Bilkent University

Department of Computer Engineering

Senior Design Project

Group T2515 - PatchMatch

Design Project Final Report

22102541, Bertan Uran, bertan.uran@ug.bilkent.edu.tr

22103867, Ekin Köylü, ekin.koylu@ug.bilkent.edu.tr

22203138, Elif Lara Oğuzhan, lara.oguzhan@ug.bilkent.edu.tr

22201668, Emre Yazıcıoğlu, e.yazicioglu@ug.bilkent.edu.tr

22203818, İlke Latifoğlu, ilke.latifoglu@ug.bilkent.edu.tr

Supervisor: Selim Aksoy

Course Instructors: Mert Bıçakçı, İlker Burak Kurt

04/05/2026

1. Introduction

Whole Slide Images (WSIs) are very large and detailed digital pathology images that contain a significant amount of visual information. While they are essential for medical research and analysis, exploring and comparing different regions within these slides can be slow and challenging when done manually. Although existing WSI viewers allow users to zoom, navigate, and view slides, they generally do not provide an easy way to find similar regions or related slides automatically.

PatchMatch is designed to solve this problem by providing a system that helps users quickly find similar regions and slides. Users can select a ROI on a slide, and the system will retrieve other regions from the dataset that look similar or share similar characteristics. This allows users to explore patterns, compare findings, and better understand the data without manually searching through large numbers of slides.

The main goal of PatchMatch is to support researchers and pathologists by making slide exploration faster, easier, and more interactive. By reducing the effort required to search and compare regions, the system helps users focus more on analysis and interpretation rather than navigation.

PatchMatch brings together several features into a single application, including:

- Interactive selection of regions directly on whole slide images
- Retrieval of similar regions and slides from the dataset
- Visualization of results to support comparison and analysis
- Display of cellular-level information to provide deeper insight
- Filtering options based on metadata or additional information
- Tools for annotation, sharing, and collaboration
- A system structure that can handle large collections of pathology slides

Overall, PatchMatch aims to provide a practical and user-friendly environment where users can efficiently explore, analyze, and compare pathology data. This report presents the system’s requirements, design, implementation, testing process, maintenance strategy, and possible future improvements.

2. Requirements Details

2.1 Functional Requirements

2.1.1 Region-to-Region Retrieval

- The system shall allow users to select a ROI from a whole slide image.
- The system shall generate an embedding representation of the selected region.
- The system shall retrieve visually similar regions from the patch database.
- The system shall rank retrieved regions according to similarity scores.
- The system shall allow users to inspect retrieved regions in their original slide context.

2.1.2 Slide-to-Slide Retrieval

- The system shall allow users to retrieve slides similar to a query slide.
- The system shall compare slides based on aggregated patch embeddings.
- The system shall rank similar slides according to similarity.
- Users shall be able to open and inspect retrieved slides.

2.1.3 Text-to-Slide Retrieval

- The system shall allow users to search slides using free-text queries through a search bar.
- The system shall support retrieval using pathology concepts, keywords, or report descriptions.
- The system shall rank slides according to text-image similarity.
- Users shall be able to refine text search using metadata filters.

2.1.4 Region Inspect

- The system shall allow users to inspect a selected region at multiple zoom levels.
- The system shall display region-level metadata and retrieval information.
- Users shall be able to analyze cellular structures within the selected region.
- The system shall support tumor cell visualization overlays.

2.1.5 Class Heatmap

- The system shall generate class activation heatmaps over a slide or region.
- The system shall visualize predicted class distributions spatially.
- Users shall be able to toggle class heatmaps on and off.

2.1.6 Similarity Map

- The system shall generate heatmaps showing similarity distributions across a slide.
- The system shall visualize areas most similar to a selected query region.
- Users shall be able to use similarity heatmaps to guide retrieval exploration.

2.1.7 Cellular-Level Metrics Visualization

- The system shall generate cellular-level quantitative maps including:
    - Density maps
    - Circularity maps
    - Cell-type distributions
- The system shall detect and visualize tumor cells.
- Users shall be able to overlay these metrics on slides and selected regions.

2.1.8 Annotation and Report Integration

- Users shall be able to annotate both slides and regions.
- Users shall be able to add findings or notes to a report.
- The system shall support storing and editing annotations.
- Annotated regions shall be shareable and revisitable.

2.1.9 Overview and Navigation

- The system shall provide an overview panel for slide navigation.
- Users shall be able to navigate through annotations, search results, and retrieved regions.
- The system shall support synchronized overview and detailed region inspection.

2.1.10 Sharing

- Users shall be able to share slides, regions, retrieval results, and annotations.
- Shared views shall preserve selected regions, overlays, and annotations.

2.1.11 View Management

- The system shall support viewing different overlays, retrieval modes, and analysis layers.
- Users shall be able to switch between raw slide, annotation, heatmap, and cellular metric views.

2.2 Non-functional Requirements

The non-functional requirements ensure that PatchMatch works well in the real world to assist pathologists. They help establish a system that is easy to use, reliable, fast, scalable, and supportable.

2.2.1 Usability

- PatchMatch should not disrupt the workflow of the pathologists, so it should be easy to learn and integrate into the existing workflow in under 30 minutes. The pathologists should be able to utilize the application following the instructions and tips provided by the application without formal training.
- The terminology given in the graphical user interface should align with the pathological terms.
- The graphical user interface should be easy to interact with, the user should be able to find similar images in no more than 5 interactions.
- The application should provide feedback after user operations within a second to indicate success or failure of operations.
- The retrieved images should be displayed in a visually easy to interpret and explainable manner.
- The images should be examined by zooming in and out just like the real-world procedure.

2.2.2 Reliability

- The system should be available during the clinical hours more than 99% percent of the time. This way, healthcare services will not halt.
- If an error happens and the system shuts down, it should save the progress up to the point of the error. The progress may consist of annotating text on the data, selecting the ROI.
- In the case of errors the user should see an informative message.
- In case of a system failure, the system should be operable back in under six hours.

2.2.3 Performance

- The initial view of a WSI should be available upon selection within 10 seconds for 95% of the requests.
- Zooming in or out should provide the new image view within 0.5 seconds on average.
- For a selected ROI, the most similar top-K images should be displayed under 5 seconds for 90% of the queries.
- For a text query, the most associated top-K images should be displayed under 5 seconds for 90% of the queries.
- The system should encode the tiles in a fast manner to create the embeddings and make them accessible.

2.2.4 Supportability

- The components like the retrieval engine, embedding pipeline should be modular to be updated or repaired without affecting the other components.
- The code should be documented, and it should be tracked using a version control system.
- The system should be monitored through logging mechanisms to identify the issues.
- Users should access support resources such as FAQs and troubleshooting guides.
- Different medical file formats should be handled ensuring compatibility.

2.2.5 Scalability

- The retrieval engine should search images from a few thousands of WSIs to a much larger dataset without compromising latency.
- Embeddings should be stored using compression or efficient vector representations to reduce storage requirements, enabling a larger number of embeddings to be stored.
- Horizontal scaling can be considered to increase the overall capacity without redesigning the system.
- Different AI models should be added and used without breaking the working logic of the previous models or requiring to reprocess all existing WSIs.

2.2.6 Security

- The system should require authenticated access to see patient data.
- Professionals should be provided access based on which data they can see. The scope of the available data should be determined based on the user.

- The communication through the network should be done in an encrypted way using a protocol like TLS.

3. Final Architecture and Design Details

3.1 Overview

PatchMatch is a Content-Based Image Retrieval system built for digital pathology. It enables pathologists and researchers to search a repository of Whole Slide Images using image queries or text queries. The system is designed around a layered, modular architecture that separates the user interface, API points, machine-learning retrieval, and data persistence into four independent layers: Frontend, Backend, ML, and Data Storage.

The Frontend Layer serves as an advanced, interactive workspace designed to streamline the digital pathology workflow. It facilitates secure authentication and role-based access control, allowing pathologists to manage their private slide libraries or collaborate via shared repositories. At its core, the frontend features a high-performance Whole Slide Image (WSI) viewer that supports seamless exploration. Beyond simple viewing, it enables users to perform searches integrating visual regions of interest (ROI), natural language queries, and clinical metadata. The interface provides real-time visualization of diagnostic heatmaps and quantitative cellular metrics, allowing for deep spatial data analysis. Users can further explore the data through collaborative annotation tools, bookmarks, and detailed note-taking. Administrative personnel have access to dedicated dashboards for user management, hardware telemetry monitoring (CPU/GPU), and model status tracking. These interfaces allow administrators to evaluate global usage patterns, assign server-side WSI root directories with automatic user folder provisioning, and trigger system-level operations such as FAISS [1] index re-building and slide library synchronization.

The ML Layer is responsible for all computationally intensive retrieval operations. It encodes user queries into dense embedding vectors using domain-specific vision-language models (CONCH [2] v1 and CONCH v1.5), performs sub-second nearest-neighbor search over pre-computed patch embeddings via FAISS [1] inner-product indices, and ranks results through adaptive percentile-based score stretching. An intent detection pipeline automatically classifies incoming queries and selects per-subclass optimal fusion weights, removing the need for manual parameter tuning. Beyond patch-level retrieval, the layer supports whole-slide comparison through TITAN [3] slide embeddings and dense region-to-region search through batched k-NN aggregation. A CellViT [4] analysis service

provides quantitative cellular metrics by streaming pre-computed cell segmentation outputs for real-time ROI querying. Each module operates independently, allowing new capabilities such as alternative encoders or scoring strategies to be integrated without disrupting existing search functionality.

Data Management and Security are implemented through a dual-storage strategy that prioritizes both computational performance and transactional integrity. High-volume, unstructured artifacts ,including raw WSIs, pre-computed embedding vectors, and CellViT-generated segmentation maps, are managed within a local high-performance file system. This centralized storage approach minimizes latency during heavy retrieval operations and ensures that the system can handle multi-gigabyte files efficiently within the institution's infrastructure. Conversely, all structured and collaborative data, such as user accounts, slide ownership records, vectorized annotations, and clinical notes, is managed within a PostgreSQL [5] database to ensure ACID [6] compliance and relational consistency. Security is woven throughout this architecture, employing bcrypt for password hashing and a multi-layered authorization model that combines Role-Based Access Control (RBAC) with resource-level permissions to safeguard sensitive diagnostic data and facilitate secure multi-user collaboration.

Together, these four layers form a pipeline that supports the full retrieval workflow, from slide ingestion and offline embedding generation, through real-time multimodal search and score fusion, to secure result presentation and collaborative annotation. It remains flexible enough to accommodate new data sources, search modalities, and deployment configurations as the project evolves.

3.2 Subsystem decomposition

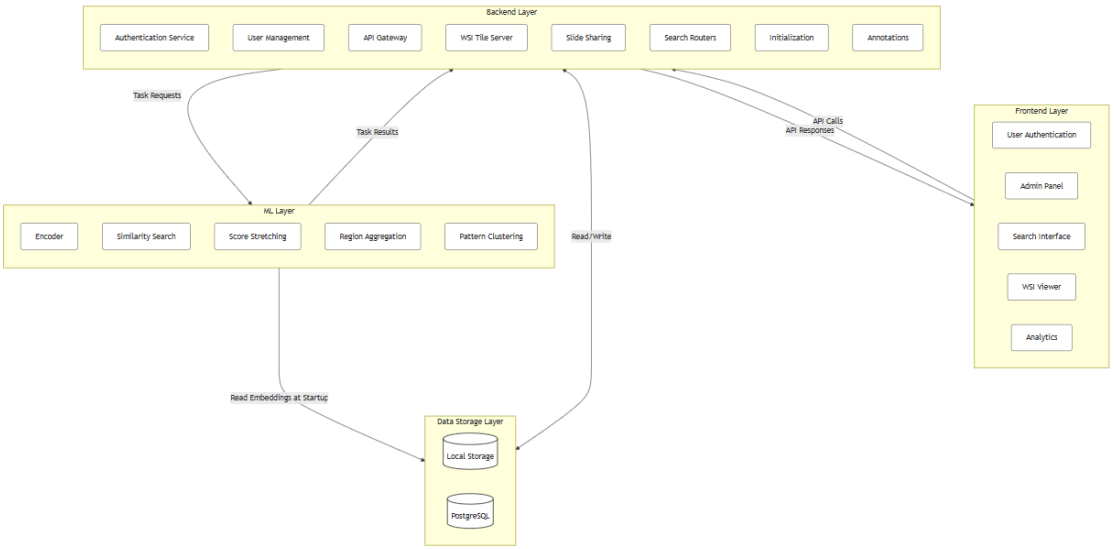


Figure 1. PatchMatch subsystem decomposition

3.2.1 Frontend Layer:

Frontend layer provides an advanced, interactive interface for pathologists to explore whole-slide images, perform multimodal searches across the repository, and analyze spatial tissue patterns through quantitative metrics.

Modules:

- Authentication: Handles secure user login and session management via JWT-based [7] authentication.
- Library & Collaboration: Provides a centralized workspace for browsing private slide libraries and a dedicated "Shared with Me" section for multi-user collaborative projects.
- Search Console: The primary retrieval interface supporting text-to-image, image-to-image, and dense region search. It displays results as ranked cards featuring adaptive similarity scores and clinical context.
- Diagnostic Viewer: Renders WSIs via OpenSeadragon [8] with IIIF-compatible tile serving. It supports high-resolution navigation, ROI selection, collaborative annotations, and real-time heatmap overlays for metrics like TIL Ratio and TSR.

- Admin Dashboards: Specialized modules for user account management, hardware telemetry monitoring (CPU/GPU), and ML model status tracking. It grants administrators the authority to assign server-side WSI root directories and trigger automatic user folder provisioning, as well as perform system-level operations like FAISS index re-building.

Data Flow:

- Frontend → Backend: Search queries, ROI coordinates, dynamic fusion weights, sharing requests, annotations, bookmarks, and clinical notes.
- Backend → Frontend: Ranked results with adaptive scores, clinical reports, IIIF [9] tiles, quantitative heatmap layers and spatial metrics, and authentication tokens.

3.2.2 Backend Layer:

Backend layer Serves as the central API gateway and orchestration engine. It manages communication between the frontend, the ML retrieval layer, and the storage systems while ensuring secure access to diagnostic data.

Modules:

- Auth & Security Service: Manages the full JWT lifecycle and enforces Role-Based Access Control (RBAC) across all API endpoints.
- WSI & IIIF Server: Responsible for high-performance, multi-resolution tile serving and metadata management for whole-slide images.
- Retrieval Orchestrator: Processes multimodal, textual, and dense region search queries. It implements Intent-Driven weight optimization to dynamically adjust search parameters.
- Analysis & Metrics Service: Handles the loading and serving of cellular-level metrics and outputs, quantitative tissue metrics (e.g., TIL Ratio), and spatial analysis heatmaps.
- Collaboration & Persistence: Manages the creation and retrieval of user-generated annotations, bookmarks, and clinical notes, as well as granular slide-sharing permissions.
- System Monitor (Init & Health): Tracks the multi-phase startup sequence (FAISS index building) and provides real-time health telemetry for hardware and model status.

Data Flow:

- Frontend → Backend: Search queries, ROI coordinates, sharing requests, annotations, bookmarks, and clinical notes.
- Backend → Frontend: Ranked results with adaptive scores, IIIF tiles, clinical reports, spatial heatmaps, quantitative metrics, and system health status.
- Backend → ML Layer: Query text for intent detection; visual patches for encoding; requests for dense region comparison and score fusion.
- Backend → Storage: Transactional data persistence (PostgreSQL); retrieval of WSIs and embeddings; management of CellViT and clustering caches.

3.2.3 ML Layer:

The ML Layer is the computational intelligence engine of the system. It hosts all deep learning models, manages high-dimensional embedding spaces, and executes the core similarity retrieval algorithms. The Backend orchestrates this layer by forwarding encoded queries and receiving ranked, score-calibrated results.

Modules:

Encoder Service: Manages the full lifecycle of the foundation models used for embedding generation. Supports a dual-encoder architecture where the image encoder and text encoder can be independently configured:

- CONCH v1 (CoCa-based, 512-d): Joint vision-language model capable of both image and text encoding.
- CONCH v1.5 (ViT-L + cross-attention pooler, 768-d): Vision-only encoder providing higher-resolution patch representations.

FAISS Index & Embeddings: Responsible for high-performance nearest-neighbor search and persistent embedding management. Builds FAISS IndexFlatIP indices (inner-product search over L2-normalized vectors, equivalent to cosine similarity) at startup from all loaded embeddings. Operates at two tiers:

- Patch-level index: All patch embeddings across all slides (typically tens of thousands of vectors), used for text-to-patch search, and dense region comparison.
- Slide-level index: One embedding per slide (TITAN slide embeddings), used for whole-slide-to-slide retrieval.

Intent Detection Engine: Processes incoming text queries to automatically determine the user's diagnostic intent and dynamically select per-subclass optimal weights. Implements a cascading detection pipeline:

- Keyword Detection: Regex matching against a curated vocabulary of breast cancer subtypes (IDC, ILC, DCIS, mucinous, papillary, metaplastic). Highest precision, fires first.
- Similarity Detection: Encodes the query with CONCH and computes cosine similarity against precomputed canonical description embeddings for each subclass (e.g., "invasive ductal carcinoma with desmoplastic stroma and irregular glandular structures"). Zero-shot fallback when keywords are absent.
- Image Probe Detection: For image-only queries, uses the BRACS linear probe to classify the query patch and maps the predicted BRACS [10] class (IC, DCIS, ADH) to a cancer subclass.

Region-to-Region Retrieval: Handles ROI-to-database comparison by executing a batched k-nearest-neighbor search for every patch in the user-selected region against the entire embedding database. Uses a max-pool-then-mean aggregation strategy:

- For each query patch, FAISS returns the top-2000 nearest database patches.
- Per query patch, the maximum similarity score per target slide is retained (max-pool).
- Across all query patches, per-slide max scores are averaged (mean aggregation) to produce the final slide-level ranking.

This avoids materializing the full Q×N dense similarity matrix (which causes OOM on large databases) while preserving fine-grained spatial similarity information. Query patches are capped at a configurable maximum (default 500) with deterministic subsampling for oversized ROIs. Each result slide includes a full per-patch heatmap with stretched scores for spatial visualization on the viewer.

Slide-to-Slide Retrieval: Provides whole-slide retrieval using TITAN, a vision-language foundation model for pathology. TITAN generates a single 768-dimensional slide embedding by attending over all CONCH v1.5 patch features via a contrastive attentional pooler with a single learned query. The retrieval pipeline:

- Loads precomputed TITAN slide embeddings and per-patch attention weights.
- Builds a FAISS inner-product index over L2-normalized slide vectors.
- At query time, retrieves the top-k most similar slides via single-vector search.

For similarity heatmap visualization, loads per-slide patch features from H5 files and computes a dense cross-slide patch similarity matrix to identify the most consistently matched regions across slides.

Region Classification & Semantic Matching: Provides diagnostic context when a user inspects a specific region.

- BRACS Linear Probe: A logistic regression classifier trained on BRACS dataset embeddings, classifying regions into 7 categories (Normal, Pathological Benign, UDH, FEA, ADH, DCIS, Invasive Carcinoma) with calibrated per-class probability scores.
- Quilt-1M [11] Semantic Index: Matches region embeddings against a curated set of 200+ breast pathology captions extracted from the Quilt-1M dataset (filtered for breast relevance using keyword scoring and prefix-based deduplication, supplemented with 40 hand-crafted fallback descriptors). Returns the top-3 most semantically similar natural-language descriptions, providing interpretable characterization of the inspected tissue.

CellViT Analysis Service: Provides quantitative cellular-level tissue analysis from CellViT model outputs. Operates in a two-phase architecture:

- Offline Precomputation: Streams raw CellViT cell_detection.json output (containing hundreds of thousands of individual cells per slide, each with centroid, contour, and type), bins cells into a spatial patch grid aligned to the embedding pipeline's 224×224 patches at 20x magnification, and computes sufficient statistics (counts and sums per patch).
- Runtime ROI Querying: When a user selects a region, retrieves only the precomputed patch statistics that overlap the ROI bounding box, aggregates them via summation of sufficient statistics, and returns both scalar metrics and spatial heatmap data.
- Classifies cells into 5 types following the CellViT type map: Neoplastic (1), Inflammatory (2), Connective/Stromal (3), Dead (4), and Epithelial (5). From these classifications, computes 4 quantitative tissue metrics:
    - Tumor Cellularity: Fraction of neoplastic cells relative to total cell count.
    - TIL Ratio: Inflammatory cells relative to neoplastic cells, quantifying immune infiltration.
    - Tumor-Stroma Ratio (TSR): Stromal and immune cells relative to the combined tumor microenvironment.

    - Mean Circularity: Average morphological regularity of tumor cell contours ($4\pi A/P^2$), computed via the Shoelace formula for area and discrete perimeter.

Data Flow:

- Backend → ML Layer: Region classifications with BRACS probabilities and Quilt-1M semantic descriptions, query text or image patch for encoding, retrieval parameters.
- ML Layer → Storage: Reads WSI files (via OpenSlide) for realtime ROI encoding, reads precomputed patch embeddings and spatial metadata, reads slide embeddings and attention weights, reads model checkpoints (CONCH v1 weights, CONCH v1.5 weights)
- ML Layer → Backend: Ranked lists of slide IDs, scores, coordinates, reports, and aggregated regions.

3.2.4 Data Storage Layer:

Data Storage Layer Persists all application data across two complementary storage systems chosen for their respective strengths in managing relational and large-scale unstructured data.

Components:

- PostgreSQL: Stores user accounts, hashed passwords, slide-sharing records, slide annotations, and clinical notes. Chosen for its reliability, ACID compliance, and efficiency in managing structured relational data.
- Local File System (Docker [12] Volumes): Stores multi-gigabyte whole-slide images (WSIs), pre-computed CONCH patch embeddings, CellViT spatial metrics (msgpack/HDF5), patch metadata, and clinical reports. Embedding files follow a per-slide directory convention and are accessed directly from the local disk to build the FAISS index in memory at startup.
- Note on Clinical Deployment: The architecture is designed to operate entirely on localized file systems and Docker volumes, eliminating reliance on external cloud providers. This native on-premise design ensures that all sensitive WSIs, diagnostic embeddings, and clinical reports reside strictly within hospital-managed servers. This approach inherently complies with strict data sovereignty and patient privacy regulations, such as KVKK and GDPR, guaranteeing that sensitive pathology data never leaves the institution's controlled infrastructure.

Data Flow:

- Backend → Data Storage Layer (PostgreSQL): Writes and reads user account data, access permissions, annotation records, and slide-sharing entries on every relevant API request.
- ML Layer → Data Storage Layer (Local File System): Reads pre-computed patch embedding files, patch metadata, and clinical report JSON files at startup to initialize the FAISS index and populate in-memory retrieval structures.
- Backend → Local File System: Reads heavy whole-slide image files directly from the disk to serve multi-resolution tiles to the frontend viewer dynamically on demand.

3.3 Hardware/software mapping

PatchMatch does not include any custom hardware components. The system runs entirely on standard server and client infrastructure. The mapping between software components and their target execution environments is as follows:

- Frontend Layer: Runs in the user's web browser on any standard workstation or clinical desktop. No installation is required beyond a modern browser such as Chrome, Firefox, Safari, or Edge. The OpenSeadragon WSI viewer runs entirely client-side as a JavaScript library.
- Backend Layer: Runs on a server-side Docker container hosting the FastAPI [13] application. In the current development environment this container is deployed on a cloud virtual machine. In a clinical deployment this would run on a hospital-managed server.
- Operates within the backend Docker container, leveraging a suite of high-performance foundation models and search indices. The system utilizes CONCH for generating high-dimensional vision-language embeddings, while CellViT is integrated for automated cellular segmentation and the generation of quantitative tissue metrics (e.g., TIL Ratio and TSR). Additionally, the layer incorporates specialized Intent Classifiers (including Quilt-1M logistic regression and BRACS-based linear probes) to dynamically optimize retrieval weights. All core models and FAISS indices are loaded into system memory at startup and remain resident to ensure sub-second query response times.
- Data Storage Layer: Relational data is managed via PostgreSQL (which can be deployed as a local container or a cloud-managed service like Neon). Large-scale artifacts, including WSIs, embeddings, and analysis outputs, are stored on the local file system, which is mounted into the Docker containers via persistent volumes.

The entire system is containerized using Docker Compose. This makes the deployment environment reproducible across development, testing, and production targets without hardware-specific configuration.

3.4 Persistent data management

PatchMatch does not implement a custom file system or database engine. However, the nature of WSI data and deep learning embeddings introduces persistent data management challenges that go beyond what a standard relational database alone can address, and the team has made deliberate storage design decisions to handle these. The core challenge is that whole-slide images are gigapixel-scale files, each potentially several gigabytes in size, and the patch embeddings derived from them are high-dimensional numerical arrays that are not suited to row-based relational storage. To address this, PatchMatch uses a split persistence strategy, which is a deliberate engineering decision for simpler infrastructure and faster startup in the current development context.

- Structured data such as user accounts, annotations, and sharing records is stored in PostgreSQL, where relational integrity and query flexibility are needed.
- Unstructured and numerical data—including Whole Slide Images (WSIs), patch embeddings, metadata, and clinical reports—are stored as flat files within a centralized local storage system. Patch embeddings are stored as NumPy (.npy) or HDF5 (.h5) files, while associated patch metadata is managed via .jsonl payload files. This naming convention and file structure allow the entire embedding corpus to be loaded into memory and indexed using FAISS during the system's initialization phase, facilitating high-speed similarity search without the overhead of repeated database queries. While the system is provider-agnostic and designed to scale to object storage (such as Google Cloud Storage or on-premise S3-compatible systems), the current deployment utilizes a high-performance local file management system mounted via Docker volumes to ensure low-latency access and simplified data sovereignty within the institution's infrastructure.

3.5 Access control and security

PatchMatch handles sensitive medical imaging data and operates in a clinical context, making access control and security integral to the system design rather than supplementary concerns.

- The system uses JWT-based authentication. When a user submits valid credentials, the backend issues a signed JWT bearer token which the frontend includes in the Authorization header of all subsequent requests. The Authentication Service validates this token on every protected endpoint through a dependency injection mechanism, meaning no protected route can be reached without a valid token. Passwords are stored as hashed values in PostgreSQL and are never transmitted or logged in plaintext.
- Two primary user roles are defined in the system. Regular Users can perform multimodal and dense region searches, create collaborative annotations, manage personal bookmarks and clinical notes, and share slides with granular permission levels. Administrators possess elevated privileges, allowing them to manage user account lifecycles, monitor real-time hardware telemetry, and oversee the data infrastructure by assigning server-side WSI root directories and orchestrating FAISS index updates. Access to slide libraries and annotations is strictly scoped to the owning user, ensuring that data remains private unless explicitly shared through the system's collaborative mechanism. However, administrators maintain global visibility over the repository to facilitate system-wide maintenance, user folder provisioning, and data governance.
- All communication between the frontend and backend is conducted over HTTPS, ensuring that queries, authentication tokens, and retrieval results are protected in transit. WSIs, embeddings, and clinical reports stored in the file storage layer are access-controlled at the infrastructure level through cloud storage bucket permissions during development. In a clinical deployment these would be further protected by hospital network access controls and storage encryption at rest in compliance with GDPR [14] and KVKK [15] requirements.

4. Development/Implementation Details

Details of the subsystems and methods can be viewed by downloading the class diagram and opening it in a web browser via the the following link: UMLClassDiagram_PatchMatch.svg

4.1 Frontend Subsystem

The frontend subsystem is a React-based single-page application built with TypeScript and Vite. Upon authentication, users are presented with a centralized workspace

that integrates a multimodal search console with a comprehensive slide library, including a dedicated section for shared collaborative projects. The search interface supports visual, textual, and dense region retrieval, presenting results as ranked cards with adaptive similarity scores and clinical context.

The WSI Viewer module, utilizing OpenSeadragon and IIIF protocols, serves as the primary diagnostic tool, supporting full-resolution navigation, ROI selection, and real-time visualization of diagnostic heatmaps and quantitative cellular metrics. To facilitate multi-user cooperation, the viewer includes integrated tools for collaborative annotation, bookmarking, and clinical note-taking. Access control is enforced through a role-based system: unauthenticated users are redirected to the login interface, while administrative dashboards, including System Status for hardware/model monitoring and for dataset statistics, and Infrastructure Tools for WSI root folder assignment and index management, are gated behind privileged user flags.

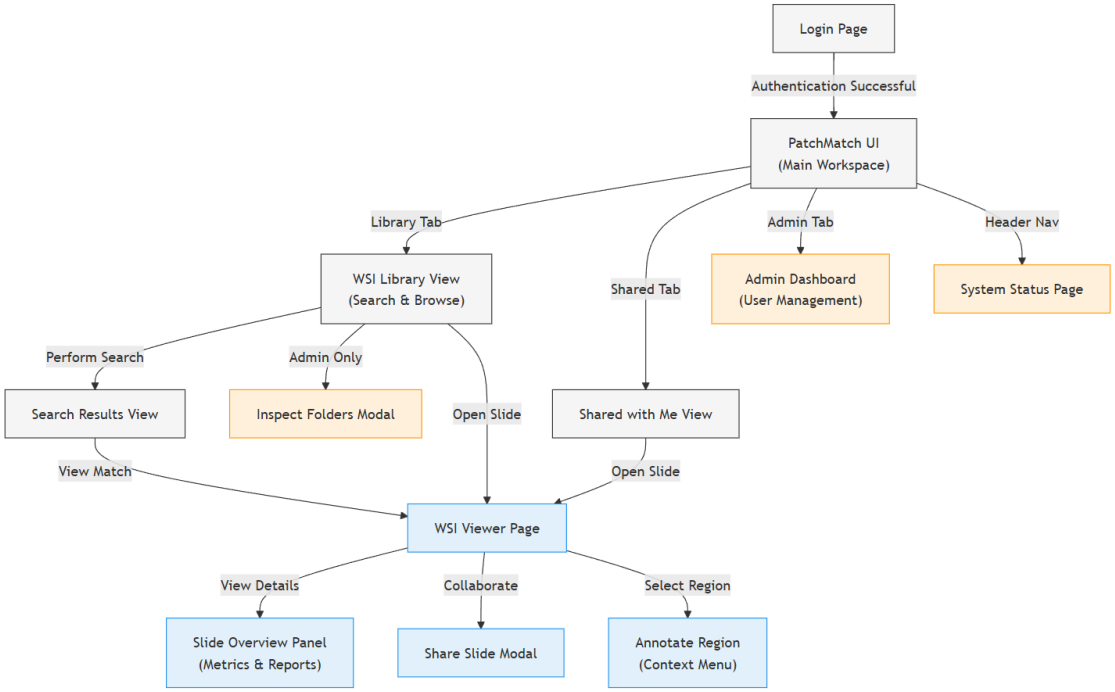


Figure 2. PatchMatch frontend subsystem

Methods

Auth Service (via useAuthStore):

- login(email, password): Submits user credentials to /auth/login as FormData, persists the returned JWT in localStorage, and invokes fetchUser() to initialize the session.

- fetchUser(): Retrieves the current user's profile and role permissions from /users/me. It triggers an automatic logout if the session token is expired (401 Unauthorized).
- logout(): Clears the authentication token from storage, resets the global auth state, and wipes the localized bookmark cache.

Search Service (via useSearchStore & useTextSearch hook):

- searchDenseRegion(params): Executes high-density region comparisons via /similarity/search-dense-region, returning ranked slides accompanied by spatial heatmap data.
- setFilters(filters): Dynamically updates the active clinical metadata filters (e.g., organ, diagnosis, stain) used to narrow search results.
- setSearchFocus(focus): Provides a one-touch interface to toggle between optimized weight presets (e.g., "Visual-only", "Report-heavy", or "Balanced").
- clearResults(): Resets the search state, including ranked results, dense heatmaps, intent detection results, and active ROI selections.

Slide & Bookmark Service (via useBookmarkStore)

- loadBookmarks(): Retrieves the user's persistent slide bookmarks from the database.
- toggleBookmark(slideId): Performs an optimistic UI update and synchronizes the addition or removal of a slide bookmark with the backend repository.

4.2 Backend Subsystem

The backend subsystem is a FastAPI application. It acts as the single API gateway between the frontend and all data and ML services. Every request passes through CORS middleware. Authenticated endpoints use a dependency-injection layer to decode JWT Bearer tokens and enforce role-based access. The Authentication Service handles secure login using bcrypt-hashed records and session management.

The WSI and IIIF services manage slide listing, metadata, and high-speed tile serving. These services are enriched with CellViT-derived cellular metrics and heatmap overlays. A dedicated sharing service enables multi-user collaboration with granular annotation permissions and clinical notes.

The Search Engine orchestrates multimodal fusion and dense region retrieval. It leverages memory-mapped HDF5 embeddings and FAISS indices for sub-second similarity matching. The Initialization Service provides real-time health telemetry and readiness

probes. At startup, a background thread handles the loading of foundation models and index building. This background process ensures the HTTP server remains responsive while the ML layer initializes. Additionally, administrators can manage WSI root folder assignment and library synchronization through specialized backend routes.

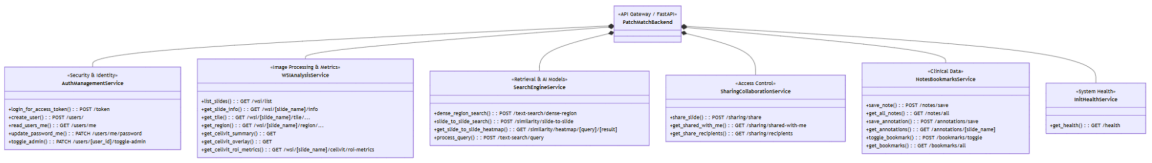


Figure 3. PatchMatch backend subsystem.

Methods

Auth & User Management Service:

- login_for_access_token (POST /token): Authenticates the user by verifying the bcrypt-hashed password against the database and returns a JWT access token.
- create_user (POST /users/): (Admin only) Persists a new user record with a hashed password and automatically provisions a private WSI subdirectory for the user based on their email.
- read_users_me (GET /users/me): Retrieves the currently authenticated user's profile.
- update_password_me (PATCH /users/me/password): Allows a user to securely update their own password.
- toggle_admin (PATCH /users/{user_id}/toggle-admin): (Admin only) Grants or revokes administrative privileges for a specific user.

WSI & Analysis Service:

- list_slides (GET /wsi/list): Returns all slides accessible to the user (private library + global root for admins), including precomputed availability flags for BRACS heatmaps, CellViT results, and sharing permissions.
- get_slide_info (GET /wsi/{slide_name}/info): Retrieves base dimensions and magnification metadata from the WSI using OpenSlide.
- get_tile (GET /wsi/{slide_name}/tile/{level}/{x}/{y}.jpg): Serves multi-resolution slide tiles dynamically, utilizing an LRU cache for performance.

- get_region (GET /wsi/{slide_name}/region/{x}/{y}/{w}/{h}.jpg): Extracts and serves an arbitrary ROI crop at the highest available resolution.
- get_cellvit_summary / get_cellvit_overlay: Retrieves automated cellular counts, standardized metrics (TIL and TSR), and spatial centroid overlays for a slide from precomputed msgpack/JSON files.
- get_cellvit_roi_metrics (GET /wsi/{slide_name}/cellvit/roi-metrics): Computes real-time cellular statistics (counts, densities, TIL/TSR) specifically bounded by a user-selected ROI.

Search Engine Service:

- dense_region_search (POST /text-search/dense-region): Performs a high-density, patch-by-patch region comparison (MaxSim), returning ranked slides along with localized similarity heatmaps.
- slide_to_slide_search (POST /similarity/slide-to-slide): Executes a global slide-level similarity search using aggregated slide embeddings.
- get_slide_to_slide_heatmap (GET /similarity/heatmap/{query}/{result}): Generates a spatial similarity heatmap explaining which regions of the result slide matched the query slide.
- process_query (POST /text-search/query): Automatically detects the intent behind a text query (clinical feature, visual concept, etc.) to optimize downstream retrieval logic.

Sharing & Collaboration Service:

- share_slide (POST /sharing/share): Shares a slide with another user via email, allowing the owner to set granular annotation access (view, comment, edit) and persisting it in the database.
- get_shared_with_me (GET /sharing/shared-with-me): Lists all slides shared with the current user, returning slide metadata and the respective permission level.
- get_share_recipients (GET /sharing/recipients): Lists all active users to populate the sharing dropdown menu.

Notes, Annotations & Bookmarks Service:

- save_note / get_all_notes (POST /notes/save, GET /notes/all): Persists and retrieves custom clinical notes, diagnostic labels, and regions of interest (with screenshots) for specific slides.

- save_annotation / get_annotations (POST /annotations/save, GET /annotations/{slide_name}): Manages persistent vector-based spatial annotations (boxes, points) drawn on the WSI for collaborative review.
- toggle_bookmark / get_bookmarks (POST /bookmarks/toggle, GET /bookmarks/all): Allows users to flag specific slides for quick access and retrieves the list of bookmarked slides.

Initialization & System Health Service:

- get_health (GET /health): Exposes real-time initialization progress, subsystem status (FAISS indexing, model loading, embedding ingestion), and hardware telemetry (CPU/Memory usage).

4.3 ML Subsystem

The ML Subsystem is the computational core responsible for all deep learning inference, embedding management, and similarity retrieval. The Encoder Service manages the lifecycle of foundation models (CONCH v1 and v1.5) and provides text and image encoding interfaces. The Search Engine Service builds FAISS inner-product indices from L2-normalized embeddings and executes similarity retrieval with adaptive percentile-based score stretching. The Intent Detection Service processes incoming queries through a cascading keyword → similarity → image probe pipeline to automatically select per-subclass optimal fusion weights. The Slide-to-Slide Retrieval Service provides whole-slide nearest-neighbor search using TITAN slide embeddings, along with cross-slide patch similarity heatmaps. The Region-to-Region Service handles dense ROI comparison by executing batched k-NN queries per patch and aggregating results via max-pool-then-mean to produce slide-level rankings with spatial heatmaps. The Region Classification Service combines a BRACS linear probe with a Quilt-1M semantic caption index to provide diagnostic context when inspecting tissue regions. The Report & Clinical Filter Service loads clinical report metadata, computes IDF-weighted keyword boosts, and extracts structured clinical entities (receptor statuses, grades) from free-text queries. The CellViT Analysis Service provides quantitative cellular-level tissue metrics by streaming pre-computed cell segmentation outputs and aggregating sufficient statistics in real time for user-selected regions.

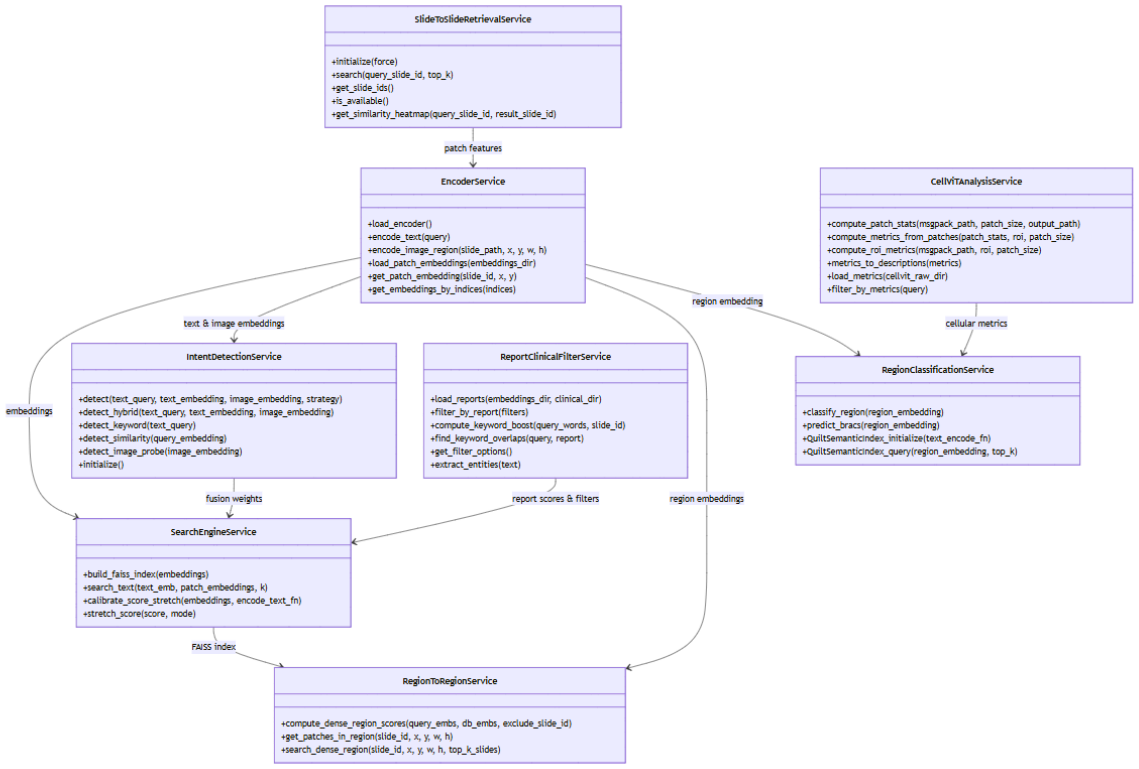


Figure 4. PatchMatch ML subsystem

Encoder Service:

- load_encoder(): Detects the available hardware accelerator (CUDA → MPS → CPU) and loads the configured encoder models. In the dual-encoder configuration, loads CONCH v1.5 (ViT-L + attentional pooler) as the image encoder and CONCH v1 (CoCa) as the text encoder; reuses the same model instance when both are set to the same type.
- encode_text(query): Tokenizes the input string using the CONCH tokenizer (max_length=128) and runs a forward pass through the text encoder, returning an L2-normalized embedding vector.
- encode_image_region(slide_path, x, y, w, h): Opens the WSI via OpenSlide, reads the specified region at approximately 20× magnification, tiles it into 224×224 patches, filters background tiles by mean-intensity thresholding, subsamples to 100 patches maximum, runs a batched forward pass through the image encoder, and returns a L2-normalized region embedding.
- load_patch_embeddings(embeddings_dir): Scans the embeddings directory for .npy and .h5 files, loads all patch embedding matrices into a single contiguous NumPy

array, parses the accompanying JSONL metadata, and builds the slide-to-patch index mapping for O(1) per-slide lookup.

- get_patch_embedding(slide_id, x, y): Retrieves the pre-computed embedding for a specific patch by matching slide ID and coordinates against the patch metadata index.
- get_embeddings_by_indices(indices): Returns the embedding sub-matrix for a list of global patch indices, used for mean-pool aggregation in multi-patch ROI queries.

Search Engine Service:

- build_faiss_index(embeddings): Constructs a FAISS IndexFlatIP (inner-product) index from the full patch embedding matrix. Falls back to NumPy matrix-multiply when the faiss-cpu package is unavailable.
- search_text(text_emb, patch_embeddings, k): Performs a single-vector FAISS search against the patch index; includes a dual-encoder dimension guard that returns empty results when text and patch embedding dimensions differ.
- calibrate_score_stretch(embeddings, encode_text_fn): Runs sample pathology queries (e.g., "invasive carcinoma", "normal breast tissue") against the full embedding matrix to compute percentile-based stretch boundaries, establishing the [lo, hi] mapping for the current dataset.
- stretch_score(score, mode): Maps a raw cosine similarity value to the [0, 1] range using the calibrated percentile boundaries, providing human-interpretable confidence scores.

Intent Detection Service:

- detect(text_query, text_embedding, image_embedding, strategy): Unified entry point that dispatches to the configured strategy (hybrid or combined) and returns the detected subclass, detection method, and confidence score.
- detect_hybrid(text_query, text_embedding, image_embedding): Cascading pipeline: keyword regex match → CONCH canonical similarity → BRACS image probe → default weights.
- detect_keyword(text_query): Matches the query against compiled regex patterns for six breast cancer subtypes (IDC, ILC, DCIS, mucinous, papillary, metaplastic). Returns the matched subclass with confidence 1.0.
- detect_similarity(query_embedding): Computes cosine similarity between the query embedding and pre-computed canonical description embeddings for each subclass. Returns the best match if similarity exceeds the configurable threshold (default 0.25).

- detect_image_probe(image_embedding): Classifies the image patch using the BRACS linear probe, maps the predicted BRACS class (IC, DCIS, ADH) to a cancer subclass via a lookup table, and returns the subclass if probe confidence exceeds 0.6.
- initialize(): Pre-computes canonical description embeddings by encoding each subclass's textual description through CONCH. Called once after the text encoder is loaded.

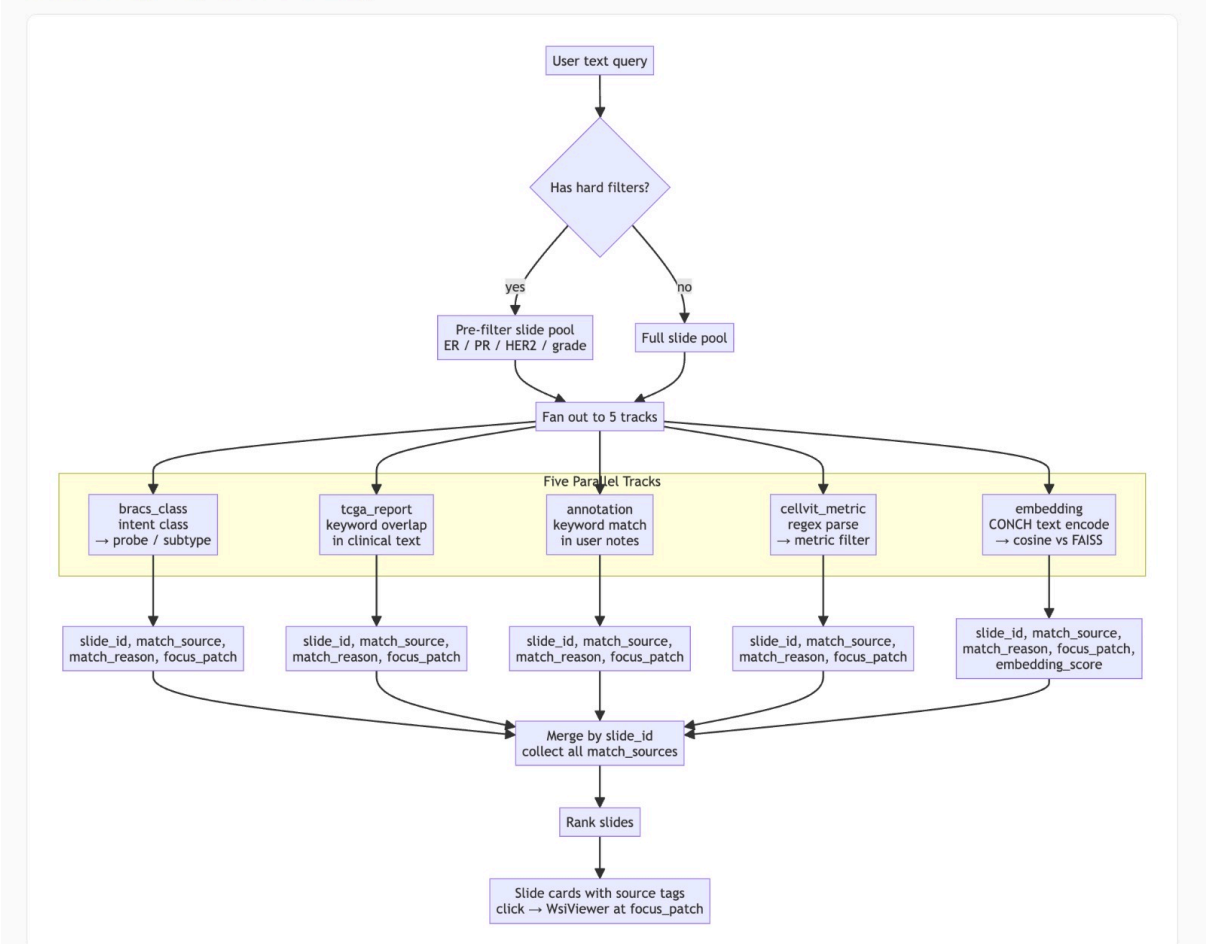


Figure 5. Text Search Pipeline

Slide-to-Slide Retrieval Service:

- initialize(force): Loads all TITAN slide embeddings (_slide.npy) and per-patch attention weights (_attn_weights.npy) from the slide embeddings directory, stacks them into a matrix, L2-normalizes, and builds a FAISS IndexFlatIP index.

- search(query_slide_id, top_k): Retrieves the query slide's embedding from the index and performs a single-vector FAISS search, returning the top-k most similar slides with cosine similarity scores.
- get_slide_ids(): Returns the list of all slide IDs currently indexed for retrieval.
- is_available(): Returns whether TITAN embeddings have been loaded and the FAISS index is ready.
- get_similarity_heatmap(query_slide_id, result_slide_id): Loads per-patch features from H5 files for both slides, computes a dense cross-slide patch similarity matrix, aggregates hit counts and scores per viewed patch, filters by minimum-hit thresholds, and returns stretched similarity values for spatial heatmap visualization.

Region-to-Region Service:

- compute_dense_region_scores(query_embeddings, db_embeddings, exclude_slide_id, top_k_slides, top_m_patches): Executes batched k-NN (k=2000) for each query patch against the FAISS index, aggregates via max-pool-then-mean to produce slide-level scores, and generates per-patch similarity heatmaps with stretched values for spatial visualization.
- get_patches_in_region(slide_id, x, y, w, h): Identifies all patch indices belonging to the given slide that fall within the specified ROI bounding box, used to extract the query embedding set for a region search.
- search_dense_region(slide_id, x, y, w, h, top_k_slides, top_m_patches): End-to-end dense region retrieval pipeline: resolves the slide ID, extracts query patch indices within the ROI, caps the query set to the configured maximum (default 500) with deterministic subsampling, retrieves the query embeddings, and delegates to compute_dense_region_scores for ranked results with per-slide heatmaps.

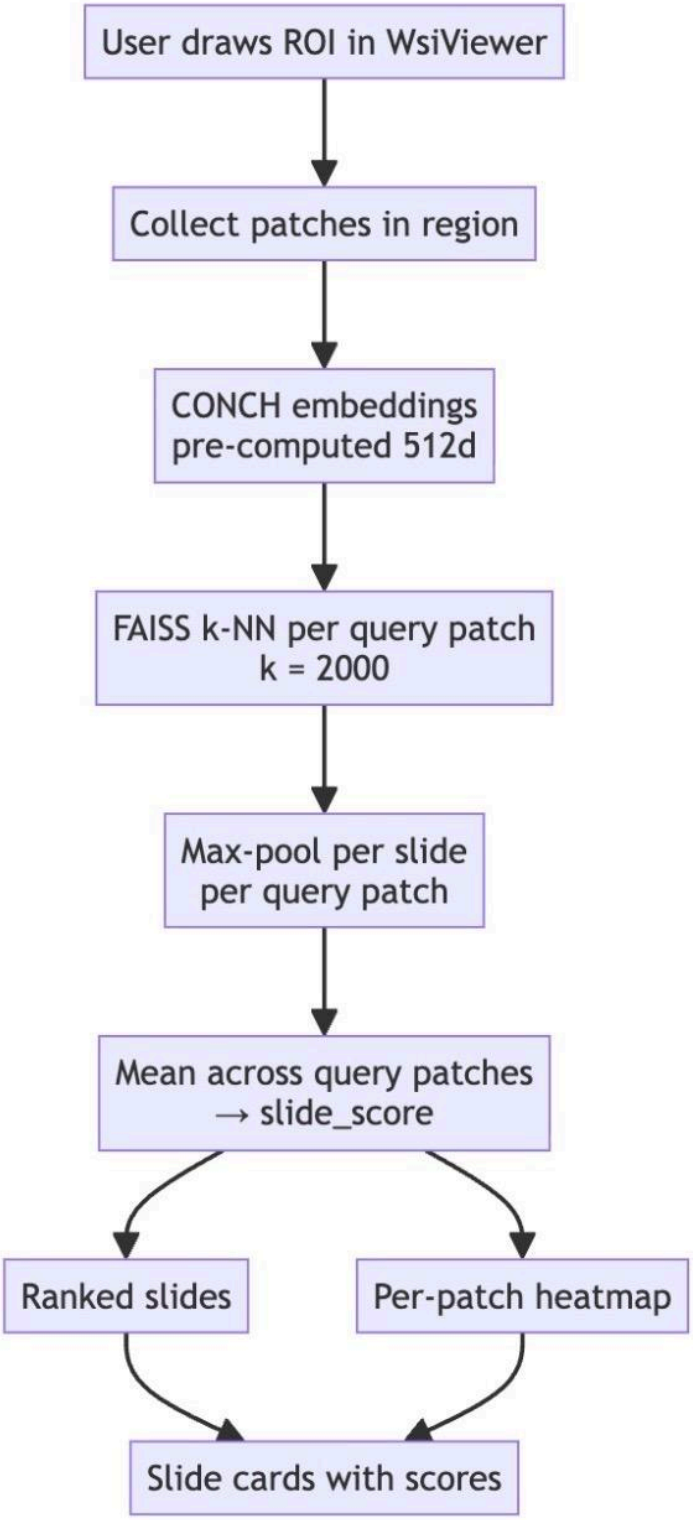


Figure 6. Region Search Pipeline

Region Classification Service:

- classify_region(region_embedding): Applies two complementary classification methods to a region embedding and returns a structured result combining BRACS class probabilities with semantic text descriptions.
- predict_bracs(region_embedding): Runs the BRACS linear probe (logistic regression trained on BRACS dataset embeddings) on the region embedding, returning the predicted class label (Normal, Pathological Benign, UDH, FEA, ADH, DCIS, or Invasive Carcinoma), a confidence score, and the full per-class probability distribution.
- QuiltSemanticIndex.initialize(text_encode_fn): Extracts up to 200 diverse breast pathology captions from the Quilt-1M dataset (filtered by keyword relevance and prefix-based deduplication, supplemented with 40 hand-crafted fallback descriptors), encodes each via the text encoder, and builds an L2-normalized caption embedding matrix.
- QuiltSemanticIndex.query(region_embedding, top_k): Computes cosine similarity between the region embedding and all caption embeddings, returning the top-k most semantically similar natural-language descriptions with their similarity scores.

Report & Clinical Filter Service:

- load_reports(embeddings_dir, clinical_dir): Loads clinical report JSON files and their corresponding report embeddings (.npy), merges BRACS metadata and probe classification labels into the report index, and computes IDF weights across all report terms.
- filter_by_report(filters): Returns the set of slide IDs whose clinical reports match the given filter criteria via case-insensitive field comparison.
- compute_keyword_boost(query_words, slide_id): Computes an IDF-weighted keyword overlap boost for a slide's clinical report against the query terms, used as an additive component in the report similarity channel (γ).
- find_keyword_overlaps(query, report): Identifies query terms that appear in the clinical report using bigram-first matching and clinical term grouping (e.g., "invasive ductal carcinoma" is recognized as a single term rather than three separate word matches).
- get_filter_options(): Returns available filter values and their occurrence counts from all loaded reports, driven by the modular filter schema configuration.

- extract_entities(text): Extracts structured clinical filters from free-text queries using regex patterns and dictionary lookups against a curated clinical vocabulary covering subtypes (IDC, ILC, DCIS), grades (I–III), and receptor statuses (ER, PR, HER2).

CellViT Analysis Service:

- compute_patch_stats(msgpack_path, patch_size, output_path): Offline precomputation step that streams the CellViT cells.msgpack file, bins each cell into a spatial patch grid by centroid coordinates, accumulates sufficient statistics (cell type counts and circularity sums per patch), and serializes the result to a pickle file for O(1) retrieval at query time.
- compute_metrics_from_patches(patch_stats, roi, patch_size): Real-time ROI querying step that retrieves only the precomputed patch statistics overlapping the ROI bounding box, aggregates via summation of sufficient statistics, and returns tumor cellularity, TIL ratio, TSR, mean circularity, along with density and circularity heatmap patches for spatial visualization.
- compute_roi_metrics(msgpack_path, roi, patch_size): Streaming alternative that reads the full msgpack file and filters cells by ROI bounds in real time. Used when precomputed patch statistics are unavailable; returns identical metric structure including individual tumor cell centroids and circularity values for overlay rendering.
- metrics_to_descriptions(metrics): Converts numeric CellViT metrics into natural-language descriptions based on clinical thresholds (e.g., tumor cellularity > 70% → "High tumor cellularity", TIL ratio > 30% → "Dense lymphocytic infiltrate").
- load_metrics(cellvit_raw_dir): Scans the CellViT output directory for per-slide metrics.json files and loads them into an in-memory dictionary for slide-level metric filtering and search track matching.
- filter_by_metrics(query): Parses one or more metric condition expressions from the query string (e.g., "til > 0.05", "total cells < 5000") and returns the set of slide IDs whose metrics satisfy all conditions. Supports operators >, <, >=, <=, ==, != and percentage notation.

Retrieval Evaluation Metrics

Precision@K (P@K) measures the fraction of the top-K retrieved slides that share the query's diagnostic label. For each query slide *i* with label *L_i*, the gallery is ranked by cosine similarity and the top-K results are inspected:

$$P@K_i = \frac{|\{j \in \text{top-}K : \text{label}(j) = L_i\}|}{K}$$

The reported value is the mean over all N queries. P@1 is equivalent to top-1 accuracy — the probability that the single best match is correct.

Mean Reciprocal Rank (MRR) captures how highly the *first* correct result is ranked. For query *i*, the reciprocal rank is 1/r where r is the 1-indexed position of the first correct hit:

$$MRR = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{\text{rank of first correct result}_i}$$

MRR = 1.0 means the top result is always correct; MRR = 0.5 means the first correct result appears at rank 2 on average.

Majority Vote Accuracy at 5 (MV@5) predicts the query label by majority vote over the top-5 retrieved slides (ties broken by the highest-ranked slide) and measures how often this prediction is correct. It is the primary metric for comparison with prior work.

Random baseline is computed by Monte Carlo simulation (30 random shuffles per query) under the BRACS class distribution, giving the expected performance of a system with no slide content knowledge.

Results

Evaluation follows leave-one-out protocol over all 546 BRACS WSIs across seven diagnostic classes: Normal, PB, UDH, FEA, ADH, DCIS, IC.

Table 1. Slide-to-slide retrieval on BRACS (N = 546, leave-one-out).

System	P@1	P@5	P@10	MRR	MV@5
Random baseline	17.5%	17.9%	17.9%	36.2%	19.6%
TITAN (raw cosine)	31.9%	28.0%	26.5%	48.1%	36.1%
CONCH v1.5 + mean pool	48.0%	41.9%	39.3%	62.1%	48.5%
CONCH v1.5 + top-20% patches	48.0%	41.8%	38.9%	61.8%	48.9%
CONCH v1.5 + adaptive patch pool	45.8%	41.2%	39.3%	60.0%	47.3%
Mean pool ⊕ TITAN (concat)	46.5%	42.3%	39.0%	61.1%	50.5%
CONCH v1.5 + mean pool + PCA-128 ★	**47.3%**	**40.8%**	**35.5%**	**61.7%**	**51.5%**

Table 2. Per-class retrieval performance — CONCH v1.5 + mean pool + PCA-128.

Class	N	P@1	P@5	MRR
IC	**132**	**82.58%**	**67.73%**	**0.880**
PB	147	48.98%	46.39%	0.683
DCIS	61	39.34%	31.80%	0.524
Normal	44	31.82%	20.45%	0.464
FEA	41	29.27%	24.39%	0.457
ADH	48	25.00%	18.33%	0.392
UDH	73	20.55%	24.93%	0.418

All systems substantially exceed the random baseline across every metric, confirming that embedding-based retrieval carries genuine diagnostic signal. PCA whitening applied to mean-pooled features yields the best overall result (MV@5 = 51.5%, P@1 = 47.3%, MRR = 61.7%), as it suppresses dominant stroma and background variance shared across all classes and equalises the contribution of remaining dimensions.

Per-class results follow the known difficulty ordering of BRACS. IC retrieves well (P@1 = 82.6%) due to morphologically distinct features, stromal invasion and marked nuclear atypia, while the atypical classes UDH, ADH, and FEA score poorly (P@1 = 20.6%, 25.0%, 29.3%). This reflects genuine histological overlap: these

three classes share ductal proliferation patterns and differ only in subtle quantitative cytological criteria, a distinction that a single global vector inevitably averages away.

4.4 Sequence and Activity Diagrams

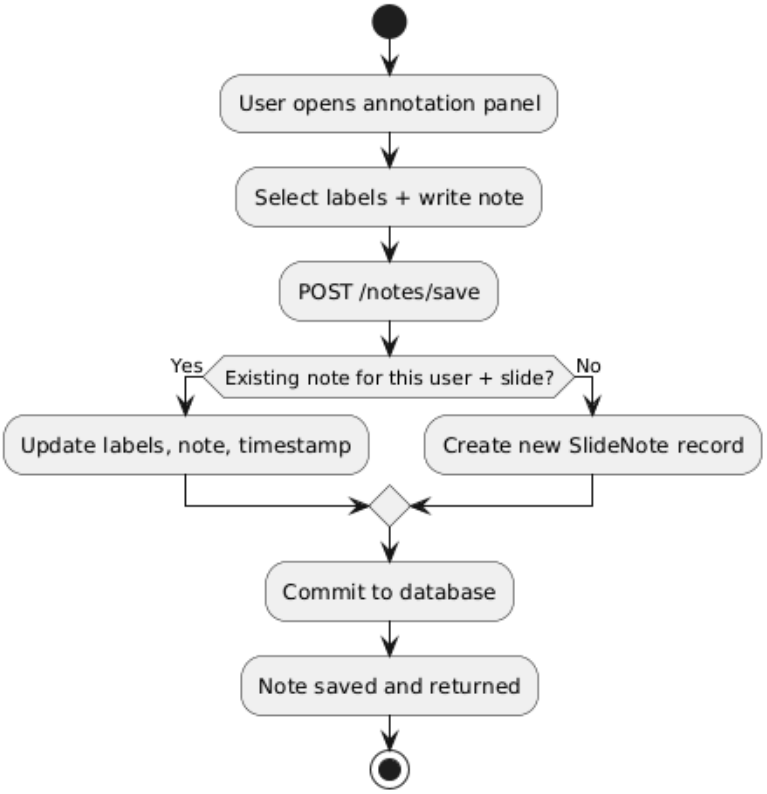


Figure 7. Slide annotation activity diagram

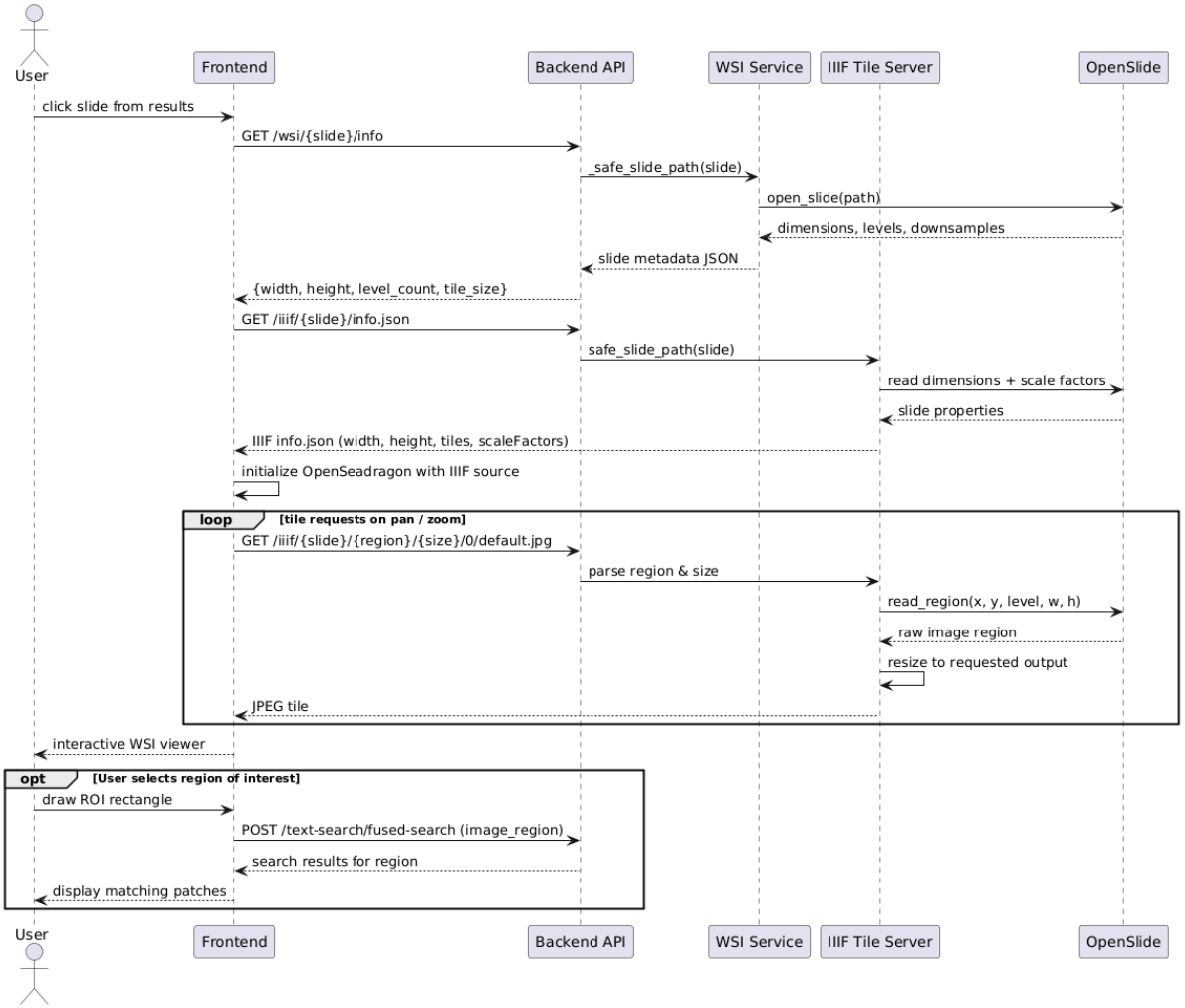


Figure 8. View WSI sequence diagram

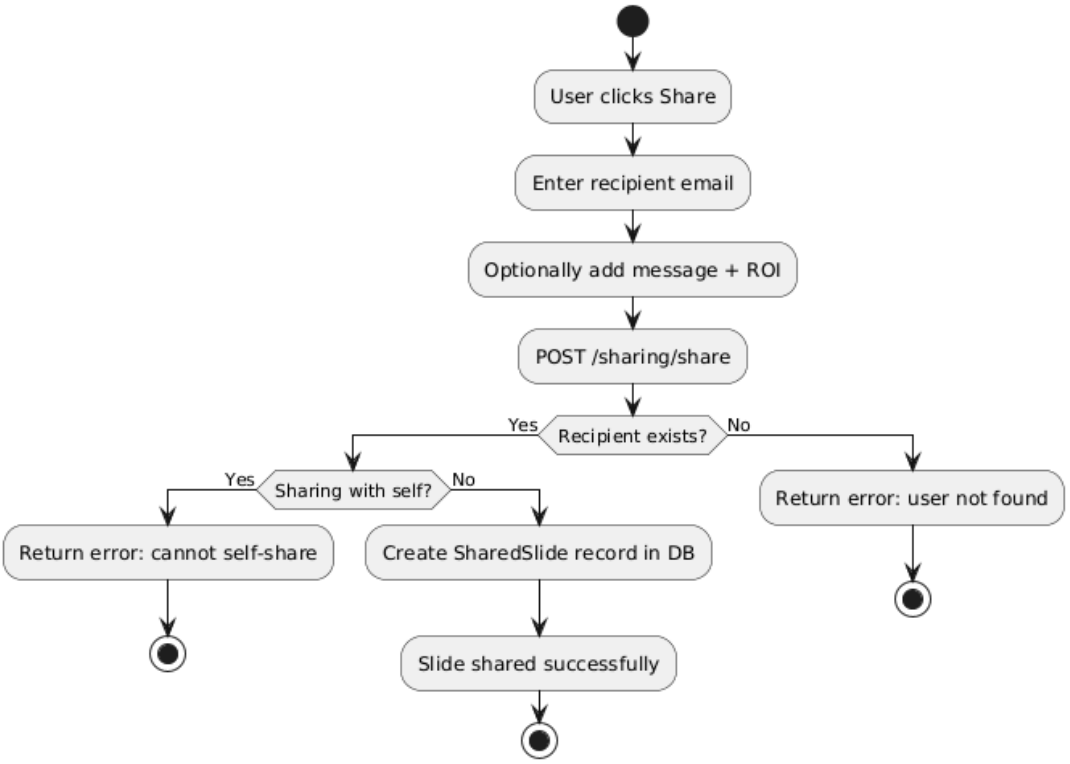


Figure 9. Slide sharing activity diagram.

5. Test Cases and Results

T-1: Signing In Successfully

Test ID	T-1	Category	Functional	Severity	Critical
Objective	Verify that a user can sign in with valid credentials				
Steps	1. Navigate to the sign in page 2. Enter a valid email and a password 3. Click the Sign In button				
Expected	• Session cookie or token is set • User is directed to the home page				
Date-Result	28.04.2026 - Passed				

T-2: Signing In with Invalid Credentials

Test ID	T-2	Category	Functional	Severity	High
Objective	Verify that signing in fails if the credentials are invalid				
Steps	1. Navigate to the sign in page 2. Enter an invalid email or a password 3. Click the Sign In button				
Expected	● Display an error message indicating that the credentials are wrong ● No session or token is created ● User is not directed to the home page				
Date-Result	28.04.2026 - Passed				

T-3: Signing In with Missing Field

Test ID	T-3	Category	Functional	Severity	Medium
Objective	Verify that signing in fails if the credentials are invalid				
Steps	1. Navigate to the sign in page 2. Enter an invalid email or a password 3. Click the Sign In button				
Expected	● Display an error message indicating that the credentials are wrong ● No session or token is created ● User is not directed to the home page				

Date-Result	28.04.2026 - Passed

T-4: Signing In with Invalid Email Format

Test ID	T-4	Category	Functional	Severity	Low
Objective	Verify that signing in fails and triggers a validation error when the email entered is missing the '@' symbol and domain.				
Steps	1. Navigate to the sign in page 2. Enter an email string missing the '@' symbol 3. Enter any password 4. Click the Sign In button				
Expected	● A frontend validation error message "Please include an '@' in the email address" is displayed ● The form does not submit to the server ● No session or token is created ● User remains on the sign in page				
Date-Result	28.04.2026 - Passed				

T-5: Viewing a WSI

Test ID	T-5	Category	Functional	Severity	Critical
Objective	Verify that a user can successfully open and view the base layer of a selected WSI from the library				
Steps	1. Log into the system and navigate to the WSI Library 2. Select a valid, fully processed WSI thumbnail 3. Click to open the slide in the main viewer interface				

Expected	• The viewer interface launches without errors • The lowest magnification level (base layer) of the WSI renders fully on the screen • Default viewer controls (zoom, pan, toggle annotations) become active and visible
Date-Result	28.04.2026 - Passed

T-6: Panning Across WSI

Test ID	T-6	Category	Functional	Severity	High
Objective	Verify that the user can smoothly pan across the WSI to view different tissue regions at the current magnification				
Steps	1. Open a WSI in the main viewer 2. Click and hold the left mouse button on the image 3. Drag the cursor in multiple directions to pan across the slide 4. Release the mouse button				
Expected	• The image tracks accurately with the cursor movement without snapping back • Panning stops immediately when the mouse button is released				
Date-Result	28.04.2026 - Passed				

T-7: Valid ROI Selection

Test ID	T-7	Category	Functional	Severity	Critical

Objective	Verify that a user can successfully draw and lock a bounding box over a specific region of the WSI
Steps	1. Log into the application with valid credentials 2. Open a WSI in the viewer 3. Select the ROI mode 4. Click and drag the cursor over a specific tissue area to draw an ROI 5. Release the cursor
Expected	● A visible bounding box is accurately drawn where the user dragged the cursor ● The selected ROI coordinates are captured by the system for querying
Date-Result	28.04.2026 - Passed

T-8: Out-of-Bounds ROI Selection

Test ID	T-8	Category	Functional	Severity	Medium
Objective	Verify that attempting to select an ROI on a blank background or outside slide boundaries prevents an invalid query				
Steps	1. Open a WSI in the viewer. 2. Attempt to draw a bounding box completely outside the tissue area 3. Click the "Find Similar Slides" button				
Expected	● The system prevents the query execution ● A helpful validation message "Please select an ROI containing tissue" is displayed ● No blank image queries are sent to the backend				
Date-Result	28.04.2026 - Passed				

T-9: Increasing Magnification (Zooming In)

Test ID	T-9	Category	Functional	Severity	High
Objective	Verify that increasing the magnification successfully loads and renders the higher-resolution WSI tiles for a targeted tissue area				
Steps	1. Open a WSI in the main viewer and locate a specific tissue structure at the base magnification 2. Position the cursor over the structure 3. Use the viewer controls to increase the magnification level to a high power				
Expected	● The viewer successfully zooms into the targeted spatial coordinates ● The system fetches and renders the high-resolution image tiles for that specific area without remaining permanently blurry or pixelated ● The UI's magnification indicator correctly updates to reflect the new zoom level				
Date-Result	28.04.2026 - Passed				

T-10: Decreasing Magnification (Zooming Out)

Test ID	T-10	Category	Functional	Severity	High
Objective	Verify that decreasing the magnification smoothly zooms out to reveal the broader tissue context without losing spatial orientation				
Steps	1. Open a WSI and zoom in to a high magnification level on a specific area 2. Use the viewer controls to decrease the magnification back to the base layer				
Expected	● The viewer smoothly zooms out to display the broader area of the slide ● The system successfully unloads the high-resolution tiles and renders the				

	lower-resolution overview tiles ● The UI's magnification indicator correctly updates to reflect the new zoom level
Date-Result	28.04.2026 - Passed

T-11: Annotating Slides

Test ID	T-11	Category	Functional	Severity	High
Objective	Verify that a Pathologist can draw and save a new annotation on a WSI				
Steps	1. Open WSI in the viewer 2. Select one of the selection modes 3. Draw a shape around a tissue or a specific group of cells 4. Enter text in the annotation label field and click "Save"				
Expected	● The drawn annotation remains persistently visible on the slide ● The label is correctly associated with the shape ● The annotation data is saved to the database				
Date-Result	28.04.2026 - Passed				

T-12: Displaying Annotations

Test ID	T-12	Category	Functional	Severity	High
Objective	Verify that clicking the "View Annotations" toggle to the "ON" position allows a user to display all annotations associated with a slide				
Steps	1. Open a WSI containing one or more existing annotations				

	2. Ensure the "View Annotations" toggle is in the "OFF" position and annotations are currently hidden 3. Click the "View Annotations" toggle in the viewer toolbar 4. Select the “Pan” mode and zoom across the WSI to inspect the annotated areas 5. Click on the annotated areas to view the associated text labels
Expected	● All saved annotation shapes become visible and overlay correctly on the WSI. ● The annotations accurately align with the specific tissue coordinates they were originally drawn on, regardless of the zoom level. ● Associated text labels are displayed correctly when an annotated shape is selected
Date-Result	28.04.2026 - Passed

T-13: Hiding Annotations

Test ID	T-13	Category	Functional	Severity	Medium
Objective	Verify that clicking the "View Annotations" toggle to the "OFF" position allows a user to hide all annotations associated with a slide				
Steps	1. Open a WSI containing one or more existing annotations 2. Ensure the "View Annotations" toggle is in the "ON" position and annotations are currently visible 3. Click the "View Annotations" toggle in the viewer toolbar 4. Select the “Pan” mode and zoom across the WSI to inspect the annotated areas				
Expected	● All saved annotation shapes and associated text labels become hidden on the WSI ● The underlying whole-slide image is completely unobstructed ● Panning or zooming the image does not cause the hidden annotations to reappear				

Date-Result	28.04.2026 - Passed

T-14: Editing Shape of an Annotation

Test ID	T-14	Category	Functional	Severity	Medium
Objective	Verify that the "Edit Annotations" extension allows a user to modify the shape and boundaries of an existing saved annotation.				
Steps	1. Open a WSI containing an existing annotation 2. Ensure the "View Annotations" toggle is in the "ON" position and the annotation is currently visible 3. Select the existing annotation shape on the slide 4. Drag the nodes to alter the shape's boundaries 5. Click "Save"				
Expected	● The system successfully updates the geometric coordinates of the shape ● The new shape boundaries are accurately displayed on the WSI ● The changes persist after refreshing the page				
Date-Result	28.04.2026 - Passed				

T-15: Editing Label of an Annotation

Test ID	T-15	Category	Functional	Severity	Low
Objective	Verify that the "Edit Annotations" extension allows a user to modify the text label of an existing saved annotation.				
Steps	1. Open a WSI containing an existing annotation 2. Ensure the "View Annotations" toggle is in the "ON" position and the				

	annotation is currently visible 3. Select the existing annotation on the slide to reveal its current text label 4. Change the text label to a new value 5. Click "Save"
Expected	● The system successfully updates the text label associated with the annotation ● The new label is accurately displayed when the annotated shape is selected ● The geometric coordinates of the shape remain completely unchanged ● The changes persist after refreshing the page
Date-Result	28.04.2026 - Passed

T-16: Sharing a WSI

Test ID	T-16	Category	Functional	Severity	Medium
Objective	Verify that the "Edit Annotations" extension allows a user to modify the text label of an existing saved annotation.				
Steps	1. Navigate to the Library. 2. Open a WSI and click the "Share" button 3. Enter the exact email of an existing registered user 4. Click "Send" button				
Expected	● A success confirmation is displayed ● The receiving user sees the shared WSI in their "Shared with Me" tab ● Both users can access the same WSI record				
Date-Result	29.04.2026 - Passed				

T-17: Sharing a WSI with Invalid Email

Test ID	T-17	Category	Functional	Severity	Low
Objective	Verify that the system handles sharing attempts with non-existent users gracefully				
Steps	1. Navigate to the Library. 2. Open a WSI and click the "Share" button 3. Enter an email address that is not registered in the system 4. Click "Send" button				
Expected	● The system prevents the share action ● An error message is displayed				
Date-Result	29.04.2026 - Passed				

T-18: Content-based Image Retrieval

Test ID	T-18	Category	Functional	Severity	Critical
Objective	Verify that submitting an ROI successfully queries the database and returns visually similar case thumbnails				
Steps	1. Define a valid ROI on the WSI 2. Click the "Find Similar Slides" button				
Expected	● The system processes the image query successfully ● Visually similar WSI patches/thumbnails are displayed in the “Retrieval Results” panel.				
Date-Result	29.04.2026 - Passed				

T-19: Valid Text-based Image Retrieval

Test ID	T-19	Category	Functional	Severity	Critical
Objective	Verify that entering clinical keywords into the search bar successfully returns relevant results				
Steps	1. Navigate to the Library 2. Enter a valid clinical keyword into the text search bar 3. Click the "Search" button				
Expected	● The system processes the text query successfully ● Slides matching the provided clinical keyword are retrieved ● Slides are displayed correctly in the Library				
Date-Result	29.04.2026 - Passed				

T-20: Invalid Text-based Image Retrieval

Test ID	T-20	Category	Functional	Severity	Medium
Objective	Verify that entering unsupported characters or invalid text queries is handled gracefully				
Steps	1. Navigate to the Library 2. Enter a string of special characters (e.g., *&^%$#) into the text search bar 3. Click the "Search" button				
Expected	● The system does not crash or throw an unhandled exception ● Specific format validation message is displayed to the user				
Date-Result	29.04.2026 - Not passed				

T-21: Viewing Retrieved Results

Test ID	T-21	Category	Functional	Severity	Critical
Objective	Verify that the retrieval results interface correctly displays the returned WSI patches along with their similarity scores and clinical metadata				
Steps	1. Execute a successful search query 2. Wait for the retrieval process to complete 3. Inspect the loaded results grid on the dashboard				
Expected	● The system displays a list of the retrieved WSI patch thumbnails in the “Retrieval Results” panel ● Each thumbnail clearly displays its similarity score ● Key clinical metadata is visible alongside or below each result				
Date-Result	29.04.2026 - Passed				

T-22: Opening Retrieved Image

Test ID	T-22	Category	Functional	Severity	Critical
Objective	Verify that clicking a specific retrieval result successfully navigates the user to the full WSI view at the exact matched coordinates				
Steps	1. Execute a search query and wait for the results to load 2. Click on the thumbnail of a specific retrieved result 3. Observe the system's navigation and the resulting viewer state				
Expected	● The system seamlessly transitions to the main WSI viewer ● The viewer automatically loads the correct Whole-Slide Image associated with that result ● The viewer automatically pans and zooms to the exact tissue coordinates (ROI) that matched the search query				

Date-Result	30.04.2026 - Passed

T-23: Fetching Associated Reports Successfully

Test ID	T-23	Category	Functional	Severity	High
Objective	Verify that retrieving WSIs via text search successfully fetches and displays the associated clinical reports for the results				
Steps	1. Navigate to the Search dashboard and select the "Text Search" mode 2. Enter a valid clinical keyword and click "Search" 3. Select one of the retrieved WSI thumbnails from the results list 4. Open or expand the "Associated Reports" panel for that specific WSI				
Expected	● The system successfully queries the database for the selected WSI ● The associated clinical report text is fetched and displayed correctly in the UI panel ● The report data accurately matches the selected WSI case				
Date-Result	30.04.2026 - Passed				

T-24: Uploading Single WSI

Test ID	T-24	Category	Functional	Severity	Critical
Objective	Verify that a System Administrator can successfully upload a single supported WSI file format				
Steps	1. Log in using System Administrator credentials. 2. Navigate to the "Library" page. 3. Click the "Inspect Folders" button in the top navigation bar.				

	4. Review the "WSI Images Folder Configuration" modal. 5. Click on the "Loose Slides", "User Folders", and "Non-Empty Folders" panels to view folder statistics. 6. Verify the "Active root" path displayed at the bottom.
Expected	● The individual file uploads successfully without timing out ● A success message is displayed to the user confirming the single upload ● The newly uploaded WSI appears correctly in the associated user's Library panel
Date-Result	30.04.2026 - Passed

T-25: Extracting Valid Tissue Patches

Test ID	T-26	Category	Functional	Severity	
Objective	Verify that the system successfully extracts and tiles valid tissue patches from a newly uploaded WSI while discarding the blank background				
Steps	1. Trigger the patch extraction service (via API or background job) for a valid WSI file 2. Monitor the extraction pipeline logs 3. Inspect the output directory or database for the generated patches				
Expected	● The service successfully divides the WSI into smaller tiles ● The background filtering algorithm successfully discards pure white/blank glass tiles ● Only tiles containing actual tissue are passed to the next stage of the pipeline				
Date-Result	30.04.2026 - Passed				

T-26: Handling Corrupted WSI Patch Extraction

Test ID	T-27	Category	Functional	Severity	High
Objective	Verify that the extraction service handles corrupted or fully blank WSI files gracefully without crashing the pipeline				
Steps	1. Trigger the patch extraction service (via API or background job) for a completely blank WSI or a corrupted file 2. Monitor the extraction pipeline logs 3. Check the status of the background worker				
Expected	● The service detects the lack of valid tissue or the file corruption ● It logs an appropriate error or warning message ● The background worker safely terminates the job for that specific file without crashing the entire service				
Date-Result	30.04.2026 - Passed				

T-27: Computing Patch Embeddings

Test ID	T-28	Category	Functional	Severity	Critical
Objective	Verify that the CONCH model successfully generates numerical feature vectors (embeddings) for a batch of tissue patches.				
Steps	1. Pass a known batch of extracted WSI patches to the embedding computation endpoint 2. Await the response from the CONCH model 3. Inspect the resulting data payload				
Expected	● The model returns a complete set of numerical vectors ● The number of output vectors perfectly matches the number of input patches ● The vectors contain valid floating-point numbers				

Date-Result	30.04.2026 - Passed

T-28: Memory Management in Computing Embeddings

Test ID	T-29	Category	Functional	Severity	High
Objective	Verify that the embedding service manages memory correctly when fed an extremely large WSI containing thousands of patches				
Steps	1. Pass an exceptionally large dataset of patches to the embedding service 2. Monitor the GPU/RAM usage on the processing server				
Expected	● The service processes the patches in manageable batches (chunking) rather than loading everything into memory at once ● The server does not throw an Out-of-Memory error ● All patches are eventually processed successfully				
Date-Result	30.04.2026 - Passed				

T-29: Storing Embeddings

Test ID	T-30	Category	Functional	Severity	Critical
Objective	Verify that computed embeddings and their associated metadata are correctly saved to the local storage				
Steps	1. Trigger the storage function with a payload of computed embeddings, patch coordinates, and the parent WSI ID 2. Query the database directly for that specific WSI ID 3. Retrieve the stored records				

Expected	• The storage successfully inserts the records without schema validation errors • The queried data exactly matches the payload sent (embeddings, X/Y coordinates, and WSI ID map correctly)
Date-Result	30.04.2026 - Passed

T-30: Handling Duplicate Embedding Insertions

Test ID	T-31	Category	Functional	Severity	Critical
Objective	Verify that the storage service prevents or safely handles duplicate embedding insertions for the same WSI				
Steps	1. Identify a WSI ID that already has embeddings stored in the database. 2. Re-trigger the storage payload for that exact same WSI ID and embedding set 3. Check the database row count for that WSI ID				
Expected	• The system detects the duplicate data • It rejects the new insertion with a "Duplicate entry" log • The database does not bloat with duplicate rows for the same spatial coordinates				
Date-Result	30.04.2026 - Passed				

T-31: Updating Similarity Index

Test ID	T-32	Category	Functional	Severity	Critical
Objective	Verify that new embeddings are successfully added to the similarity index (FAISS) making them retrievable in search.				

Steps	1. Ensure a new set of embeddings has just been stored 2. Trigger the "Update Similarity Index" job 3. Perform a region-of-interest (ROI) search from the frontend using a patch from the newly indexed WSI
Expected	● The indexing job completes successfully without errors ● The frontend search retrieves the newly indexed WSI patch in the top results ● This confirms the index mapping is perfectly synced with the database
Date-Result	30.04.2026 - Passed

T-32: Updating Similarity Index without Crashing

Test ID	T-33	Category	Functional	Severity	Critical
Objective	Verify that an ongoing index update does not block or crash active search queries from pathologists				
Steps	1. Initiate a large "Update Similarity Index" job in the background 2. While the index is actively building/updating, execute multiple image retrieval searches from the frontend UI 3. Observe the frontend search results and response times				
Expected	● The frontend searches execute successfully against the old index without throwing a timeout or connection error ● The background index update finishes seamlessly ● The system hot-swaps to the new index without dropping user traffic				
Date-Result	30.04.2026 - Passed				

T-33: Registering User with Valid Credentials

Test ID	T-34	Category	Functional	Severity	Critical
Objective	Verify that a System Administrator can successfully register a new user with valid credentials.				
Steps	1. Log in using System Administrator credentials 2. Navigate to the "Admin" page 3. Click the “Register User” button 4. Enter the name of the user 5. Enter a valid, unique email address 6. Enter a secure password that meets system requirements 7. Click the "Create Account" button				
Expected	● The system successfully creates the new user account ● A success message is displayed on the screen ● The new user is visible in the active users list ● The new user can successfully log into the system with the provided credentials				
Date-Result	30.04.2026 - Passed				

T-34: Registering User with Existing Email

Test ID	T-35	Category	Functional	Severity	High
Objective	Verify that the system prevents the registration of a new user if the provided email address already exists in the database.				
Steps	1. Log in using System Administrator credentials 2. Navigate to the "Admin" page 3. Click the “Register User” button 4. Enter the name of the user 5. Enter an email address that is already registered to an existing user 6. Enter a valid password				

	7. Click the "Create Account" button
Expected	● The system rejects the registration attempt ● A validation error message is displayed ● No duplicate account is created in the database
Date-Result	30.04.2026 - Passed

T-35: Changing Password

Test ID	T-36	Category	Functional	Severity	High
Objective	Verify that the System Administrator can successfully update the password for an existing user account.				
Steps	1. Log in using System Administrator credentials 2. Navigate to the "Admin" page 3. Click the "Change Password" button in one of the “Registered Users” table entries. 4. Enter a new, valid password and confirm it. 5. Click "Update Password" 6. Attempt to log in as that user with the old password, then the new password				
Expected	● The system successfully updates the user's password ● Logging in with the old password fails ● Logging in with the new password succeeds ● If the user was currently logged in during the change, their active session is terminated				
Date-Result	30.04.2026 - Passed				

T-36: Granting Admin

Test ID	T-37	Category	Functional	Severity	High
Objective	Verify that a System Administrator can successfully grant admin access to a standard user				
Steps	1. Log in using System Administrator credentials 2. Navigate to the "Admin" page 3. Locate a standard user in the "Registered Users" table 4. Click the "Grant Admin" button for that user 5. Log out of the current Admin account 6. Log in as the newly modified user and check for access to the "Admin" panel				
Expected	● The system successfully updates the user's role to Administrator in the database ● The modified user can now view the "Admin" tab in the navigation menu ● The modified user can successfully access and interact with the administrative functions on the Admin page				
Date-Result	30.04.2026 - Passed				

T-37: Revoking Admin

Test ID	T-38	Category	Functional	Severity	High
Objective	Verify that a System Administrator can successfully revoke admin access from an existing administrator account				
Steps	1. Log in using System Administrator credentials 2. Navigate to the "Admin" page 3. Locate another user who currently has Admin privileges in the "Registered Users" table 4. Click the "Revoke Admin" button for that user 5. Log out of the current Admin account				

	6. Log in as the modified user and attempt to access the "Admin" panel
Expected	● The system successfully reverts the user's role back to a standard user ● The "Admin" tab is no longer visible in the modified user's navigation menu ● Attempting to manually navigate to the Admin page URL redirects the user or displays an "Access Denied" message
Date-Result	30.04.2026 - Passed

T-38: Removing User

Test ID	T-39	Category	Functional	Severity	High
Objective	Verify that the System Administrator can successfully remove an existing user from the system				
Steps	1. Log in using System Administrator credentials 2. Navigate to the "Admin" page 3. Click the "Delete User" button in one of the "Registered Users" table entries 4. Confirm the deletion prompt 5. Attempt to log in using the removed user's credentials				
Expected	● The user account is successfully removed from the "Registered Users" list. ● The removed user's subsequent login attempts fail with an appropriate "Invalid credentials" message ● The user's previously created annotations and uploaded WSIs remain in the system, attributed to a deleted user, ensuring medical records are not lost				
Date-Result	30.04.2026 - Passed				

T-39: Logging Out

Test ID	T-40	Category	Functional	Severity	High
Objective	Verify that a logged-in user can successfully log out, terminating their active session and restricting access to protected pages.				
Steps	1. Log into the application with valid credentials. 2. Click the "Log Out" button. 3. Attempt to navigate back to the previous protected page using the browser's "Back" button 4. Attempt to access a protected URL directly				
Expected	● User is redirected to the Sign In page ● The authentication token/session is destroyed ● The user cannot access protected pages via the "Back" button. ● Direct navigation to a protected URL forces a redirect back to the Sign In page.				
Date-Result	30.04.2026 - Passed				

T-40: Search Response Time

Test ID	T-41	Category	Non-Functional	Severity	Critical
Objective	Verify that the system retrieves and renders search results within an acceptable clinical workflow timeframe				
Steps	1. Execute a complex query 2. Measure the round-trip time from the button click to the results rendering on screen				
Expected	● The CONCH model computes the query embedding, searches the index,				

	and returns results in under 7.0 seconds ● The UI remains responsive (no freezing) while waiting for the payload
Date-Result	30.04.2026 - Passed

T-41: WSI Viewer Client Rendering & Memory Management

Test ID	T-42	Category	Non-Functional	Severity	Critical
Objective	Verify that deep zooming and rapid panning on a WSI maintain a smooth framerate without causing memory leaks in the browser				
Steps	1. Open a multi-gigabyte WSI in the viewer 2. Rapidly pan and zoom in/out continuously for 2 minutes 3. Monitor browser tab RAM usage and visual framerate				
Expected	● Tile rendering occurs smoothly without severe visual tearing or stuttering ● Browser RAM usage stabilizes due to effective garbage collection of off-screen tiles				
Date-Result	30.04.2026 - Passed				

T-42: Backend Similarity Index Memory Efficiency

Test ID	T-43	Category	Non-Functional	Severity	Critical
Objective	Verify that the backend Similarity Index (FAISS) efficiently manages RAM when holding a massive dataset of patch embeddings				

Steps	1. Load the backend vector database with 100,000+ computed patch embeddings 2. Monitor the active RAM utilization of the indexing service
Expected	● The indexing service memory footprint remains within the server's allocated limits and does not cause an Out-Of-Memory crash ● Search times do not degrade exponentially as the index grows
Date-Result	30.04.2026 - Passed

T-43: Concurrent Search Scalability (Load Testing)

Test ID	T-44	Category	Non-Functional	Severity	Critical
Objective	Verify that CONCH model can handle multiple concurrent search requests from different pathologists without dropping connections				
Steps	1. Use a load-testing tool to simulate 20 users executing multimodal searches simultaneously 2. Monitor the API error rate and average response times				
Expected	● The system successfully queues and processes all requests ● 0% of requests return a 500 Internal Server Error or timeout ● Response times may increase under load, but remain within defined acceptable limits				
Date-Result	30.04.2026 - Passed				

T-44: Background Processing Resource Isolation

Test ID	T-45	Category	Non-Functional	Severity	Critical
Objective	Verify that intensive background tasks (like WSI patch extraction) do not degrade the performance of the frontend search features				
Steps	1. Initiate a massive batch upload of 10 WSIs to trigger heavy backend patch extraction and embedding computation 2. While extraction is running, have a user perform standard text and image searches on the frontend				
Expected	● Frontend search requests execute normally without significant lag ● The system properly isolates resources between background workers and user-facing APIs				
Date-Result	30.04.2026 - Passed				

T-45: Automatic Session Timeout

Test ID	T-46	Category	Non-Functional	Severity	High
Objective	Verify that the system enforces an automatic session timeout after a period of user inactivity, adhering to medical data security standards				
Steps	1. Log in as a Pathologist 2. Leave the application completely idle (no mouse clicks, scrolling, or API calls) for the defined timeout period (15 minutes) 3. Attempt to navigate to a new page or execute a search				
Expected	● The user is automatically logged out and redirected to the login screen ● The active session token is invalidated on the backend				

	• A message explains that the session expired due to inactivity is displayed
Date-Result	30.04.2026 - Passed

T-46: Backend API Role-Based Access Control

Test ID	T-47	Category	Non-Functional	Severity	High
Objective	Verify that standard users cannot access backend administrative APIs directly via URL or API manipulation				
Steps	1. Log in as a standard Pathologist. 2. Use a tool like Postman to attempt a direct API call to an admin-only endpoint				
Expected	• The backend strictly rejects the request • A 403 Forbidden or 401 Unauthorized HTTP status code is returned • The action is logged as an unauthorized access attempt				
Date-Result	30.04.2026 - Passed				

T-47: Clinical Audit Logging

Test ID	T-48	Category	Non-Functional	Severity	High
Objective	Verify that the system actively logs crucial user actions to maintain an audit trail for clinical accountability				

Steps	1. Log in, open a specific WSI, execute a multimodal search, and log out 2. Access the system's backend database or log management console
Expected	● Logs accurately record the user ID, timestamp, the ID of the WSI accessed, and the parameters of the executed search ● Passwords and sensitive personal data are strictly excluded from or hashed in the logs
Date-Result	30.04.2026 - Passed

T-48: UI Contrast for Annotations and Heatmaps

Test ID	T-49	Category	Non-Functional	Severity	Critical
Objective	Verify that the explainability heatmaps and annotation colors remain clearly visible against standard H&E tissue stains				
Steps	1. Open a densely stained H&E WSI 2. Draw several annotations and toggle the similarity heatmap 3. Visually inspect the contrast and readability				
Expected	● The default colors for annotations and heatmaps contrast sufficiently with the underlying tissue ● The pathologist does not have to strain to distinguish the highlighted regions from the physical cell structures				
Date-Result	30.04.2026 - Passed				

T-49: Long-Running Process UI Feedback

Test ID	T-50	Category	Non-Functional	Severity	High
Objective	Verify that the system provides clear, continuous feedback during long-running processes like WSI uploads or index updates				
Steps	1. Initiate a 20GB WSI file upload 2. Observe the UI during the upload process				
Expected	● A persistent, accurate progress bar or percentage indicator is displayed ● The user is not left wondering if the system has frozen				
Date-Result	30.04.2026 - Passed				

T-50: Viewer Network Interruption Recovery

Test ID	T-51	Category	Non-Functional	Severity	High
Objective	Verify that the WSI viewer handles sudden network drops gracefully without crashing the browser tab				
Steps	1. Open a WSI in the viewer 2. Disable the device's internet connection 3. Attempt to pan to an unloaded region of the slide				
Expected	● The application does not crash. ● A non-intrusive error message "Network connection lost, reconnecting..." appears ● The viewer successfully resumes loading tiles once the connection is restored without requiring a full page refresh				

Date-Result	30.04.2026 - Passed

T-51: Background Worker Error Recovery (Corrupted File Handling)

Test ID	T-52	Category	Non-Functional	Severity	High
Objective	Verify that a corrupted WSI file does not crash the entire backend extraction queue				
Steps	1. Upload a deliberately corrupted file disguised as a WSI 2. Upload a valid WSI immediately afterward 3. Monitor the background worker queue				
Expected	● The background worker fails gracefully on the corrupted file, marking it as "Failed" in the database ● The worker does not permanently die; it successfully picks up and processes the next valid WSI in the queue				
Date-Result	30.04.2026 - Passed				

6. Maintenance Plan and Details

Once PatchMatch is deployed, a structured maintenance strategy will be followed to ensure the system remains stable, efficient, and up to date over time. The system will follow a continuous release cycle, where small improvements, bug fixes, and minor feature updates are released on a weekly basis. This allows the team to quickly address user feedback and maintain a responsive development process. In addition to these incremental updates, major updates will be scheduled approximately twice a year, focusing on larger improvements such as introducing new retrieval models, enhancing visualization features, or expanding system capabilities.

Handling bugs will be prioritized based on severity. Critical bugs, especially those affecting retrieval accuracy, system availability, or data integrity, will be addressed and deployed as soon as possible. Less critical issues will be grouped into regular update cycles. Logging and monitoring tools will be used to detect errors early and reduce downtime.

Since PatchMatch relies heavily on large-scale image data and embeddings, database maintenance is an essential part of the system lifecycle. Regular backups will be performed to prevent data loss, and consistency checks will ensure that slide data, embeddings, and metadata remain synchronized. As new slides and patches are added, the embedding index will be updated periodically to maintain retrieval performance. Optimization techniques such as indexing and caching will be applied to keep query times low.

The system infrastructure will be continuously monitored to ensure stable performance, especially when handling large WSIs and computationally expensive retrieval tasks. If usage increases, scaling strategies such as expanding storage, optimizing search structures, or distributing workloads will be applied to maintain responsiveness.

User feedback will play an important role in maintenance. Feedback collected from researchers and users will be analyzed regularly to identify usability issues and potential improvements. This will guide future updates and help prioritize new features.

Performance optimization will also be an ongoing process. Retrieval speed, UI responsiveness, and system load will be monitored, and bottlenecks will be addressed through improvements in query processing, model efficiency, and frontend rendering. As the system evolves, updates to machine learning models (such as embedding models or cellular analysis models) will be incorporated carefully. When models are updated, corresponding data (e.g., embeddings or cellular metrics) will be recomputed if necessary to ensure consistency across the system.

Security and data handling practices will be maintained throughout the system's lifecycle. Access control mechanisms will be monitored, and updates will be applied when necessary to ensure safe handling of medical data.

Finally, the system will follow a continuous testing approach, where new updates are validated before deployment. This includes functional testing, regression testing, and performance testing to ensure that updates do not negatively affect existing features. Overall, this maintenance plan ensures that PatchMatch remains reliable, scalable, and adaptable, while continuously improving based on both system performance and user needs.

7. Other Project Elements

7.1. Consideration of Various Factors in Engineering Design

7.1.1 Constraints

PatchMatch operates at the intersection of artificial intelligence, medical imaging, and clinical practice. The following sections discuss how public health, safety, security, welfare, global, cultural, social, environmental, and economic factors were considered during both the analysis and design phases of the project.

Public Health Effect Level: 9

PatchMatch directly serves the public health domain by supporting pathologists in identifying and comparing tissue patterns. During the analysis phase, interviews with a pathologist confirmed that enabling faster case comparison could meaningfully accelerate the diagnostic workflow, reducing both the time required to reach a conclusion and the likelihood of ordering additional tests due to uncertainty. The decision to design PatchMatch strictly as a decision-support tool rather than an autonomous diagnostic system was also driven by public health considerations; an incorrect automated diagnosis could directly harm a patient. Explainable retrieval through heatmaps and similarity scores was designed into the system specifically so that clinicians can critically evaluate results rather than blindly accept them. The 99% uptime requirement during clinical hours was set because system unavailability during active diagnostic sessions could delay patient care.

Safety Effect Level: 9

Any automated diagnostic output is deliberately excluded from the system. PatchMatch retrieves similar cases and presents similarity scores, but it does not classify tissue or suggest diagnoses. This boundary was established during the analysis phase and is enforced throughout the design through the absence of any diagnostic label generation in the ML layer. Score stretching was designed partly for safety reasons as well: raw cosine similarities clustering between 0.2 and 0.7 could be misread as low confidence across the board, potentially causing a clinician to dismiss relevant results. Stretching these to an interpretable 0–1 range reduces the risk of misinterpretation. The system also displays clinical report information alongside retrieval results so that pathologists have full context

when comparing cases, reducing the risk of visually similar but clinically dissimilar cases being treated as equivalent.

Security Effect Level: 10

Security is a core of the system's design, focusing on the protection of medical imaging data and secure user collaboration. JWT-based authentication with bcrypt password hashing ensures that credentials are never stored or transmitted in plaintext. Role-Based Access Control (RBAC) is strictly enforced: regular users are restricted to their own private directories and slides explicitly shared with them, while administrative functions such as user management and infrastructure configuration—are gated behind privileged flags.

The system implements a privacy-by-design storage architecture. Whole-slide images and associated metadata are stored within the institution’s controlled local environment. When a pathologist uploads a slide, it is placed in a dedicated local directory, ensuring that the data remains under the user's or institution's physical control. Any clinical context or diagnostic notes are similarly persisted locally. This approach minimizes data exposure risks and ensures that sensitive imaging assets remain on-premise, directly supporting compliance with GDPR and KVKK regulations.

Welfare Effect Level: 7

The design prioritizes the welfare of both pathologists and patients. For pathologists, usability is optimized through a simplified workflow where complex multimodal retrieval results are achieved in fewer than five interactions with sub-second feedback times, significantly reducing cognitive load. The inclusion of System Status dashboards provides transparency into model residency and hardware health, ensuring trust in the system's operational state.

For patients, welfare is safeguarded by the system’s role as a decision-support tool rather than an autonomous diagnostic engine. This is reinforced by explainability features, such as diagnostic heatmaps and quantitative cellular metrics, which provide visual justification for retrieval results. By maintaining a local storage architecture and emphasizing 'Retrieval-as-Evidence,' the system reduces the risk of diagnostic overreliance and ensures that final clinical decisions remain firmly in the hands of the human expert.

Global Factors Effect Level: 6

Digital pathology is a globally practiced discipline and PatchMatch was designed with international deployment in mind. The dataset used during development is drawn from

TCGA, a publicly available US-based cancer genomics archive, which introduces a known geographic and demographic bias. This was acknowledged as a limitation during the analysis phase and documented in the system. The compliance framework was designed to satisfy both GDPR for European deployments and KVKK for Turkish deployments, with the architecture being flexible enough to accommodate other regional data protection regimes. The use of open standards such as IIIF for tile serving and support for common WSI formats including SVS and NDPI ensures that the system is not locked to any particular scanner manufacturer or geographic market.

Cultural Factors Effect Level: 3

Cultural factors had a limited but non-zero effect on the design. The user interface terminology was aligned with standard pathological vocabulary used internationally, rather than region-specific clinical language, to support adoption across different institutional contexts. The system was designed to avoid making clinical recommendations that could conflict with locally established diagnostic protocols or medical culture. Beyond these interface and framing considerations, the core retrieval and ML pipeline is not materially affected by cultural factors.

Social Factors Effect Level: 7

PatchMatch has meaningful social implications in terms of access to diagnostic quality. One of the stated motivations of the project is that pathologists in under-resourced institutions could use the system to compare their cases against a curated reference archive, effectively extending the reach of expert knowledge. The collaborative features, slide sharing, annotations, and notes, were designed to support knowledge transfer between pathologists across institutions. The educational use case was also explicitly considered during analysis: medical students and trainees can use the system to explore visual patterns and build case familiarity without requiring access to a physical slide library. These social motivations influenced the decision to support multiple user roles and to design the interface to be learnable without formal training.

Environmental Factors Effect Level: 5

Environmental factors had a moderate indirect effect on design decisions. The embedding generation pipeline is computationally intensive, requiring GPU acceleration and significant processing time for large slide collections. During the analysis phase, the decision to pre-compute all patch embeddings offline rather than on demand was motivated partly by the desire to avoid repeated energy-intensive inference at query time. Storing embeddings

as compressed NumPy files and caching clustering results to disk with fingerprint-based invalidation also reduces unnecessary recomputation. In a large-scale clinical deployment, the energy consumption of running continuous GPU inference servers would be a legitimate environmental consideration, and the architecture's support for batch preprocessing rather than live inference keeps this footprint lower than an on-demand design would.

Economic Factors Effect Level: 8

Economic considerations significantly influenced the system's technical architecture. The entire stack is built on open-source frameworks, including FastAPI, React, PyTorch, and PostgreSQL, to eliminate licensing fees and ensure accessibility for resource-constrained clinical institutions. High-performance foundation models such as CONCH and CellViT were selected as accessible, research-aligned alternatives to proprietary, paid vision-language APIs.

Data management choices also reflect cost-efficiency: patch embeddings are stored in memory-mapped HDF5 files and indexed via FAISS, bypassing the recurring subscription costs associated with commercial vector databases. The hybrid storage strategy using PostgreSQL for relational metadata and the local file system for high-volume binary blobs optimizes existing infrastructure usage. Furthermore, the containerized architecture (Docker) allows institutions to scale their hardware investments, such as GPU servers and storage arrays, independently as their diagnostic volume grows, preventing the need for massive upfront capital investment.

Table 1. Various factors in engineering design and their effects.

Factor	Effect Level (0-10)	Effect
Public Health	9	PatchMatch directly supports clinical diagnostic workflows. Retrieval latency targets, uptime requirements during clinical hours, and the decision-support framing were all shaped by the need to avoid disrupting or endangering patient care.
Safety	9	The exclusion of automated diagnosis, the design of explainable similarity scores, and

		score stretching for interpretability were all driven by the need to prevent clinical misuse or overreliance on system outputs.
Security	10	Authentication architecture, role-based access control, bcrypt password hashing, audit logging, and the clinical deployment note requiring on-premise storage were all direct consequences of handling sensitive patient medical data.
Welfare	7	Usability requirements including 30-minute onboarding, 5-interaction retrieval, and one-second feedback were designed to protect pathologist workflow. Patient welfare was addressed through anonymization requirements and the decision-support framing.
Global Factors	6	Multi-regulation compliance covering both GDPR and KVKK, open format support for SVS and NDPI, and IIIF adoption were all motivated by the need to support international deployment without geographic lock-in.
Cultural Factors	3	The system uses internationally standardized pathology terminology in the interface. No further cultural localization was required given that English is the established language of medical communication globally.
Social Factors	7	Collaboration features, multi-role user support, and the educational use case were motivated by the goal of making expert-level case comparison accessible to pathologists, trainees, and researchers across institutions with varying resources.
Environmental Factors	5	Pre-computed embeddings, batch inference

		rather than on-demand inference, and fingerprint-based clustering cache were designed to reduce unnecessary computation and energy use. Environmental impact is limited but was considered in pipeline design.
Economic Factors	8	The open-source stack including FastAPI, React, PyTorch, FAISS, and PostgreSQL, flat file embedding storage, and modular architecture were all chosen to minimize infrastructure costs and make the system accessible to institutions with limited budgets.

7.1.2 Standards

Medical Software Development Standards

- The system follows IEC 62304:2006/AMD1:2015 [16] to ensure safe medical software development. As a decision-support system, PatchMatch aligns with the principles of safety class-based development and traceability.
- The system follows IEC 82304-1:2016 [17], which focuses on usability, validation, and release practices for health software.

Medical Imaging and Data Exchange Standards

- The system considers DICOM Supplement 145 [18] to support future interoperability with hospital PACS systems.
- The system is designed to be compatible with commonly used digital pathology formats including SVS and NDPI through the use of OpenSlide, enabling integration with existing clinical infrastructures.
- HL7 standards [19] are considered for the exchange of clinical metadata and pathology reports when integrating with hospital information systems.
- The WSI tile serving component is implemented using an IIIF-compatible interface, which is an open standard for serving large image collections in an interoperable and resolution-independent manner. This ensures that the viewer layer can be integrated with other IIIF-compliant platforms in the future.

Quality and Risk Management Standards

- ISO 13485:2016 [20] is considered for development practices as it emphasizes quality management, documentation, and controlled change processes in medical software.
- ISO 14971:2019 [21] guides the management of risks such as incorrect retrieval results, misinterpretation of similarity scores, and overreliance on system outputs in clinical decision-making.

Regulatory and Compliance Considerations

- The system is reviewed in line with the EU Medical Device Regulation (EU) 2017/745 [22], which covers clinical decision-support software.
- The EU Artificial Intelligence Act (EU) 2024/1689 [23] is considered to promote transparent and responsible use of AI in medical settings. PatchMatch's explainability features, score transparency, and decision-support framing directly address the transparency obligations outlined in this regulation.
- Regulatory pathways in other regions, such as the FDA Class II 510(k) process [24] in the United States, are taken into account for future deployment.

Data Protection and Privacy Standards

- PatchMatch follows GDPR (EU) 2016/679 [25] and KVKK Law No. 6698 [26] requirements to process medical images and patient data lawfully.
- To protect patient data, the system implements JWT-based authenticated access, role-based authorization, bcrypt password hashing, and audit logging. Anonymization is required for any data used in research or educational contexts.

Documentation and Modeling Standards

- UML 2.5.1 [27] is used for modeling system architecture, subsystem interactions, navigational flows, and dynamic behavior throughout this report.
- IEEE 830-1998 [28] principles guide the structure and clarity of the software requirements specification, with functional and non-functional requirements clearly separated and written in a testable and verifiable form.

Software Engineering and API Standards

- The backend API is structured following REST architectural principles [29], with clearly separated endpoints, stateless request handling, and standard HTTP status codes, ensuring predictable and interoperable client-server communication.
- OpenAPI Specification 3.0 [30] is used to document the backend API, enabling automatic generation of interactive API documentation and supporting future integration with external systems or hospital middleware.
- The frontend is built following React component architecture conventions and TypeScript strict typing [31], which enforces interface contracts between components and reduces the risk of integration errors during development.
- Docker and Docker Compose [12] are used following containerization best practices to ensure reproducible builds and consistent deployment environments across development, testing, and production targets.

7.2. Ethics and Professional Responsibilities

The PatchMatch project emphasizes the importance of ethical and professional responsibilities as it operates within the medical domain. These principles guide both the analysis and future development of the system to meet the requirements of responsible use, transparency, and user trust.

Patient Data Privacy, Confidentiality, and Security

- Access Control: Only authorized users will have access to WSIs, metadata, and reports.
- Data Protection: Medical images, embeddings, and reports will be stored and transmitted to ensure authorized access.
- Anonymization and De-identification: The system will support anonymization or de-identification of data when for research or educational purposes.
- Compliance: Data handling practices comply with GDPR [14] and KVKK [15] to ensure ethical and lawful processing of patient data.

Clinical Responsibility and Risk Awareness

- Decision-Support Role: PatchMatch is a decision support system and will not provide automated diagnoses or treatment recommendations.
- Human Oversight: Clinical users will have full responsibility for interpretation and final decision-making based on retrieved results.

- Risk Mitigation: The system will clearly communicate its limitations to reduce the risk of overreliance or misinterpretation in clinical workflows.

Explainability and Transparency

- Explainable Retrieval Results: The system will provide clear explanations for similarity-based retrieval results, such as visual heatmaps or shared semantic features.
- Transparency of System Behavior: Users will be informed that retrieval results are generated through automated methods and should be interpreted critically.

Bias Awareness and Fair Use

- Bias Recognition: The system will acknowledge that retrieval results may reflect biases present in training data or institutional archives.
- Responsible Interpretation: Retrieval results are intended to support decision-making and should not be interpreted as definitive evidence.
- Fair Use: The system supports diverse datasets and avoids misleading or unrepresentative patterns.

Professional Conduct and Software Quality

- Quality Assurance: The development process will include systematic testing and validation to ensure robustness.
- Documentation: Code, requirements, and design decisions will be documented to support maintainability.

Responsible Use of Artificial Intelligence

- Responsible AI Deployment: Artificial intelligence components shall be used to assist users rather than replace professional judgment.
- Transparency in AI Use: Users shall be informed when AI-based methods are involved in retrieval and explanation processes.

7.3. Teamwork Details

7.3.1. Contributing and functioning effectively on the team to establish goals, plan tasks, and meet objectives

Ekin Köylü, Emre Yazıcıoğlu, and İlke Latifoğlu developed the machine learning pipeline, including literature review, model selection, patch extraction, and similarity-based retrieval implementation. Embedding generation was carried out collaboratively by the entire team. Elif Lara Oğuzhan contributed to both backend and frontend development by implementing API integrations, managing data flow between system components, and developing the user interface. She ensured that frontend components correctly interacted with backend services. Bertan Uran worked on the database and frontend components. He designed and implemented the database schema for storing embeddings and metadata, handled data organization and retrieval, and contributed to frontend development and integration. İlke Latifoğlu and Bertan Uran also implemented validation and testing procedures to ensure the correctness and robustness of the pipeline outputs.

7.3.2. Helping creating a collaborative and inclusive environment

The team maintained a collaborative and inclusive working environment by fostering open communication, shared decision-making, and active participation from all members. Regular meetings were held to discuss progress, identify challenges, and align on upcoming tasks, ensuring that everyone remained informed and involved. Feedback was continuously exchanged, allowing ideas to be refined collectively and issues to be addressed early.

Tasks and responsibilities were assigned transparently, and team members were encouraged to contribute beyond their primary domains when needed. This created an environment where knowledge was shared across different components of the system, reducing dependency on single individuals and improving overall coordination. Additionally, the team ensured that all viewpoints were considered during technical and design decisions, promoting inclusivity and balanced contributions throughout the project lifecycle.

7.3.3. Taking lead role and sharing leadership on the team

Leadership was shared dynamically within the team, with members taking initiative based on their areas of expertise and the needs of each project phase. Ekin Köylü and İlke Latifoğlu primarily led the literature review and model selection process, and took

responsibility for adapting the chosen models to the project's specific use case. Emre Yazıcıoğlu led the development of the pipeline in which these models were utilized, focusing on system integration, execution flow, and ensuring that the ML components functioned reliably within the overall architecture. On the system side, Elif Lara Oğuzhan took initiative in coordinating frontend–backend interaction, leading decisions related to API usage, data flow, and interface behavior. Bertan Uran led the database-related decisions and contributed to frontend integration, ensuring that data storage, retrieval, and presentation were consistent with system requirements.

This distribution allowed leadership to shift naturally across different components, with each member guiding decisions and implementation within their domain while maintaining alignment with the overall system.

7.3.4. Meeting objectives

The team followed a structured development process with clearly defined milestones and iterative progress tracking. Regular meetings were conducted to monitor progress, address emerging issues, and maintain synchronization between the machine learning pipeline, backend, database, and frontend development.

Documentation, including system design decisions, pipeline configurations, and interface requirements, was maintained throughout the project to support coordination and consistency. Integration phases were carefully managed to ensure that independently developed components functioned correctly as a unified system. Potential issues were identified early during development and resolved collaboratively, minimizing delays. This approach enabled the team to complete all major components, including the ML pipeline, system integration, and user interface, in alignment with the defined project scope and timeline.

7.4 New Knowledge Acquired and Applied

Throughout the development of PatchMatch, we gained significant new knowledge in both theoretical and practical aspects of software engineering and artificial intelligence. Since the project involves advanced topics such as whole slide image processing, deep learning-based retrieval, and large-scale system design, it required us to go beyond the knowledge covered in standard coursework.

One of the main areas in which we developed new understanding was artificial intelligence and deep learning, particularly in representation learning for medical images. In this project, we used the CONCH model, which is a foundation model designed for pathology images. Learning how to use CONCH required us to understand how pretrained models generate embeddings and how these embeddings can be used for similarity-based retrieval. We explored model documentation, experimented with different configurations, and integrated the model into our pipeline to extract meaningful features from image patches.

During this process, Prof. Selim Aksoy provided valuable guidance, especially in understanding how to approach representation learning and retrieval problems. His feedback helped us better interpret model outputs, refine our retrieval strategy, and make more informed design decisions throughout the project.

We also gained experience in working with whole slide images (WSIs), which are significantly larger and more complex than standard images. We learned how to use tools such as OpenSlide and tile-based rendering techniques to efficiently display and navigate these images. This required understanding how to manage large-scale data and ensure smooth interaction within a web-based environment.

Another important area of learning was modern web development frameworks. On the frontend side, we worked with React and integrated libraries such as OpenSeadragon to build an interactive viewer for high-resolution slides. This helped us improve our skills in designing responsive and user-friendly interfaces. On the backend side, we developed new knowledge in API design and system integration. We implemented backend services that coordinate communication between the frontend, retrieval system, and data storage. This involved designing RESTful APIs, handling requests efficiently, and ensuring reliable data flow.

We also learned how to work with vector similarity search systems, such as FAISS, which allowed us to perform efficient nearest-neighbor search on high-dimensional embeddings. Understanding how indexing and similarity computations work was essential for building a scalable retrieval system.

In addition, integrating cellular-level analysis tools (such as CellViT) enabled us to extract biological features such as cell density and morphology. We incorporated these features into the system to enhance analysis and provide more meaningful insights.

Overall, we acquired new knowledge through a combination of independent research, experimentation, and guidance from instructors. We applied this knowledge throughout the development process, resulting in a system that combines multiple advanced techniques into a cohesive and functional platform.

8. Conclusion and Future Work

The PatchMatch system successfully demonstrates how patch-level retrieval can be integrated with whole slide image visualization to support efficient and interactive exploration of pathology data. The system is capable of performing core functionalities such as region-based retrieval, slide comparison, visualization of similarity patterns, and integration of cellular-level metrics. Throughout the development process, the objectives of building a functional and scalable retrieval system have been achieved.

Although the current version provides a strong foundation, there is significant room for further improvement and expansion. If the project is continued, one of the main directions would be to enhance the quality of retrieval results. This can be achieved by experimenting with more advanced embedding models and improving ranking strategies based on domain-specific knowledge.

Another important improvement area is cellular-level analysis. While the system already integrates basic metrics such as density and circularity, these can be extended with more detailed biological features, better tumor cell detection, and richer visualizations. Making these outputs more interpretable and useful for domain experts would significantly increase the system's value.

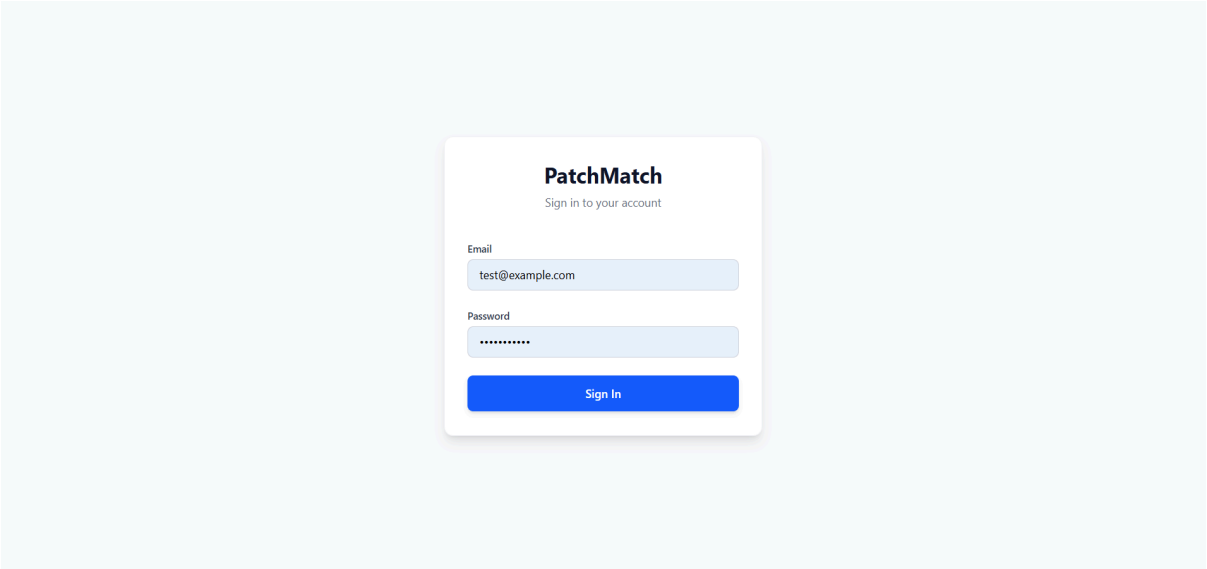
Scalability is another key direction. As larger datasets are introduced, improvements in indexing, distributed processing, and storage management will be necessary to maintain performance.

Finally, with further development and validation, PatchMatch could evolve into a more comprehensive tool that supports not only retrieval but also decision support and research workflows in computational pathology. By continuing to refine both the technical and user-facing aspects, the system has strong potential to become a practical and impactful platform in the field.

9. User Manual

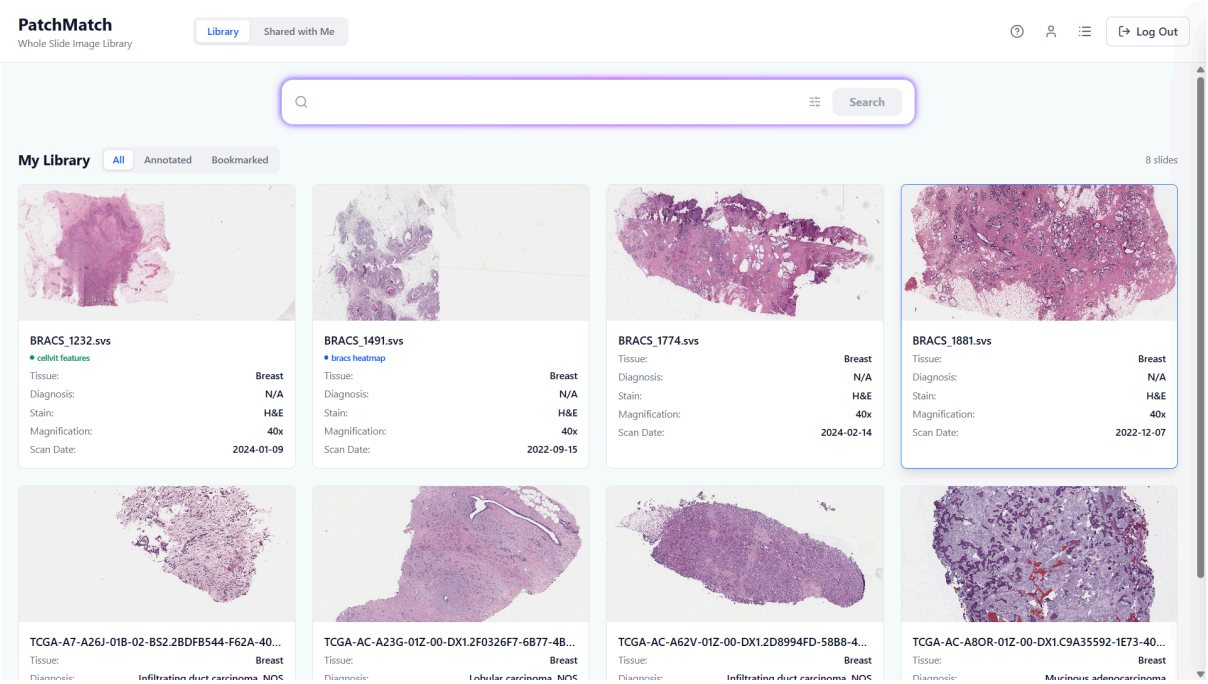
9.1 Login Page

The Login Page provides secure access to the system through user authentication. Users enter their credentials to log in and access their personal workspace. This ensures that sensitive slide data, annotations, and analysis results are protected and only accessible to authorized users. The page may also support session management to keep users logged in securely during their interaction with the system.



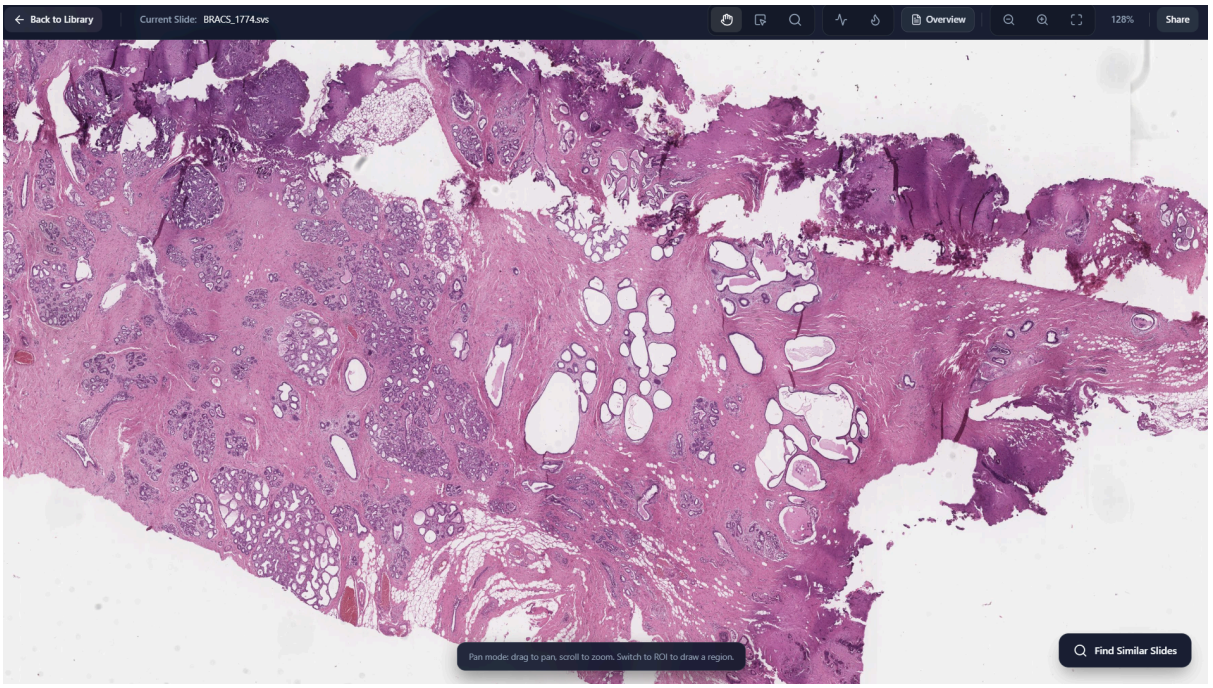
9.2 Library Page

The Library Page serves as the central interface for managing and accessing whole slide images. Users can browse available slides, search using keywords, and select slides for analysis. To ensure efficient organization, the page features dedicated tabs: 'All' for the full dataset, 'Annotated' for slides with existing findings, and 'Bookmarked' for quickly accessing prioritized images. Additionally, users can toggle between their personal 'Library' and the 'Shared with Me' section to collaborate on slides shared by other team members. This page is designed to provide an organized overview, allowing users to efficiently locate and open relevant slides for further exploration.



9.3 WSI Viewer

The WSI Viewer enables interactive exploration of high-resolution pathology slides. Users can zoom in and out, pan across the slide, and seamlessly navigate between different magnification levels. The viewer ensures smooth rendering of large images and provides the foundation for performing all region-based analysis tasks within the system.

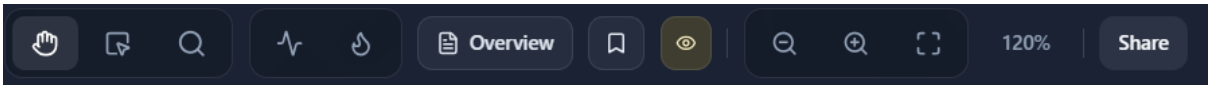


9.4 Navigation Bar

The Navigation Bar provides a centralized set of tools for slide exploration, analysis, and visualization:

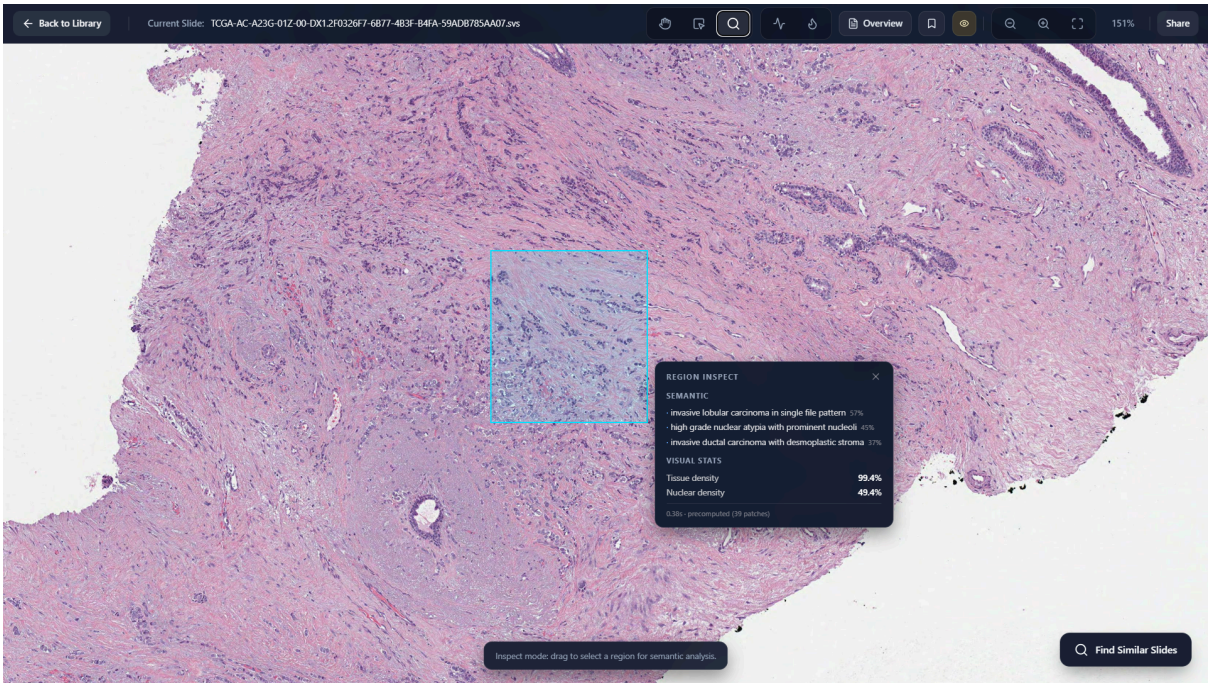
- Hand Tool: Allows for free navigation and panning across the slide.
- Selection Tool: Used to draw rectangles to select specific regions for analysis or annotation.
- Inspect Mode (Magnifying Glass): Activates a specialized mode to click and analyze specific local regions for morphological features.
- Cellular Analysis (Pulse Icon): Initiates the full-slide cellular processing to generate both global heatmaps and diagnostic metrics.
- Class Heatmap (Flame Icon): Toggles the tissue-level classification map, visualizing the spatial distribution of predicted diagnostic categories (e.g., IC, PB) across the slide.

- Overview: Opens the Slide Report Panel, aggregating clinical data and analysis results.
- Bookmark: Saves or bookmarks the current slide to the user's collection for quick access from the library's "Bookmarks" filter.
- Annotation Toggle (Eye Icon): Manages the visibility of all user-created annotations and drawings, allowing users to quickly hide them to see the underlying tissue clearly.
- Zoom Controls (+/-): Adjusts the magnification level of the image incrementally.
- Fit to Screen: Automatically adjusts the zoom level to fit the entire slide within the current viewing window for a complete overview.
- Share: Generates a link to share the current view or analysis with other users.



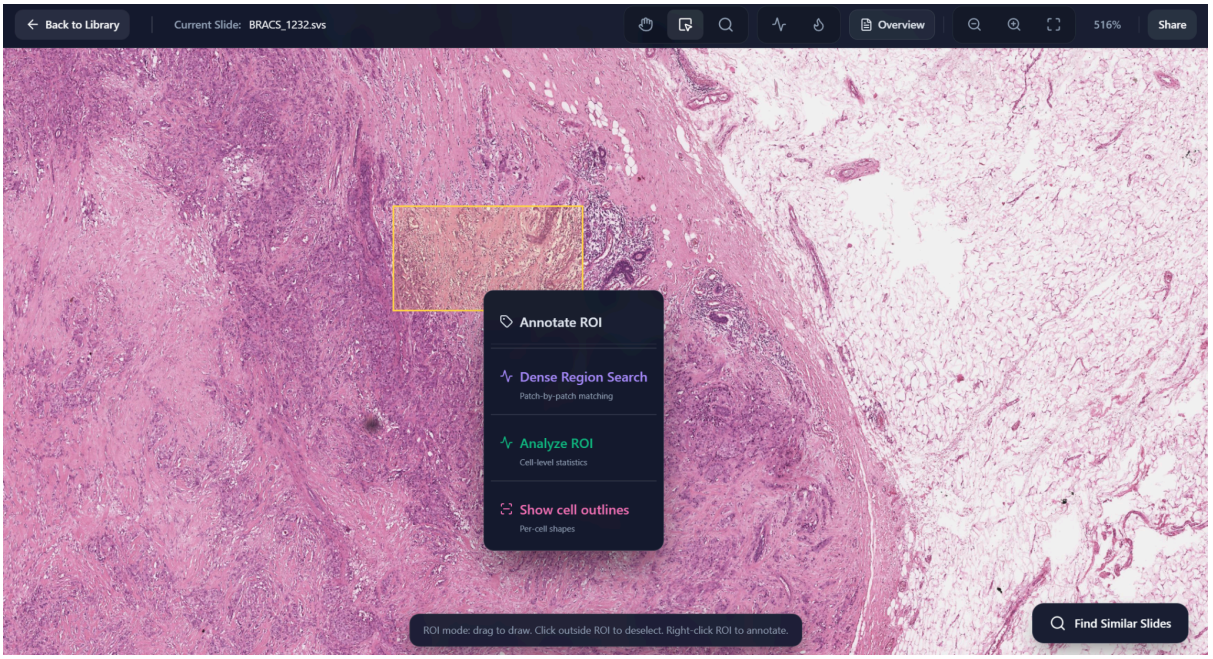
9.5 Inspect a Region

Users can select and inspect it in greater detail through a dedicated analysis panel. This feature allows for viewing high-magnification snapshots of the selected area and provides automated semantic descriptions (such as 'solid growth pattern', 'nuclear atypia', or 'cribriform pattern') derived from the vision model's analysis of the tissue morphology. Users can also review the predicted diagnostic class distribution (e.g., Normal, Benign, DCIS, Invasive) for the specific region and attach clinical notes. This supports precise visual and morphological assessment, facilitating more informed diagnostic decisions.



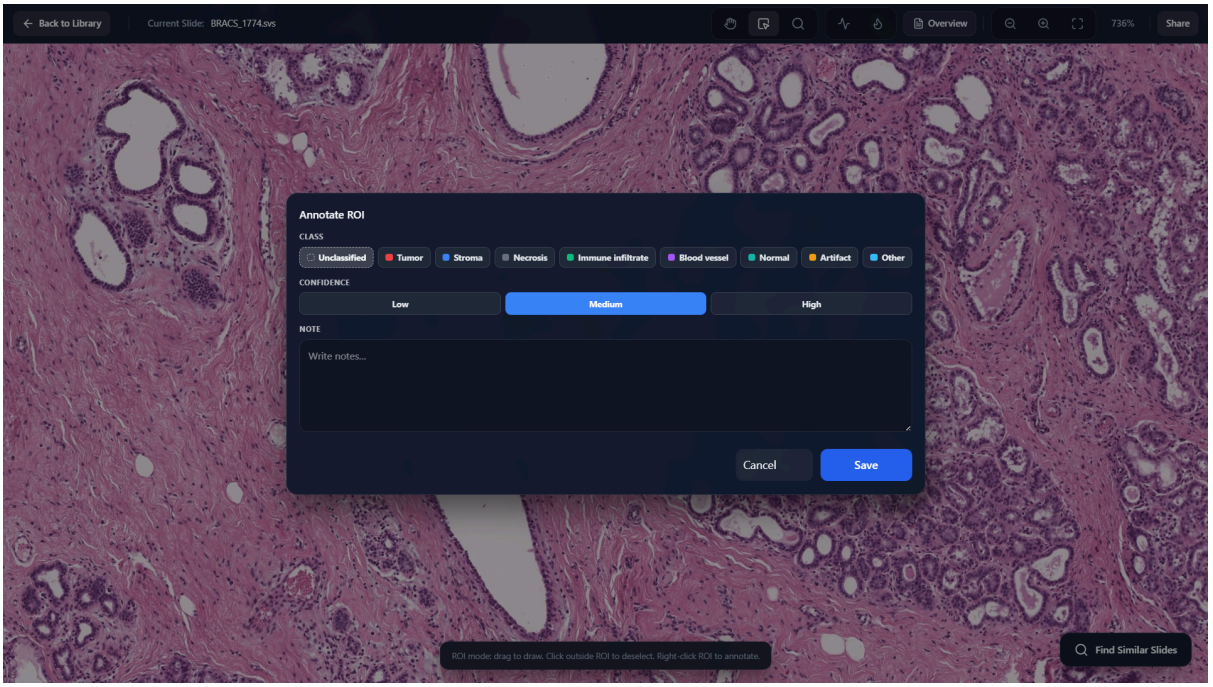
9.6 Select a Region

Users can select a ROI directly on the slide using the selection tool. This functionality allows users to focus on specific areas to Annotate, Search, or Analyze the region. The selected region serves as the input for further operations, including saving detailed notes, performing dense region searches across the library, or computing localized cellular metrics.



9.7 Annotate a Region

The system allows users to add annotations to selected regions, including descriptive notes or labels. These annotations can be used to highlight important findings, document observations, or communicate insights to other users. Annotated regions can be stored and revisited later as part of the analysis workflow.



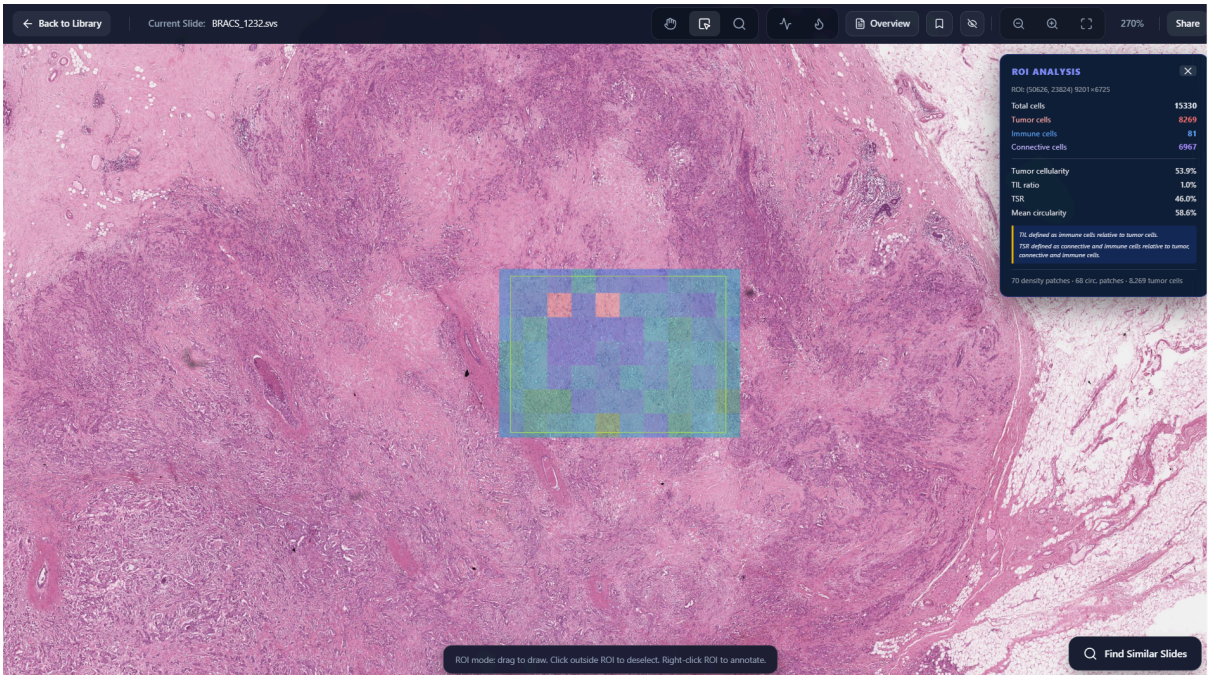
9.8 Show Cellular Level Metrics of a Region

For any selected ROI, the system provides a comprehensive breakdown of cellular metrics to assist in detailed visual analysis. The system automatically categorizes cells within the region into Tumor, Immune, and Connective (stromal) types.

Based on this analysis, the system displays key diagnostic indicators for the selected region, including:

- Cell Counts: Displays the exact number of tumor, immune, and connective cells, along with the overall tumor cellularity.
- Clinical Ratios: Automatically calculates the Tumor-Infiltrating Lymphocytes (TIL) and Tumor-Stroma (TSR) ratios.
- Cell Morphology: Measures the physical characteristics of tumor cells, such as their circularity.
- Visual Heatmaps: Overlays interactive spatial heatmaps for cell density and tumor circularity, helping you instantly spot dense cell clusters or abnormal structures on the slide.

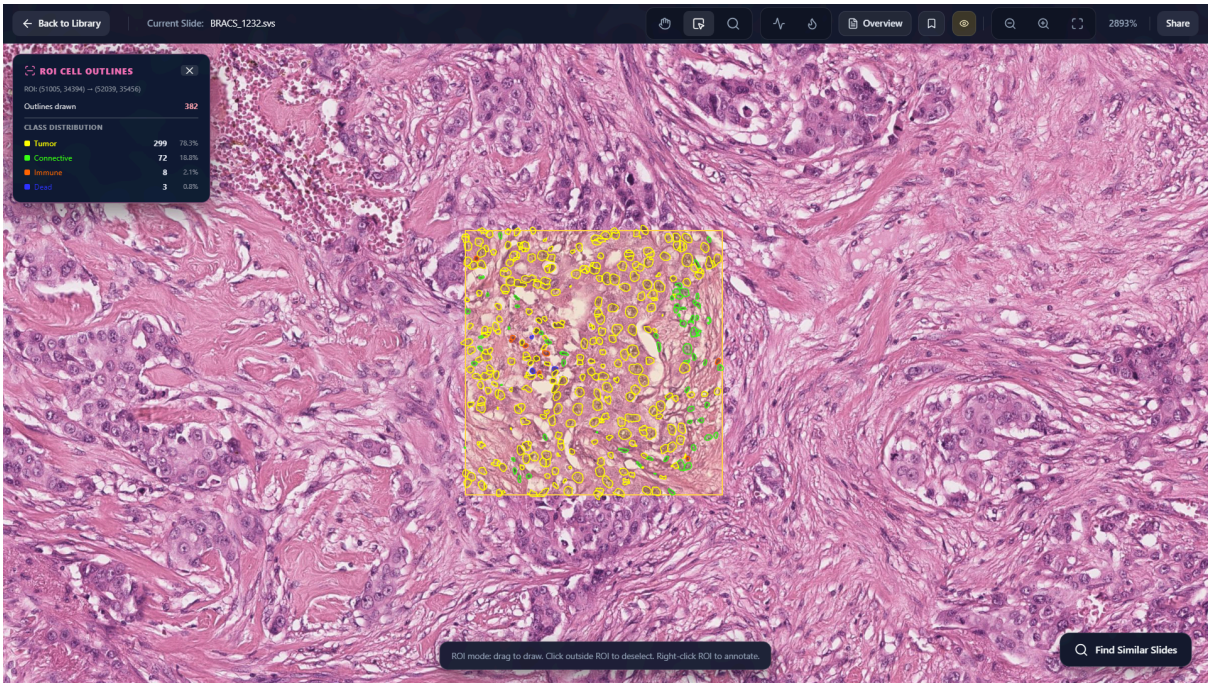
These quantitative measurements eliminate the need for manual cell counting, allowing users to support their visual observations with immediate, actionable data regarding the tissue's biological composition.



9.9 Outline the Cells

The system detects and draws precise boundaries around individual cells within a selected region. To facilitate rapid visual analysis, each cell outline is color-coded according to its classification: Tumor, Immune, Connective, and Dead cells.

Alongside this visual overlay, the "ROI Cell Outlines" panel provides an instant summary of the region. It displays the total number of cells drawn and a detailed class distribution, showing exact counts and percentages for each cell type. This combined view helps users easily observe structural patterns and verify the biological composition of the area at a glance.

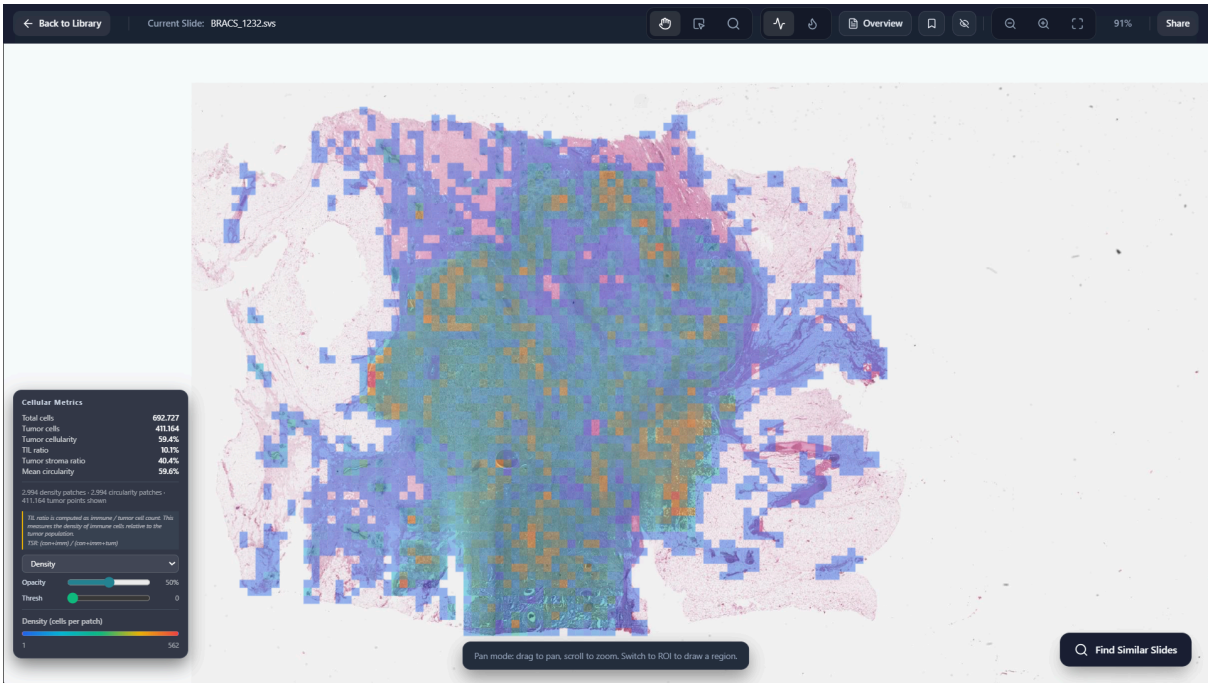


9.10 Show Cellular Level Metrics of a Slide

Users can view aggregated cellular metrics for the entire slide, offering a global perspective on tissue characteristics.

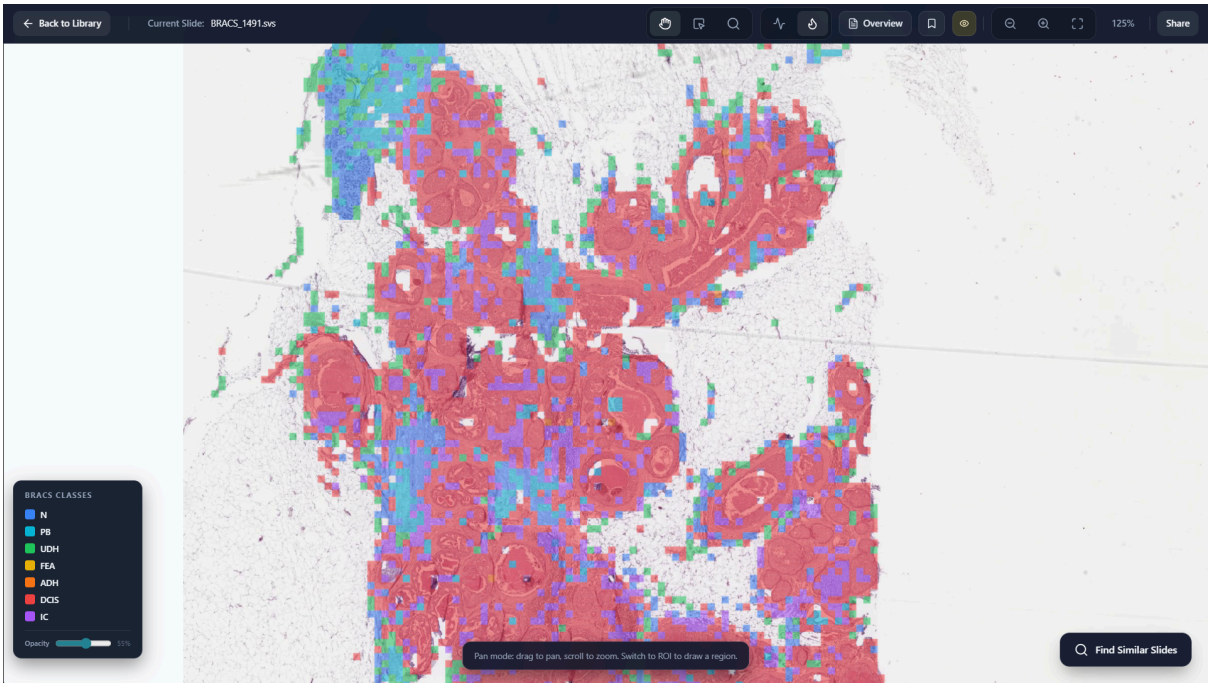
The "Cellular Metrics" panel provides a comprehensive whole-slide summary. It displays exact counts for total and tumor cells, alongside key clinical indicators such as Tumor Cellularity, TIL Ratio, Tumor-Stroma Ratio, and Mean Circularity.

In addition to numerical data, the panel serves as a control center for the global heatmap overlay. Users can toggle the visualization between cell "Density" and "Circularity", and fine-tune the display using "Opacity" and "Threshold" sliders. A dynamic color scale at the bottom of the panel helps users easily interpret the heatmap values (e.g., highlighting patches with the highest cell density), making slide-level comparisons and structural analysis highly intuitive.



9.11 Show Class Heatmap of a Slide

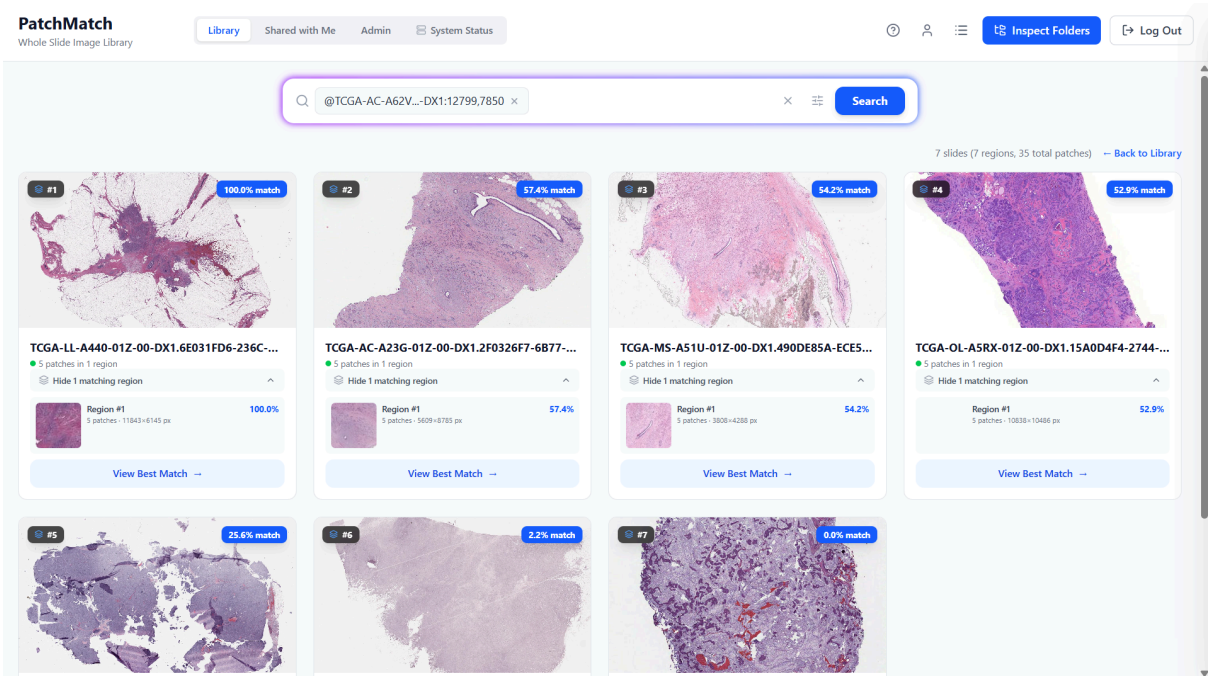
The class heatmap visualizes the spatial distribution of diagnostic categories across the entire slide. By processing the tissue at a patch-level, the system generates a color-coded overlay that represents the predicted classification for each area. The map highlights specific tissue types including Normal, Benign (PB, UDH), Atypical (FEA, ADH), and Carcinoma (DCIS, IC). This allows users to quickly identify high-risk regions and evaluate model predictions in real-time. The opacity of the overlay can be adjusted via the legend slider to facilitate a direct comparison between the classification results and the original H&E stained tissue.



9.12 Search a Region to Retrieve Similar Regions with Slides

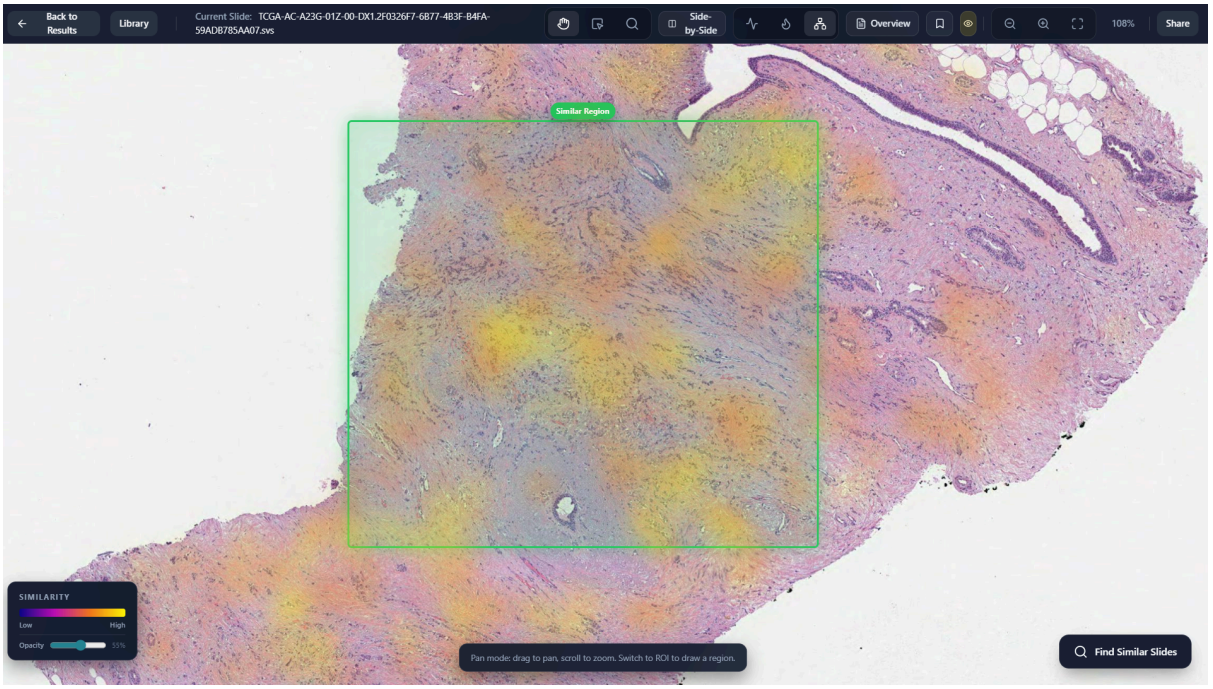
Users can initiate a specialized similarity search by selecting a region within the WSI Viewer and clicking the Dense Region Search button from the selection menu. This tool performs a patch-by-patch comparison between the query area and the entire slide library to find visually and morphologically similar regions.

The system ranks results based on their match quality, ensuring that slides with patterns most closely matching the selected region appear first. Each search result displays a similarity score and allows users to jump directly to the "Best Match" area in the corresponding slide. This facilitates efficient cross-slide comparison and helps identify recurrent pathological patterns across different cases.



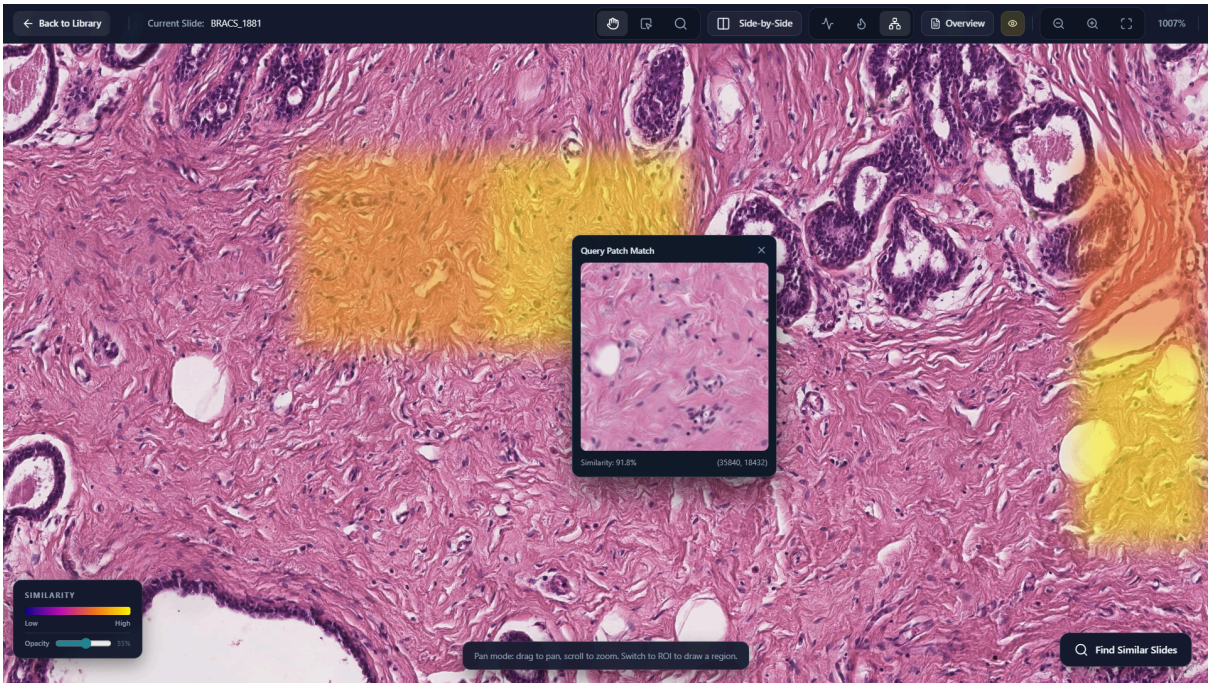
9.13 View a Retrieved Region along with Slide

Selecting View Best Match opens the slide with a Similarity Heatmap overlay. The green bounding box identifies the specific area on this slide that most closely matches your original query region. As indicated by the Similarity Legend in the bottom-left panel, vibrant colors highlight areas of high morphological similarity. This visual map allows users to instantly identify where patterns similar to the query appear across the entire tissue, with the green frame pinpointing the top result.



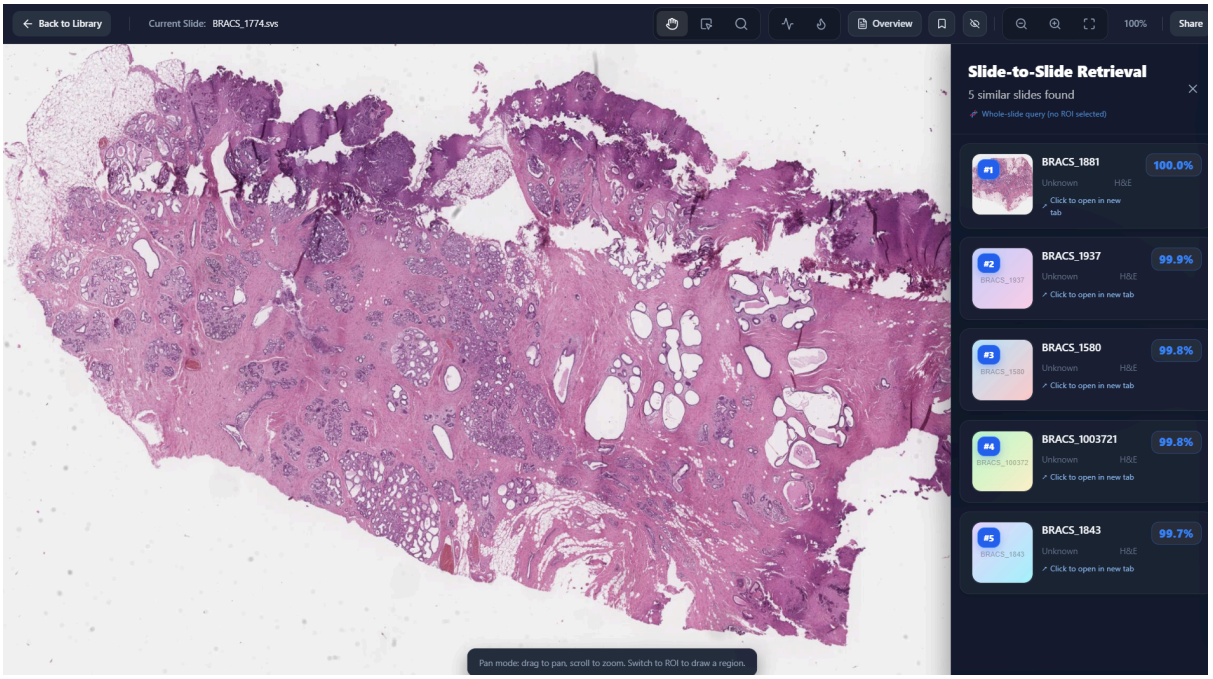
9.14 Show Query Patch Match

While examining a retrieved slide, users can click on any area covered by the similarity heatmap to see the exact corresponding match from the original query. The Query Patch Match popup displays the specific image patch from your original query region that was found to be most similar to the location you clicked on the current slide. This feature provides direct evidence for the similarity score, allowing users to visually verify which part of the query slide corresponds to the matching area in the retrieved slide. It includes the similarity percentage and coordinates for precise morphological correlation.



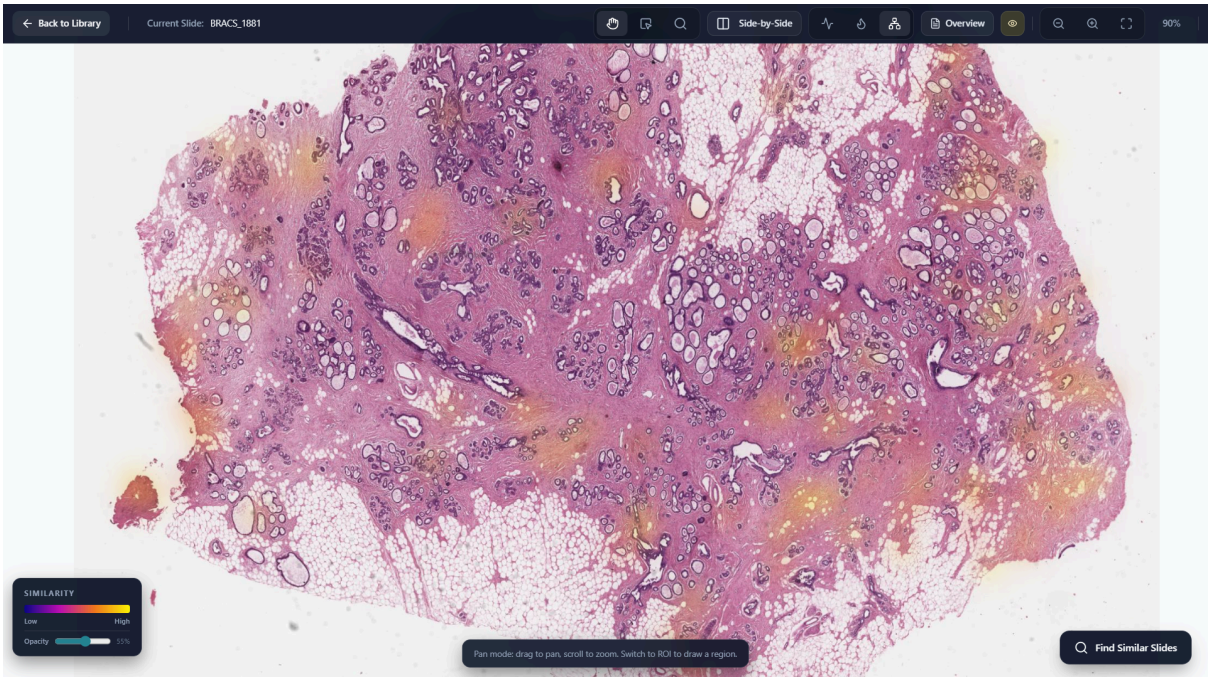
9.15 Search a Slide to Retrieve Similar Slides

Users can initiate a global similarity search by clicking the Slide-to-Slide Retrieval button in the navigation bar. This feature identifies slides in the library that are most similar to the current case based on overall tissue architecture and cellular patterns. The search results are displayed in a dedicated right-side panel, ranked by their global similarity scores. While this panel is active, the viewer highlights representative regions with light blue frames on the current slide. These frames indicate the specific areas that the system identified as key landmarks for determining the similarity between the current slide and the rest of the dataset. Users can click on any slide in the results panel to further explore the matching cases.



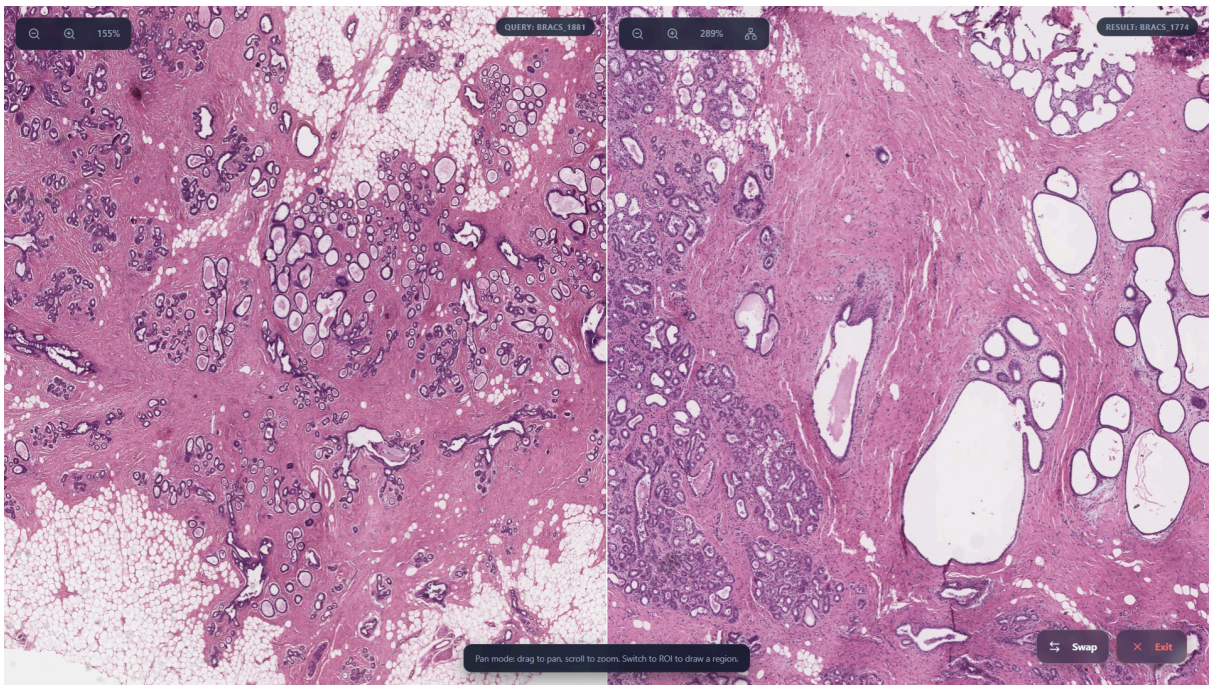
9.16 View Retrieved Slides with optional Similarity Map

When viewing a slide retrieved through global slide-to-slide search, users can toggle an optional Similarity Map to understand the basis of the match. This map visualizes how different areas of the current slide compare to the original query slide, highlighting the specific tissue structures that contributed most to the overall similarity score. By displaying these shared morphological characteristics as a heat-coded overlay, the system provides a clear visual comparison that justifies why the two slides were ranked as related cases. This helps pathologists quickly pinpoint the regions of interest that define the similarity between the query and the retrieved result.



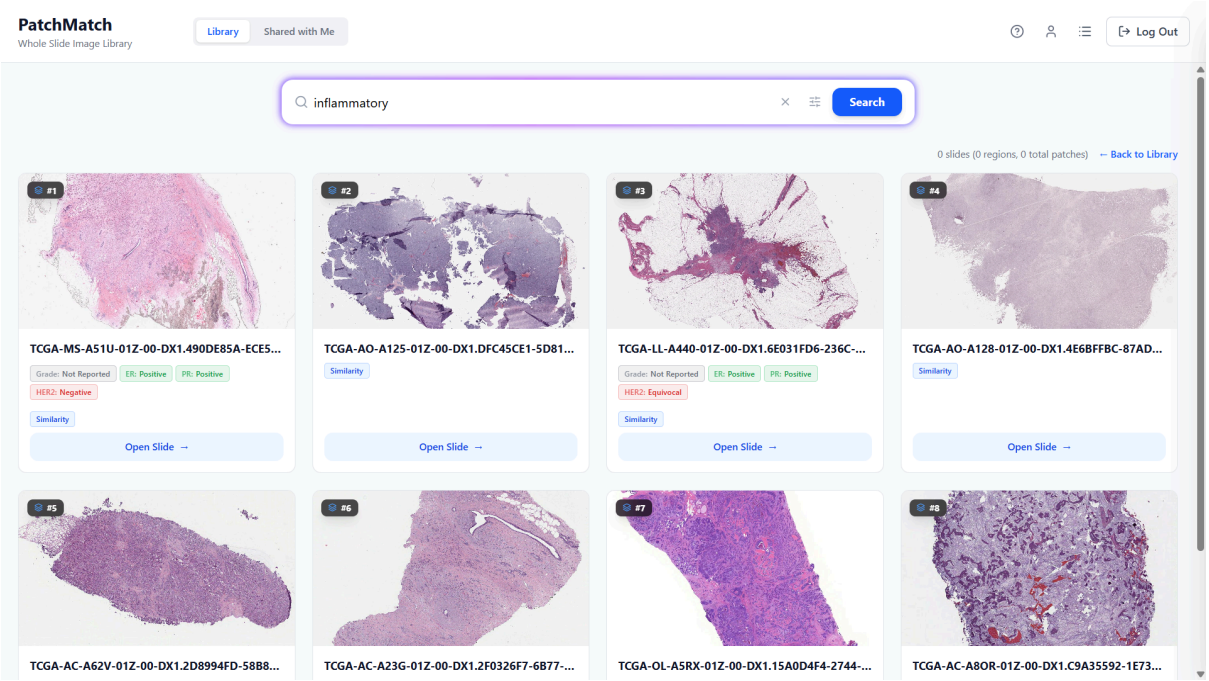
9.17 Side-by-Side View

The Side-by-Side View provides a powerful comparative workspace, typically accessed after performing a slide-to-slide search. This mode allows pathologists to visually correlate the original query case with a retrieved result in a split-screen interface, displaying the Query slide on the left and the Result slide on the right. Each side can be navigated independently, enabling the simultaneous examination of tissue architecture and cellular patterns at different magnifications. Users can utilize the Swap tool at the bottom right to quickly switch slide positions or click Exit to return to the standard single-slide viewer. This specialized view is essential for validating similarity results and performing comprehensive differential diagnosis across multiple cases.



9.18 Search Slides with Text

The system supports a powerful text-based search that allows you to find slides using natural language. Instead of just searching by filename, you can retrieve slides by typing diagnostic terms (e.g., 'invasive carcinoma'), clinical details from reports, or even your own saved notes and annotations. The search also understands descriptive queries about tissue appearance (e.g., 'dense nuclei') and can filter slides based on cellular metrics (til > 0.05). This integrated approach helps you find the exact cases you need quickly, drawing information from every part of the slide's record.

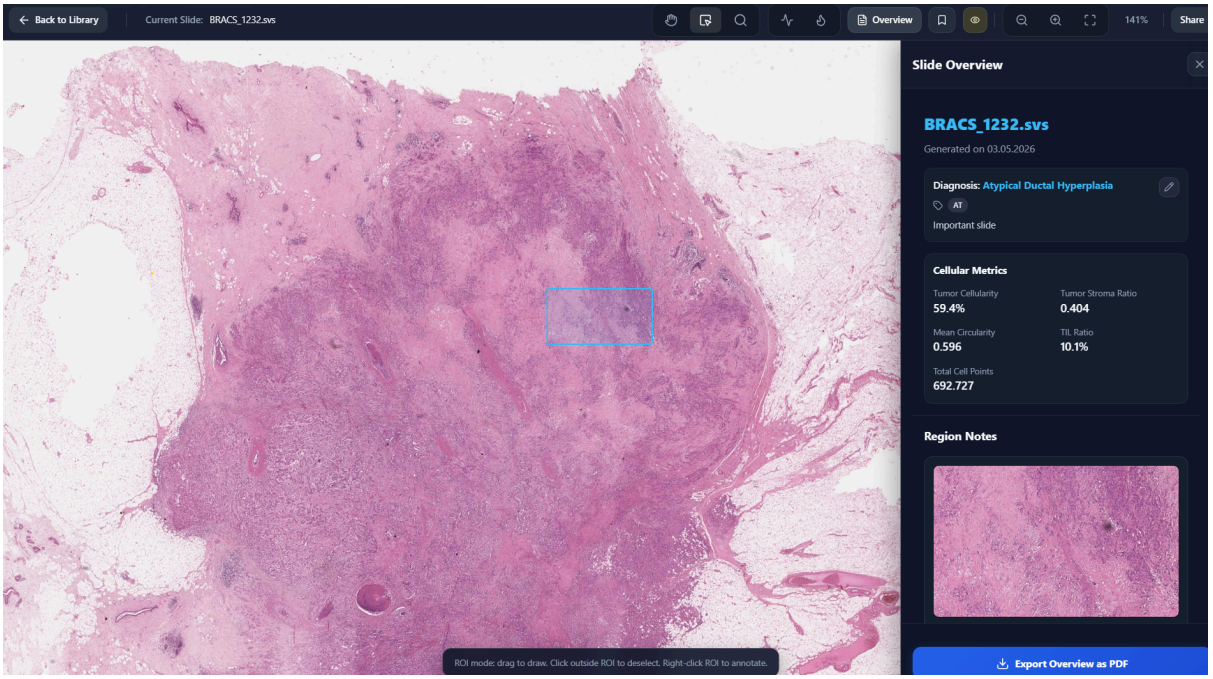


9.19 Slide Overview

The Slide Overview panel acts as a comprehensive clinical report for the current case. It aggregates data from both global automated analysis and manual user annotations to provide a unified diagnostic view:

- Global Diagnostics & Metrics: The top section displays the slide name, manually entered diagnosis, and automated Cellular Metrics for the entire slide (such as Tumor Cellularity, TIL, and TSR).
- Region Notes & Annotations: As users select and annotate specific areas, they are added to the Region Notes section. Each entry includes a high-resolution screenshot of the region, its localized cellular metrics, and the user's custom notes or diagnosis for that specific area.
- Reporting: All information within this panel, including the global statistics and individual region annotations, can be consolidated and downloaded as a professional documentation via the Export Overview as PDF button.

This panel ensures that all findings, from high-level slide statistics to granular regional observations, are organized and ready for clinical review or sharing.



9.20 Export Overview

The Export Overview feature allows users to download the complete contents of the Slide Overview panel as a professional PDF document. By clicking this button, all global cellular metrics, clinical notes, and annotated regions are consolidated into a portable file, providing a static version of the slide's digital report.

BRACS_1232.svs

Generated on 03.05.2026

AT

Important slide

Cellular Metrics

Tumor Cellularity
59.4%

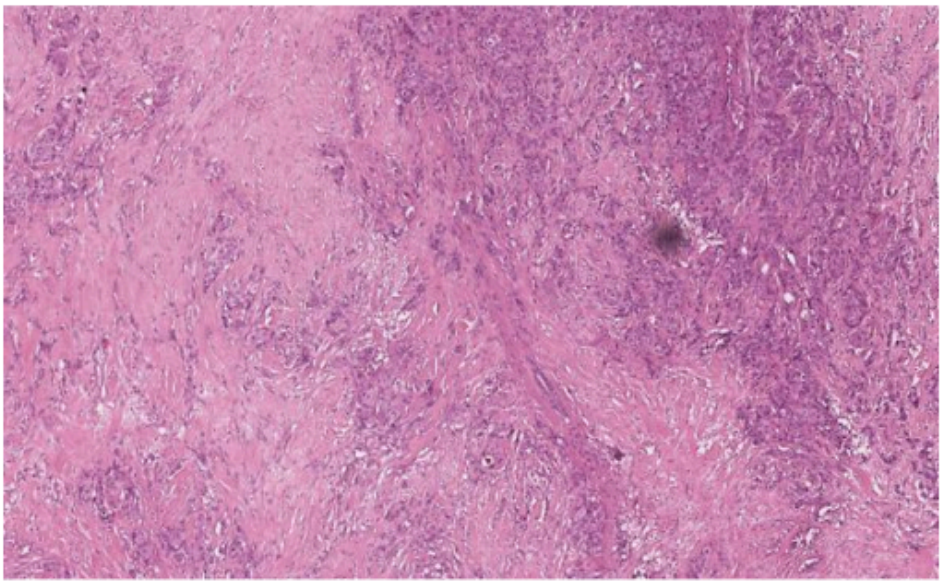
Tumor Stroma Ratio
0.404

Mean Circularity
0.596

TIL Ratio
10.1%

Total Cells
692.727

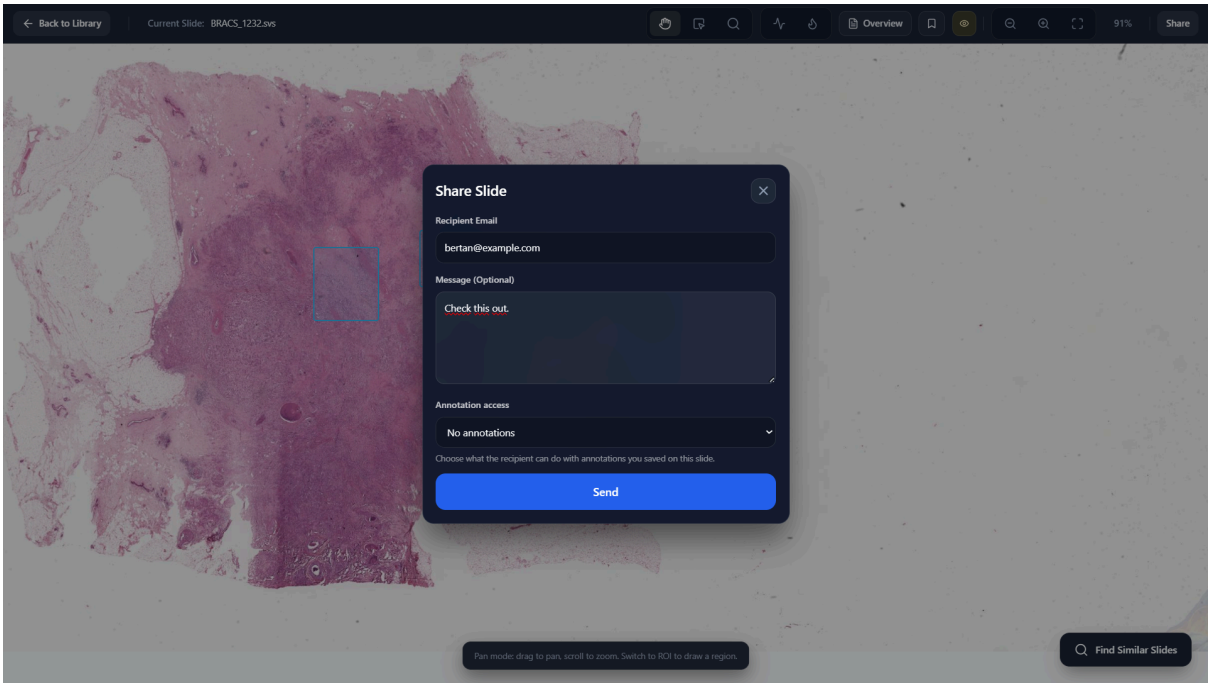
Region Notes



important region

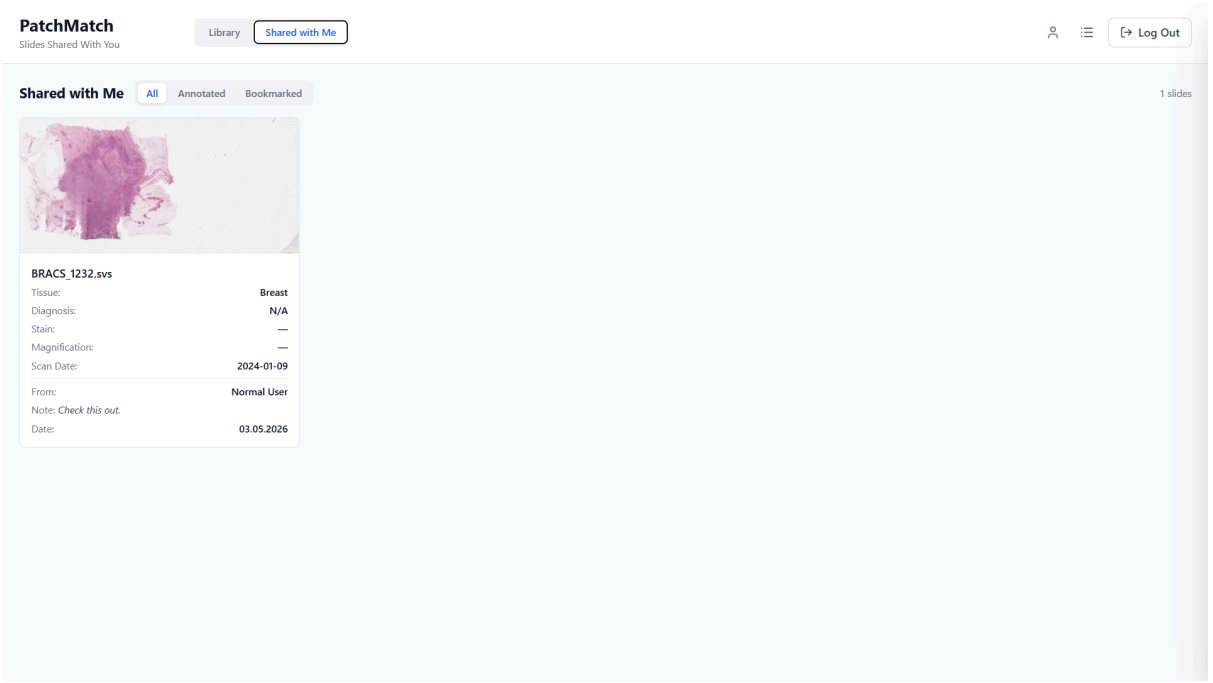
9.21 Share a Slide

Users can share slides or analysis results with others by generating a shareable link or granting access permissions. This feature facilitates collaboration and allows multiple users to work on the same data.



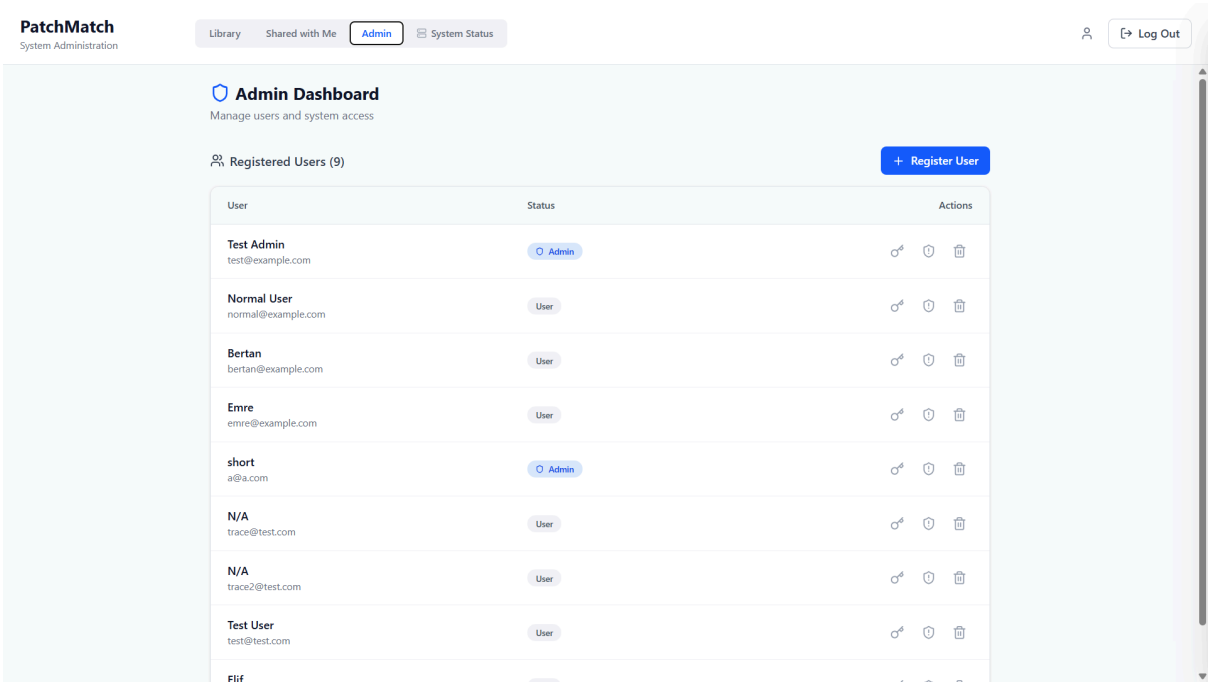
9.22 Shared with Me Page

This page displays slides and resources that have been shared with the user. It allows users to easily access collaborative content and continue working on shared analyses.



9.23 Admin Dashboard

The Admin Dashboard provides administrative control over the system. Administrators can manage users, configure system settings, monitor performance, and control available features. This ensures the system operates efficiently and securely.

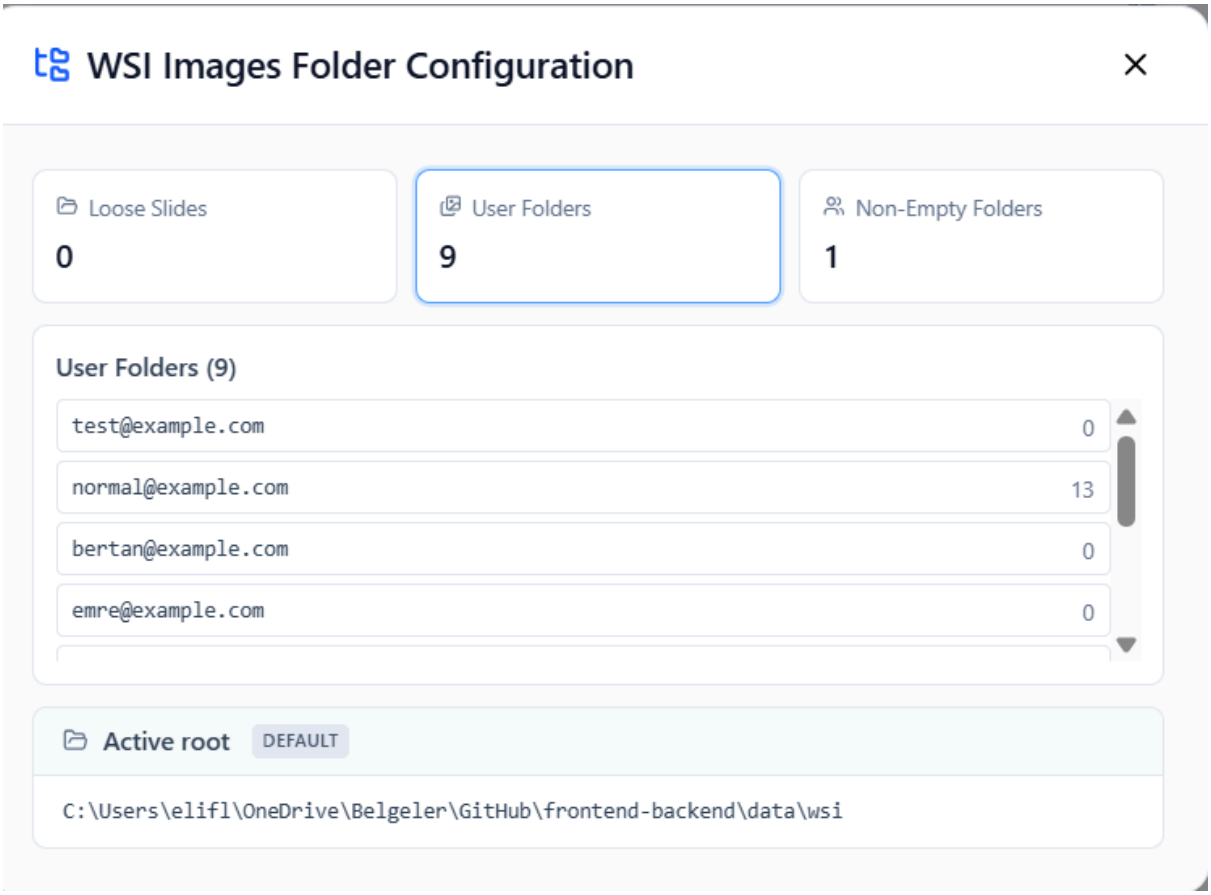


9.24 Inspect Folders

The Inspect Folders modal provides an administrative overview of how Whole Slide Images (WSI) are organized within the system's storage. It allows users to monitor the distribution of files across different categories:

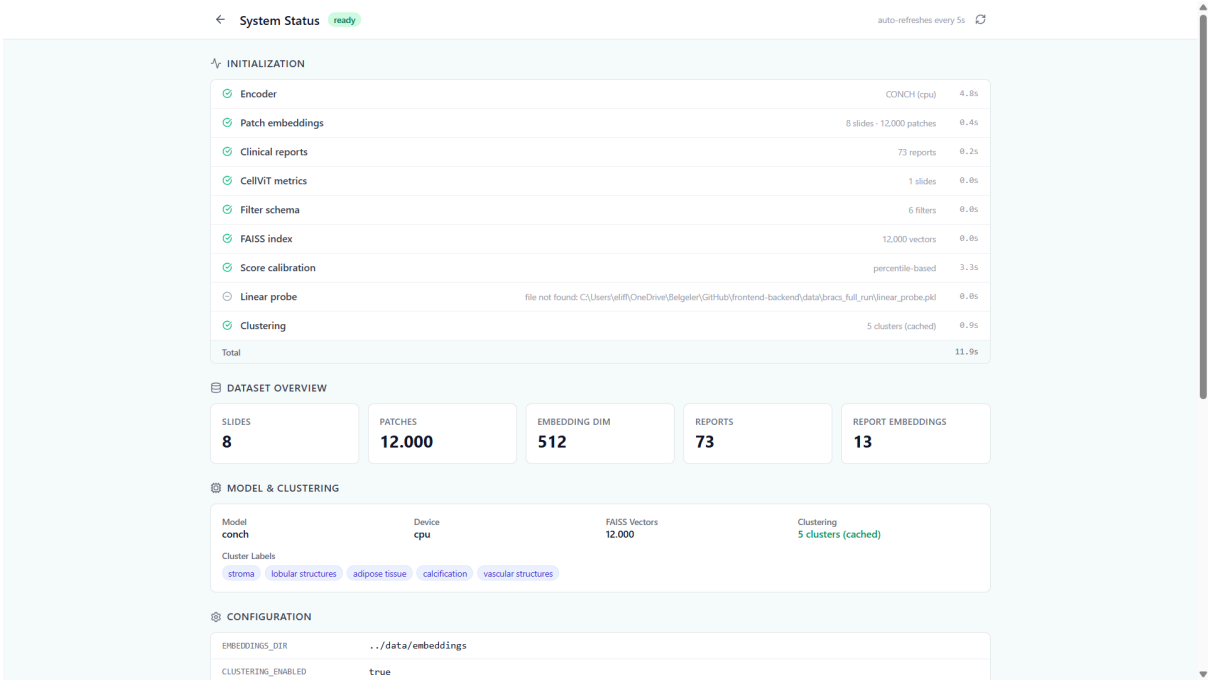
- Folder Statistics: Displays the total count of "Loose Slides" (slides not assigned to a specific user folder), "User Folders," and "Non-Empty Folders" currently detected by the system.
- User Directory Overview: Provides a detailed list of all user-specific folders, showing the number of slides associated with each account or directory.
- Active Root Path: Displays the current system path (Active Root) where the application is looking for slide data, ensuring transparency regarding the data source.

This tool is essential for verifying that newly added slides are correctly recognized and categorized within the library's file system.



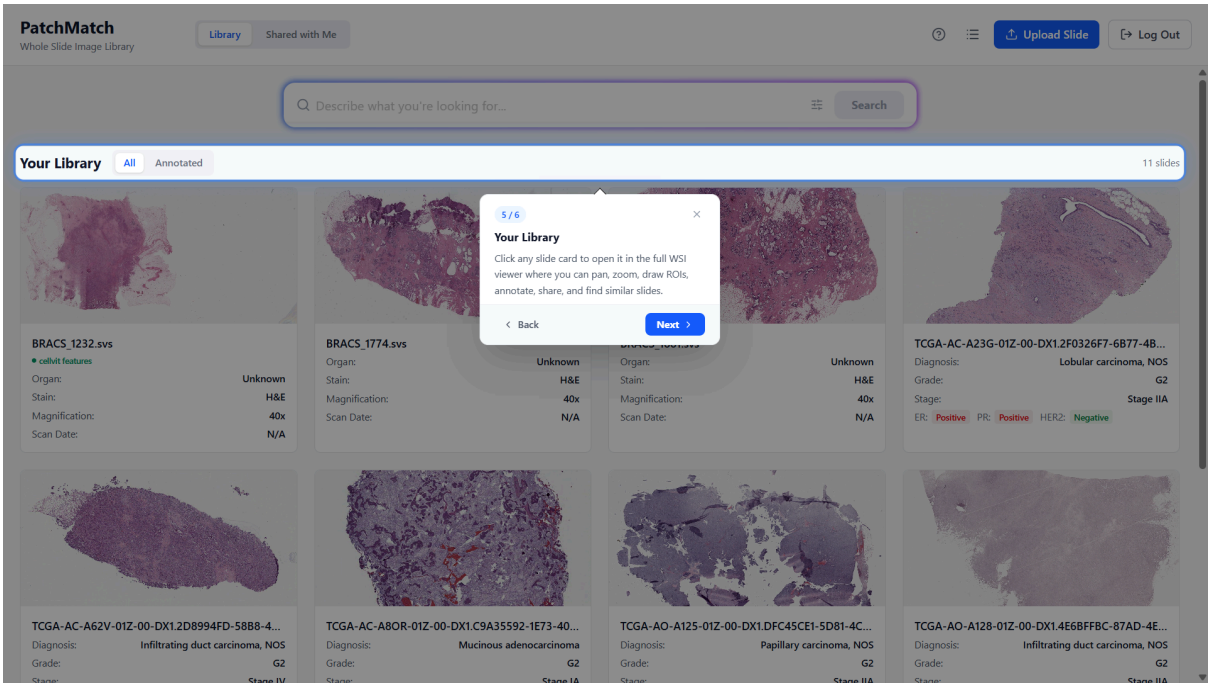
9.25 System Status Page

The System Status Page, accessible only to administrators, provides a real-time health check of the application's core components. It monitors the Initialization of AI models and search indices, displays Dataset Statistics (total slides, patches, and reports), and verifies the current system Configuration. This page ensures that all background services are fully operational and ready for analysis tasks.



9.26 Take a Tour

The “Take a Tour” feature provides an interactive introduction to the system. It guides new users through the main functionalities and interface components, helping them quickly become familiar with how to use the platform.



10. Glossary

IFF: International Image Interoperability Framework

WSI (Whole Slide Image): Complete digitized microscopy slides with gigapixel sizes (50,000 × 50,000 pixels) that pathologists examine and that PatchMatch searches through.

ROI (Region of Interest): A specific area of a WSI that the user selects for retrieval. Instead of comparing entire slides, users can select a small region to find similar patches.

API (Application Programming Interface): A set of rules that lets different software talk to each other. PatchMatch uses APIs to connect the frontend with the backend and to access external services.

KVKK (Kişisel Verilerin Korunması Kanunu): Turkey's data protection law. Requires storing patient data securely and getting consent before processing medical images.

GDPR (General Data Protection Regulation): Europe's data privacy law. Similar to KVKK. Both laws guide how to store and protect WSI data.

FastAPI: A modern Python framework for building APIs that handle requests between the frontend and backend services, like retrieval and embedding generation.

React: A JavaScript library for building user interfaces that is used in this project to create the interactive image viewer, where users can navigate and annotate WSIs, and view retrieval results.

PostgreSQL: An open-source relational database. Stores user accounts, WSI metadata, annotations, and session information for PatchMatch.

Google Cloud: A cloud platform that provides servers and storage. PatchMatch uses it to host the backend and therefore handle the computational demands of processing large images.

LLM (Large Language Model): AI models trained on text that can understand and generate language that can help interpret pathology report descriptions.

UML (Unified Modeling Language): A standard way to visualize software design through diagrams that is used for class diagrams, sequence diagrams, and use case models in this report.

11. References

[1] M. Y. Lu, B. Chen, D. F. K. Williamson, R. J. Chen, I. Liang, T. Ding, G. Jaume, I. Odintsov, L. P. Le, G. Gerber, A. V. Parwani, A. Zhang, and F. Mahmood, "A visual-language foundation model for computational pathology," Nature Medicine, vol. 30, pp. 863–874, Mar. 2024. [Online]. Available: https://doi.org/10.1038/s41591-024-02856-4

[2] M. Douze, A. Guzhva, C. Deng, J. Johnson, G. Szilvasy, P.-E. Mazaré, M. Lomeli, L. Hosseini, and H. Jégou, "The Faiss library," arXiv preprint arXiv:2401.08281, 2024. [Online]. Available: https://arxiv.org/abs/2401.08281

[3] T. Ding et al., "Multimodal Whole Slide Foundation Model for Pathology," arXiv preprint arXiv:2411.19666, 2024. [Online]. Available: https://arxiv.org/abs/2411.19666

[4] F. Hörst, M. Rempe, L. Heine, C. Seibold, J. Keyl, G. Baldini, S. Ugurel, J. Siveke, B. Grünwald, J. Egger, and J. Kleesiek, "CellViT: Vision Transformers for precise cell segmentation and classification," Medical Image Analysis, vol. 94, p. 103143, May 2024. [Online]. Available: https://doi.org/10.1016/j.media.2024.103143

[5] The PostgreSQL Global Development Group, "PostgreSQL: The World's Most Advanced Open Source Relational Database," PostgreSQL. [Online]. Available: https://www.postgresql.org/

[6] T. Haerder and A. Reuter, "Principles of transaction-oriented database recovery," ACM Computing Surveys, vol. 15, no. 4, pp. 287–317, Dec. 1983. [Online]. Available: https://doi.org/10.1145/289.291

[7] M. Jones, J. Bradley, and N. Sakimura, "JSON Web Token (JWT)," RFC 7519, IETF, May 2015. [Online]. Available: https://www.rfc-editor.org/rfc/rfc7519

[8] OpenSeadragon Contributors, "OpenSeadragon: An open-source, web-based viewer for high-resolution zoomable images," OpenSeadragon. [Online]. Available: https://openseadragon.github.io/

[9] International Image Interoperability Framework Consortium, "IIIF: International Image Interoperability Framework," IIIF Consortium. [Online]. Available: https://iiif.io/

[10] N. Brancati, A. M. Anniciello, P. Pati, D. Riccio, G. Scognamiglio, G. Jaume, G. De Pietro, M. Di Bonito, A. Foncubierta, G. Botti, M. Gabrani, F. Feroce, and M. Frucci, "BRACS: A Dataset for BReAst Carcinoma Subtyping in H&E Histology Images," Database, vol. 2022, p. baac093, 2022. [Online]. Available: https://doi.org/10.1093/database/baac093

[11] Quilt Data, Inc., "Quilt: Manage data like code," Quilt Data. [Online]. Available: https://quiltdata.com/

[12] Docker, Inc., "Docker: Accelerated Container Application Development," Docker. [Online]. Available: https://www.docker.com/

[13] S. Ramírez, "FastAPI," FastAPI. [Online]. Available: https://fastapi.tiangolo.com/

[14] Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data (General Data Protection Regulation), Official Journal of the European Union, L 119, pp. 1–88, 4 May 2016. [Online]. Available: https://eur-lex.europa.eu/eli/reg/2016/679/oj

[15] Kişisel Verilerin Korunması Kanunu (Law on the Protection of Personal Data), Law No. 6698, Republic of Türkiye, 24 Mar. 2016. [Online]. Available: https://www.kvkk.gov.tr/Icerik/6649/Personal-Data-Protection-Law

[16] Medical Device Software — Software Life Cycle Processes, IEC 62304:2006/AMD1:2015, International Electrotechnical Commission, Geneva, Switzerland, 2015. [Online]. Available: https://www.iso.org/standard/64686.html

[17] Health Software — Part 1: General Requirements for Product Safety, IEC 82304-1:2016, International Electrotechnical Commission, Geneva, Switzerland, 2016. [Online]. Available: https://www.iso.org/standard/59543.html

[18] DICOM Standards Committee, Working Group 26, Pathology, "Supplement 145: Whole Slide Microscopic Image IOD and SOP Classes," NEMA, 2010. [Online]. Available: https://www.dicomstandard.org/News-dir/ftsup/docs/sups/sup145.pdf

[19] Health Level Seven International, "HL7 Standards," HL7. [Online]. Available: https://www.hl7.org/implement/standards/

[20] Medical Devices — Quality Management Systems — Requirements for Regulatory Purposes, ISO 13485:2016, International Organization for Standardization, Geneva, Switzerland, 2016. [Online]. Available: https://www.iso.org/standard/59752.html

[21] Medical Devices — Application of Risk Management to Medical Devices, ISO 14971:2019, International Organization for Standardization, Geneva, Switzerland, 2019. [Online]. Available: https://www.iso.org/standard/72704.html

[22] Regulation (EU) 2017/745 of the European Parliament and of the Council of 5 April 2017 on medical devices (Medical Device Regulation), Official Journal of the European Union, L 117, pp. 1–175, 5 May 2017. [Online]. Available: https://eur-lex.europa.eu/eli/reg/2017/745/oj

[23] Regulation (EU) 2024/1689 of the European Parliament and of the Council of 13 June 2024 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act), Official Journal of the European Union, L series, 12 July 2024. [Online]. Available: https://eur-lex.europa.eu/eli/reg/2024/1689/oj

[24] U.S. Food and Drug Administration, "Classify Your Medical Device — Class II Devices," FDA. [Online]. Available: https://www.fda.gov/medical-devices/overview-device-regulation/classify-your-medical-device

[25] Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 (General Data Protection Regulation), Official Journal of the European Union, L 119, pp. 1–88, 4 May 2016. [Online]. Available: https://eur-lex.europa.eu/eli/reg/2016/679/oj

[26] Personal Data Protection Law, Law No. 6698, Republic of Türkiye, Official Gazette No. 29677, 7 Apr. 2016. [Online]. Available: https://www.kvkk.gov.tr/Icerik/6649/Personal-Data-Protection-Law

[27] OMG Unified Modeling Language Specification, Version 2.5.1, Object Management Group, Dec. 2017. [Online]. Available: https://www.omg.org/spec/UML/2.5.1/

[28] IEEE Recommended Practice for Software Requirements Specifications, IEEE Std 830-1998, IEEE, New York, NY, USA, 1998. [Online]. Available: https://standards.ieee.org/ieee/830/1222/

[29] R. T. Fielding, "Architectural Styles and the Design of Network-based Software Architectures," Ph.D. dissertation, Univ. of California, Irvine, 2000. [Online]. Available: https://ics.uci.edu/~fielding/pubs/dissertation/top.htm

[30] OpenAPI Initiative, "OpenAPI Specification, Version 3.0.3," The Linux Foundation, 2020. [Online]. Available: https://spec.openapis.org/oas/v3.0.3

[31] Microsoft Corporation, "TypeScript: JavaScript With Syntax For Types," Microsoft. [Online]. Available: https://www.typescriptlang.org/