TAHAI Web Services Browser

Mission Tabs, Mission Control, IT Docs / PSA Boundary, and Security Guardrails

Canonical project-source specification for multi-chat, multi-pass development

Version: 1.0 - Prepared: April 26, 2026

Purpose

This document is intended to be uploaded into project sources and treated as the hard guardrail source for future TAHAI Web Services Browser passes. It defines what must be built, what must not be built, and how security, UX, repo hygiene, verification, and integration boundaries must be enforced without drift.

This document is intentionally strict. It uses MUST, MUST NOT, SHOULD, and MAY in the ordinary RFC sense. If a future chat or pass conflicts with this document, this document wins unless Justin explicitly supersedes it in a newer project-source document.

Table of Contents

1. Non-negotiable source of truth

This section is the shortest possible hardset summary to paste into any new chat before coding begins. It is not optional.

```
PROJECT: TAHAI Web Services Browser
CANONICAL LOCAL PATH: C:\dev\browser\app
CANONICAL PUBLIC REPO: https://github.com/JTAHAI/tahai-web-services-browser
CURRENT SOURCE LANE: 1.8.x public open-source browser
LICENSE / ATTRIBUTION: Apache-2.0, NOTICE, TRADEMARKS.md remain in place

BUILD GOAL: Build a clean Chromium-compatible IT/DevOps command browser.
UNIQUE SYSTEM: TAHAI Mission Control = Mission Tabs + Mission Views + Mission Tools + Mission Evidence.

BROWSER PROJECT SCOPE ONLY:
- Implement browser-side UX, state, local persistence, validated links, export, redaction, and IT Docs /
PSA reference contracts.
- Do not implement IT Docs backend features in this repo.
- Do not implement PSA connectors in this repo.
- Do not store PSA/API/provider secrets in this repo or in Mission JSON.
- Route future PSA writeback through IT Docs-authorized server-side connectors only.

SECURITY RULE:
Open-source means attackers can read the code. Defense must not rely on obscurity.
Never trust renderer input. Never trust local mission files. Never trust remote page content. Never store
integration secrets in the browser.
```

1.1 Existing handoff constraints preserved

Constraint	Hard rule
Real source only	Fix actual Electron source, renderer templates, CSS, package resources, and installer scripts. No blind runtime DOM hacks.
No generated artifacts	Do not commit generated installers, release zips, dist, node_modules, runtime profiles, caches, secrets, certs, .env files, or local data.
Public repo materials	Keep Apache-2.0 LICENSE, NOTICE, TRADEMARKS.md, SECURITY.md, docs/code-signing-policy.md, and docs/privacy-policy.md in place.
Verification before artifacts	Run source verification before returning deltas or installers. Be explicit when Windows-only packaging/signing cannot be verified.
Navigation parity remains required	Back, forward, reload, address bar, menu, shortcuts, mouse buttons, and future panes must route to the active tab or active pane.

1.2 Canonical product sentence

Preserve this positioning

Chrome is a browser. Edge is a productivity browser. TAHAI is an IT/DevOps command browser. Normal mode stays clean. Ops Mode becomes Mission Control.

2. Product thesis and scope boundaries

TAHAI Web Services Browser should one-up Chrome and Edge by making common browser amenities operationally aware for IT engineers, DevOps teams, builders, support desks, incident responders, and documentation-heavy technical operators.

2.1 What this browser is

- A Chromium-compatible browser shell focused on IT, DevOps, cloud, provider-console, documentation, and BYOK workflows.
- A clean browser in normal mode, with an optional Ops Mode for Mission Control.
- A workspace browser that remembers operational context, not just URLs.
- A local-first tool that may link to IT Docs and PSA records only through explicit, authorized references.
- A public open-source project that must be safe to inspect, fork, audit, and build without leaking secrets.

2.2 What this browser is not

- Not a full IT Docs backend.
- Not a PSA connector host.
- Not a secret vault.
- Not a cloud-provider credential manager.
- Not a replacement for Chromium DevTools.
- Not a hidden automation platform that clicks admin consoles without explicit user intent.
- Not a place to store bearer tokens, OAuth refresh tokens, PSA API keys, or customer secrets.

2.3 Three layers of responsibility

Layer	Responsible for	Forbidden from doing
Browser	Mission tabs, panes, UX, local state, local evidence metadata, redaction, export, validated deep links, IT Docs/PSA references.	Storing secrets, making direct PSA API calls, bypassing IT Docs authorization, scraping tokens from pages.
IT Docs	Authentication, org authorization, projects, runbooks, evidence storage, audit log, server-side PSA connector authority.	Trusting browser-provided org/project/ticket IDs without server-side checks.
PSA	Ticket/project/client authority through vendor API and org-approved integration.	Being called directly from the open-source browser.

3. Hard architecture boundaries

Architectural sentence

Mission Tabs are local browser-side operational context. IT Docs is the trusted documentation and authorization layer. PSA systems are downstream integrations accessed only through IT Docs-authorized server-side connectors.

3.1 Allowed architecture

```
Browser Mission Control
  -> Local mission model and local persistence
  -> IT Docs authentication/status API
  -> IT Docs mission/reference/evidence API
      -> IT Docs server-side authorization
      -> Optional IT Docs PSA connector
          -> PSA ticket/project/client API
```

3.2 Forbidden architecture

```
Browser -> PSA API directly
Browser stores PSA token
Browser stores OAuth refresh token
Browser stores provider cloud token
Browser stores raw cookie/auth header from a webview
```

```
Browser exposes generic shell/file/eval IPC to renderer
Browser lets untrusted page content call Mission APIs
```

3.3 Trust zones

Zone	Trust level	Rules
Main process	Privileged	Smallest possible attack surface. Owns filesystem writes, validated IPC, app-command handling, external-open wrapper, and mission persistence.
Preload	Privileged bridge	Expose minimal typed APIs through contextBridge. No raw ipcRenderer exposure. No direct Node APIs to renderer.
Renderer shell	Semi-trusted app UI	Can request mission actions only through allowlisted APIs. Must validate again in main. Must not receive secrets.
Webviews / remote pages	Untrusted	No Mission APIs. No Node integration. No direct filesystem. No privileged IPC. No access to mission JSON or exports.
Mission files	Untrusted local input	Validate schema, size, URL protocols, enums, and IDs before use. Never execute content from mission files.
IT Docs API	Trusted only after HTTPS + auth + server authorization	Every org/project/runbook/evidence/ticket reference requires server-side authorization.

4. Mission Tabs canonical model

Mission Tabs are not ordinary tab groups. They are named operational workspaces with role-aware tabs, pane layouts, notes, evidence metadata, and optional external references.

4.1 Required mission properties

Property	Required?	Description	Security rule
schemaVersion	Yes	Integer schema version, starting at 1.	Reject unknown versions unless a safe migrator exists.
missionId	Yes	UUID generated locally.	Reject non-UUID values. Never trust as authorization.
name	Yes	Human label such as Cloudflare DNS Migration.	Length limit. Escape on render. No HTML injection.
missionType	Yes	Enum: deployment, incident, support, documentation, migration, audit, admin, generic.	Unknown types map to generic or are rejected.
mode	Yes	local, itdocs-linked, or psa-linked.	Mode is display state only; server still authorizes actions.
tabs	Yes	Array of mission tab references.	Validate URL, title length, role enum, active flag.
layout	Yes	Mission Control layout and pane assignments.	Validate layout enum and pane IDs.
notes	Optional	Local operator notes.	Redaction scanner runs before export/sync.
timeline	Optional	Local mission timeline events.	No auth headers, cookies, tokens, or secrets.
links.itDocs	Optional	Opaque references to IT Docs org/project/runbook/evidence objects.	References only. No IT Docs tokens.
links.psa	Optional	Opaque references to PSA ticket/project/client display metadata.	References/deep links only. No PSA tokens.

4.2 Mission tab roles

Role	Meaning	Examples
primary-console	The main admin/work surface.	AWS Console, Microsoft 365 Admin, Cloudflare dashboard.
logs	Deployment, application, cloud, or system logs.	GitHub Actions, CloudWatch, Vercel logs, status output.
docs	Reference documentation.	Vendor docs, internal docs, API docs.
runbook	Step-by-step operational procedure.	IT Docs runbook, local markdown, deployment checklist.
ticket	Ticket or work-order context.	PSA ticket deep link, help desk case.
monitoring	Observability/status dashboard.	Status page, uptime monitor, metrics dashboard.
evidence	Proof/capture area.	Evidence pack panel, captured screenshots, headers/cert summaries.
live-target	The live system being validated.	Production site, staging URL, API endpoint.
vendor-portal	Third-party provider surface.	Registrar, firewall vendor, domain provider.
tool	Local OpsTool surface.	DNS lookup, TLS inspector, JWT decoder.

4.3 Mission lifecycle

```
Create local mission
  -> Add tabs / assign roles / optionally choose layout
  -> Save local mission
  -> Optional: sign in to IT Docs
      -> Link org/project/runbook after server authorization
      -> Optional: link PSA ticket only if IT Docs says org integration is active
  -> Capture notes/evidence locally
  -> Preview redaction
  -> Export local packet or explicitly sync approved items to IT Docs
  -> Optional: request IT Docs server-side PSA writeback
  -> Close/archive mission
```

4.4 Mission states

State	UI label	Allowed actions	Disallowed actions
local-only	Local Only	Create/rename/save tabs/layout/notes/evidence; export local packet.	IT Docs writeback; PSA link; PSA writeback.
signed-in	Signed In	List authorized orgs/projects/runbooks through IT Docs API.	Assume access from stale local IDs.
org-selected	Org Selected	Attach mission to authorized org; query capabilities.	PSA actions unless capability says enabled.
itdocs-linked	IT Docs Linked	Save mission reference; append approved notes/evidence to IT Docs.	Direct PSA API calls.
psa-available	PSA Available	Show link-ticket UI if IT Docs grants capability.	Store PSA credentials.
psa-linked	PSA Linked	Store display metadata and validated deep link; request IT Docs server writeback.	Automatic ticket spam or direct browser-to-PSA write.
session-expired	Session Expired	Keep local state; prompt re-auth.	Writeback; silent sync.
permission-denied	Permission Denied	Show clear message; keep local-only data.	Retry loops; leaking object IDs in errors.

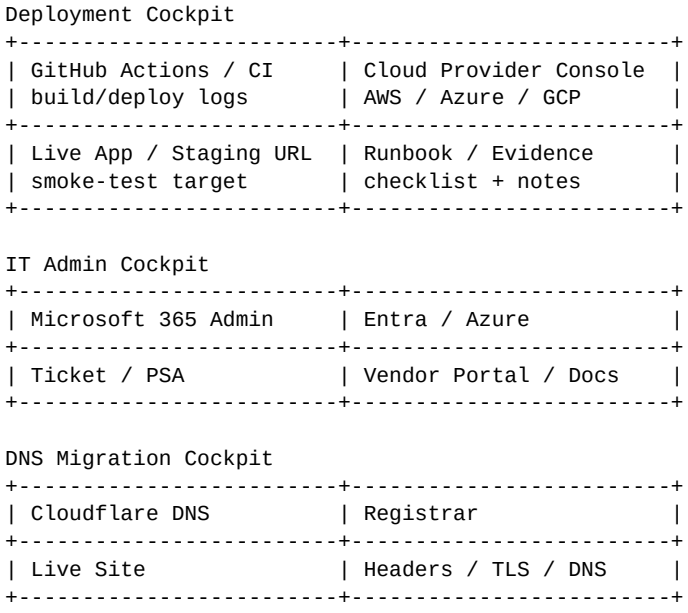
5. Mission Control multi-view UX

Mission Control is the browser-side work surface that lets large monitors, gaming displays, TV-style NOC displays, and support desks use more than a single page or a two-pane split. It must stay clean when off.

5.1 Required view modes

Mode	Purpose	Guardrails
1-Up	Normal browser behavior.	Default. No clutter. All ordinary browser controls behave normally.
2-Up	Split view for console + docs, logs + live target, or ticket + admin console.	Back/forward/reload/address bar route to active pane.
3-Up	Operational triad, such as console + logs + runbook.	No awkward hidden panes; use responsive layout.
4-Up	Signature Quad Ops View for deployment, incident, DNS migration, and IT admin cockpits.	Active pane must be clear. Pane controls must not obscure content.
Focus Pane	Temporarily maximize one pane without losing layout.	Restore must return exactly to prior layout.
Command View	Keyboard-first operation through Ctrl+K.	Commands must indicate target mission/pane before destructive actions.

5.2 Signature Quad layouts



5.3 Active pane routing rules

- Back, forward, reload, stop, address-bar navigation, and command-palette navigation MUST target the active pane when in multi-view mode.
- The active pane MUST have a visible but subtle focus marker.
- Mouse Back/Forward buttons MUST apply to the active pane or active tab, not only the shell window.
- Alt+Left and Alt+Right MUST remain compatible with the active pane/tab model.
- If the active pane cannot go back/forward, the action MUST safely no-op without error noise.
- Opening a new URL from the address bar MUST not unexpectedly replace another pane.

5.4 Keyboard shortcuts

Shortcut	Action	Requirement

Ctrl+K	Open Operator Command Center.	Always available. Commands are mission-aware.
Ctrl+Alt+1..4	Focus pane 1 through 4.	No effect outside Mission Control if pane does not exist.
Ctrl+Alt+Q	Toggle Quad View.	Must preserve current mission context.
Ctrl+Alt+S	Toggle Split View.	Must preserve current mission context.
Ctrl+Alt+F	Focus/maximize active pane.	Must restore previous layout on second press.
Ctrl+Alt+C	Capture active pane.	Must run redaction awareness before export/sync.
Ctrl+Alt+E	Export mission packet.	Must preview and warn on secrets.
Ctrl+Alt+R	Restore mission layout.	Must not open unsafe URLs without validation.

6. One-up Chrome and Edge feature map

Every parity feature must be upgraded into an IT/DevOps mission-aware feature. The goal is not more clutter. The goal is more context.

Browser amenity	Chrome/Edge style baseline	TAHAI one-up implementation
Tabs / groups	Group related pages.	Mission Tabs with roles, mission type, timeline, layout, notes, and optional IT Docs/PSA references.
Split view	Two pages side by side.	Mission Control 1-Up, 2-Up, 3-Up, 4-Up, Focus Pane, and NOC/war-room layouts.
Collections	Saved pages and research.	Evidence Packs with URL/title/timestamp, screenshots, notes, headers/cert/DNS summaries, and export/redaction.
Command palette / shortcuts	Open commands and pages.	Operator Command Center: start mission, launch cockpit, check DNS/TLS, capture panes, export handoff.
DevTools	Developer inspection.	Keep Chromium DevTools; add OpsTools for IT/DevOps workflows without polluting DevTools.
Profiles	Personal/work browser profiles.	Admin Console Profiles: AWS, Azure/Entra, M365, Google Workspace, Cloudflare, GitHub, Vercel/Firebase, firewall/VPN.
History	Visited pages.	Mission Timeline: operational events, evidence captures, layout changes, exports, explicit syncs.
Side panel	Bookmarks/history/reading list.	Runbook Rail: checklist, notes, evidence, commands, validation steps, rollback plan, export.
Screenshots	Capture a page or region.	Operational Capture: active pane, all panes, workspace packet, metadata, notes, redaction.
Session restore	Restore previous tabs.	Restore Mission: panes, tabs, roles, notes, evidence, checklist state, profile, last focus.
Bookmarks	Folders of links.	Launch Recipes: open consoles/docs/tools/checklists into the right mission layout.
Downloads	List downloaded files.	Artifact Shelf: configs, exported runbooks, logs, installers, screenshots, evidence packs, with risk warnings.
Security warning	Standard browser warnings.	Operator Safety: secret detection, redaction warnings, mixed production/staging hints, untrusted link guards.
New tab page	Search/news/recent sites.	Ops Launchpad: recent missions, DevOps workspace, IT admin, incident response, documentation sprint, provider consoles.

6.1 Operator Command Center command taxonomy

Command family	Examples	Guardrails
Mission	Start mission, restore mission, archive	No destructive delete without

	mission, rename mission, change mission type.	confirmation.
View	Switch 1-Up/2-Up/3-Up/4-Up, focus pane, send tab to pane, return pane to tabs.	Never lose unsaved mission state.
Capture	Capture active pane, capture all panes, add note, create evidence entry.	Redaction scan before export/sync.
OpsTools	DNS, TLS, headers, redirects, JWT, JSON, YAML, CIDR, curl builder.	Local-first where possible; no secret logging.
IT Docs	Link org/project/runbook, append note, create evidence metadata.	Only when signed in and authorized by IT Docs.
PSA	Open ticket, link ticket, append mission summary through IT Docs.	No direct PSA API calls or stored PSA secrets.
Export	Markdown, HTML, evidence packet, sanitized handoff.	Default to preview and redaction.

7. IT Docs integration contract - browser side only

Boundary

The browser may link a Mission to IT Docs objects only when the user is signed into an active account and IT Docs authorizes the selected org/project/runbook/evidence object. This repo implements only the browser-side client contract and disabled/placeholder UI states until the server API exists.

7.1 Browser may store IT Docs references only

Allowed local field	Meaning	Security rule
itDocs.orgId	Opaque org identifier returned by IT Docs.	Never treat as authorization. Server validates every request.
itDocs.orgName	Sanitized display name.	Escape on render; do not use as key.
itDocs.projectId	Opaque project ID.	Server-side object authorization required.
itDocs.runbookId	Opaque runbook/document ID.	Server-side object authorization required.
itDocs.evidencePackId	Opaque evidence/reference ID.	Created only by IT Docs API.
itDocs.deepLink	Validated HTTPS link to IT Docs page.	Must be allowlisted to configured IT Docs origins.

7.2 Browser must not store IT Docs secrets

- No raw IT Docs access tokens in Mission JSON.
- No refresh tokens in Mission JSON.
- No copied Cookie headers.
- No Authorization headers.
- No Cognito/OAuth secrets.
- No org secrets, invite secrets, or billing/Stripe metadata beyond safe display state.

7.3 IT Docs capability query

```
Browser asks IT Docs:
GET /api/browser/mission-capabilities

Expected browser interpretation:
- signedIn: boolean
- activeOrgs: array of safe display refs
- canCreateMissionReference: boolean
- canAppendEvidence: boolean
- canAppendRunbookNote: boolean
- psaProvidersAvailable: array of provider capability refs
```

```
Important:
The browser displays capabilities. It does not grant capabilities.
Every write still requires server-side authorization.
```

7.4 IT Docs UI states

State	Browser copy	Behavior
Not signed in	Sign in to TAHAI IT Docs to link this mission.	Local-only mission continues working.
Signed in, no org	Select an IT Docs org to link this mission.	No writeback until org selected and authorized.
Signed in, org denied	Your account cannot link this mission to that org.	Keep local mission. Do not retry noisily.
Linked	Mission linked to IT Docs.	Allow explicit sync/capture buttons.
Offline	Offline. Local mission changes are saved locally.	Queue nothing by default unless explicit offline queue exists later.
Session expired	Session expired. Sign in again to sync.	Disable writeback until re-auth.

8. PSA integration contract - browser side only

PSA integration must be designed as a browser-side reference model and UI contract only. Actual PSA credentials, tokens, API calls, object authorization, and writeback live server-side inside IT Docs connectors later.

8.1 PSA allowed references

Field	Allowed value	Hard rule
psa.provider	autotask, connectwise, halo, syncro, zendesk, freshservice, generic, etc.	Enum or server-provided safe value.
psa.ticketId	Opaque ID.	Do not infer tenant or authorization.
psa.ticketDisplayKey	Safe display string such as CHG-2041.	Sanitize and length limit.
psa.ticketTitle	Sanitized title.	Escape on render and export.
psa.ticketDeepLink	Validated HTTPS URL.	Must pass allowlist and URL parser.
psa.status	Safe display status.	Display only. Server authority wins.

8.2 PSA forbidden data

Forbidden in browser repo or mission state	Reason
PSA API keys	Public open-source browser must not carry integration secrets.
PSA OAuth refresh tokens	Long-lived secret exposure risk.
PSA client secrets	Server-side only.
PSA base credentials	Server-side only.
Vendor tenant secrets	Server-side only.
Raw ticket attachments containing secrets without redaction	Export/sync must be deliberate and previewed.

8.3 PSA writeback rule

```
Allowed browser action:
User clicks: Append Mission Summary to PSA Ticket
Browser sends approved summary to IT Docs API
IT Docs validates user/org/ticket permission
IT Docs server-side PSA connector writes ticket note
Browser receives success/failure display result

Forbidden:
Browser fetch('https://psa-vendor/api/...')
Browser stores PSA token
```

```
Browser automatically appends every timeline event to ticket
Browser writes ticket notes without explicit user action
```

9. Defense-in-depth security guardrails

Because the browser is open source, assume attackers can read the code, craft mission files, inspect local storage, tamper with IPC payloads, host malicious pages, and attempt to poison exports. Defense must be layered.

9.1 Threat model

Threat	Attack example	Required mitigation
Malicious web page	A remote page tries to call browser privileged APIs.	No Mission APIs exposed to webviews. Node off. Context isolation on. Strict preload bridge.
IPC abuse	Renderer sends malformed mission:save payload.	Allowlisted channels, schema validation, sender validation, size limits.
Mission file tampering	Attacker edits local JSON to include javascript: URL.	Treat files as untrusted. Validate URL protocols and schema before opening.
Secret leakage	Operator pastes bearer token into note.	Redaction scanner, export warning, preview before sync/export.
External-open abuse	Malicious URL triggers shell.openExternal command behavior.	Central safeOpenExternal wrapper; block unknown protocols.
Direct PSA abuse	Browser tries to write vendor ticket directly.	No PSA client code, no PSA tokens, grep/validator gate.
Object ID tampering	User changes projectId or ticketId locally.	Server-side IT Docs authorization on every object access.
Supply-chain risk	Dependency adds postinstall or malicious package.	npm audit/Dependabot, lockfile discipline, no unreviewed packages for core security paths.
Export injection	HTML export includes unsanitized tab title.	Escape all output. Markdown/HTML generators sanitize content.
Path traversal	Mission export path tries ../../	Main process restricts export to user-selected path or app-approved directory.

9.2 Security invariants

- Never rely on obscurity.
- Never trust renderer input.
- Never trust local mission files.
- Never trust remote page content.
- Never store third-party integration secrets in the browser.
- Never expose raw Electron, Node, filesystem, shell, eval, or IPC primitives to untrusted contexts.
- Never use local mission state as proof of authorization.
- Never auto-write to PSA or IT Docs without explicit user action and server authorization.
- Always validate payload shape, enum values, URLs, sizes, and object references.
- Always make security states clear without making local work impossible.

9.3 Defense layers

Layer	Required control
UX	Disabled/clear states for sign-in, permission denied, PSA unavailable, session expired. No broken mystery buttons.
Renderer	Typed API only. No raw IPC. Escape all dynamic text. No secrets in DOM outside user-entered text.
Preload	contextBridge exposes narrow functions. No ipcRenderer.on raw callback leak.
Main process	Schema validation, URL validation, safe external open, filesystem restrictions, IPC sender validation.
Webview/session	No Node integration. Verify webview options. Limit new

	windows and external links.
Storage	Schema versioning, size limits, path restrictions, no secrets, migration discipline.
Export	Redaction scan, preview, explicit confirmation, sanitized output.
Server boundary	IT Docs validates org/project/runbook/ticket permissions on every write.
Repo	Secret scan, generated-artifact exclusion, dependency review, release blockers.

10. Electron, IPC, webview, and URL hardening

Mission Tabs must obey Electron security recommendations and keep privileged work in the main process behind a minimal, validated API bridge. Electron recommends no Node integration for remote content, context isolation, sandboxing, limiting navigation/windows, validating IPC senders, and avoiding shell.openExternal with untrusted content.

10.1 Electron webPreferences guardrail

Setting / practice	Required stance	Exception process
nodeIntegration	false for every renderer or webview that can load remote content.	No exception for Mission Tabs.
contextIsolation	true.	No exception.
sandbox	true where compatible; documented exception if app shell requires a current workaround.	Exception must be named in SECURITY.md or release notes.
enableRemoteModule	false / not used.	No exception.
webSecurity	Do not disable.	No production exception.
allowRunningInsecureContent	false.	No production exception.
experimentalFeatures	false unless a specific reviewed feature is required.	Document risk and reason.
allowpopups on webview	Do not use by default.	Open via controlled window handler only.
shell.openExternal	Only through safeOpenExternal(url).	No direct calls with user/web content.

10.2 IPC allowlist

Allowed channel family	Examples	Validation required
mission:create	Create local mission.	Name length, mission type enum.
mission:rename	Rename mission.	Mission ID UUID, name length, escape on render.
mission:add-active-tab	Add active tab to mission.	Active tab exists, URL protocol allowed.
mission:set-tab-role	Assign role.	Role enum, mission/tab ID valid.
mission:set-layout	Set 1/2/3/4/focus layout.	Layout enum, pane assignment valid.
mission:save-local	Persist local mission.	Full schema validation and size limits.
mission:load-local	Load mission by ID.	App-owned path, schema validation.
mission:export-markdown	Export packet.	Redaction scan, sanitized output, user-selected path.
mission:link-itdocs	Attach IT Docs refs.	Signed-in state + server authorization result.
mission:get-auth-state	Query UI status.	No token return; display-safe state only.
tool:dns-lookup	Run DNS lookup tool.	Input domain validation; no shell injection.
tool:tls-summary	TLS/cert summary.	HTTPS host validation; timeouts.

10.3 Forbidden IPC patterns

```
execute
exec
shell
run-command
```

```
eval
open-url-anywhere
read-file-any
write-file-any
save-any-path
load-any-path
psa:direct-fetch
secret:get
cookie:get-all
auth:get-token
```

10.4 URL protocol policy

Protocol	Default	Notes
https:	Allowed	Preferred for all remote mission tabs and deep links.
http:	Restricted	Allowed for local/dev only if user explicitly enters or dev mode allows. Warn for external HTTP.
file:	Blocked for mission tabs by default	Only allow a designed local-doc feature with path validation.
tahai-browser:	Internal only	Parse as data. Do not execute commands from URL.
tahai-itdocs:	Internal/deep-link placeholder only	Validate and translate to HTTPS IT Docs origin if implemented.
javascript:	Blocked	Always.
data:	Blocked	Always for mission tabs/deep links.
vbscript:	Blocked	Always.
ftp:	Blocked or external only after warning	No privileged open.
unknown custom protocols	Blocked	Only allow explicitly reviewed protocols.

11. Mission files, storage, schema, and validation

Local mission files are useful, but they are untrusted. Validation must be centralized and reusable. Any new pass that edits the schema must also update tests and migration rules.

11.1 Canonical schema sketch

```
{
  "schemaVersion": 1,
  "missionId": "uuid",
  "name": "Cloudflare DNS Migration",
  "missionType": "migration",
  "mode": "local-only",
  "createdAt": "ISO-8601 timestamp",
  "updatedAt": "ISO-8601 timestamp",
  "tabs": [
    {
      "tabId": "uuid",
      "role": "primary-console",
      "url": "https://dash.cloudflare.com/",
      "title": "Cloudflare",
      "pinned": false,
      "paneId": "pane-1"
    }
  ],
  "layout": {
    "type": "quad",
    "activePaneId": "pane-1",
    "panes": [
      { "paneId": "pane-1", "role": "primary-console", "tabId": "uuid" }
```

```
      ]
    },
    "notes": [],
    "timeline": [],
    "links": {
      "itDocs": null,
      "psa": null
    }
  }
```

11.2 Validation rules

- Reject files over the configured mission-file size limit.
- Reject schemaVersion values that are unknown and not migratable.
- Reject unknown top-level executable/control fields.
- Reject tab URLs with blocked protocols.
- Reject invalid UUIDs for missionId, tabId, paneId, and eventId.
- Reject unknown role, layout, missionType, or mode enums.
- Escape all names, titles, notes, and ticket display values on render and export.
- Never load a path outside the application mission directory unless selected by the user through a safe file picker.
- Never execute data from a mission file.

11.3 Storage path discipline

Storage item	Allowed location	Forbidden
Mission JSON	App user data mission directory or user-selected import/export path.	Repo tree, dist, release, node_modules, public docs folder.
Evidence images	App user data evidence directory or explicit export packet.	Repo tree or cloud sync without explicit user action.
Temp export drafts	App temp/user data temp directory.	Committed source tree.
Config defaults	Packaged resources without secrets.	.env files or real tenant/customer secrets.
Signed-in display state	Safe display cache only.	Tokens, cookies, Authorization headers.

12. Evidence, export, and redaction guardrails

Evidence is a differentiator, but evidence can leak secrets. Exports must be safe by default and intentional when not redacted.

12.1 Evidence entry model

Field	Meaning	Rule
eventId	UUID event identifier.	Generated locally.
kind	url, screenshot, note, header-summary, tls-summary, dns-summary, checklist, export.	Enum only.
sourceTabId / paneId	Where evidence came from.	Must reference existing mission entity.
url	Source URL if applicable.	Validate/sanitize; no auth fragments copied.
title	Page/title display.	Escape on render/export.
createdAt	Timestamp.	ISO-8601.
operatorNote	User note.	Redaction scan.
metadata	Tool-specific safe metadata.	No raw cookies, auth headers, or tokens.

12.2 Redaction classes

Class	Examples	Default behavior

Bearer/API tokens	Bearer ..., x-api-key, ghp_..., github_pat_...	Warn and redact by default before export/sync.
Cloud keys	AWS access key patterns, secret access key labels.	Warn and redact by default.
JWT-looking strings	Header.payload.signature with base64url segments.	Warn; allow user to keep only if explicitly confirmed.
Private keys	BEGIN PRIVATE KEY / RSA PRIVATE KEY blocks.	Block export until removed or explicitly acknowledged with high-risk warning.
Cookies/auth headers	Cookie:, Set-Cookie:, Authorization:.	Never include in automated evidence metadata.
Emails	User/customer addresses.	Redact in sanitized handoff; optional in internal export.
IP addresses	IPv4/IPv6.	Redact in client/public handoff; optional in internal export.
Account/tenant IDs	AWS account IDs, Azure tenant IDs, org IDs.	Warn and optionally redact depending on export profile.

12.3 Export profiles

Profile	Use	Rules
Internal Markdown	Personal/local working notes.	Warn on secrets, allow explicit unredacted export.
Sanitized Handoff	Client/vendor/coworker handoff.	Redact sensitive classes by default; include safe metadata.
Incident Packet	Incident documentation.	Include timeline, screenshots, URLs, checks; redact tokens and secrets.
Change Record	Deployment/migration proof.	Include before/after, validation steps, timestamps, evidence.
IT Docs Sync	Authorized IT Docs evidence.	Server authorizes target; redaction preview required.
PSA Ticket Note	Ticket summary through IT Docs connector.	Short summary; no automatic attachments unless explicit.

13. Open-source repo hygiene and supply-chain gates

The public repo must remain safe and credible. The browser may be distributed unsigned while signing is pending, but the source cannot contain secrets, generated release artifacts, or customer data.

13.1 Do not commit list

```
node_modules/
dist/
release/
*.zip generated release bundles
*.exe installers
*.msi installers
*.dmg / *.AppImage / generated packages
runtime profiles
browser cache
cookies
localStorage/sessionStorage dumps
.env / .env.*
*.pem / *.key / *.p12 / *.pfx
id_rsa / id_ed25519
customer screenshots with secrets
real PSA sandbox credentials
cloud provider access keys
generated mission/evidence local files unless intentionally sanitized fixtures
```

13.2 Required public repo materials

File	Required status
LICENSE	Apache-2.0 preserved.
NOTICE	Attribution preserved.
TRADEMARKS.md	TAHAI name/mark boundaries preserved.
SECURITY.md	Security reporting and support posture preserved.
CONTRIBUTING.md	Contributor expectations preserved.
CODE_OF_CONDUCT.md	Community expectations preserved.
SUPPORT.md	Support boundaries preserved.
docs/code-signing-policy.md	Unsigned preview / future signing posture preserved.
docs/privacy-policy.md	Privacy posture preserved and updated only deliberately.
docs/known-issues.md	Known issues kept accurate.
docs/mission-tabs-security-spec.md	This spec should be converted into or referenced by a repo doc.

13.3 Secret scan patterns

Pattern family	Examples to scan
Environment / cloud	AWS_ACCESS_KEY_ID, SECRET_ACCESS_KEY, AZURE_CLIENT_SECRET, GOOGLE_APPLICATION_CREDENTIALS.
HTTP auth	Authorization:, Bearer , x-api-key, api_key, access_token, refresh_token.
OAuth	client_secret, refresh_token, id_token, cognito secrets.
Private material	BEGIN PRIVATE KEY, .pem, .key, .p12, .pfx, id_rsa.
PSA	connectwise secret, autotask secret, halo token, psa_api_key.
Generated data	cookies, Local Storage, IndexedDB, browser profile, session restore.

14. Verification commands and release gates

Every pass must return either a source delta zip, a full source zip when explicitly requested, or commands. Do not claim Windows-installed app behavior unless it was actually verified on Windows.

14.1 Baseline commands

```
Set-Location C:\dev\browser\app
npm ci
npm run verify:public-repo
npm run verify:release-blockers
```

14.2 Packaging commands

```
Set-Location C:\dev\browser\app
Remove-Item .\release -Recurse -Force -ErrorAction SilentlyContinue
$env:CSC_IDENTITY_AUTO_DISCOVERY = "false"
npm run package:win:release
npm run release:friend:zip
```

14.3 New mission-tabs security verifier

```
npm run verify:mission-tabs-security
```

14.4 verify:mission-tabs-security must check

- No forbidden IPC channel names are present.
- No direct PSA API URL/client code exists in browser source.
- Mission schema exists and rejects unknown protocol/role/layout/mode values.
- Mission model does not include token/secret fields.

- No shell.openExternal call exists outside a safe validation wrapper.
- No webPreferences enable nodeIntegration for remote content.
- No contextIsolation false in production renderer/webview paths.
- No sandbox false without documented exception.
- No raw ipcRenderer exposure through contextBridge.
- Redaction fixtures pass for bearer tokens, AWS key patterns, JWT-like strings, emails, IPs, private key blocks, Authorization/Cookie headers.
- Mission export escapes dynamic content.
- Generated artifacts and local mission/evidence files remain excluded from source.

14.5 Manual Windows installed-app gates

Gate	Manual test
Navigation parity	Back/forward/reload/address bar route to active tab or pane. Mouse Button 4/5 work and safely no-op when unavailable.
Mission drawer	Create/rename mission, add tab, assign role, save, close, restore.
Layout routing	1-Up/2-Up/4-Up routing targets active pane correctly.
Keyboard	Ctrl+K, Ctrl+Alt+1..4, Ctrl+Alt+Q/S/F function without breaking ordinary browsing.
Local-only	Mission features work without sign-in and display Local Only status.
Disabled integration	IT Docs/PSA buttons are clear and disabled when not signed in or not available.
Export safety	Secret-like content triggers warning/redaction preview.
No console noise	No unhandled promise rejections, DOM injection errors, or missing resource errors.

15. Pass plan and definition of done

The sequence below is intentionally bounded. Do not inflate pass counts. Each pass must deliver a clear source delta and verification result.

15.1 Versioned implementation roadmap

Version	Primary goal	Required deliverables
1.8.2	Mission Tabs local model	Create/rename mission, add/remove tab, assign role, save/restore local mission, drawer UI, mission security verifier skeleton.
1.8.3	Mission Control layouts	1-Up/2-Up/3-Up/4-Up, active pane tracking, navigation routing to active pane, keyboard shortcuts.
1.8.4	Launch Recipes and Admin Console Profiles	DevOps/IT templates, profile launch cards, mission start presets, no secret-bearing defaults.
1.8.5	Runbook Rail and Mission Timeline	Local notes/checklist/timeline, mission activity events, Markdown export skeleton.
1.8.6	Evidence Pack v1	Capture URL/title/timestamp, active pane metadata, all-pane metadata, export packet.
1.8.7	Redaction and export safety	Redaction scanner, export profiles, preview/warnings, sanitized handoff.
1.8.8	OpsTools Pack 1	DNS, TLS, headers, redirects, JSON/YAML formatter, JWT decoder, CIDR calculator, curl builder.
1.8.9	IT Docs link placeholder/client contract	Signed-in state detector, capabilities UI, IT Docs reference model, disabled states if unavailable.
1.9.0	PSA reference support	PSA reference display model, validated deep links, IT Docs-routed writeback command

		stubs only.

15.2 Definition of done for every pass

- No blind runtime DOM hacks.
- Actual TypeScript/Electron/renderer/CSS/source changes only.
- No generated release artifacts or local data committed.
- No secrets added.
- Existing browser navigation still works.
- Existing DevOps and IT Tools cards still work.
- Public repo verification passes or failure is explicitly reported.
- Release-blocker verification passes or failure is explicitly reported.
- Mission-specific verifier passes once introduced.
- Delta zip unfolds directly at repo root without extra nesting.
- Any Windows-only behavior not testable in ChatGPT is explicitly marked for local verification.

15.3 Stop conditions

Stop condition	Required response
Security invariant conflict	Stop coding, state conflict, propose safest bounded alternative.
Need IT Docs backend support	Implement browser-side placeholder/contract only; do not create backend in this repo.
Need PSA credentials/API call	Do not implement in browser. Route through IT Docs contract.
Cannot run npm ci/build/package	Report exact limitation and provide local commands. Do not claim verified packaging.
Renderer/webview privilege temptation	Refactor through main/preload allowlisted API.
Feature creates clutter	Hide under Ops Mode, side rail, command palette, or workspace template.

16. Code implementation map

Exact file names may vary with repo evolution, but the implementation should remain grouped and reviewable. Do not scatter Mission logic across unrelated runtime injection blocks.

16.1 Recommended source organization

Area	Recommended files / modules	Purpose
Types	src/shared/mission-types.ts	Mission schema interfaces, enums, state types, export profiles.
Validation	src/shared/mission-validators.ts	Pure validation for URLs, schema, roles, layout, sizes.
Redaction	src/shared/redaction.ts	Secret scanner and redaction profiles.
Main persistence	src/main/mission-store.ts	Validated load/save/import/export.
Main IPC	src/main/mission-ipc.ts	Allowlisted IPC handlers.
External open	src/main/safe-open-external.ts	Central URL/protocol validation wrapper.
Preload bridge	src/preload/mission-api.ts or existing preload module	Narrow contextBridge Mission API.
Renderer model	src/renderer/mission-state.ts	UI state/store and command routing.
Renderer UI	src/renderer/app.ts plus templates/CSS	Drawer, mission tabs, role selector, layout controls.
Verification	scripts/verify-mission-tabs-security.mjs	Static security and schema checks.
Docs	docs/mission-tabs-security-spec.md	Repo-readable version of this guardrail PDF.

16.2 Implementation rules

- Mission APIs must be typed. Do not pass loosely shaped objects through IPC without validation.
- Mission validation must be shared between import, save, restore, export, and integration link paths.
- Renderer UI can optimistically display local state only after main process acceptance for persistence-sensitive actions.
- All dynamic titles, notes, URLs, org names, ticket titles, and evidence labels must be escaped before insertion into HTML or export.
- Any direct webview control must target active pane or active tab deterministically.
- Any new command palette command must include a stable command ID, display title, target scope, and disabled reason when unavailable.
- Tests/verification must fail closed when schema/security assumptions cannot be confirmed.

17. New-chat handoff template

Use this exact handoff style to prevent drift across multiple chats and numerous passes.

```
We are continuing TAHAI Web Services Browser from the latest verified delta.

Repo:
C:\dev\browser\app

Public repo:
https://github.com/JTAHAI/tahai-web-services-browser

Current product direction:
TAHAI is an IT/DevOps command browser. Normal mode stays clean. Ops Mode opens Mission Control.
Mission Control = Mission Tabs + Mission Views + Mission Tools + Mission Evidence.

Hard scope:
Only browser-side work in this repo. IT Docs and PSA integrations are browser-side contracts/references
only.
No IT Docs backend code. No PSA connector code. No direct PSA API calls. No PSA/API/provider secrets in
browser code or mission files.

Security rules:
Open-source posture. Never trust renderer input, mission files, or remote page content. No blind DOM
hacks. No raw IPC exposure. No shell.openExternal without validated wrapper. No generated
artifacts/secrets/runtime data in source.

Verification:
npm ci
npm run verify:public-repo
npm run verify:release-blockers
npm run verify:mission-tabs-security   # once introduced

Next pass goal:
[Insert one bounded version target from the roadmap, e.g., 1.8.2 Mission Tabs local model]

Acceptance:
- Actual source changes only
- Existing navigation/devops/IT tools still work
- Security verifier updated
- Delta zip unfolds at repo root
- Windows-only installed-app verification explicitly stated if not run
```

18. Appendices

18.1 Mission type enum

```
missionType:
- deployment
- incident
- support
- documentation
- migration
- audit
- admin
- development
- security-review
- generic
```

18.2 Layout enum

```
layout.type:
- single
- split-horizontal
- split-vertical
- triple
- quad
- focus
- command
```

18.3 Mode enum

```
mode:
- local-only
- signed-in
- org-selected
- itdocs-linked
- psa-available
- psa-linked
- offline
- session-expired
- permission-denied
```

18.4 Minimum test cases

Test	Expected result
Create mission with valid name/type.	Mission created and saved locally.
Create mission with oversized name.	Rejected with clear validation error.
Import mission with javascript: URL.	Rejected; no navigation.
Import mission with unknown role.	Rejected or migrated to generic only if safe.
Mission file contains token field.	Rejected or stripped; warning logged without secret echo.
Ctrl+Alt+Q in normal browser.	Quad view opens with active pane indicated.
Back button in quad view.	Only active pane navigates back.
Capture note with Bearer token.	Redaction warning appears before export/sync.
Not signed in and click Link IT Docs.	Prompt to sign in; no error stack.
PSA unavailable for org.	PSA controls disabled with clear reason.
PSA deep link uses unknown protocol.	Blocked.
Raw ipcRenderer exposed in preload.	Verifier fails.
Direct PSA API URL found in source.	Verifier fails.

18.5 External standards consulted

Source	Use in this spec
Electron Security Tutorial - https://electronjs.org/docs/latest/tutorial/security	Electron hardening: Node integration, context isolation, sandbox, permission requests, webSecurity, navigation/window limits, shell.openExternal, IPC sender validation.

OWASP API Security Top 10 2023 - API1 Broken Object Level Authorization - https://owasp.org/API-Security/editions/2023/en/0xa1-broken-object-level-authorization/	Reason every org/project/runbook/ticket object reference must be authorized server-side by IT Docs.
OWASP Session Management Cheat Sheet - https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html	Session and token protection posture: do not expose/copy session identifiers into mission state.
OWASP Secrets Management Cheat Sheet - https://cheatsheetseries.owasp.org/cheatsheets/Secrets_Management_Cheat_Sheet.html	No secrets in source, mission files, fixtures, or export defaults.

18.6 Final anti-drift clause

Anti-drift clause

Future passes must improve the browser toward the Mission Control system without breaking normal browser simplicity, navigation parity, open-source repo hygiene, or the IT Docs/PSA security boundary. When uncertain, choose the safer local-only browser-side implementation and document what must wait for IT Docs server support.