Smart Capture

Session Context & Handoff Document

This document captures the full state of the Smart Capture project so that a future session can resume development without loss of context.

Project	Smart Capture — Chrome Extension (Manifest V3)
Location	~/Desktop/SmartCapture
License	BSD 3-Clause (Michael Koster)
Status	v1.0 complete, builds with zero errors, untested in browser
Source	~3,200 lines of TypeScript across 32 files
Build Output	~100 KB bundled JavaScript in dist/

What This Project Is

Smart Capture is a Chrome extension (Manifest V3) that replaces traditional bookmarks. When the user presses a keyboard shortcut (**Ctrl+Shift+S** / **Cmd+Shift+S**), it captures the current page's URL along with rich, auto-extracted metadata: page basics (title, OG tags, favicon), a content summary (excerpt, keywords, reading time), page type classification with type-specific fields, and user annotations (tags, notes, highlighted text). Each capture is stored as a structured JSON block with a UUID and ISO-8601 timestamp.

Design Decisions

These decisions were confirmed with the user at the start of the project:

1. **Form factor**: Chrome extension (not userscript or desktop app)
2. **Browser target**: Chrome Manifest V3 only (not cross-browser)
3. **Storage**: Pluggable backend via IStorageBackend interface, with chrome.storage.local (default) and IndexedDB as initial implementations
4. **Metadata categories**: All four — page basics, content summary, classification, and user annotations
5. **UI framework**: Vanilla TypeScript (no React/Preact) to keep the bundle small
6. **Build tool**: esbuild (bundled with tsx) — npm registry was inaccessible in the build environment, so we use esbuild directly via a shell script rather than Vite

Build System

The npm registry was blocked in the original build environment. The project has two build paths:

Primary (used in practice):

```
bash build.sh
```

- Uses esbuild (bundled with the globally-installed tsx package)
- Bundles 4 entry points: service-worker (ESM), content-script (IIFE), popup (ESM), options (ESM)
- Copies manifest.json, HTML (with .ts→.js patching), CSS, and icons to dist/
- Icons generated via sharp from SVG templates

Secondary (for standard Node.js environments):

```
npm install && npm run build
```

- Uses Vite with a custom plugin that copies manifest and icons
- Configured in vite.config.ts with rollup multi-entry
- Requires npm registry access to install vite and typescript

Type checking: `tsc --noEmit --project tsconfig.json`

Output: The `dist/` folder contains everything Chrome needs to load the extension.

Architecture Summary

```
src/
■■■ types/           # TypeScript interfaces (the schema contract)
■   ■■■ capture.ts   # Capture, PageType enum, 8 type-specific field interfaces
■   ■■■ storage.ts   # IStorageBackend, QueryFilter, StorageStats
■   ■■■ config.ts    # ExtensionConfig, CaptureProfile, DEFAULT_CONFIG
■   ■■■ chrome.d.ts  # Minimal Chrome API type declarations
■■■ background/
■   ■■■ service-worker.ts   # Initializes storage, keyboard shortcut, 14 message types
■   ■■■ capture-engine.ts   # Ensures content script, requests extraction, assembles Capture
■■■ content/
■   ■■■ content-script.ts      # Injected into pages; 3 message handlers
■   ■■■ metadata-extractor.ts  # OG tags, Twitter cards, meta tags, JSON-LD
■   ■■■ content-parser.ts      # Main content finder, keywords, reading time
■■■ classifier/
■   ■■■ classifier.ts          # Runs 3 detectors, picks extractor
■   ■■■ detectors/             # schema.org, URL patterns, DOM heuristics
■   ■■■ extractors/            # article, product, video, repo, docs, recipe
■■■ storage/
■   ■■■ chrome-storage.ts      # In-memory cache + chrome.storage.local
■   ■■■ indexeddb-storage.ts   # IDB with 4 indexes
■   ■■■ storage-factory.ts     # Cached backend factory
■■■ ui/
■   ■■■ popup/                 # Capture button, tags, notes, recent list
■   ■■■ options/               # Storage backend, toggles, export/import
■■■ utils/                    # UUID, date, sanitizer, logger
```

Message Protocol

The extension uses Chrome's message passing. All responses have the shape `{ success: boolean, ... }`.

Service worker handles 14 message types:

Message	Direction	Purpose
`capture_current_tab`	popup → SW	Trigger full capture of active tab
`update_annotations`	popup → SW	Update tags/notes on existing capture
`get_capture`	popup → SW	Retrieve single capture by ID
`query_captures`	popup → SW	Query with filters, sort, pagination
`get_recent`	popup → SW	Get latest N captures (default 10)
`get_all_tags`	popup → SW	Get sorted unique tag list
`get_stats`	popup → SW	Get storage statistics
`delete_capture`	popup → SW	Delete a capture by ID
`export_all`	options → SW	Export all captures as array
`import_captures`	options → SW	Import array of captures (merge)
`clear_all`	options → SW	Delete all captures

Message	Direction	Purpose
`get_config`	options → SW	Get current ExtensionConfig
`update_config`	options → SW	Update config (re-inits if backend changed)

Content script handles 3 message types:

Message	Direction	Purpose
`extract_all`	SW → content	Extract basics + summary + classification + selected text
`get_selected_text`	SW → content	Get window.getSelection() text
`ping`	SW → content	Health check before extraction

Keyboard shortcut flow: Ctrl+Shift+S → `chrome.commands.onCommand` in service worker → validates tab isn't chrome:// → `engine.capture(tabId)` → badge shows ✓ or ✗ for 2 seconds.

Classification Pipeline

Detection runs three layers in priority order and stops at the first confident match:

1. **SchemaOrgDetector** (confidence: 0.95) — Parses JSON-LD blocks, maps @type to PageType. Handles @graph arrays.
2. **UrlPatternDetector** (confidence: 0.85–0.9) — 24 regex rules for known domains (GitHub, YouTube, Amazon, StackOverflow, etc.).
3. **HeuristicDetector** (confidence: 0.5–0.8) — Counts DOM signals for 6 page types. Minimum score of 3 required.

Supported page types and extracted fields:

Page Type	Extracted Fields
Article	author, published/modified dates, section, publisher, reading time
Product	name, price, currency, rating, reviews, availability, brand, SKU
Video	title, duration, channel, upload date, views, thumbnail, embed URL
Repository	owner, name, stars, forks, language, license, topics
Documentation	section title, breadcrumb, TOC, version, framework
Recipe	name, prep/cook/total time, servings, calories, ingredients, cuisine
Social Post	Detection works; no extractor yet (returns empty fields)
Forum Thread	Detection works; no extractor yet (returns empty fields)

Storage Interface

Storage is pluggable via the `IStorageBackend` interface. Two backends are included:

Backend	Underlying API	Capacity	Notes
Chrome Storage	chrome.storage.local	~10 MB	In-memory Map cache, sync flush
IndexedDB	smart-capture DB v1	50+ MB	4 indexes (createdAt, type, tags, starred)

Interface methods: `init()`, `save()`, `get()`, `query()`, `delete()`, `getStats()`, `export()`, `import()`, `clear()`

Known Limitations and Gaps

These are intentional scope boundaries for v1:

1. **No SocialPostExtractor class** — SocialPostFields interface defined, detection works, extraction defaults to {}.
2. **No ForumThreadExtractor class** — Same situation as social posts.
3. **IndexedDB queries are in-memory** — query() fetches all records then filters. Fine for hundreds; needs cursors for 10,000+.
4. **IndexedDB stats return 0** — storageUsedBytes and storageQuotaBytes unavailable.
5. **No data migration between backends** — Switching abandons old data. Must export/import manually.
6. **English-only keyword extraction** — Stop word list is English only.
7. **No test suite** — vitest configured but no test files exist.
8. **No duplicate detection** — Same URL captured twice creates two captures.
9. **Favicon as URL only** — Not stored as data URIs; breaks if site goes down.
10. **Content script injection path mismatch** — capture-engine.ts references 'src/content/content-script.js' but built file is 'content-script.js'. Works because manifest auto-injects, but manual fallback path is wrong.
11. **No capture editing UI** — Tags/notes editable at capture time only. update_annotations message exists but UI doesn't expose it for existing captures.

Suggested Next Steps

In rough priority order:

1. **Test in Chrome** — Load dist/ as unpacked extension, test capture flow on various page types.
2. **Fix content script injection path** — Change 'src/content/content-script.js' to 'content-script.js' in capture-engine.ts.
3. **Add SocialPostExtractor and ForumThreadExtractor** — Follow existing extractor pattern.
4. **Add duplicate detection** — Check canonical URL before saving.
5. **Add capture editing** — Click existing captures to edit tags/notes.
6. **Write tests** — Unit tests for classifiers, extractors, storage backends.
7. **Data migration on backend switch** — Auto export/import when switching.
8. **Internationalization** — Configurable stop words, locale-aware dates.

File-by-File Reference

File	Lines	Role
src/types/capture.ts	175	Core schema: Capture, PageType enum, 8 field interfaces
src/types/storage.ts	65	IStorageBackend interface, QueryFilter, StorageStats
src/types/config.ts	35	ExtensionConfig, CaptureProfile, defaults
src/types/chrome.d.ts	75	Minimal Chrome API declarations
src/background/service-worker.ts	175	Init, command listener, 14 message handlers
src/background/capture-engine.ts	95	CaptureEngine: capture() and updateAnnotations()
src/content/content-script.ts	55	Message listener dispatching to extractors
src/content/metadata-extractor.ts	105	OG/Twitter/meta/link extraction
src/content/content-parser.ts	115	Main content finder, keywords, reading time
src/classifier/classifier.ts	75	Detector pipeline + extractor dispatch
src/classifier/types.ts	20	DetectionResult, PageDetector, TypeExtractor
detectors/schema-detector.ts	65	JSON-LD @type mapping, @graph arrays
detectors/url-detector.ts	75	24 regex rules for known domains
detectors/heuristic-detector.ts	110	DOM signal scoring for 6 page types
extractors/base-extractor.ts	50	getMeta(), getJsonLd(), nestedString(), toNumber()
extractors/article-extractor.ts	55	Author, dates, reading time
extractors/product-extractor.ts	100	Price, rating, availability, brand
extractors/video-extractor.ts	85	Duration (ISO 8601), channel, views
extractors/repository-extractor.ts	70	GitHub URL parsing, star count handling
extractors/documentation-extractor.ts	70	Breadcrumbs, TOC, version detection
extractors/recipe-extractor.ts	65	Times, servings, calories, ingredients
src/storage/chrome-storage.ts	145	In-memory cache + chrome.storage.local flush
src/storage/indexeddb-storage.ts	165	IDB with 4 indexes, full query support
src/storage/storage-factory.ts	30	Cached factory function
src/ui/popup/popup.ts	195	Capture flow, tag autocomplete, recent list
src/ui/popup/popup.html	85	Popup markup
src/ui/popup/popup.css	240	Popup styles (indigo primary)
src/ui/options/options.ts	115	Config persistence, export/import/clear

File	Lines	Role
src/ui/options/options.html	95	Options page markup
src/ui/options/options.css	185	Options page styles
src/utils/uuid.ts	15	crypto.randomUUID() with fallback
src/utils/date.ts	25	ISO timestamps, relative time
src/utils/sanitizer.ts	30	HTML escaping, truncation
src/utils/logger.ts	40	Tagged logger with level filtering
manifest.json	40	MV3 manifest: commands, permissions, content_scripts
build.sh	45	esbuild-based build script
vite.config.ts	70	Alternative Vite build config
Total	**~3,200**	