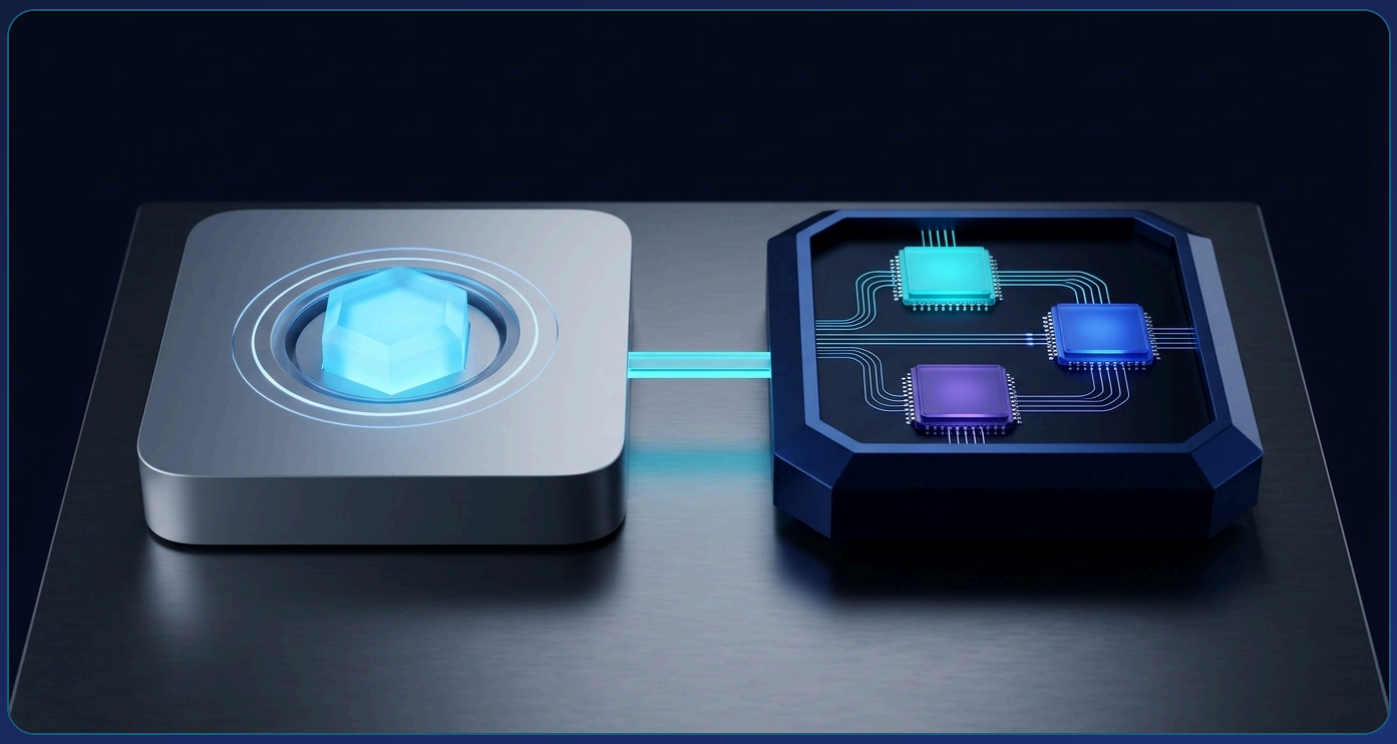
DESKTOP OS & PERSONAL OPS • CROSS-PLATFORM FIELD GUIDE

Mac vs. Windows AI Power-User Playbook

Master both sides of modern desktop AI—comparing Apple Silicon unified memory, Apple Intelligence, Shortcuts, and Raycast against Windows 11 Copilot+ PCs, Click to Do, opt-in Recall, PowerToys Command Palette, MLX, DirectML/ONNX NPUs, and WSL2 CUDA passthrough.

Apple Intelligence & PCC | Copilot+ & Click to Do | Unified RAM vs. 40+ TOPS NPU | Raycast & PowerToys Parity | MLX vs. DirectML & WSL2 | Recall & Privacy Guardrails



Silicon-Matched Compute

Size models to Apple Silicon unified memory bandwidth with `MLX`, or offload background AI onto 45–80 TOPS Windows NPUs and `WSL2` CUDA GPUs.

Mirrored Daily Launchers

Keep identical keyboard muscle memory across Raycast (now on both Mac & Windows), Apple Shortcuts, PowerToys Command Palette, and Advanced Paste.

Verified Privacy Boundaries

Enforce cryptographic Private Cloud Compute checks on macOS alongside BitLocker, Windows Hello ESS, and strict app/site Recall exclusions.

Two Philosophies of Personal AI Silicon & Compute

macOS and Windows 11 solve on-device intelligence from opposite hardware directions. Apple pools CPU, GPU, and Neural Engine access into one high-bandwidth unified memory slab, while Microsoft orchestrates specialized execution providers across dedicated 40+ TOPS NPUs, discrete GPUs, and Linux WSL2 containers.

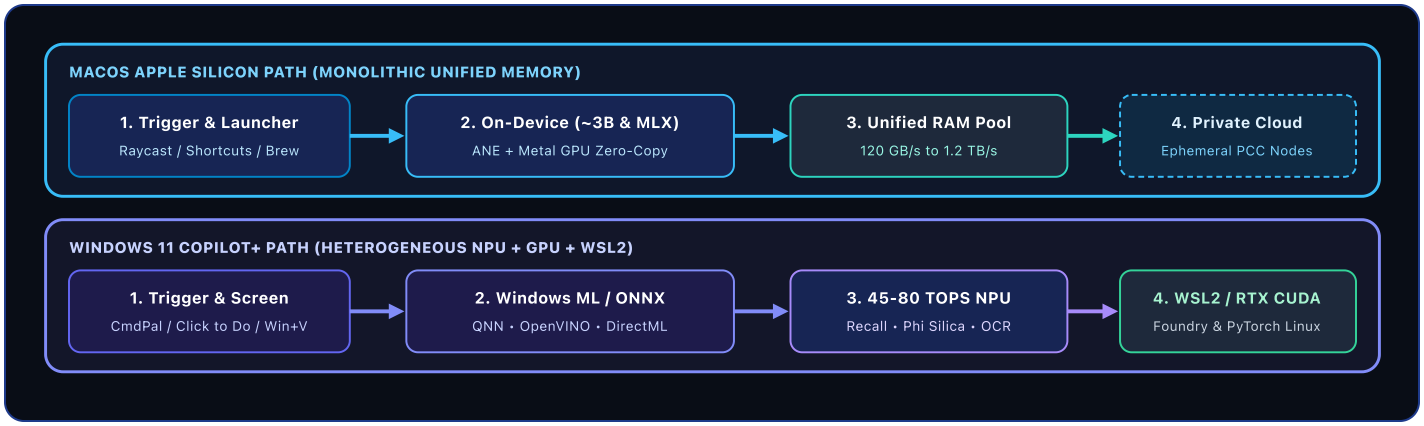
Quick Navigation

1. Two Philosophies of AI Silicon p. 2
2. Hardware Tiers & Decision Matrix p. 3
3. Dual-OS Launcher & Screen Tour p. 4
4. Mac Stack: Apple AI & Raycast p. 5
5. Hotkey Parity & 5-Part Formula p. 6
6. Windows Copilot+ & PowerToys p. 7
7. Compute: MLX vs. DirectML & WSL2 p. 8
8. 5 Copy-Paste Dual-OS Recipes p. 9
9. Privacy: PCC, Recall & Admin Policy p. 10
10. Cross-OS Synergy Pipeline p. 11
11. Troubleshooting & Pre-Flight Check p. 12
Golden Rule: Match Work to Silicon All

The Architectural Split: Unified Pool vs. Modular Execution Providers

Local token speed rarely bottlenecks on raw math cores; it bottlenecks on how fast weights stream out of memory. On Apple Silicon, one shared pool feeds Metal and `mlx-lm` without copying data across a PCIe slot. On Windows 11 Copilot+ hardware, Windows ML and ONNX Runtime dynamically route lightweight always-on models to an ultra-efficient NPU while heavier inference targets DirectML, TensorRT, or WSL2 CUDA.

Architecture: Side-by-Side Execution Paths Across macOS & Windows 11



Core Rule: Stop Fighting Your Machine's Silicon

Use Apple Silicon unified memory and `mlx_lm` when you want large local LLMs (14B–70B) running silently on a laptop. Use Windows Copilot+ NPUs for low-wattage always-on desktop vision (Click to Do, Recall, local OCR) and lean on discrete NVIDIA CUDA inside WSL2 for heavy batch training or high-throughput local serving.

Silicon Tiers, Onboarding & Mac-vs-Windows Matrix

Before tuning launchers or local models, check what hardware tier you are standing on. Unified memory bandwidth determines large-model speed on macOS, while Copilot+ certification and discrete VRAM govern NPU and CUDA workloads on Windows 11.

1 macOS Readiness (Sequoia & Tahoe)

- **Silicon Floor:** Apple M1 or newer with **16 GB+ unified RAM**; 24–64 GB unlocks smooth 14B–32B 4-bit MLX weights alongside Xcode or Docker.
- **Bandwidth Ladder:** Base M4/M5 (~120–150 GB/s), Pro (273–307 GB/s), Max (546–614 GB/s), and Mac Studio Ultra (~1.2 TB/s) scale token rates linearly.
- **Native AI Stack:** Enable `System Settings → Apple Intelligence & Siri`; verify matching system/Siri language and FileVault encryption.
- **Power Layer:** Install Homebrew (`brew`), Raycast (or Alfred Powerpack), and Python `mlx-lm` for zero-copy Metal inference.

2 Windows 11 Copilot+ Readiness (24H2+)

- **Copilot+ Floor: 40+ TOPS NPU** (Snapdragon X Elite 45 / X2 80 TOPS, Intel Lunar Lake 48 TOPS, AMD Strix Point/Halo 50 TOPS), 16 GB RAM, 256 GB SSD.
- **Security Prerequisites:** Opt-in Recall requires active **BitLocker / Device Encryption** and **Windows Hello Enhanced Sign-in Security (ESS)** biometrics.
- **Power Layer:** Install **PowerToys** (Command Palette + Advanced Paste), **Raycast for Windows v2.0**, and **Microsoft Foundry Local**.
- **Linux Escape Hatch:** Run `wsl --install` with host-only NVIDIA Studio/Game Ready drivers for near-native CUDA inside Ubuntu.

90-Second Cross-OS Parity Bootstrap (Terminal One-Liners)

macOS Terminal: `brew install --cask raycast && pip3 install -U mlx-lm` | **Windows PowerShell:** `winget install --id Microsoft.PowerToys -e && winget install --id Microsoft.FoundryLocal -e && winget install --id 9PFXXSHC64H3 -e`

Hardware & Ecosystem Decision Matrix (2025–2026 Silicon)

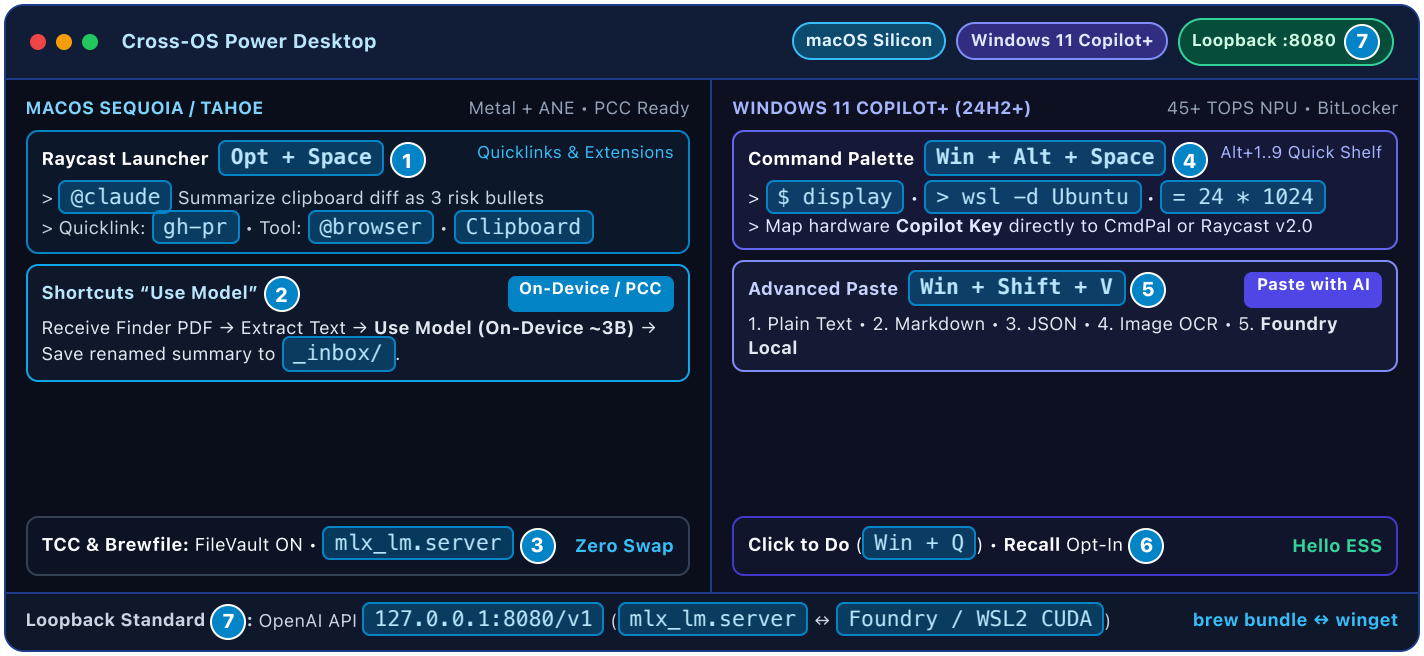
Machine Class	Memory Bandwidth	Accelerator Topology	Native OS AI Features	Best Power-User Role
MacBook Air / Pro *M4 or M5 Base / Pro*	120–307 GB/s unified memory	Apple GPU + ~35–38 TOPS ANE via Metal & MLX	Writing Tools, Shortcuts Use Model, Private Cloud Compute	Silent battery-friendly daily coding, Raycast AI & 7B–14B MLX models.
Mac Studio / Max *M4/M5 Max & Ultra*	546 GB/s to 1.2 TB/s unified	High-core Metal GPU sharing 64–512 GB RAM	Full Apple Intelligence + massive zero-copy local context	Running 32B–120B+ local reasoning models & LoRA fine-tuning offline.
Copilot+ Ultrabook *Snapdragon X / Lunar Lake*	100–228 GB/s LPDDR5X	45–80 TOPS dedicated NPU (QNN / OpenVINO EP)	Click to Do, Opt-in Recall, Phi Silica, Studio Effects	All-day office ops, screen OCR, instant PowerToys Paste with AI.
AMD Strix Halo PC *Ryzen AI Max (256-bit)*	~256–273 GB/s quad-channel	50 TOPS XDNA 2 NPU + large shared iGPU pool	Copilot+ NPU suite + Windows ML / DirectML / ROCm	Windows counterpart to Mac Pro-tier unified memory for big local LLMs.
Windows RTX Tower *Desktop + WSL2 Ubuntu*	120 GB/s system + 500–1,000+ GB/s GDDR6X/7	Discrete NVIDIA Tensor Cores + TensorRT EP / CUDA	PowerToys AI + Foundry Local + native WSL2 Linux CUDA	Heavy PyTorch training, vLLM multi-user serving, image/video diffusion.

Smart Two-Machine Strategy: Portable Mac + Windows Workstation (or Vice Versa)

You no longer need to abandon favorite workflows when switching desks. With Raycast v2.0 now running natively on Windows 11 alongside PowerToys Command Palette and Foundry Local, one standardized shortcut scheme and Git-backed dotfile setup travels cleanly across both operating systems.

Annotated Tour: Mirrored Mac & Windows AI Controls

Although Apple and Microsoft build radically different system UIs, power users rely on seven parallel control points—mapping global launchers, clipboard transforms, on-screen vision overlays, and local model endpoints into one coherent daily rhythm.



1 **Raycast Launcher (Mac & Win v2.0):** Unified `Opt+Space` command bar running multi-model Quick AI, Quicklinks, and React/TS Extensions on both platforms.

2 **Shortcuts "Use Model" Action:** Native macOS Sequoia/Tahoe automation step routing text through On-Device Apple models, PCC, or opt-in ChatGPT.

3 **TCC & Homebrew Provisioning:** Declarative `Brewfile` app bundles paired with strict macOS Privacy & Security permissions and local MLX acceleration.

4 **PowerToys Command Palette:** Fast keyboard launcher (`Win+Alt+Space`) with `$` settings jumps, WSL commands, and `Alt+1..9` pinned shelf slots.

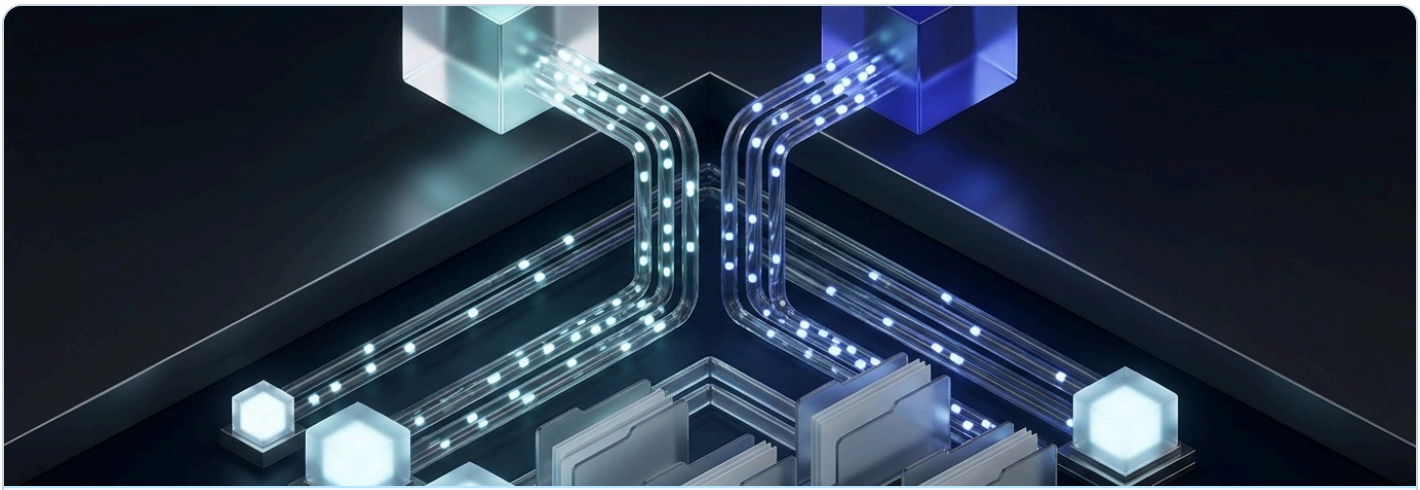
5 **Advanced Paste (`Win+Shift+V`):** Converts clipboard data to Plain Text, Markdown, JSON, local image OCR, or AI transforms via Foundry Local / Phi Silica.

6 **Click to Do & Recall Guardrails:** Press `Win+Q` for instant on-screen NPU actions while keeping opt-in Recall protected behind BitLocker, ESS, and exclusions.

7 **Shared Loopback Endpoint Standard:** Expose `mlx_lm.server` on macOS or Foundry Local / Ollama on Windows behind `127.0.0.1` so scripts, editor plugins, and Raycast AI work identically on every desk.

macOS Power Stack: Apple Intelligence, Shortcuts & Raycast

On Apple Silicon, native privacy meets effortless keyboard velocity. Combine system-wide Writing Tools and deterministic Shortcuts automations with Raycast Quicklinks, Alfred offline scripts, and reproducible Homebrew environments.



Keyboard-First Orchestration: Clean macOS launchers turn fleeting clipboard selections, Finder files, and shell outputs into deterministic Shortcuts or multi-model Raycast extensions without touching a browser tab.

Four Pillars of a Modern macOS AI Setup

1 On-Device Models vs. PCC Escalation

Apple's ~3B on-device foundation model handles instant Writing Tools rewrites and summaries locally. Larger reasoning steps escalate automatically to **Private Cloud Compute (PCC)** nodes running Apple Silicon with cryptographic attestation and zero persistent storage.

2 Shortcuts "Use Model" Pipelines

In macOS Sequoia/Tahoe Shortcuts, pair deterministic triggers (Finder Quick Actions, folder watches, or `shortcuts run`) with the built-in **Use Model** block—explicitly choosing On-Device, Private Cloud Compute, or ChatGPT with zero API key overhead.

3 Raycast AI, Quicklinks & Extensions

Use Raycast (`Opt+Space`) when you want model choice (Claude, GPT, Gemini, or local Ollama/MLX), dynamic **AI Quicklinks** with `{selection}` or `{clipboard}` tokens, and `@tool` mentions across GitHub, Linear, and your active browser tab.

4 Alfred Powerpack & Homebrew `Brewfile`

Prefer zero-account local speed? **Alfred** remains unmatched for instant offline file filters and custom shell Script Filters. Anchor your CLI tools, casks, and fonts in a versioned `Brewfile` synced via `brew bundle`.

```
# ~/.Brewfile — Declarative macOS AI power-user setup (apply with: brew bundle --global)
cask "raycast"            # Global launcher, AI Quicklinks, Extensions & Clipboard History
cask "orbstack"           # Lightweight Linux containers on Apple Silicon
brew "mlx"                # Apple Silicon array runtime; install LLM tools via: pip3 install mlx-lm

# Trigger a saved Apple Intelligence Shortcuts workflow directly from zsh or Raycast Script Command:
pbpaste | shortcuts run "Clean Clipboard to Markdown" | pbcopy
```

Pro Tip: Keep Triggers Deterministic, Let the Model Handle Only Ambiguity

Do not ask an AI agent to rename files or move folders blindly. Let macOS Shortcuts or zsh gather exact file metadata first, pass only extracted text into **Use Model** (On-Device), and validate output format before writing changes back to disk.

Cross-OS Hotkey Parity & the 5-Part Workflow Formula

Switching between a MacBook Pro and a Windows 11 Copilot+ desk feels effortless when your muscle memory stays identical. Standardize four core shortcut pairs, then compose every desktop AI action from the same five-part blueprint.

macOS Muscle-Memory Map

- **Command Launcher:** `Opt + Space` (Raycast) or `Cmd + Space` (Spotlight)
- **Clipboard & Paste AI:** `Cmd + Shift + V` (Raycast Clipboard / Paste Plain)
- **Screen Region AI / OCR:** `Cmd + Shift + 5` → Shortcut / Raycast OCR
- **Package Provisioning:** `brew install` & `brew bundle dump`

Windows 11 Copilot+ Mirror Map

- **Command Launcher:** `Win + Alt + Space` (Command Palette) or Raycast v2.0
- **Clipboard & Paste AI:** `Win + Shift + V` (PowerToys Advanced Paste)
- **Screen Region AI / OCR:** `Win + Q` / `Win + Click` (Click to Do overlay)
- **Package Provisioning:** `winget import -i winget.json`

The 5-Part Cross-OS Desktop AI Workflow Formula

[1. Hotkey Trigger] Instant muscle-memory chord (`Opt+Space` / `Win+Shift+V`) [2. Local Context] pulls selected text, clipboard, or screen OCR, [3. Silicon Runtime] routes to MLX / PCC on Mac or NPU / Foundry on Win, [4. Structured Transform] outputs clean JSON, Markdown, or diff summary, [5. Privacy & Approval Gate] after checking app exclusions and human review before paste.

Workflow Element	What It Controls	What Fails When Skipped
1. Hotkey Trigger	Binds one memorable chord (launcher or paste menu) to fire without touching the mouse.	Opening a browser chat tab breaks deep focus and slows quick tasks by 5x.
2. Local Context	Grabs exact clipboard text, active file path, or Click to Do screen selection automatically.	You waste time manually copying paragraphs or re-explaining which file you mean.
3. Silicon Runtime	Matches task size to local NPU/MLX for speed and privacy, or frontier cloud for deep logic.	Sensitive logs leak to external APIs or tiny NPUs choke on 64K-token codebases.
4. Structured Transform	Forces strict output shape (JSON schema, table, Conventional Commit, or bullet checklist).	Chatty prose spills into your terminal or Slack channel, forcing manual cleanup.
5. Privacy Gate	Enforces password-manager exclusions, PII scrubbing, and clipboard preview before pasting.	Secrets, API keys, or confidential client details accidentally enter history or logs.

```
// Portable Raycast AI Command & PowerToys Advanced Paste Prompt (Identical Across Mac & Windows)
SYSTEM: Convert the clipboard snippet into valid JSON with keys: "summary" (1 sentence), "risks" (array),
"next_action".
RULES: Strip emails, bearer tokens, and internal IP addresses as [REDACTED]. Output raw JSON only, no Markdown fences.
```

Parity Tip: Keep Outputs Machine-Readable by Default

Asking both Raycast Quick AI and PowerToys Advanced Paste for strict JSON or Markdown bullet lists prevents chatty preambles, letting you paste straight into terminal pipes, Git commit messages, Linear tickets, or Obsidian notes without manual trimming.

Windows Copilot+: Click to Do, PowerToys & Recall

Windows 11 Copilot+ PCs combine 40+ TOPS NPUs with Microsoft's sharpest utility kit yet—pairing Click to Do visual screen intelligence with PowerToys Command Palette, Advanced Paste AI, Raycast for Windows v2.0, and opt-in Recall.

1 Click to Do (`Win + Q`)

Press `Win + Q` or `Win + Click` to freeze an interactive overlay over your screen. Select any visible paragraph or image to summarize, rewrite, extract tables, blur backgrounds, or launch directly into apps using local NPU models.

2 Command Palette (CmdPal)

Press `Win + Alt + Space` (or remap the Copilot key) to summon PowerToys Command Palette. Search apps, run `$` system settings jumps, execute shell commands, calculate math, and pin top actions to an `Alt+1..9` quick shelf.

3 Advanced Paste AI

Press `Win + Shift + V` to transform clipboard text into Plain Text, Markdown, JSON, audio/video transcodes, or local image OCR. Enable **Paste with AI** backed by **Foundry Local**, Ollama, or Phi Silica for 100% offline reformatting.

Windows 11 AI Power-Tool Matrix

Windows Utility	Trigger / Hotkey	Compute Engine	Power-User Workflow Sweet Spot
Click to Do *Copilot+ Exclusive*	`Win + Q` or `Win + Click`	On-device 40+ TOPS NPU (Phi Silica / Vision)	Grab unselectable text from Zoom screen-shares, screenshots, or PDFs and summarize instantly.
Windows Recall *Strictly Opt-In*	Tray Icon / Recall Search	NPU Semantic Index + VBS Encrypted Store	Locate a design mockup, terminal error, or web page you glanced at three days ago (with strict exclusions).
PowerToys Command Palette	`Win + Alt + Space` or Copilot Key	Native WinUI 3 + Extensible Plugins	Launch apps, switch Windows settings via `$`, jump into WSL distros, and trigger custom scripts fast.
PowerToys Advanced Paste	`Win + Shift + V`	Local OCR + Foundry Local / Ollama / Cloud	Turn messy HTML tables into clean Markdown or extract text from snipped error dialogs before pasting.
Raycast for Windows v2.0	`Alt + Space` (customizable)	Multi-Model Cloud + Local OpenAI Endpoints	Share identical Quicklinks, Dictate to AI, and React/TS Extensions directly with your macOS setup.

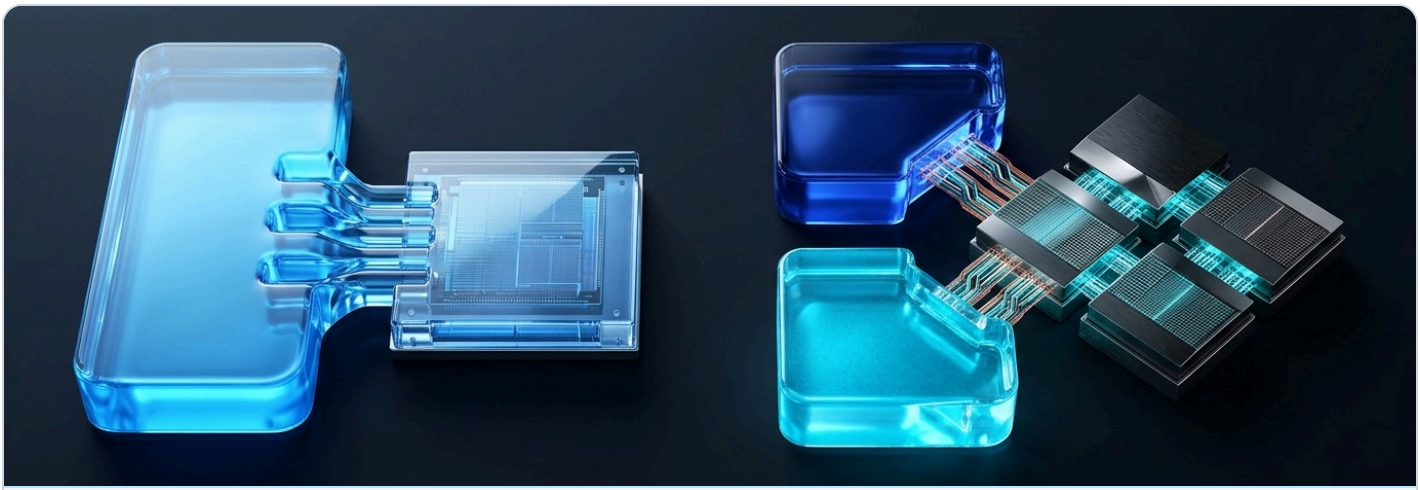
```
# PowerShell Workstation Bootstrap: install PowerToys, Foundry Local, Raycast v2.0 & WSL2 Ubuntu
winget install --id Microsoft.PowerToys -e
winget install --id Microsoft.FoundryLocal -e
winget install --id 9PFXXSHC64H3 -e
wsl --install -d Ubuntu-24.04
```

Copilot+ Best Practice: Remap the Hardware Copilot Key to Command Palette or Raycast

If you rarely open the web-style Copilot sidebar, open `Settings → Personalization → Text input` (or PowerToys Keyboard Manager) and bind your dedicated Copilot key straight to Command Palette or Raycast for Windows—turning a static key into a true command hub.

Local AI Acceleration: MLX, DirectML, ONNX & WSL2

Running private models fast requires speaking your hardware's native dialect. On Mac, Apple's `mlx` framework compiles straight onto Metal and unified memory; on Windows, execution providers split across NPUs, DirectML GPUs, Foundry Local, and WSL2 Linux CUDA.



Memory Topology Dictates Speed: Apple Silicon avoids CPU-to-GPU copies via one unified memory pool (left), while Windows routes specialized sub-graphs across modular NPU, GPU, and discrete VRAM lanes (right).

Local Acceleration Runtimes & Execution Providers

Runtime / Provider	Target Silicon	Key Strength	When Power Users Choose It
Apple MLX (`mlx-lm`)	Apple Silicon M1–M5+ (Metal + ANE)	Zero-copy unified memory & native 4-bit quants	Running 7B–70B+ local LLMs, LoRA fine-tuning, and OpenAI-compatible `mlx_lm.server` on macOS.
Windows ML / ORT (QNN & OpenVINO)	Qualcomm, Intel & AMD NPUs (40–80 TOPS)	Ultra-low battery draw for continuous background AI	Always-on voice, live captions, local embeddings, and Phi Silica desktop apps unplugged from power.
ONNX DirectML / TensorRT for RTX EP	Any DirectX 12 GPU or NVIDIA RTX	Broad vendor GPU reach; TensorRT unlocks Tensor Cores	Cross-vendor Windows desktop apps (DirectML) or max-speed local inference on GeForce RTX cards.
Microsoft Foundry Local	Windows CPU, GPU & Copilot+ NPU	One-command model catalog & PowerToys integration	Pulling quantized Phi-4 Mini or Qwen ONNX models offline for Advanced Paste AI and local scripts.
WSL2 GPU Passthrough	Windows + NVIDIA CUDA (or AMD ROCm)	Near-native Linux kernel PyTorch & vLLM speed	Running Linux-first ML stacks (PyTorch, vLLM, ComfyUI, Docker GPU containers) on a Windows host.

```
# macOS (Apple Silicon): Launch an OpenAI-compatible Metal server on :8080 with 4-bit MLX weights
mlx_lm.server --model mlx-community/Qwen2.5-Coder-14B-Instruct-4bit --port 8080

# Windows 11: Run Foundry Local on NPU/GPU, or verify near-native CUDA inside WSL2 Ubuntu (host driver only)
foundry model run phi-4-mini
wsl -d Ubuntu-24.04 -- python3 -c "import torch; print(torch.cuda.is_available(), torch.cuda.get_device_name(0))"
```

5 Real-World Copy-Paste Cross-OS AI Recipes

Wire these five high-leverage recipes into Raycast AI Commands, macOS Shortcuts, PowerToys Advanced Paste, Click to Do, or Foundry Local so you execute identical daily cleanups and developer checks on both operating systems.

Recipe & Goal	macOS Execution (Shortcuts / Raycast / MLX)	Windows 11 Execution (PowerToys / Click to Do / Foundry)
1. Clipboard PII Scrub → JSON *Safe log sharing*	Save Raycast AI Command bound to `Cmd+Opt+J` targeting local `mlx_lm.server`; mask tokens/emails and copy valid JSON.	Open Advanced Paste (`Win+Shift+V`) → **Paste with AI** (backed by Foundry Local) with strict JSON redaction rule.
2. Screen Region OCR → Action List *Grab slide tables*	Trigger Raycast Screen Capture OCR or Shortcuts *Extract Text from Image* → **Use Model (On-Device)** → Markdown checklist.	Press `Win+Q` (Click to Do) over any video frame or dialog → copy OCR text → `Win+Shift+V` Paste as Markdown.
3. Offline PDF Digest & Tag *Zero cloud leak*	Finder Quick Action → Shortcuts *Get Text from PDF* → **Use Model (On-Device)** → save summary & 3 tags to Obsidian.	Pipe extracted PDF text via PowerShell to `foundry model run phi-4-mini` locally on NPU/GPU and emit YAML frontmatter.
4. Git Diff → Risk & Commit Check *Pre-PR sanity*	Run `git diff --staged | pbcopy` then fire Raycast Quick AI on local 14B MLX coder model for a 3-bullet regression scan.	In WSL2 Ubuntu or PowerShell: pipe `git diff --staged | clip.exe`, then run `Win+Shift+V` AI prompt or local Ollama/Foundry script.
5. Voice Braindump → Clean Spec *Hands-free capture*	Raycast Dictation or macOS Voice Memos transcript → Apple Intelligence Writing Tools *Key Points* or Custom Prompt.	`Win+H` Windows Voice Typing or Raycast v2.0 *Dictate to AI* → format structured user story + acceptance bullets.

```
# Portable Python Client: works identically against macOS mlx_lm.server (:8080) or Windows Foundry/Ollama
import os, sys, urllib.request, json

base = os.getenv("LOCAL_AI_URL", "http://127.0.0.1:8080/v1")
payload = {
    "model": os.getenv("LOCAL_AI_MODEL", "mlx-community/Qwen2.5-Coder-14B-Instruct-4bit"),
    "messages": [
        {"role": "system", "content": "Extract 3 concise action items with owners from the input."},
        {"role": "user", "content": sys.stdin.read()},
    ],
    "temperature": 0.1,
}
req = urllib.request.Request(f"{base}/chat/completions", data=json.dumps(payload).encode(),
                             headers={"Content-Type": "application/json"})
print(json.loads(urllib.request.urlopen(req).read())["choices"][0]["message"]["content"])
```

Standardize One Env Var Across Both OSes

Export `LOCAL_AI_URL` and `LOCAL_AI_MODEL` in both `~/.zshrc` and your PowerShell `$PROFILE`. Every helper script in your dotfiles then works transparently whether powered by Metal MLX or Windows Foundry/WSL2.

Never Store Cloud API Keys Inside Synced Quicklinks

If you sync Raycast Quicklinks or PowerToys scripts across machines via Git or cloud drive, read cloud credentials exclusively from macOS Keychain, Windows Credential Manager, or an untracked `.env` file.

Privacy Guardrails: PCC, Recall Encryption & Admin Policy

System-wide assistants touch your screen, clipboard, and local files—making encryption locks, exclusion lists, and enterprise MDM policies essential before enabling continuous OS intelligence on work machines.

1 Apple Intelligence & Private Cloud Compute

- **On-Device Priority:** Simple summarization and notification triage stay inside Apple Silicon RAM with zero network calls.
- **Verifiable PCC Offload:** Heavy requests route to hardened Apple Silicon cloud nodes running signed, publicly inspectable images with ephemeral memory and zero Apple access.
- **External Extension Gate:** ChatGPT or third-party integrations prompt explicitly per request unless pre-approved, and admins can lock them down via MDM.
- **Self-Audit Report:** Inspect `System Settings → Privacy & Security → Apple Intelligence Report` to export exact request telemetry.

2 Windows Recall Security & Exclusion Vault

- **Strict Opt-In Only:** Recall ships disabled until explicitly opted in during setup or Windows Settings—never silent capture by default.
- **BitLocker + Hello ESS Lock:** Snapshots live inside a Virtualization-based Security (VBS) enclave requiring **BitLocker** and biometric **Windows Hello ESS** liveness before viewing.
- **App & Site Exclusions:** Block password managers, HR portals, banking tabs, and Edge/Chrome InPrivate sessions; click the tray icon anytime to Pause or Wipe snapshots.
- **Zero Cloud Upload:** Recall snapshots stay strictly on your encrypted local SSD unless you manually share an item.

Cross-OS Security, Encryption & Admin Governance Matrix

Security Layer	macOS / Enterprise MDM Control	Windows 11 / Intune & Group Policy Control
Disk & Enclave Encryption	FileVault full-disk encryption + Apple Secure Enclave biometric Touch ID / Watch unlock.	BitLocker / Device Encryption + TPM 2.0 + Windows Hello Enhanced Sign-in Security (ESS) VBS enclave.
Screen Capture Consent	Explicit TCC `Screen & System Audio Recording` prompt per app; no silent system-wide capture.	Recall tray indicator always visible when active; `AllowRecallEnablement` CSP disables or removes feature.
Snapshot / Log Blocking	Per-app TCC toggles + disable Apple Intelligence Writing Tools / External Extensions in MDM profile.	Intune OMA-URI `./Device/Vendor/MSFT/Policy/Config/WindowsAI/DisableAIDataAnalysis` stops all snapshots.
Sensitive App Filtering	Add password managers and SSH terminals to Raycast *Ignore Applications* & Clipboard exclusions.	Add 1Password, KeePass, VDI clients, and banking URLs to Recall exclusions + enable sensitive info filter.
Air-Gapped Local Mode	Run `mlx_lm.server` bound strictly to `127.0.0.1` with outbound firewall block if needed.	Run Foundry Local or Ollama offline; verify Windows Firewall blocks local loopback ports externally.

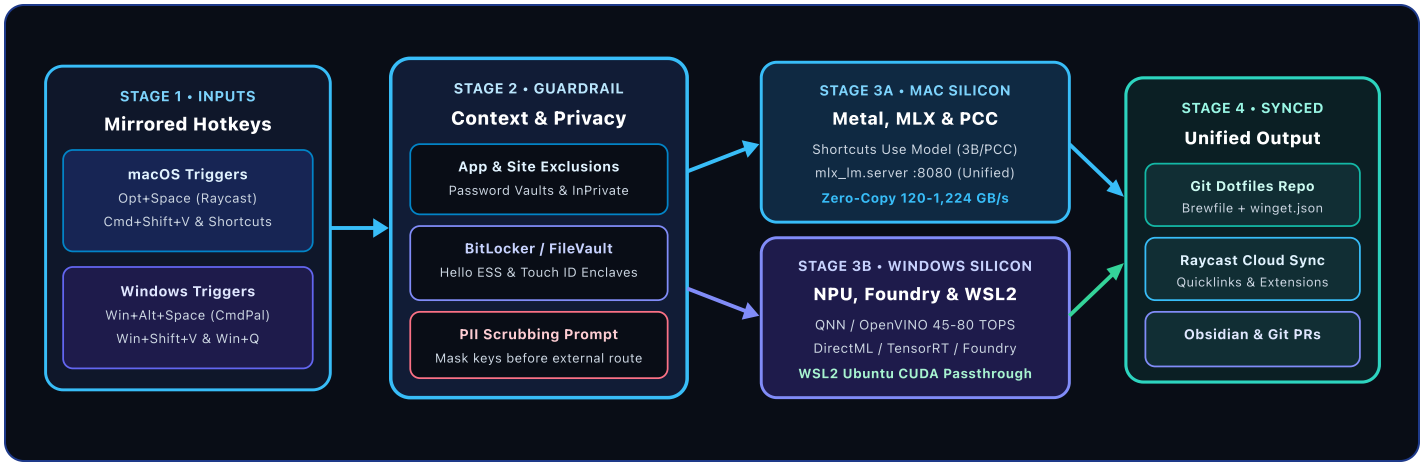
```
# Quick Encryption & Policy Audit (Run on every client laptop before enabling OS AI features)
# macOS:                                          # Windows 11 PowerShell (Admin):
fdesetup status                                   manage-bde -status C:
spctl --status                                    Get-ItemProperty
"HKLM:\SOFTWARE\Policies\Microsoft\Windows\WindowsAI" -ErrorAction SilentlyContinue
```

Consultant & Regulated Work Watchout: Protect Clipboard History & Screen Memory

Clipboard managers (Raycast, Alfred, PowerToys Advanced Paste) and Recall can accidentally cache copied production credentials or patient/financial records. Configure automatic clipboard exclusion for password managers on both OSes, and verify Recall is paused or disabled by policy on client VDI machines.

Cross-Tool Synergy: One Brain Across Both Operating Systems

Whether you carry an Apple Silicon laptop on the road and sit down at a Windows 11 RTX/Copilot+ tower at home, unifying hotkeys, Git-managed dotfiles, and standard `localhost` AI endpoints turns two distinct machines into one seamless workspace.



Four Habits That Keep Dual-OS Power Users Fast

1 Unify Your Launcher via Raycast Cloud

With Raycast v2.0 available on both macOS and Windows 11, enable encrypted Cloud Sync so your custom Quicklinks, snippets, and AI Commands appear automatically on every machine you touch.

2 Keep OS-Native Superpowers Separate

Don't force Mac patterns onto Windows or vice versa: embrace Shortcuts "Use Model" + Metal MLX on Mac, and pair PowerToys Advanced Paste + Click to Do + WSL2 CUDA on Windows.

3 Standardize Localhost Ports & CLI Wrappers

Point editors, Obsidian plugins, and terminal helpers at an OpenAI-compatible `LOCAL_AI_URL` so switching from `mlx_lm.server` to Foundry Local or WSL2 vLLM requires zero code changes.

4 Track Provisioning in One Git Repo

Commit your `Brewfile`, `winget.json`, `.zshrc`, and PowerShell `$PROFILE` side by side so spinning up a fresh Mac or Copilot+ laptop takes one reproducible command.

Close the Cross-OS Loop: One Prompt Library, Any Silicon Backend

Treat your prompt recipes, redaction rules, and dotfiles as portable infrastructure. Whether inference lands on an M5 Max unified memory pool, a 48 TOPS Intel NPU, or a WSL2 CUDA card, your workflows stay sharp and repeatable.

Troubleshooting Clinic & Cross-OS Pre-Flight Checklist

Use this field clinic whenever local models swap-thrash on macOS, Recall refuses activation on Windows, hotkeys collide, or WSL2 PyTorch fails to detect your GPU—then lock in your setup with the 6-point pre-flight check.

Symptom / Issue	Root Cause	10-Second Fix
1. Mac local `mlx_lm` crawls or triggers heavy SSD swap	Model weights + KV cache exceed ~75% of unified memory, starving macOS WindowServer.	Check Activity Monitor memory pressure, drop from 8-bit to 4-bit `mlx-community` weights, or cap context length.
2. Apple Intelligence or Shortcuts "Use Model" missing	Older macOS release, unsupported Intel chip, or mismatched System vs. Siri language.	Confirm Apple Silicon (M1+), update to macOS Sequoia/Tahoe, align System & Siri region/language, and verify MDM allows it.
3. Windows Recall toggle grayed out or blocked	Missing 40+ TOPS NPU, BitLocker off, Hello ESS disabled, or Intune `DisableAIDataAnalysis`.	Verify Copilot+ 24H2+ PC, run `manage-bde -status` to enable BitLocker, enroll Windows Hello ESS biometrics, or check GPO.
4. `Win + Shift + V` opens notifications instead of Advanced Paste	Windows system shortcut conflict steals the notification-focus chord before PowerToys hooks it.	Open PowerToys Settings → Advanced Paste, verify activation hotkey or remap cleanly to `Ctrl + Alt + V`.
5. DirectML / ONNX model silently falls back to slow CPU	Missing vendor Execution Provider (QNN/OpenVINO/TensorRT) or unsupported FP16 graph op.	Update NPU/GPU drivers via Windows Update, install the matching ORT EP package, or pull tested graphs via `foundry model`.
6. WSL2 PyTorch reports `cuda.is_available() == False`	Someone installed a Linux NVIDIA driver inside WSL2, overwriting Microsoft's `DXGKRNL` stub.	Remove Linux NVIDIA driver packages inside Ubuntu, run `wsl --update`, and install the latest Windows-host NVIDIA driver only.
7. `Opt+Space` types non-breaking space or `Win+Alt+Space` hits Game Bar	macOS Option-key character input or Windows Xbox Game Bar intercepts launcher chord.	Disable Spotlight shortcut on Mac (rebind Raycast to `Opt+Space`) and turn off conflicting Game Bar shortcuts on Windows.

6-Point Cross-OS AI Power-User Pre-Flight Checklist

- [x] **1. Lock Encryption & Biometrics:** Confirm FileVault + Touch ID on Mac and BitLocker + Windows Hello ESS on Windows 11.
- [x] **2. Audit Screen & App Exclusions:** Add password vaults, banking URLs, and VDI apps to Raycast clipboard ignore & Recall exclusions.
- [x] **3. Mirror Launcher & Paste Chords:** Align Raycast / Command Palette and Advanced Paste hotkeys so muscle memory holds across desks.
- [x] **4. Size Models to Bandwidth:** Fit 4-bit MLX inside 70% of Mac unified RAM; keep always-on Copilot+ screen ops on the 40+ TOPS NPU.
- [x] **5. Verify Local & WSL2 Endpoints:** Test `mlx_lm.server` on macOS and confirm `foundry` + WSL2 `torch.cuda.is_available()` on Windows.
- [x] **6. Back Up Declarative Manifests:** Commit `Brewfile`, `winget.json`, Shortcuts, and Raycast Cloud exports (excluding API secrets).

Final Takeaway: Master the Tradeoffs, Own Both Platforms

Apple Silicon wins on quiet unified-memory capacity and verifiable Private Cloud Compute; Windows 11 Copilot+ wins on modular 40+ TOPS NPU features, Click to Do, PowerToys speed, and raw WSL2 CUDA versatility. With mirrored launchers and disciplined privacy locks, you get the best of both worlds.