VerifiedSignal

Use Case Specifications & Screen Wireframes

Document Intelligence Platform

Version 1.0 · January 2025

Introduction

VerifiedSignal is a document intelligence platform that enables users to upload, analyse, and explore documents with automated scoring across eight quality dimensions: factuality, AI generation probability, logical fallacies, pseudoscience indicators, fictional content likelihood, and source provenance.

This document describes the primary end-to-end use cases and presents wireframe mockups of every major screen. It is intended as a reference for product design, engineering, and stakeholder review.

Scope of this document

- User registration and authentication
- Document upload and ingestion pipeline
- Document review with inline scoring and alerts
- Collection statistics and metadata analytics
- Library search with keyword and semantic modes
- Report generation and PDF download
- Account management and billing
- Security and session management

Use Case 1 — Arrival and Login

A new or returning user navigates to verifiedsignal.io and signs in to reach their personal dashboard. This is the entry point for all authenticated workflows.

User goal

Access the VerifiedSignal application and reach the personal dashboard without friction, whether logging in for the first time or returning after a session gap.

Workflow steps

1. **Open browser.** User navigates to verifiedsignal.io. The marketing landing page loads as a static asset from CloudFront — target load time under 1.5 seconds.
2. **Navigate to login.** User clicks "Log in" in the top navigation. The React SPA transitions to /login without a full page reload.
3. **Enter credentials.** User enters their email address and password. The form validates client-side before submission.
4. **Authenticate.** FastAPI passes credentials to Supabase Auth. On success, the access token is stored in memory and the refresh token is set as an httpOnly cookie.
5. **Dashboard loads.** The user is redirected to /dashboard. Recent documents, collection stats, and KPI metrics are fetched and rendered. An SSE connection is opened for live pipeline updates.

Error states

- Invalid credentials: inline error message shown below the password field — no page redirect.
- Unverified email: user reaches dashboard but sees a verification banner and cannot upload until confirmed.
- Rate limiting: after 5 failed attempts in 15 minutes, login is temporarily blocked with a clear message.

Wireframe — Login screen

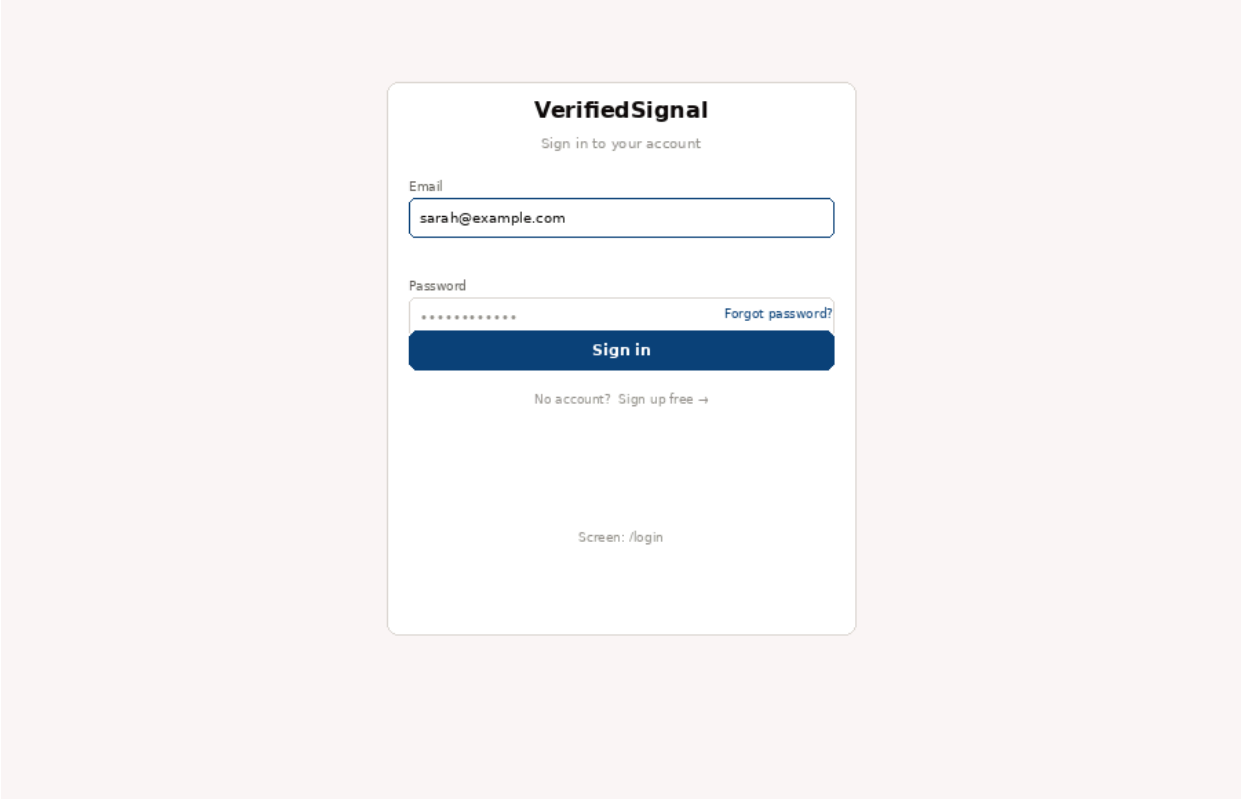

Figure 1.1 — /login — Centered card with email, password, and forgot password link

Wireframe — Dashboard

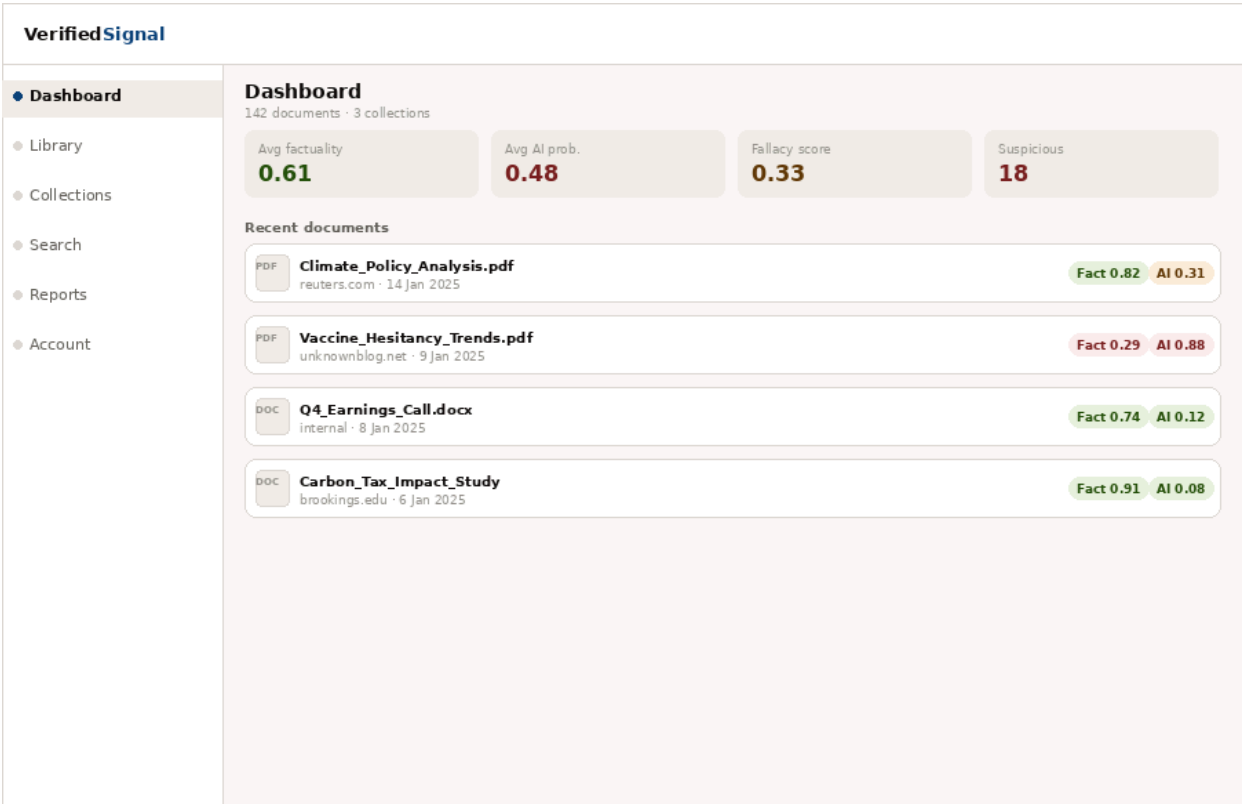

Figure 1.2 — /dashboard — KPI metrics, recent documents with score badges, sidebar navigation

Use Case 2 — Document Upload

The user uploads one or more documents (PDF, DOCX, TXT, HTML) or submits URLs for ingestion. The system validates, stores, and queues each document for pipeline processing, then streams progress back in real time.

User goal

Submit a batch of documents and know that they are safely stored and being analysed — without waiting for processing to complete before returning to other tasks.

Workflow steps

1. **Open upload panel.** User clicks "+ Upload" in the top bar or sidebar. A panel opens with two tabs: File upload (drag-and-drop zone) and URL submission.
2. **Select files.** User drags files onto the drop zone or uses the file picker. Thumbnails appear per file showing name, size, and a validation status.
3. **Duplicate check.** Content hash of each file is compared against the existing library. Duplicates are flagged with an amber warning badge and can be skipped or re-ingested.
4. **Upload begins.** On submission, the API issues presigned S3/MinIO URLs. Files are uploaded directly from the browser to object storage — bypassing the API server. On completion, the API creates document records and enqueues pipeline jobs.

5 **Monitor progress.** The panel transitions to a pipeline progress view. Each document shows its current stage via SSE. Scores stream in as they complete — factuality may be visible before AI detection finishes.

6 **Ingestion complete.** All documents show "Complete" with score badges. A summary shows counts of processed, failed, and skipped documents. User is prompted to assign them to a collection.

Technical notes

- Accepted file types: PDF, DOCX, TXT, HTML, MD. Maximum 50 MB per file, 10 files per batch.
- The presigned URL pattern means large files never pass through the API server, preventing memory pressure.
- If the user navigates away mid-pipeline, processing continues in the background and a notification is sent on completion.

Wireframe — Upload panel

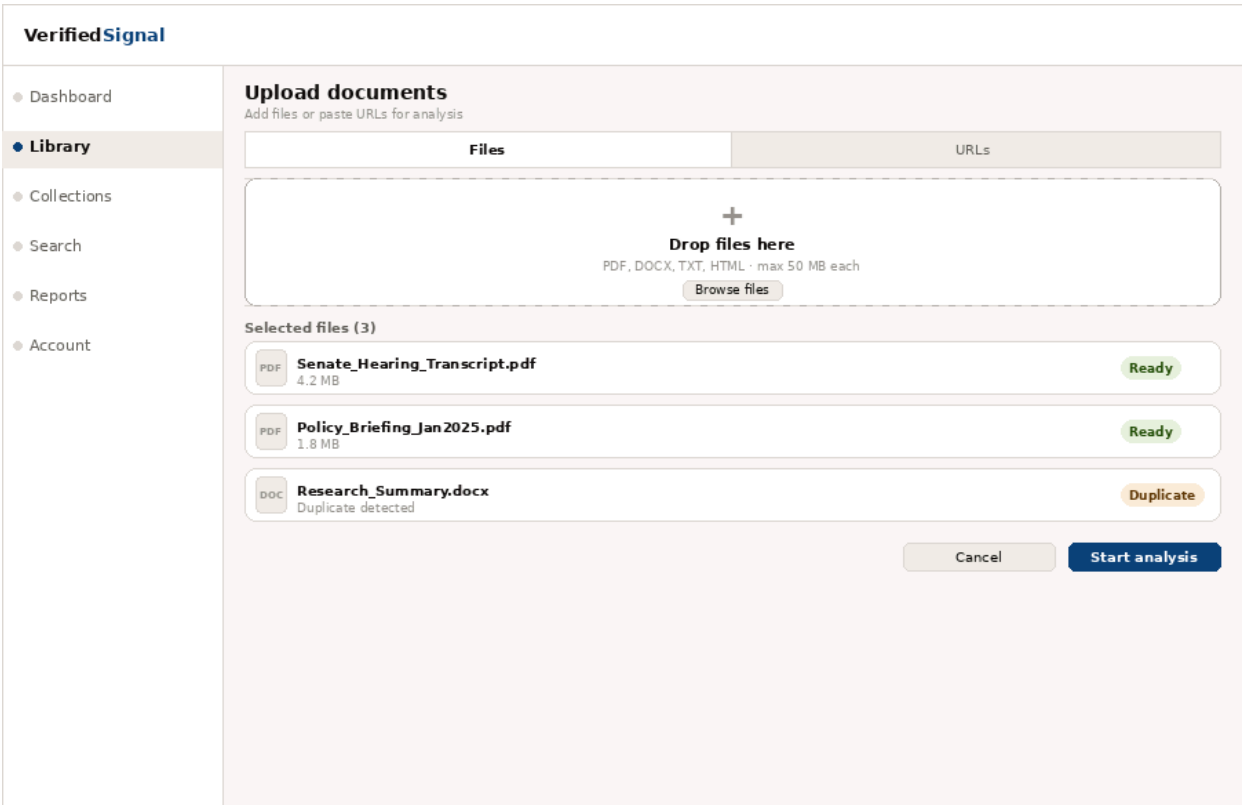

Figure 2.1 — /library/upload — File drop zone, selected files list, duplicate badge

Wireframe — Pipeline progress

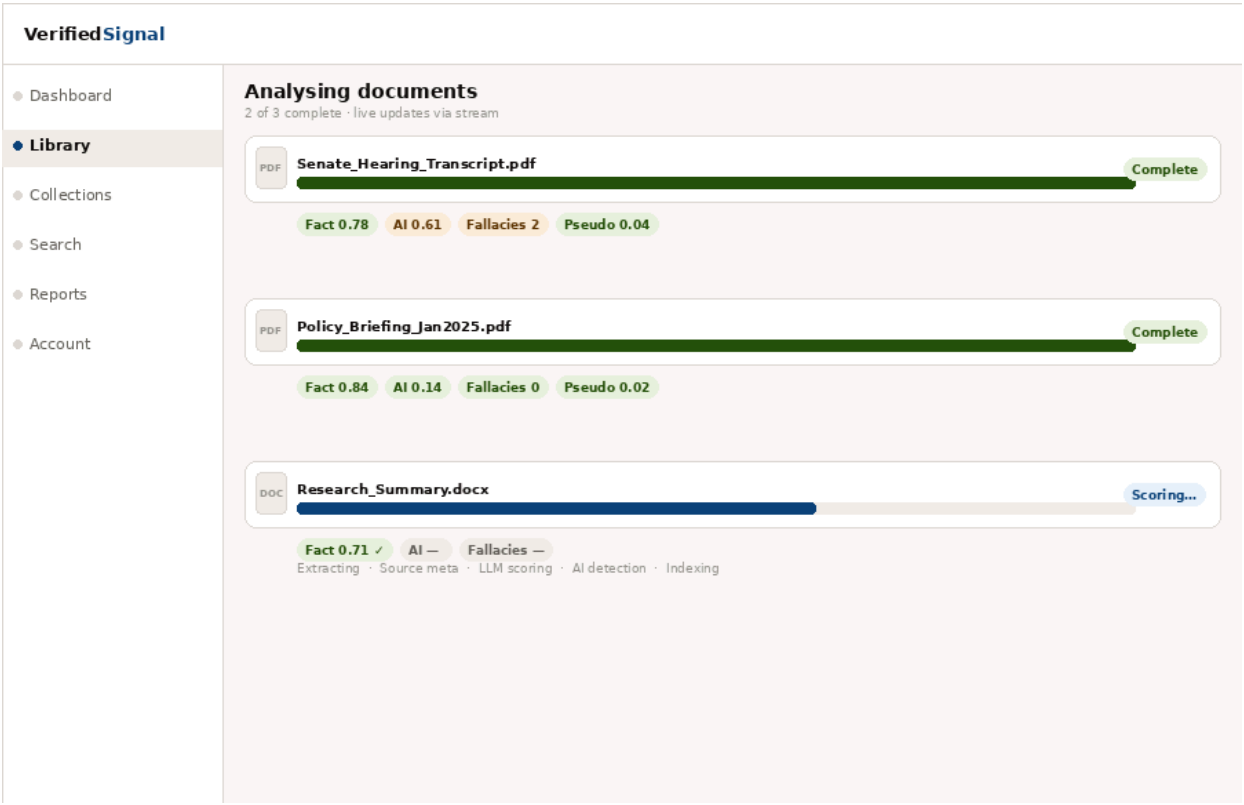

Figure 2.2 — Pipeline progress view — Live stage labels, streaming score badges per document

Use Case 3 — Document Review with Scoring

The user opens a document to read it. Scoring metadata, inline fallacy highlights, and AI detection alerts are visible throughout the reading experience without requiring additional clicks.

User goal

Read a document with critical intelligence overlaid — understanding not just what the document says, but how reliable, fallacious, or AI-generated it is, all visible while reading.

Workflow steps

1. **Open document.** User clicks a document in the library. The detail page loads at /documents/:id with a two-panel layout: document text left, metadata panel right.
2. **Alert bar.** If any score exceeds a danger threshold (e.g. AI probability > 0.75), a persistent red alert bar appears above the content with score badges visible at all times.
3. **Read with highlights.** Flagged passages are underlined in amber. Hovering over a highlight reveals a tooltip naming the specific fallacy type (e.g. "Appeal to authority") with a brief explanation.
4. **Review score panel.** The right panel shows all 8 scores as labelled bar gauges with colour coding. Each score row expands to show the LLM rationale text.
5. **Explore keywords.** Switching to the Keywords tab in the right panel shows auto-extracted terms ranked by TF-IDF. Clicking a keyword highlights every occurrence in the document.

6 **Focus mode.** User clicks "Focus mode" to expand the document full-width. The score alert bar remains pinned at the top of the viewport during scrolling.

Score dimensions displayed

- Factuality confidence (0.0 – 1.0) with rationale
- AI generation probability with model guess
- Logical fallacy score with named fallacy types and flagged passages
- Pseudoscience score with named indicators
- Fictional content likelihood
- Source provenance: domain, publication date, author, archive history

Wireframe — Document reader

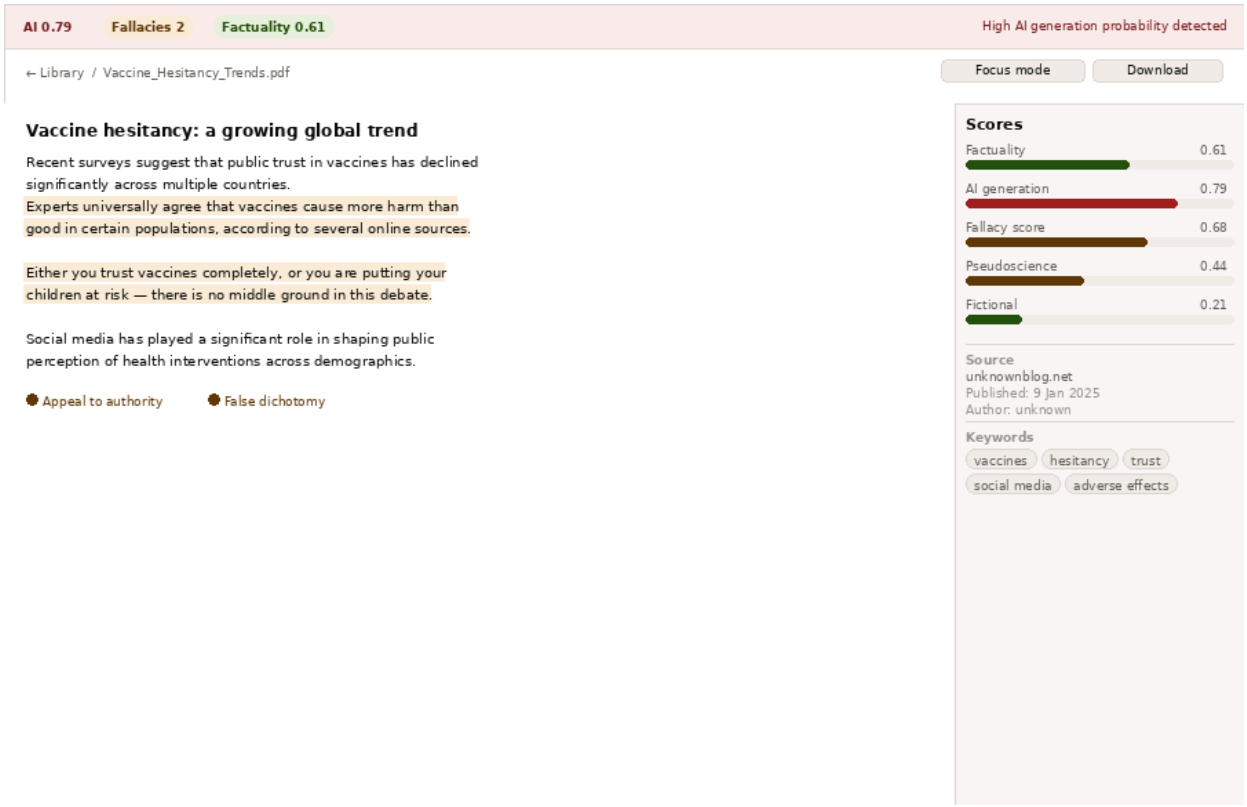

Figure 3.1 — /documents/:id — Alert bar, inline highlights, score panel, keywords tab

Use Case 4 — Statistics and Metadata Analytics

The user explores aggregated statistics about a collection of documents — score distributions, fallacy type breakdowns, source domain analysis, and month-over-month trends — all powered by Elasticsearch aggregations.

User goal

Understand the collective quality and characteristics of a document library at a glance, and drill into specific patterns (e.g. which sources are most reliable, which fallacies are most prevalent).

Workflow steps

1. **Open collection.** User navigates to Collections and selects a named collection. The overview tab loads with KPI metric cards — avg factuality, avg AI probability, avg fallacy score, and suspicious document count.
2. **Review distributions.** Score distribution histograms show the spread of factuality and AI probability across documents. Tail clusters (very high or very low) are immediately visible.
3. **Explore fallacy breakdown.** The fallacy section ranks detected fallacy types by frequency across the collection. Clicking a fallacy type filters the library to documents containing it.
4. **Trend analysis.** The Trends tab plots AI probability and factuality scores month-over-month as dual line charts, revealing quality drift over time.
5. **Source analysis.** The Sources tab shows a ranked table of domains with average scores per domain, revealing which sources are consistently reliable or problematic.
6. **Export data.** User downloads the aggregation data as JSON or CSV for use in external tools. The API endpoint GET /collections/:id/export is also available for programmatic access.

Wireframe — Collection statistics dashboard

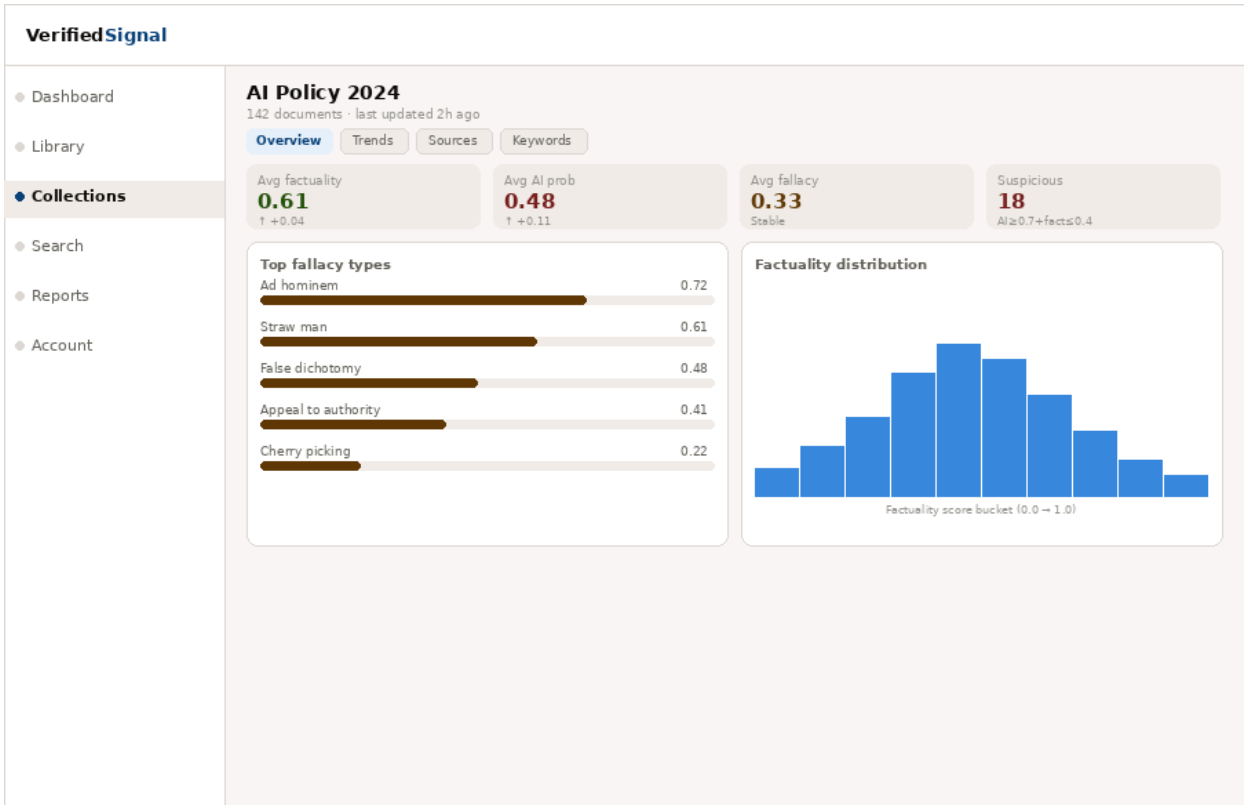

Figure 4.1 — /collections/:id/analytics — KPI cards, fallacy bar chart, factuality histogram

Use Case 5 — Library Search

The user searches across their document library using keyword queries, fuzzy matching for typos, and semantic search for conceptual similarity — all powered by Elasticsearch.

User goal

Find specific documents or passages within a large library quickly and accurately, even when the exact wording is uncertain or the query is conceptual rather than literal.

Workflow steps

1. **Open search.** User clicks the Search icon in the sidebar or presses Cmd+K. The search view opens at /search with a full-width input and recent searches shown as clickable pills.
2. **Keyword search.** User types a query and submits. Full-text results appear ranked by relevance, with matching phrases highlighted in yellow within 200-character snippets.
3. **Fuzzy matching.** Typos (e.g. "goverment") are corrected automatically. A "Did you mean?" suggestion appears above results. Phrase searches use double quotes for exact matching.
4. **Apply filters.** The filter panel allows narrowing by factuality score range, fallacy types (multi-select), AI probability range, source domain, and date range. Active filters appear as dismissible pills.
5. **Semantic search.** User switches to Semantic mode. The query is embedded as a vector and matched against document embeddings in Elasticsearch using knn search. Finds conceptually related documents without exact keyword overlap.
6. **Save search.** User saves the current query and filters as a named saved search for future reuse.

Search modes

- Keyword: BM25 full-text search across title, content, and author fields.
- Semantic: knn vector search using sentence-transformers embeddings.
- Hybrid: Combined BM25 and knn using Elasticsearch's reciprocal rank fusion.

Wireframe — Search

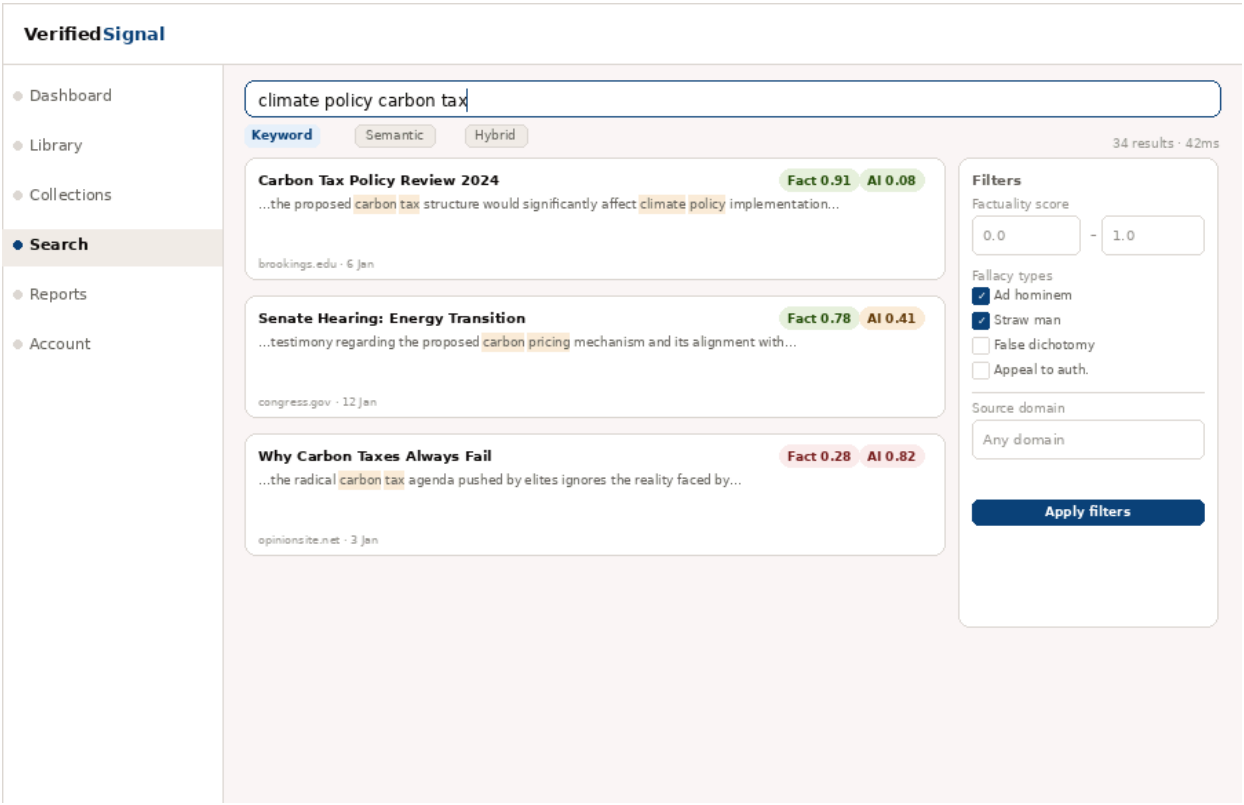

Figure 5.1 — /search — Keyword results with highlights, score badges, filter panel, mode selector

Use Case 6 — Report Generation and Download

The user configures and generates an intelligence report about a collection or custom document set. The report is available as a paginated HTML preview and a downloadable PDF.

User goal

Produce a shareable, professional summary of a document collection — suitable for briefing stakeholders, documenting research findings, or archiving an analysis run.

Workflow steps

1. **Open report builder.** User navigates to Reports and clicks "New report". A configuration panel opens with scope selection, date range, and report name.
2. **Configure scope.** User selects a collection (or manually picks individual documents) and sets an optional date range to filter by ingestion or publication date.
3. **Select sections.** User toggles which sections to include: executive summary, score distributions, top fallacies, suspicious documents table, source breakdown, keyword trends.
4. **Generate.** On submission, a background job runs Elasticsearch aggregations, calls the LLM for the executive summary, and compiles the report. Progress is streamed via SSE.
5. **Preview.** The completed report renders in a paginated HTML preview. All charts and tables are visible before download. The user can return to configuration and regenerate.

6 **Download PDF.** The server renders the report to PDF server-side (Playwright or WeasyPrint) and streams it to the browser. Reports are saved in history and re-downloadable for 90 days.

Report sections

- Executive summary: LLM-generated plain-language overview of key findings.
- Score distributions: histograms for all 8 dimensions.
- Suspicious documents: table of documents with AI prob ≥ 0.7 and factuality ≤ 0.4.
- Source breakdown: average scores per domain, sorted by document count.
- Keyword trends: word cloud and time-series frequency chart.

Wireframe — Report builder

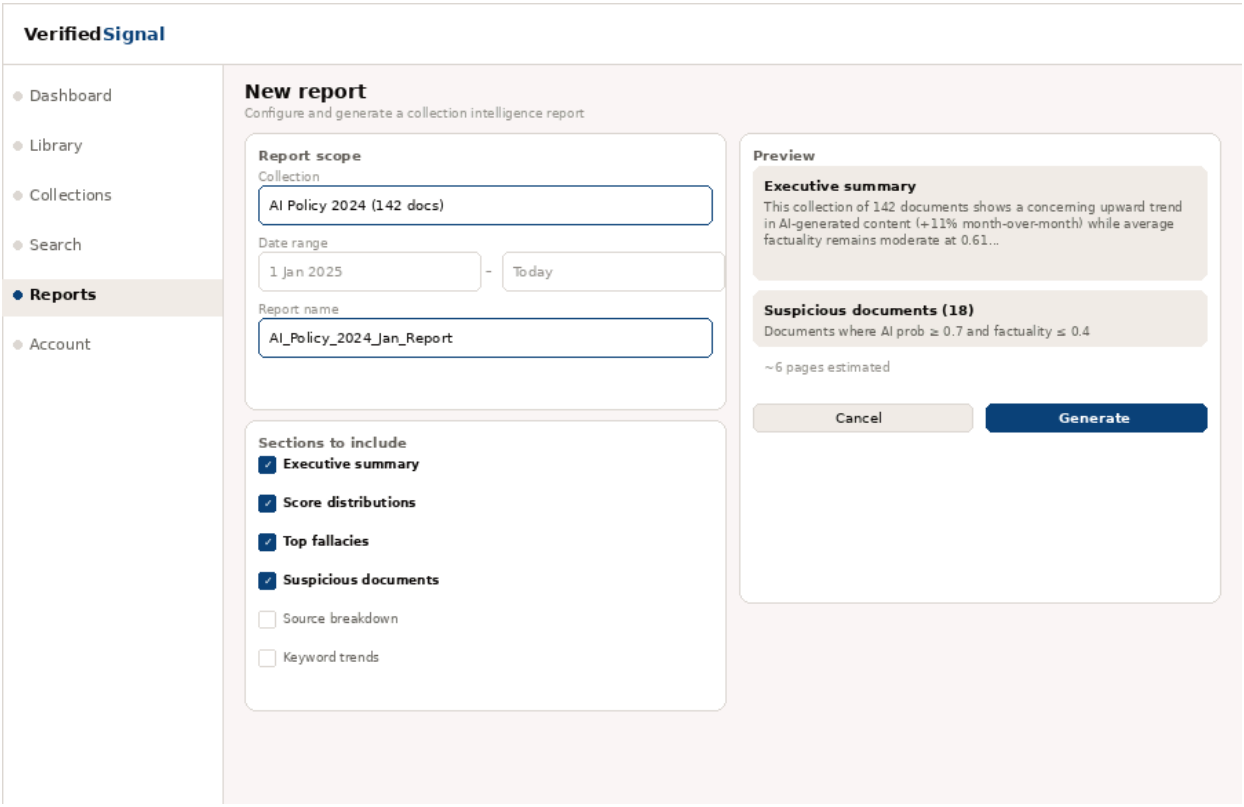

Figure 6.1 — /reports/new — Scope config left, live section preview right

Use Case 7 — Account and Billing

The user reviews their subscription plan, monitors usage against limits, manages payment methods, and downloads invoices.

User goal

Stay informed about subscription status and usage, update billing details without friction, and access invoices on demand.

Workflow steps

1. **Open account.** User clicks their avatar in the sidebar. The Account section opens at /account with sub-pages: Profile, Billing, Usage, and Security.
2. **View plan.** The Billing page shows all three plan tiers side-by-side. The current plan is highlighted with a blue border and a "Current plan" badge.
3. **Monitor usage.** Progress bars show documents used and storage consumed against plan limits. A warning banner appears when 80% of the document limit is reached.
4. **Update payment.** User clicks "Update payment method". A Stripe-powered modal opens. Card details are handled entirely by Stripe — VerifiedSignal never touches raw card data.
5. **Download invoice.** Invoice history shows date, plan, amount, and status. Clicking the PDF icon downloads the invoice generated by Stripe.

Wireframe — Account and billing

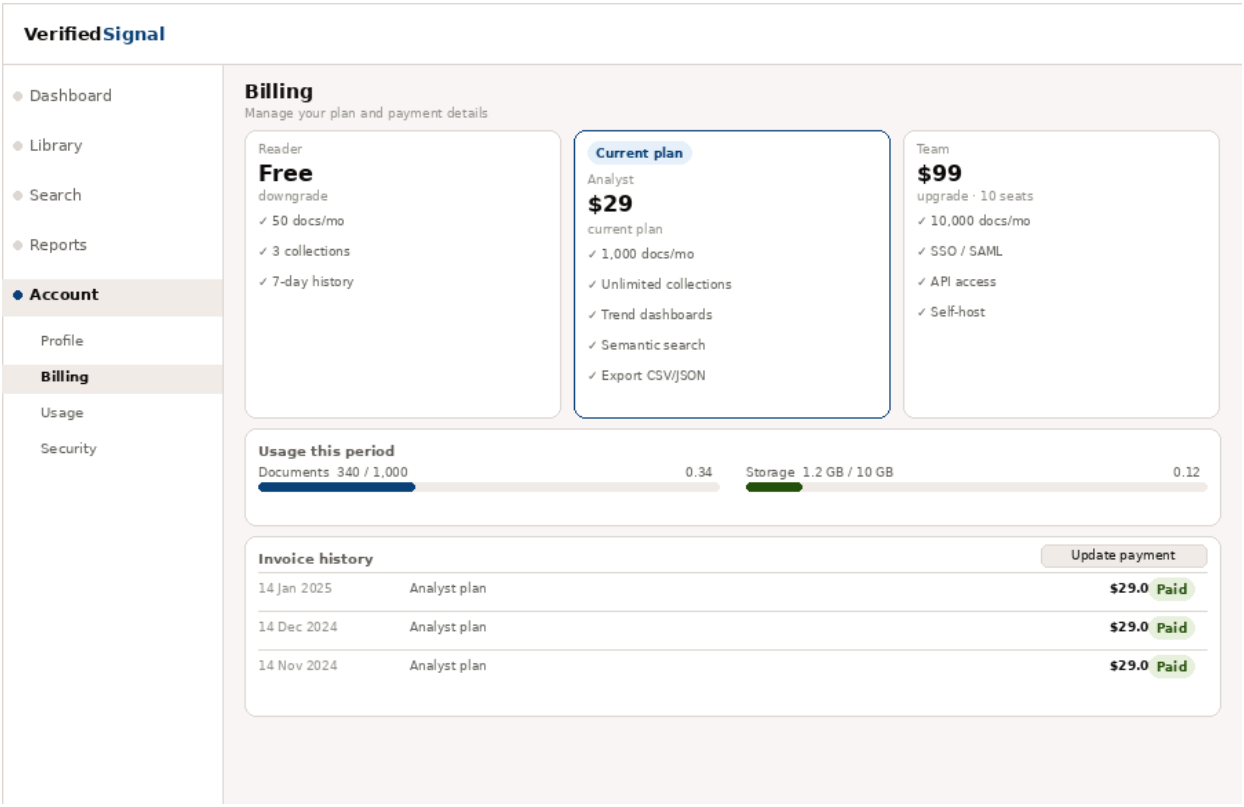

Figure 7.1 — /account/billing — Plan cards, usage bars, invoice history with PDF download

Use Case 8 — Security and Session Management

The user manages their password, enables two-factor authentication, and reviews and revokes active sessions across devices.

User goal

Maintain control over account access across multiple devices, and be able to immediately revoke access from any device that is lost, stolen, or no longer in use.

Workflow steps

1. **Open security settings.** User navigates to Account → Security. Three sections are visible: Change password, Two-factor authentication, and Active sessions.
2. **Change password.** User enters their current password and a new password (minimum 8 characters). Confirmation field must match. Update requires current password validation.
3. **Enable 2FA.** User clicks "Enable 2FA". Options are TOTP (authenticator app, e.g. Authy) or email OTP. Once enabled, the status badge changes to green.
4. **Review sessions.** The active sessions list shows browser, operating system, approximate location, and last active time for each session.
5. **Revoke a session.** User clicks "Revoke" on a specific session. The refresh token for that session is invalidated immediately. That device must re-authenticate on next request.
6. **Sign out all others.** User clicks "Sign out all other sessions" to revoke all refresh tokens except the current device in one action.

Security design notes

- Refresh tokens are stored as httpOnly cookies — not accessible to JavaScript, preventing XSS extraction.
- Access tokens are stored in memory only — never in localStorage or sessionStorage.
- Token revocation is immediate — the refresh token is invalidated server-side.
- Failed login attempts are rate-limited: 5 attempts per 15-minute window.

Wireframe — Security settings

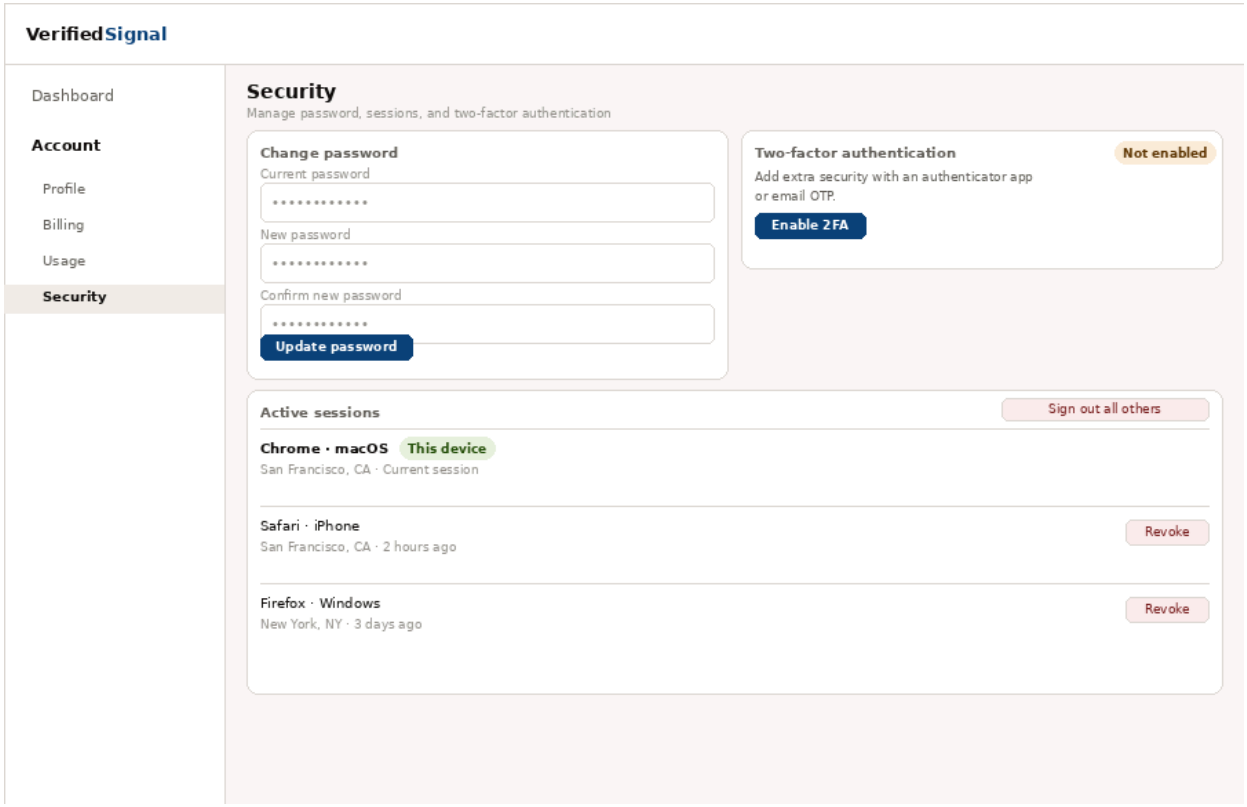

Figure 8.1 — /account/security — Password change, 2FA setup, active sessions with per-session revoke

Appendix — Screen Inventory

The following table lists every wireframe included in this document.

Figure	Screen	Route	Use case
1.1	Login	/login	Use Case 1
1.2	Dashboard	/dashboard	Use Case 1
2.1	Upload panel	/library/upload	Use Case 2
2.2	Pipeline progress	/library/upload (processing)	Use Case 2
3.1	Document reader	/documents/:id	Use Case 3
4.1	Collection analytics	/collections/:id/analytics	Use Case 4
5.1	Search	/search	Use Case 5
6.1	Report builder	/reports/new	Use Case 6
7.1	Account — billing	/account/billing	Use Case 7
8.1	Account — security	/account/security	Use Case 8

End of document.