NEMO SYNTHESIS ENGINE - TECHNICAL DOCUMENTATION v2.0

Official Developer & User Guide (With CLI Cleanup)

TABLE OF CONTENTS

---

1. INSTALLATION & SETUP

System Requirements

Minimum

- Python 3.10+
- 200MB disk space
- 100MB RAM
- Windows 10+, macOS 10.14+, Linux (any)
- Admin/sudo privileges (for keyboard interception)

Recommended

- Python 3.11+
- 1GB disk space
- 300MB RAM
- SSD storage
- 5MB/sec internet (for Gemini API, optional)
- External microphone (for better speech recognition)

Installation Methods

Method 1: From downloadnemo.com (Recommended)

```
# 1. Download from https://downloadnemo.com
# 2. Extract ZIP file
# 3. Run setup
python setup.py install
```

```
pip install -r requirements.txt
nemo setup
```

Method 2: From GitHub

```
git clone https://github.com/torresjchristopher/nemo.git
cd nemo
pip install -e . --force-reinstall --no-deps
nemo setup
```

Method 3: From pip

```
pip install nemo
nemo setup
```

First-Run Setup

```
nemo setup
```

Interactive wizard configures:

1. **AI Model Selection**
    - Gemini (Google Cloud, multimodal, supports screenshots)
    - Claude (Anthropic, reasoning)
    - Ollama (Local, completely offline)
2. **API Credentials** (if using cloud)
    - Stores encrypted at `~/.nemo/credentials.json`
    - Never logged or transmitted
3. **Button Mapping**
    - RIGHT SHIFT: Speech-to-text (default)
    - RIGHT ALT: Gemini voice + screenshot (default)
    - RIGHT ALT + LEFT: Rewind (default)
    - RIGHT ALT + RIGHT: FORWARD prediction (default)
4. **Audio Settings**
    - Microphone device selection
    - TTS voice (male/female/neutral)
    - Speech rate (0.5x - 2.0x)
    - Audio output device
5. **Privacy Settings**
    - Verification level (quick/full)
    - Log retention (never/24h/7d)

---

2. CONFIGURATION

Config File

```
~/.nemo/nemo_config.json
```

Example Configuration

```
{
  "version": "1.0.0",
  "ai_model": "gemini",
  "synthesis": {
    "keystroke_dimensions": 35,
    "temporal_buffer_seconds": 300,
    "prediction_confidence_threshold": 0.75,
    "forward_prediction_horizon_seconds": 5
  },
  "buttons": {
    "speech_to_text": "right shift",
    "gemini_voice_ai": "right alt",
    "rewind": "right alt+left",
    "forward": "right alt+right"
  },
  "gemini": {
    "api_key": "encrypted",
    "screenshot_enabled": true,
    "video_recording_enabled": false,
    "context_level": "normal"
  },
  "voice": {
    "microphone_device": "default",
    "speech_recognition_engine": "google",
    "tts_engine": "pyttsx3",
    "tts_voice": "female",
    "tts_rate": 1.0,
    "tts_pitch": 1.0
  },
  "security": {
    "audit_level": "full",
    "log_retention_hours": 0,
    "verify_on_startup": false
  }
}
```

Environment Variables

```
# API Keys
export GEMINI_API_KEY="your-api-key"
export CLAUDE_API_KEY="your-api-key"

# Nemo Home Directory
export NEMO_HOME="/custom/path"

# Logging
export NEMO_LOG_LEVEL="INFO"  # DEBUG, INFO, WARNING, ERROR
```

```
# Settings
export NEMO_ADMIN_MODE="true"  # Enable for testing
```

---

3. CLI COMMANDS (Streamlined)

Core System Commands

nemo buttons start

```
nemo buttons start
# Starts 4-button hotkey system
# Initializes keyboard interception
# Requires: Admin/sudo privileges
# Output: Shows active hotkeys, waits for input
```

nemo buttons stop

```
nemo buttons stop
# Gracefully stops hotkey system
# Clears audio buffers
# Closes microphone
```

nemo buttons test

```
nemo buttons test
# Test each hotkey individually
# Press each button when prompted
# Verifies detection and response
```

Configuration Commands

nemo config

```
nemo config
# Interactive configuration wizard
# Same as first-run setup
```

nemo config gemini

```
nemo config gemini

# Options:
#   --screenshot ON/OFF          Enable/disable screenshot capture
#   --video ON/OFF               Enable/disable video recording
#   --api-key YOUR_KEY           Set Gemini API key
#   --context-level minimal/normal/full
#                                What context to capture
#   --preview                    Preview screenshot before sending
```

nemo config speech-to-text

```
nemo config speech-to-text

# Options:
#   --auto-paste ON/OFF          Auto-insert transcribed text
#   --real-time ON/OFF           Show live transcription
#   --read-highlighted ON/OFF    Read selected text aloud
#   --timeout-seconds N          Listen timeout (default: 10)
#   --language LANG              Language (en, es, fr, de, etc)
```

nemo config forward

```
nemo config forward

# Options:
#   --enabled ON/OFF             Enable forward prediction
#   --prediction-count N         Show N predictions (1-5)
#   --confidence-threshold 0-1   Minimum confidence
#   --horizon-seconds N          How far ahead to predict
#   --learning-mode ON/OFF       Learn from corrections
```

nemo config buttons

```
nemo config buttons

# Options:
#   --map BUTTON ACTION          Remap button to action
#   --reset                      Reset to defaults
#   Example: nemo config buttons --map "right shift" "tts"
```

nemo config audio

```
nemo config audio

# Options:
#   --microphone-list            Show available microphones
#   --microphone N               Select microphone by index
#   --speaker-list               Show available speakers
#   --speaker N                  Select speaker by index
#   --test-mic                   Test microphone recording
#   --test-speaker               Test speaker output
```

Voice & TTS Commands

nemo tts speak

```
nemo tts speak "Your text here"
# Convert text to speech
# Plays via default audio device
# No recording, no storage
```

nemo tts test

```
nemo tts test
# Plays test phrase to verify TTS works
# Tests current voice settings
```

nemo tts voices

```
nemo tts voices
# List available TTS voices
# Shows: name, gender, language
```

Synthesis Commands

nemo synthesize

```
nemo synthesize

# Options:
#   --context "Your context"     Provide context for synthesis
#   --json                       Output as JSON
#   --verbose                    Show detailed analysis
# Returns: Predicted next actions with confidence scores
```

nemo forward

```
nemo forward

# Predict next action immediately
# Options:
#   --count N                    Show N predictions
#   --confidence-threshold 0-1   Minimum confidence
#   --horizon-seconds N          How far ahead (default: 5)
```

nemo rewind

```
nemo rewind

# Options:
#   --minutes N                  How far back to infer
#   --json                       Output as JSON
# Returns: Synthesis of what you were doing N minutes ago
```

Security & Verification

nemo security verify

```
nemo security verify
# Runs 8-point security audit
# Checks:
```

```
#   ✓ Temp directories (no files)
#   ✓ Cache directories (clean)
#   ✓ Memory state (no strings)
#   ✓ Log files (no sensitive data)
#   ✓ Credentials storage (encrypted)
#   ✓ Clipboard (not captured)
#   ✓ Network traffic (no exfil)
#   ✓ Behavioral verification (synthesis only)
# Returns: PASS/FAIL for each check
```

nemo security report

```
nemo security report
# Generate detailed security report
# Saves to ~/.nemo/security_report.json
# Lists all findings and recommendations
```

nemo security audit [PATH]

```
nemo security audit
# Audit custom directory
# Options:
#   --recursive                  Scan subdirectories
#   --scan-memory                Check process memory
#   --network-capture            Capture network traffic
```

System Status & Info

nemo status

```
nemo status
# Show current system status
# Displays:
#   - Hotkey system running/stopped
#   - AI model connected
#   - Microphone detected
#   - Uptime
#   - Active synthesis count
```

nemo version

```
nemo version
# Show installed version
# Check for updates
```

nemo info

```
nemo info
# System information
# Displays:
#   - Python version
#   - OS and architecture
#   - Installed dependencies
#   - Configuration paths
```

nemo health

```
nemo health
# Health check on all components
# Tests:
#   - Microphone
#   - TTS engine
#   - AI model connection
#   - Keyboard interception
#   - File permissions
```

Logging & Debugging

nemo logs

```
nemo logs

# Options:
#   --tail N                        Show last N lines
#   --follow                        Follow log in real-time
#   --level LEVEL                   Filter by level
#   --since TIMESTAMP               Logs since timestamp
#   --json                          Output as JSON
```

nemo logs clear

```
nemo logs clear
# Clear all log files
# Options:
#   --confirm                       Skip confirmation
```

---

4. HOTKEY SYSTEM (KEYBOARD LIBRARY)

Architecture

Primary Library: `keyboard` (Windows/Linux) + `pynput` fallback (macOS)

Key Names Used:

- `'right shift'` - RIGHT SHIFT key
- `'right alt'` - RIGHT ALT key
- `'left'` - LEFT ARROW
- `'right'` - RIGHT ARROW

Hotkey Detection

Press-Hold-Release Detection (via keyboard.hook())

```
# When you press RIGHT SHIFT:
# 1. DOWN event fires → recording starts
# 2. Debounced (ignores key repeat)
# 3. Starts listening to microphone
# 4. When released, UP event fires → recording stops
```

Requirements

Windows

- Run PowerShell as Administrator
- keyboard library requires admin privileges for system-level capture
- No additional dependencies

macOS

- Grant microphone permission
- Grant accessibility permission (System Preferences > Security)
- May require: `sudo` when first running

Linux

- Run with sudo first time (for keyboard hook)
- sudo dmesg | grep -i deny # Check for denials

---

5. GEMINI INTEGRATION & SCREENSHOT CONTROL

Configuration

Enable/Disable Screenshot Capture

```
# Enable screenshot with Gemini
nemo config gemini --screenshot ON
# Gemini can now see your screen when you hold RIGHT ALT

# Disable screenshot
nemo config gemini --screenshot OFF
# Gemini only gets voice, not visuals

# Check current status
nemo config gemini --status
```

Context Levels

```
# Minimal: Just your voice transcription
nemo config gemini --context-level minimal

# Normal: Voice + screenshot (if enabled)
nemo config gemini --context-level normal
```

```
# Full: Voice + screenshot + window title + file context
nemo config gemini --context-level full
```

How It Works

When you hold RIGHT ALT:

1. Voice starts recording (always)
2. IF screenshot enabled: captures current screen
3. Both sent to Gemini API
4. Gemini processes and responds
5. Response played as speech
6. **Everything local is cleared immediately**

Example Workflows

```
# Analyze code with screenshot
Hold RIGHT ALT
Say: "What's wrong with this?"
Screenshot shows code on screen
Gemini responds with analysis

# Understand data in spreadsheet
Hold RIGHT ALT
Say: "Summarize this month"
Screenshot shows spreadsheet data
Gemini responds with summary

# Get help with document
Hold RIGHT ALT
Say: "How should I rewrite this paragraph?"
Screenshot shows document
Gemini provides suggestions
```

API Settings

```
{
  "gemini": {
    "api_key": "your-key-here",
    "model": "gemini-2.0-flash",
    "max_tokens": 1024,
    "temperature": 0.7,
    "screenshot_enabled": true,
    "video_recording_enabled": false,
    "context_level": "normal",
    "timeout_seconds": 30
  }
}
```

---

6. VOICE SYSTEM

Speech-to-Text (RIGHT SHIFT)

How It Works

```
Hold RIGHT SHIFT
  ↓
Microphone starts recording
  ↓
Live transcription displays (real-time)
  ↓
Release RIGHT SHIFT
  ↓
Final text inserted at cursor
  ↓
Audio buffer cleared (no file created)
```

Configuration

```
nemo config speech-to-text

# Options:
#   --auto-paste ON/OFF          Insert text automatically
#   --real-time ON/OFF           Show partial transcription
#   --read-highlighted ON/OFF    Read selected text aloud
#   --timeout-seconds 5          How long to listen
#   --language en                Language code
```

Supported Languages

```
en (English)
es (Spanish)
fr (French)
de (German)
zh (Chinese)
ja (Japanese)
pt (Portuguese)
ru (Russian)
... and 20+ more
```

Text-to-Speech Output

Engines

pyttsx3 (Local, offline, no API key)

- Voices: Male, Female, Neutral
- Speed: 0.5x - 2.0x
- Installed by default

Google Cloud TTS (Optional, higher quality)

- Premium voices

- 100+ languages
- Requires API key

Configure TTS

```
nemo config audio

# Select microphone device
nemo config audio --microphone-list
nemo config audio --microphone 0

# Select speaker device
nemo config audio --speaker-list
nemo config audio --speaker 0

# Test TTS
nemo config audio --test-speaker
```

Audio Data Invisibility

Speech-to-Text Flow:

```
Microphone → RAM Buffer → Google Speech API (if cloud)
→ Text extracted → RAM buffer cleared → Audio gone
```

Text-to-Speech Flow:

```
Text → TTS Engine → RAM Audio Buffer → Speakers
→ Buffer cleared → Audio gone
```

Zero file creation at any point.

---

7. SECURITY VERIFICATION

8-Point Audit System

Run anytime: `nemo security verify`

Check 1: Temp Directory

```
# Verifies: No audio files in Windows temp or /tmp
# Expected: 0 .wav, .mp3, .flac files
# Data Invisibility: Audio never written to disk
```

Check 2: Cache Directory

```
# Verifies: ~/.nemo/cache is clean
# Expected: No audio or keystroke data
# Data Invisibility: Cache cleared between sessions
```

Check 3: Memory Forensics

```
# Verifies: No audio strings in process memory
# Expected: No partial audio file names
# Data Invisibility: Audio buffer overwritten after use
```

Check 4: Log Files

```
# Verifies: Logs don't contain sensitive data
# Expected: No API calls, no personal data
# Data Invisibility: Sensitive info not logged
```

Check 5: Credentials Storage

```
# Verifies: API keys encrypted
# Expected: credentials.json is encrypted, not plaintext
# Data Invisibility: Keys stored securely
```

Check 6: Clipboard Monitoring

```
# Verifies: Clipboard not captured
# Expected: No clipboard data in logs
# Data Invisibility: Clipboard not read/stored
```

Check 7: Network Analysis

```
# Verifies: No data exfiltration
# Expected: Only intentional API calls (Gemini, Claude)
# Data Invisibility: No unexpected data transmission
```

Check 8: Behavioral Verification

```
# Verifies: System behaves as documented
# Expected: Synthesis only, no recording
# Data Invisibility: Confirmed through runtime analysis
```

Running Full Security Audit

```
nemo security verify --full

# Output:
# ✓ Check 1: Temp directories clean
# ✓ Check 2: Cache clean
# ✓ Check 3: Memory forensics passed
# ✓ Check 4: Logs verified
# ✓ Check 5: Credentials encrypted
# ✓ Check 6: Clipboard not accessed
# ✓ Check 7: Network traffic verified
# ✓ Check 8: Behavioral verification passed
#
# RESULT: ALL CHECKS PASSED ✓
```

---

8. TROUBLESHOOTING

Issue: "Failed to import required modules"

Cause: Keyboard library not found

Solution:

```
# Install with admin privileges
python -m pip install keyboard

# Run PowerShell as Administrator and retry
nemo buttons start
```

Issue: Microphone not detected

Solution:

```
# List available microphones
nemo config audio --microphone-list

# Select specific microphone
nemo config audio --microphone 1

# Test microphone
nemo config audio --test-mic
```

Issue: RIGHT SHIFT not firing

Cause:

- Not running as admin (Windows)
- Key repeat debouncing issue
- Keyboard library not hooked

Solution:

```
# Windows: Run as Administrator
sudo python -m nemo.cli buttons start

# Test key detection
nemo buttons test
# Press RIGHT SHIFT when prompted
```

Issue: Transcription timeout

Cause: Microphone not picking up sound

Solution:

```
# Test mic at system level
# System Preferences → Sound → Check input level
# Adjust speech recognition sensitivity
nemo config speech-to-text --energy-threshold 4000

# Increase timeout
nemo config speech-to-text --timeout-seconds 15
```

Issue: Gemini screenshot not captured

Cause: Screenshot disabled in config

Solution:

```
# Enable screenshots
nemo config gemini --screenshot ON

# Test with verbose output
nemo config gemini --preview
# Will show screenshot before sending
```

Issue: Logs too large

Solution:

```
# Check log retention
nemo config

# Clear old logs
nemo logs clear --confirm

# Set to never log (for privacy)
# Edit ~/.nemo/nemo_config.json:
#   "log_retention_hours": 0
```

---

9. DEVELOPMENT GUIDE

Project Structure

```
nemo/
├── core/
│   ├── cli.py                 # Main CLI interface
│   ├── buttons_start_new.py   # Hotkey system
│   └── config.py              # Configuration manager
├── systems/
│   └── task-screen-simulator/
│       ├── keyboard_hotkeys.py    # Hotkey detection
│       ├── voice_input.py         # Speech-to-text
│       ├── tts_engine.py          # Text-to-speech
│       └── screen_analyzer.py     # Screen context
├── synthesis/
│   ├── keyboard_synthesizer.py   # Behavior signature
│   ├── temporal_inference.py     # Rewind/Forward
│   └── intent_detector.py        # Intent classification
│
```

```
└── security/
    └── audit.py               # Security verification
```

Adding a New CLI Command

```
# In nemo/core/cli.py

@app.command()
def mycommand(
    param1: str = typer.Argument(..., help="Description"),
    param2: bool = typer.Option(False, help="Description")
):
    """Command description."""
    # Implementation
    pass
```

Testing Hotkeys

```
# Test RIGHT SHIFT
nemo buttons test
# Press RIGHT SHIFT
# Expected: [DEBUG] RIGHT SHIFT PRESSED (DOWN)
#           [DEBUG] RIGHT SHIFT RELEASED (UP, held X.XXs)

# Test complete workflow
nemo buttons start
# Hold RIGHT SHIFT and speak
# Observe: Real-time transcription
# Release RIGHT SHIFT
# Expected: Final transcribed text
```

Debug Mode

```
# Set log level to DEBUG
export NEMO_LOG_LEVEL=DEBUG
nemo buttons start

# Verbose output for diagnostics
nemo buttons start --verbose

# Follow logs in real-time
nemo logs --follow
```

QUICK REFERENCE

Most Important Commands

```
# Setup NEMO first time
nemo setup

# Start hotkey system (MAIN COMMAND)
```

```
nemo buttons start

# Configure Gemini screenshot
nemo config gemini --screenshot ON/OFF

# Configure speech-to-text
nemo config speech-to-text

# Test all components
nemo health

# Verify data invisibility
nemo security verify
```

Configuration Files

```
~/.nemo/nemo_config.json      # Main config
~/.nemo/credentials.json      # Encrypted API keys
~/.nemo/nemo.log             # Application logs
~/.nemo/security_report.json # Last security audit
```

Environment Variables

```
GEMINI_API_KEY      # Gemini API key
NEMO_HOME           # Override Nemo home directory
NEMO_LOG_LEVEL      # DEBUG, INFO, WARNING, ERROR
```

SUPPORT & RESOURCES

- **GitHub Repository**: https://github.com/torresjchristopher/nemo
- **Issue Tracker**: Report bugs and feature requests
- **Documentation**: Full guides at docs/
- **Security**: Responsible disclosure to security@nemo.local

---

NEMO v1.0.0 - The revolutionary personal AI with data invisibility.

Your data. Your rules. Your future.