
2	CacheManager to Cache import	FIXED
3	SafeModifier to CodeValidator	FIXED
4	SandboxExecutor to ExecutionSandbox	FIXED
5	SIGTERM Windows compatibility	FIXED
6	resource module Android fallback	FIXED
7	Missing dependencies in install.sh	FIXED
8	Self-Modification Bridge	FIXED

2. One-Command Setup for Termux

Copy and paste this entire command block into Termux. It will install everything automatically:

```
pkg update && pkg upgrade -y
pkg install python git -y
pip install click colorama python-dotenv pyyaml requests tqdm schedule typing-extensions rich loguru cryptography
httpx psutil
cd ~
git clone --depth 1 https://github.com/71261121/auto-jarvis-.git jarvis_v14_ultimate
cd jarvis_v14_ultimate
mkdir -p ~/.jarvis/data ~/.jarvis/cache ~/.jarvis/backups
echo "export OPENROUTER_API_KEY='your-key-here'" >> ~/.bashrc
python main.py
```

3. Self-Modification Features - Now Fully Functional

JARVIS can now modify its own code. The AI has access to file operations through command tags:

Command	Description	Example
[READ:path]	Read any file	[READ:main.py]
[MODIFY:path]	Modify files with backup	[MODIFY:core/utils.py]
[CREATE:path]	Create new files	[CREATE:helpers.py]
[DELETE:path]	Delete with backup	[DELETE:old_file.py]
[ANALYZE:path]	Code analysis	[ANALYZE:main.py]
[LIST:path]	List directory	[LIST:core/]
[SEARCH:pattern]	Search codebase	[SEARCH:def hello]

4. Example Prompts to Test Self-Modification

After starting JARVIS, try these prompts to test the self-modification capabilities:

Read a file: "Read the main.py file and tell me what it does"

Modify code: "Add a new command called debug to main.py that enables debug mode"

Create file: "Create a new file called utils.py with string helper functions"

Analyze code: "Analyze the code_analyzer.py file for issues"

List directory: "List all files in the core directory"

Add feature: "Add a feature to export chat history to JSON"

5. Project File Structure

```
jarvis_v14_ultimate/
├── main.py # Main entry point
├── core/
│ ├── self_mod/
│ │ ├── bridge.py # Self-modification bridge
│ │ ├── code_analyzer.py # Code analysis
│ │ ├── safe_modifier.py # Safe modifications
│ │ └── backup_manager.py # Backup system
│ ├── ai/
│ │ └── openrouter_client.py
│ └── cache.py, events.py, etc.
├── security/
│ └── sandbox.py # Fixed for Termux
├── interface/
│ └── cli.py, commands.py
└── install/
└── install.sh # Updated dependencies
```

6. Requirements (All Included)

Package	Version	Purpose
click	>=8.0.0	CLI framework
colorama	>=0.4.0	Terminal colors
python-dotenv	>=0.19.0	Environment vars
requests	>=2.26.0	HTTP client
rich	>=12.0.0	Rich output
cryptography	>=3.4.0	Encryption
httpx	>=0.24.0	Async HTTP
psutil	>=5.9.0	System monitoring

7. Troubleshooting

ModuleNotFoundError: Run: pip install <missing_package>

API Key Error: Set: export OPENROUTER_API_KEY='your-key'

Permission Denied: Run: chmod +x main.py

Memory Error: Close other apps, JARVIS optimized for 4GB RAM

psutil not working: Optional - JARVIS works without it

VERIFICATION COMPLETE

All 8 critical fixes have been applied. The project is ready for deployment on Termux/Android.