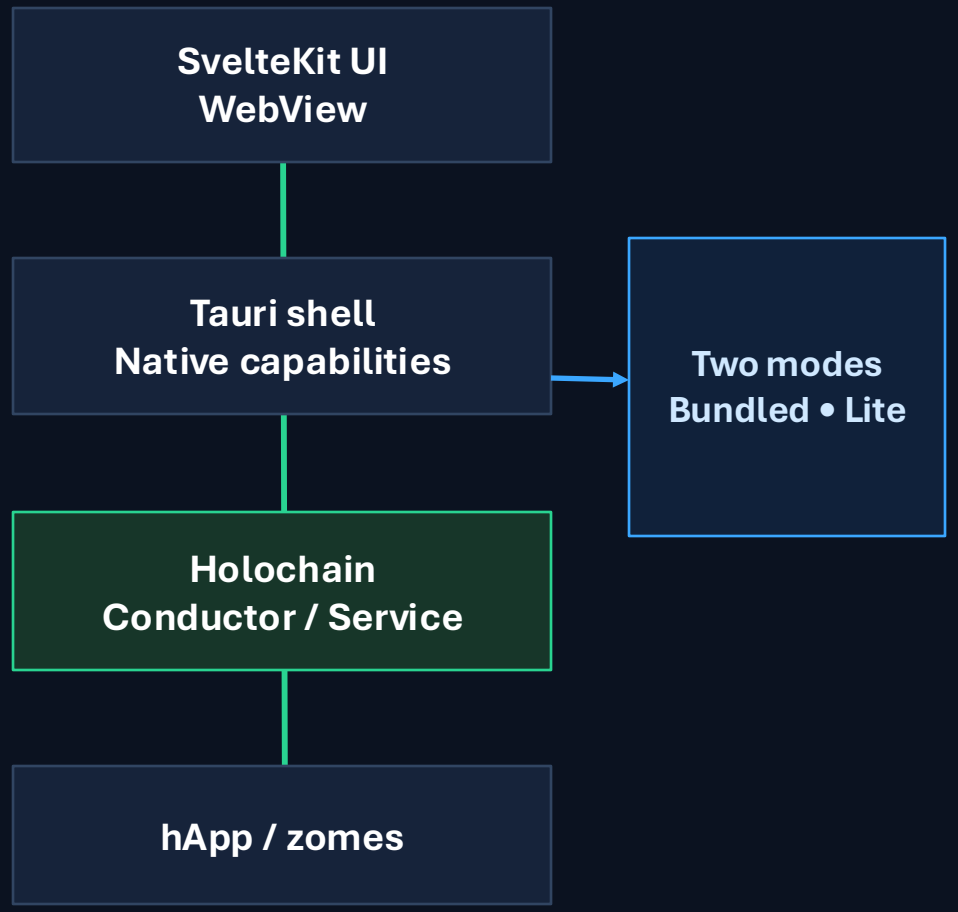
Same Holochain idea, Different app shells.

Volla Messages Tauri Architecture

How the Tauri/Web app runs Holochain in two modes: bundled conductor and lite service runtime.



Tauri in plain words

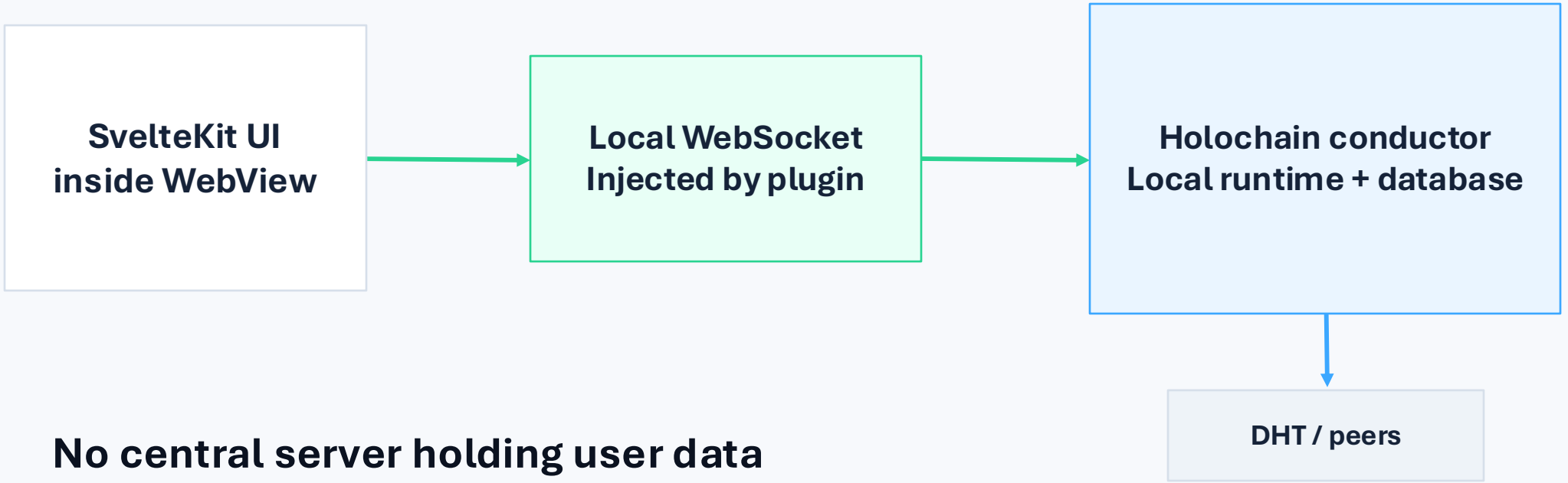
Think of Tauri as the native frame around a web app.

Without Tauri it behaves like a website.
With Tauri it becomes an installable app with native capabilities.

For Volla Messages, the web UI stays familiar while Tauri handles the native shell and Holochain integration.

Where Holochain fits

The UI talks to a local conductor — not to a central backend.

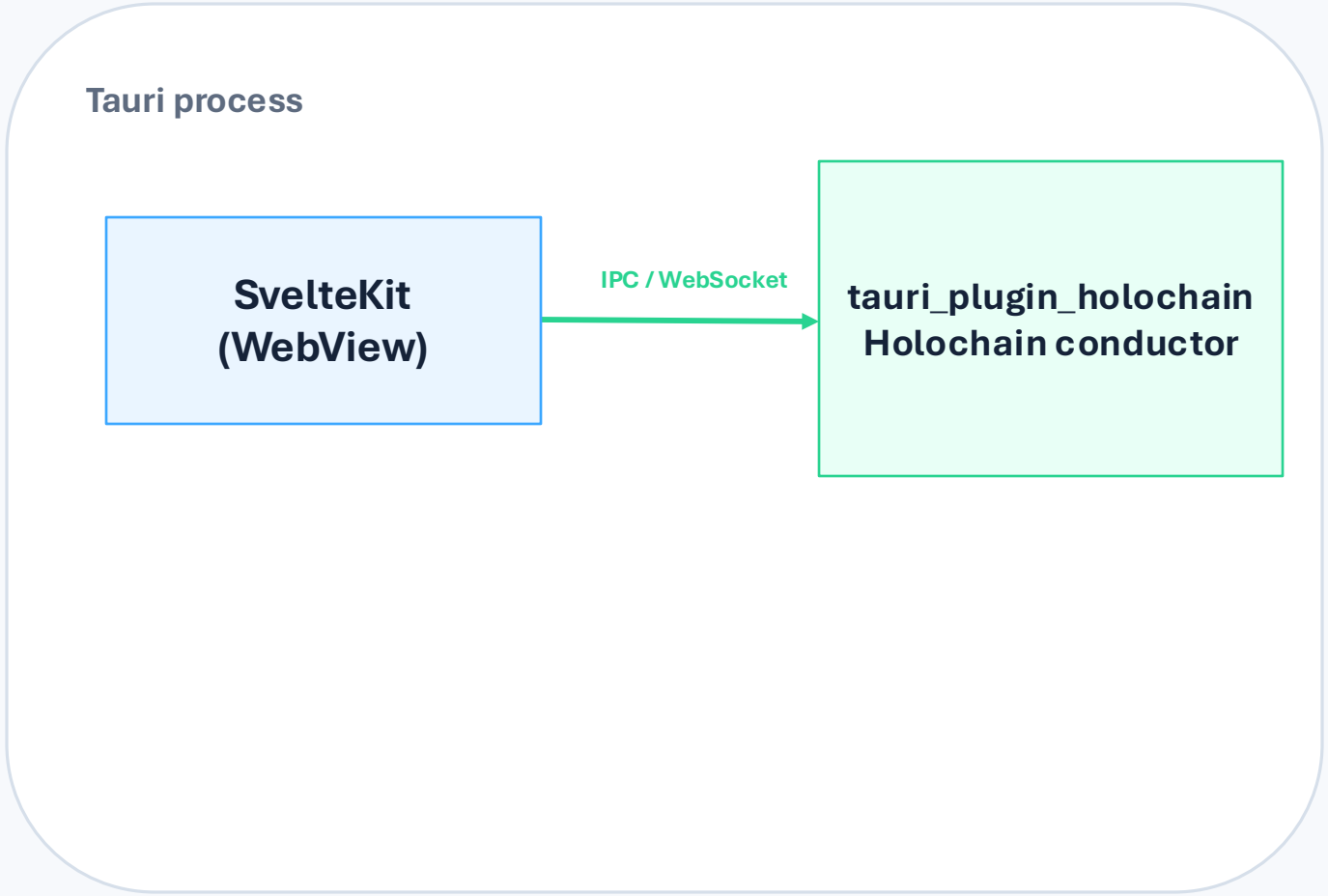


No central server holding user data

Messages, keys and app state live locally first
Holochain synchronizes through peer-to-peer networking
The frontend uses normal async calls such as callZome()

Bundled build — the app is the conductor

A self-contained app: Tauri starts a full Holochain conductor inside its own process.



Rich / bundled

Starts its own conductor

Uses configured storage path

Owns network config

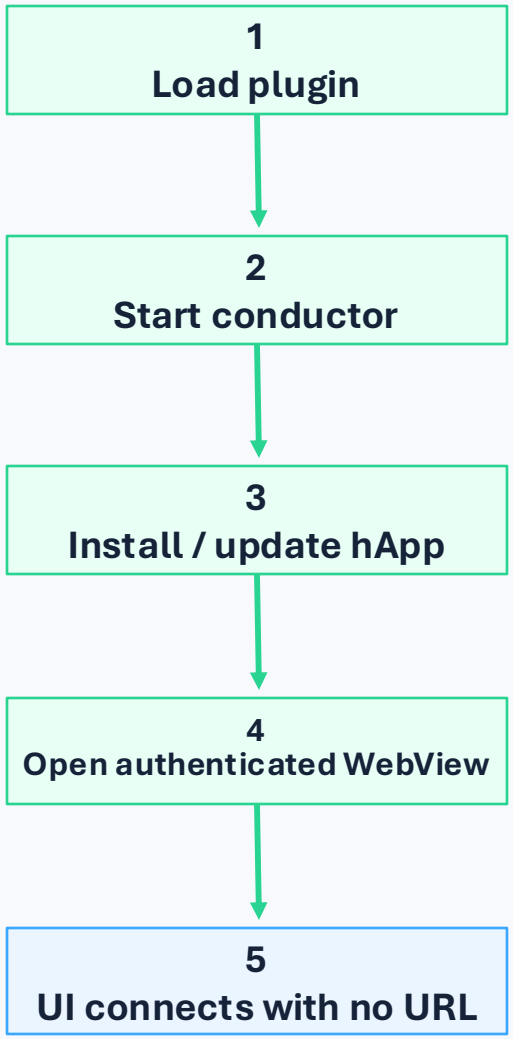
Injects the right app socket into the UI

Outbound network

wss://relay2.volla.tech
https://iroh-relay.volla.tech

Bundled startup sequence

What happens before the UI is ready.



The UI never knows the conductor URL.

holochain_bundled.rs loads tauri_plugin_holochain

The plugin starts the conductor with storage and network config

It emits holochain://setup-completed when ready

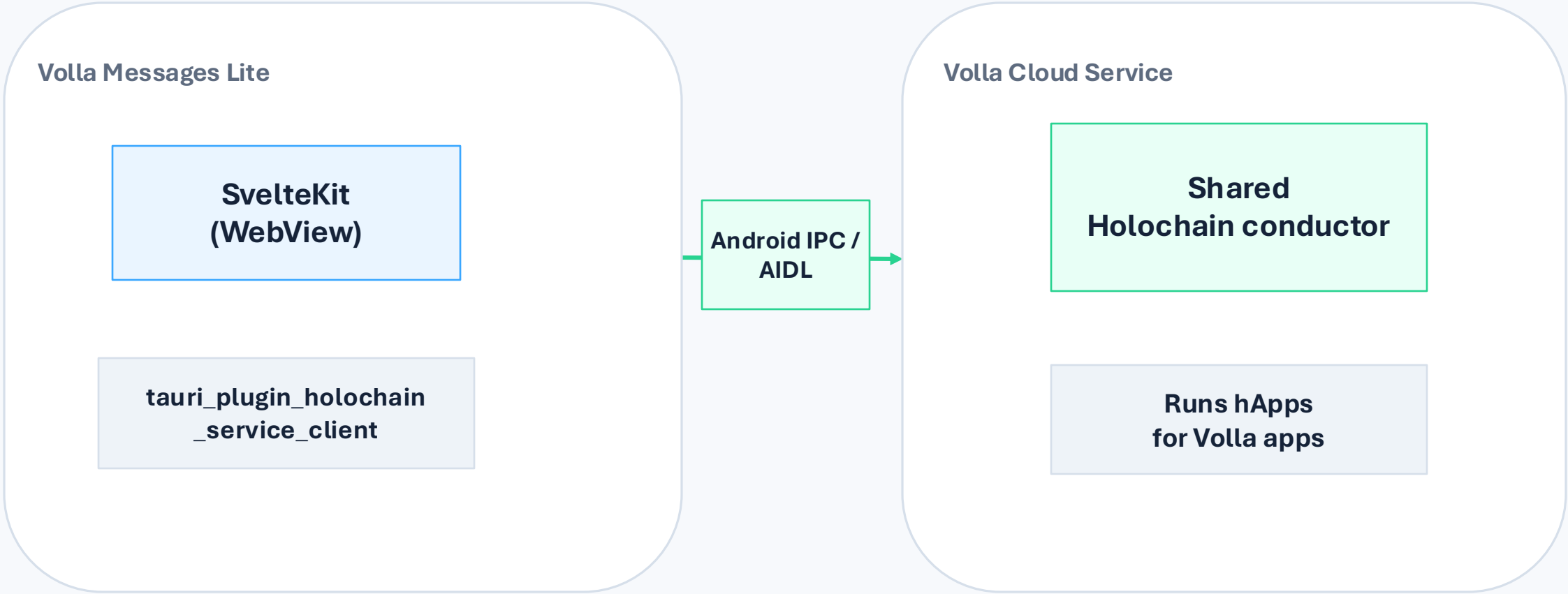
The app opens AdminWebsocket to install or update the hApp

The created WebView is already authenticated for AppWebsocket.connect()

Takeaway
The plugin hides conductor setup from the SvelteKit UI.

Lite build — app is a client

The conductor is external: Volla Cloud Service, always on.



Service mode: same UI, different Rust side

From the frontend perspective, both modes look almost identical.

holochain_service.rs loads the service-client plugin instead of the bundled conductor plugin.

Same frontend API
AppWebsocket.connect() • callZome() • signals

The difference is entirely below the UI: bundled mode starts a conductor; lite mode asks the Volla Cloud Service to run it.

Key difference in one slide

Bundled is self-contained. Lite is shared and OS-native.

Dimension	Bundled build	Lite / service build
Who runs conductor	The Tauri app itself	VCS always-on service
Network config	Set in holochain_bundled.rs	Owned by the service
Startup time	Slower — conductor boots	Faster - service already running
Battery / resources	Higher — in-process conductor	Lower — shared across apps
Code size	More setup / splashscreen logic	Much smaller; conductor is service problem

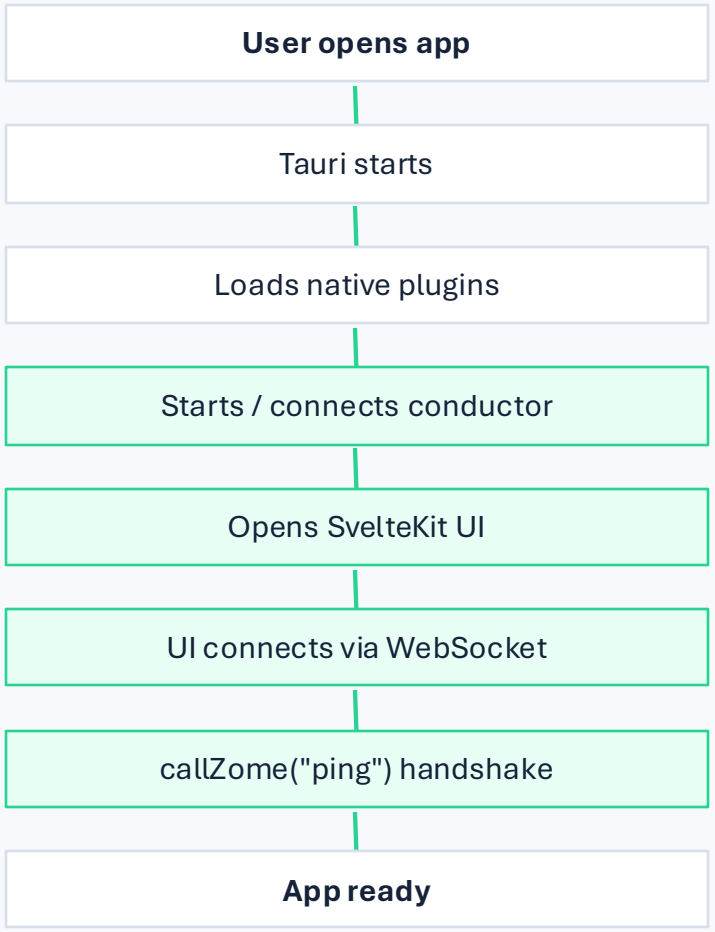
“VCS makes the lite Tauri app lighter: the app knocks on the service door and asks it to run the hApp.”

Repo walkthrough

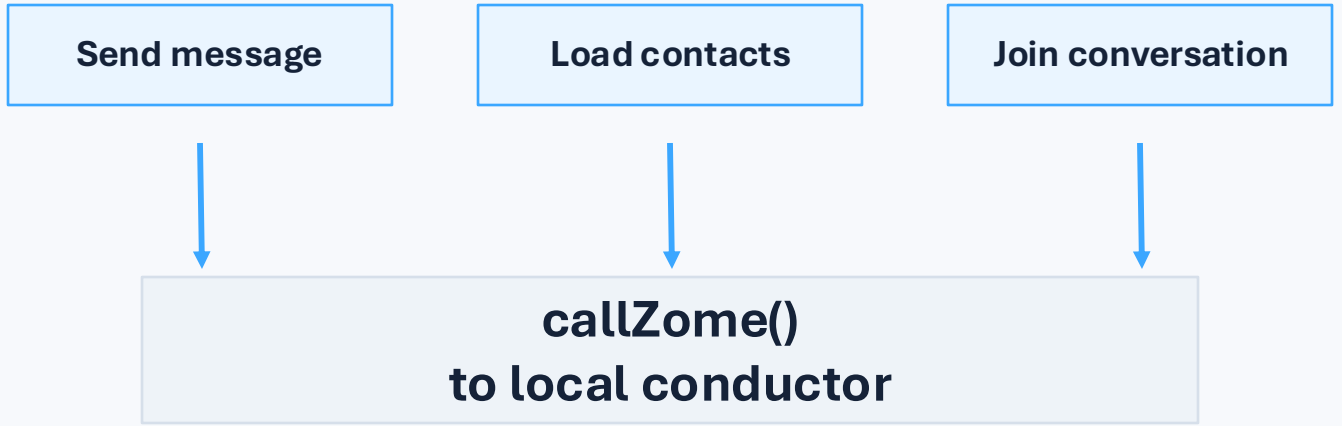
File	What it does
`src-tauri/src/lib.rs`	Lists native capabilities: camera, notifications, clipboard, etc.
`src-tauri/src/builder/holochain_bundled.rs`	Bundled mode: starts the conductor inside the app and wires relay/bootstrap config.
`src-tauri/src/builder/holochain_service.rs`	Lite mode: delegates setup to the Volla Cloud Service runtime.
`src-tauri/src/config.rs`	App ID and DNA are compiled into the binary.
`ui/src/routes/+layout.svelte`	UI handshake: connects to Holochain and loads app state.

The one startup diagram to remember:

After startup, most user actions become zome calls to the local conductor.



After the app is ready:



Thank You!