Bilkent University

Department of Computer Engineering

Senior Design Project

T2515

PatchMatch

Analysis and Requirement 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

19/12/2025

Contents

Analysis and Requirement Report

PatchMatch: An Intelligent Whole-Slide Image Retrieval System for Digital Pathology

1. Introduction

In recent years, improvements in digital pathology have replaced traditional microscope studies with the digital field which enables the acquisition and storage of WSIs. These images provide visual detail about tissue samples at high resolutions. Although digital pathology has improved accessibility in the fields of health, education, and research, it has significant challenges related to the efficient analysis of WSIs. Manually browsing large collections of WSIs to identify the clinically similar cases becomes impractical.

Current digital pathology platforms generally focus on visualizing WSIs instead of content-based retrieval, and pathologists often rely on manual searches to compare a new WSI with previous samples. In systems where retrieval is available, slide-level similarity searches are performed frequently instead of region-level. As capturing localized pathological patterns provides more accurate case comparison, slide level similarity searches are insufficient. Additionally, platforms that combine WSI and textual information to provide users with explainable reasoning are less frequent in existing systems.

To address these challenges, the PatchMatch project develops an intelligent WSI retrieval system that empowers pathologists to quickly find visually and clinically similar cases from large-scale digital pathology databases. By leveraging state-of-the-art computer vision and scalable search methods, our system transforms high-resolution pathology slides into efficient, comparable representations, enabling instant case-based retrieval and visualization. This innovation supports faster, more consistent, and explainable diagnoses while reducing workload for specialists, enhancing research reproducibility, and improving overall diagnostic confidence in medical practice.

This report outlines the analysis, requirements, and design considerations for PatchMatch. The report serves as a contractual reference between stakeholders and developers and provides a solid foundation for the system design and implementation phases to be carried out in CS492.

2. Current System

The digital pathology ecosystem includes enterprise platforms, research prototypes, and AI assistance tools. These systems support important aspects of digital pathology workflows such as image storage, visualization, and diagnostic assistance. However, they cannot fully address the need for fast, explainable, and flexible similarity retrieval. An analysis of representative systems is presented below to examine the functionalities offered by current solutions and the limitations associated with their use:

1. **Sectra Digital Pathology:**
   A PACS-grade diagnostic workstation that provides viewing, annotation, teaching workflows. It uses third-party AI tools to provide explainability in addition to annotation. However, multimodal fusion and text-to-image search are not core features, and similarity-based image retrieval is not supported [1].
2. **Iron Mountain:**
   A service for secure storage and management of WSIs. Although it provides archival capabilities and metadata-based retrieval, it doesn’t focus on CBIR, explainability mechanisms, or multimodal analysis [2].
3. **Lagotto (Huron):**
   A patented CBIR system designed to find similar WSIs. It links retrieved slides to associated reports and metadata with large archives. However, its retrieval mechanism is designed based on slide-level similarity instead of ROI level retrieval. In addition, Lagotto does not provide explainable similarity reasoning, text-to-image retrieval, multi-resolution embeddings, or personalized retrieval sessions [3].
4. **SMILY:**
   SMILY is a deep-learning–based reverse image search system for histopathology based on ROI-driven patch-level similarity queries. As it is a research prototype, it lacks production-level infrastructure and clinical integration. In addition, it does not provide explanations for similarity and session-level personalization [4].
5. **Yottixel:**
   A WSI search engine that uses Bunch of Barcodes (BoB) indexing to support large-scale, primarily slide-level similarity retrieval. In addition to the lack of ROI level matching, it also lacks explainability, multimodal fusion, text-based queries, or personalized exploration [5].

6. **Fast Pathology:**
   An open-source platform built on the FAST framework. It enables the real-time visualization and deployment of deep learning models for WSI, providing classification, segmentation, and detection capabilities. However, FastPathology does not support similarity-based retrieval functionality. In addition, explainability is limited to model prediction and multimodal fusion, and personalized retrieval workflows are not supported [6].
7. **Ibex Medical Analytics:**
   A task-specific AI system for detection and grading in digital pathology supporting predefined diagnostic tasks. It is designed to support predefined diagnostic tasks rather than general-purpose similarity retrieval or case comparison across different datasets [7].

Limitations of the Current System

The existing digital pathology solutions present the following key limitations:

- **Limited Similarity-Based Retrieval:** Most systems do not provide CBIR. If it is available as in Lagotto, similarity search is based on the slide-level comparisons rather than ROI analysis which provides more accurate case comparison.
- **Lack of Explainable Retrieval:** Existing systems provide limited insight into why retrieved cases are considered similar, as explainability is often tied to task-specific model outputs rather than retrieval-based reasoning.
- **Absence of Multimodal Querying:** Current systems rarely support multimodal search which considers image content and textual information.
- **Limited Personalization and Exploration:** Personalized retrieval sessions, cohort-based exploration, and session-level analysis are not commonly supported.
- **Fragmentation Between Research and Clinical Systems:** Research prototypes lack clinical readiness even if they demonstrate successful retrieval techniques. It is therefore difficult to translate these improvements into clinical practice as enterprises prioritize workflow integration and storage over advanced retrieval capabilities.

These limitations highlight the need for an integrated similarity retrieval system that combines region-level and slide-level search, explainable retrieval, and multimodal analysis within a clinically usable framework for digital pathology.

3. Proposed System

3.1 Overview

The PatchMatch system aims to be an advanced, similarity-based retrieval platform for digital pathology. The system uses deep learning and indexing to find visually similar cases, both across entire slides and within specific regions. PatchMatch addresses the limitations of existing digital pathology systems by offering flexible, explainable, and multimodal similarity exploration in a clinically usable system.

The proposed solution supports multiple forms of similarity analysis to address different diagnostic and research needs

- **Whole-Slide Similarity Exploration:** This feature allows users to upload a whole-slide image and browse visually similar cases in the database. The results are shown together with relevant metadata, reports, and similarity scores to help users compare cases more easily.
- **Region-Level and Multimodal Similarity Analysis:** Users can select specific ROIs within a slide and compare them against stored regions from other WSIs. The system also supports text-based and combined image–text queries to enable pathology report information to complement visual similarity.

The PatchMatch architecture is built to ensure scalability, reliability, and clinical usability, and includes the following main components:

- **Data Ingestion and Preprocessing Module:** Responsible for ingesting WSIs and associated metadata, performing tiling and multi-resolution processing, and preparing data for feature extraction.
- **Feature Extraction and Indexing Engine:** Responsible for computing feature embeddings for image patches and slides, and organizing these embeddings into a scalable vector database to efficiently compare similarities.
- **User Interaction and Visualization Layer:** Responsible for providing an interactive viewer, ROI selection, and explainable visualizations to support informed interpretation of results.

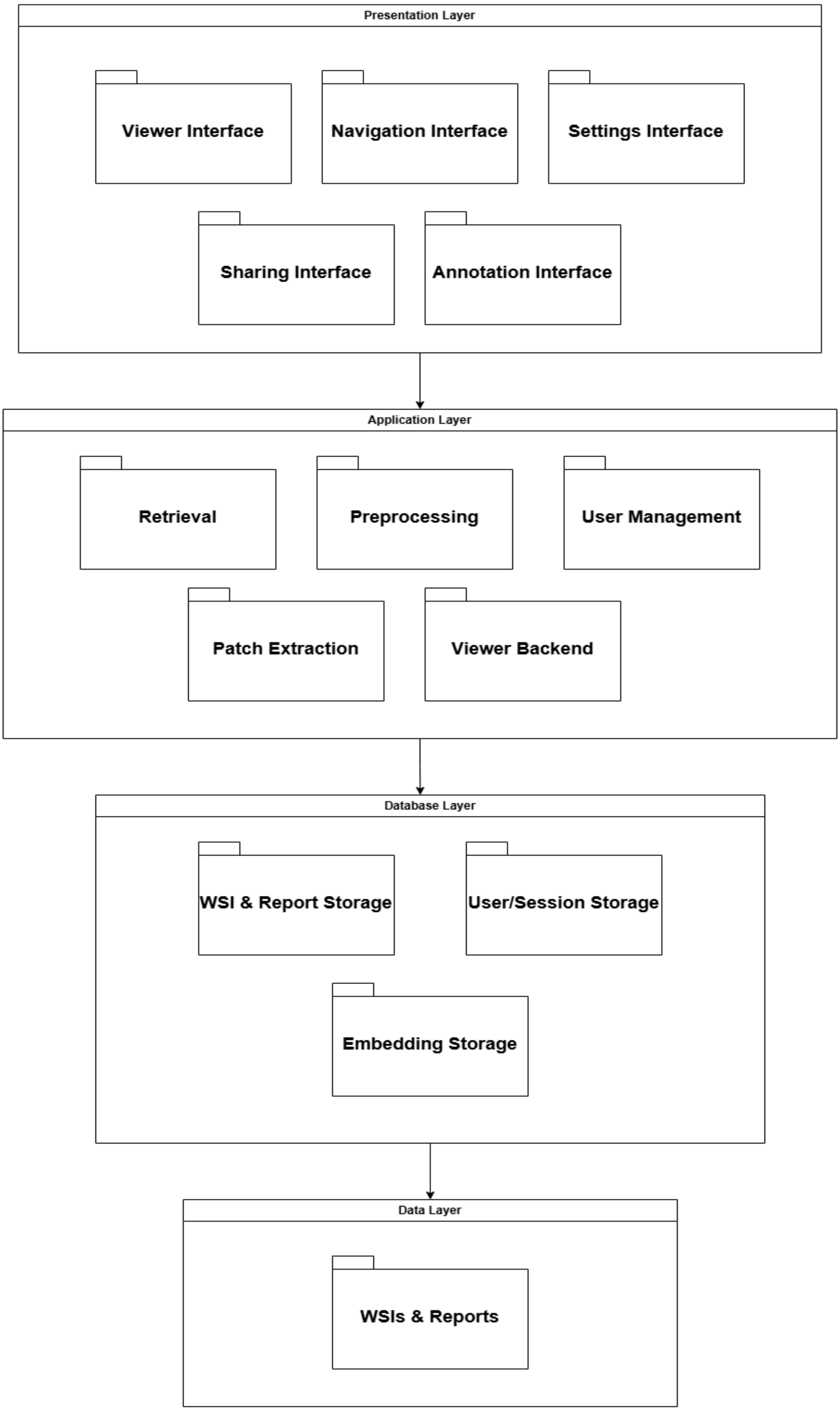

Figure 1. High Level System Diagram

3.2 Functional Requirements

The functional requirements of the PatchMatch system provide the core operations and capabilities that the software must perform. These requirements ensure the efficient management, retrieval, analysis, and interpretation of WSIs and associated textual information. They also support explainability, multimodal interaction, and collaborative workflows.

Whole Slide Image Management

- The system must allow authorized users to upload, store, and manage whole-slide images (WSIs).
- Each WSI must be associated with relevant metadata, such as staining type, tissue origin, and related reports.
- Access to WSIs and metadata must be restricted based on user authorization.

Viewer: Navigation, Zoom and Selection

- The system must provide an interactive viewer for opening and exploring WSIs.
- Users must be able to zoom smoothly across multiple magnification levels and pan across the slide.
- Users must be able to select one or more regions of interest (ROIs) for retrieval, analysis, or annotation.

Feature Extraction of Patches

- The system must extract image patches from whole-slide images (WSIs) to support localized analysis.
- Feature embeddings must be computed for extracted patches using deep learning–based methods and stored in the system.
- The computed embeddings must be used to compare the visual characteristics of patches against embeddings of patches previously stored in the WSI repository.

Multi-Resolution Embeddings

- The system must generate multi-resolution representations of WSIs.
- These representations must capture both fine-grained cellular features and higher-level tissue structures.
- Multi-resolution embeddings must support retrieval across different magnification levels.

Indexing and Database

- The system must stıre feature embeddings in a vector-based database.
- The indexing mechanism must enable efficient similarity search over large WSI collections.
- The system must support scalable retrieval as the database grows.

Image-to-Image Retrieval (Whole Slide)

- Given a query WSI, the system must retrieve the top-*k* most visually similar WSIs.
- Retrieved results must be presented with thumbnails, key metadata, and similarity scores.
- The presentation must allow users to compare retrieved slides efficiently.

ROI-to-ROI Retrieval

- Given a user-selected ROI, the system must retrieve visually similar ROIs or patches from other WSIs.
- Each retrieved ROI must be displayed together with its parent slide and spatial location.
- Similarity scores must be provided to indicate retrieval relevance.

Text-to-Image Retrieval

- The system must allow users to enter free-text queries describing visual or pathological patterns.
- The system must retrieve WSIs or ROIs whose visual content and associated reports semantically match the query.
- Retrieval must consider both visual features and textual information.

Image + Text Paired Retrieval

- The system must support paired queries consisting of an image (WSI or ROI) and a short textual description.
- Both inputs must be jointly considered during retrieval.
- Retrieved results must satisfy both visual similarity and text-based constraints.

Multimodal Fusion with Reports

- The system must combine image embeddings with text embeddings extracted from pathology reports.

- Similarity computation must account for both visual characteristics and semantic report information.
- Multimodal fusion must improve retrieval relevance and interpretability.

Explainable Retrieval

- For each retrieved result, the system must provide an explanation and reason of similarity.
- Explanations may include visual indicators such as highlighted regions or heatmaps.
- The system must also present shared visual or semantic features, such as similar patterns or report keywords.
- Explanations must address both image-based and report-based similarity factors when available.

User Annotation

- The system must allow users to annotate WSIs and ROIs.
- Users must be able to draw regions and add textual comments.
- Annotations must be stored persistently and accessible for later review.

Data Sharing

- The system must allow users to share selected images, retrieval results, and annotations with other users.
- All shared content must follow the authorization and access control rules.
- The system must support collaborative workflows for clinical, educational, and research purposes.

3.3 Nonfunctional 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.

3.3.1 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.

3.3.2 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 uploading the data, annotating text on the data, selecting the region of interest.
- In the case of errors the user should see an informative message.
- The data of WSIs should be preserved in a non-corrupted way. If data corruption happens, the administrator and the uploader of the file should be notified.
- In case of a system failure, the system should be operable back in under six hours.

3.3.3 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 region of interest, the most similar top-K images should be displayed under 7 seconds for 90% of the queries.
- For a text query, the most associated top-K images should be displayed under 7 seconds for 90% of the queries.
- The system should encode the tiles in a fast manner to create the embeddings and make them accessible.

3.3.4 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.
- Operational parameters such as the “K” in the top-K most similar image retrievals should be adjustable.
- Users should access support resources such as FAQs and troubleshooting guides.
- Different medical file formats should be handled ensuring compatibility.

3.3.5 Scalability

- The retrieval engine should search images from a few thousands of WSIs to a much larger dataset without compromising latency.
- Multi-resolution embeddings should be stored using compression or vector representations to decrease the amount of required storage space. This way, more embeddings can be stored with an increased amount of data.
- 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.

3.3.6 Security

- The system should require authenticated access to see patient data.
- The WSIs, embeddings, and associated reports should be stored in an encrypted manner.
- 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.

3.4 Pseudo Requirements

3.4.1 Platform and Accessibility

- PatchMatch should be accessible from commonly used modern web browsers such as Chrome, Firefox, Safari, and Edge since it is a web-based application.
- High-resolution monitors typically used in digital pathology workflows should be supported.
- A web browser should be enough to access the application, no special software should be downloaded.

3.4.2 Technology and Implementation Assumptions

- The system should rely on server-side processing for WSI handling and embedding computation.
- Machine learning frameworks should be used to achieve deep learning-based feature extraction while ensuring reproducibility and maintainability.
- A dedicated vector storage like Qdrant should be used for managing embeddings efficiently.
- Metadata, annotations and access control information should be stored in a separate structured database like PostgreSQL.

3.4.3 Data and File Handling

- The whole-slide images should be provided in commonly used digital pathology formats, such as SVS and NDPI.
- Original WSIs should be treated as read-only artifacts. All derived data including tiles, embeddings and annotations should be generated and stored separately.
- The relationship between the patches and the associated WSIs should be preserved.

3.4.4 Model and Retrieval Assumptions

- Multiple AI models may coexist within the platform to achieve targeted performance.
- Retrieval results should be based on similarity metrics.
- Explainability features should not change the retrieval results, only help the user visualize the similar parts.

3.4.5 Clinical Usage Assumptions

- PatchMatch should not be a diagnostic tool, it should only assist the pathologists.
- The system assumes integration into existing pathology workflows without replacing standard diagnostic procedures.

3.4.6 Legal, Ethical, and Compliance Assumptions

- The system must comply with medical data protection regulations.
- Patient-identifying information must be anonymized prior to data preprocessing.
- Access to data is assumed to be governed by role-based authorization policies and audit mechanisms.

3.4.7 Deployment and Operational Environment

- The system assumes deployment on reliable server infrastructure, which may include GPU-enabled resources for efficient embedding computation.
- The system assumes potential integration with external hospital systems through secure interfaces.

3.4.8 Development and Evolution

- The system must evolve iteratively, incorporating user feedback from pathologists and researchers.
- Documentation and monitoring must be maintained throughout the system lifecycle.

3.4.9 Ethical and Research Context

- Limitations arising from data distribution or model behavior are assumed to be documented and communicated to users.
- Research use of PatchMatch is assumed to clearly distinguish exploratory findings from clinical decision-making.

3.5 System Models

3.5.1 Scenarios

Login

- Use case name: Login
- Actor: User
- Entry condition: The user clicks the login button.
- Exit condition: The user enters the correct password and username.
- Flow of events:

    1. The user enters his/her username.
    2. The user enters his/her password.
    3. If password is incorrect or user does not exist:
        a. The user fails to login.
    4. Else:
        a. The user logs in.

Upload Image

- Use case name: Upload WSI
- Actor: User
- Entry condition: The user is logged in and has an svs file.
- Exit condition: The user uploads the image to the system and the image is ready to be shown.
- Flow of events:
    1. The user clicks Upload Image.
    2. The user selects an svs file from the folders.
    3. The image is uploaded.

View Uploaded Image

- Use case name: View WSI
- Actor: User
- Entry condition: The user is logged in and uploads an image.
- Exit condition: The user clicks the back button.
- Flow of events:
    1. The user uploads an image.
    2. The user selects the image from uploaded images.
    3. The image is shown.
    4. The user may choose to move, pan, zoom in/out on the image.

View Retrieved Image

- Use case name: View Retrieval Results
- Actor: User
- Entry condition: The user asks for a retrieval for an uploaded image/ROI.
- Exit condition: The user clicks the back button.

- Flow of events:
    1. The user selects an image from the retrieved image.
    2. The image is shown.
    3. The user may choose to move, pan, zoom in/out on the image.

Select Region of Interest

- Use case name: Select ROI
- Actor: User
- Entry condition: The user uploads an image.
- Exit condition: The user performs a retrieval.
- Flow of events:
    1. The user selects an uploaded image.
    2. The image is shown.
    3. The user selects a part of the image.

Image-to-Image Retrieval

- Use case name: Retrieve Images Using Images
- Actor: User, AI Model, System
- Entry condition: The user uploads an image.
- Exit condition: The system retrieves the most similar image.
- Flow of events:
    1. The user selects an uploaded image.
    2. The user asks for a retrieval.
    3. The model embeds the patches of the image.
    4. The model looks for all patch embeddings.
    5. The system shows the most similar image.

ROI-to-Image Retrieval

- Use case name: Retrieve Images Using ROIs
- Actor: User, AI Model, System
- Entry condition: The user uploads an image.
- Exit condition: The system retrieves the most similar image.
- Flow of events:
    1. The user selects an uploaded image.

2. The user chooses a region of interest.
3. The user asks for a retrieval.
4. The model performs patch embedding of the ROI.
5. The model looks for all patch embeddings.
6. The system shows the most similar image.

Text-to-Image Retrieval

- Use case name: Search Images Using Text
- Actor: User, AI Model, System
- Entry condition: The user enters a text.
- Exit condition: The system shows the most relevant image.
- Flow of events:
    1. The user enters a text and asks for a retrieval.
    2. The model gets the text and performs a text embedding.
    3. The model retrieves the most relevant image given the text.
    4. The system shows that image.

Image + Text Pair Retrieval

- Use case name: Retrieve Images Using Image and Text
- Actor: User, AI Model, System
- Entry condition: The user enters a text and uploads an image.
- Exit condition: The system shows the most relevant image.
- Flow of events:
    1. The user selects an uploaded image.
    2. The user may or may not choose a region of interest.
    3. The user enters a text.
    4. The model performs a patch and text embedding on this pair.
    5. The model finds the most relevant image.
    6. The system shows the image.

Annotate Image

- Use case name: Annotate WSI
- Actor: User
- Entry condition: The user uploads an image.

- Exit condition: The user finishes the annotation.
- Flow of events:
    1. The user selects an uploaded image.
    2. The user selects a region of the image.
    3. The user writes annotations on the region.

Share Image

- Use case name: Share Content
- Actor: User, E-mail Service
- Entry condition: The user uploads an image.
- Exit condition: The e-mail is sent.
- Flow of events:
    1. The user selects an uploaded image.
    2. The user clicks the share button.
    3. The user enters the e-mail of the person.
    4. The user may or may not choose to write a message.
    5. The user sends the image to the chosen e-mail.

3.5.2 Use Case Model

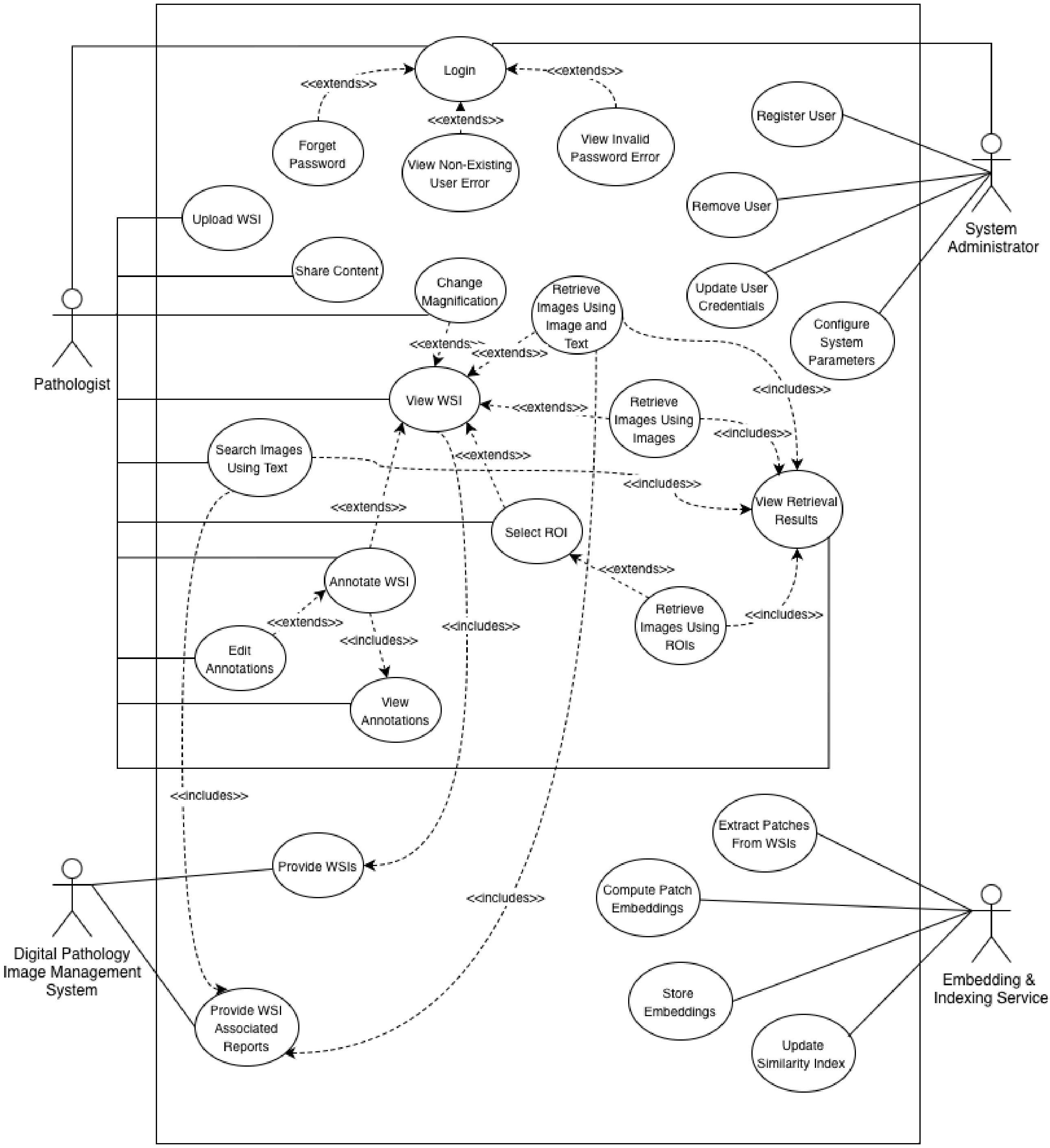

Figure 2. Use Case Diagram

3.5.3 Object and Class Model

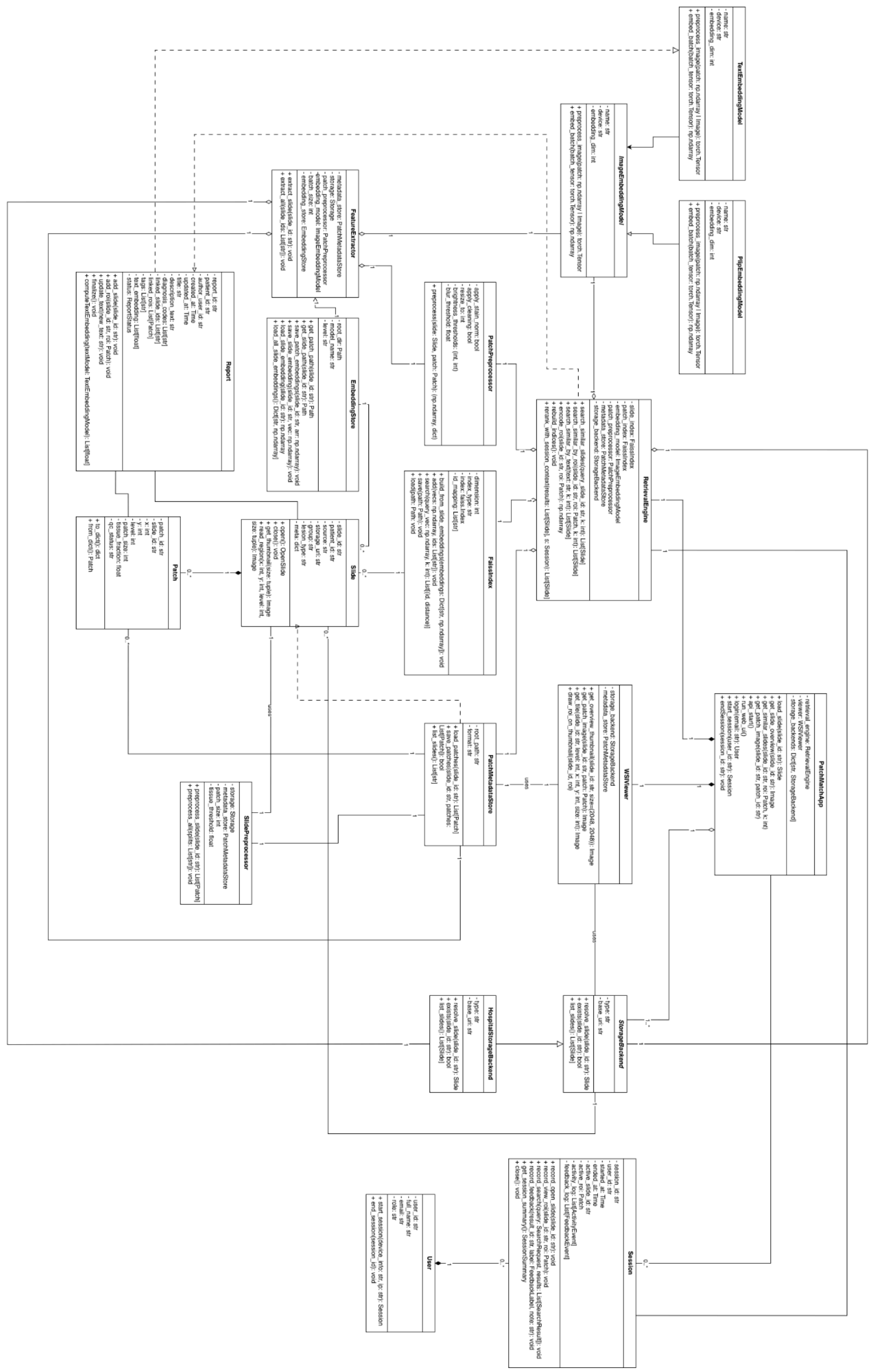

Figure 3. Object and Class Model

Note: The full resolution of the diagram is available at:

https://drive.google.com/file/d/1gXXw8Y11o7R2zlg6X7swtlKgPT5iLthJ/view?usp=sharing

3.5.4 Dynamic Models

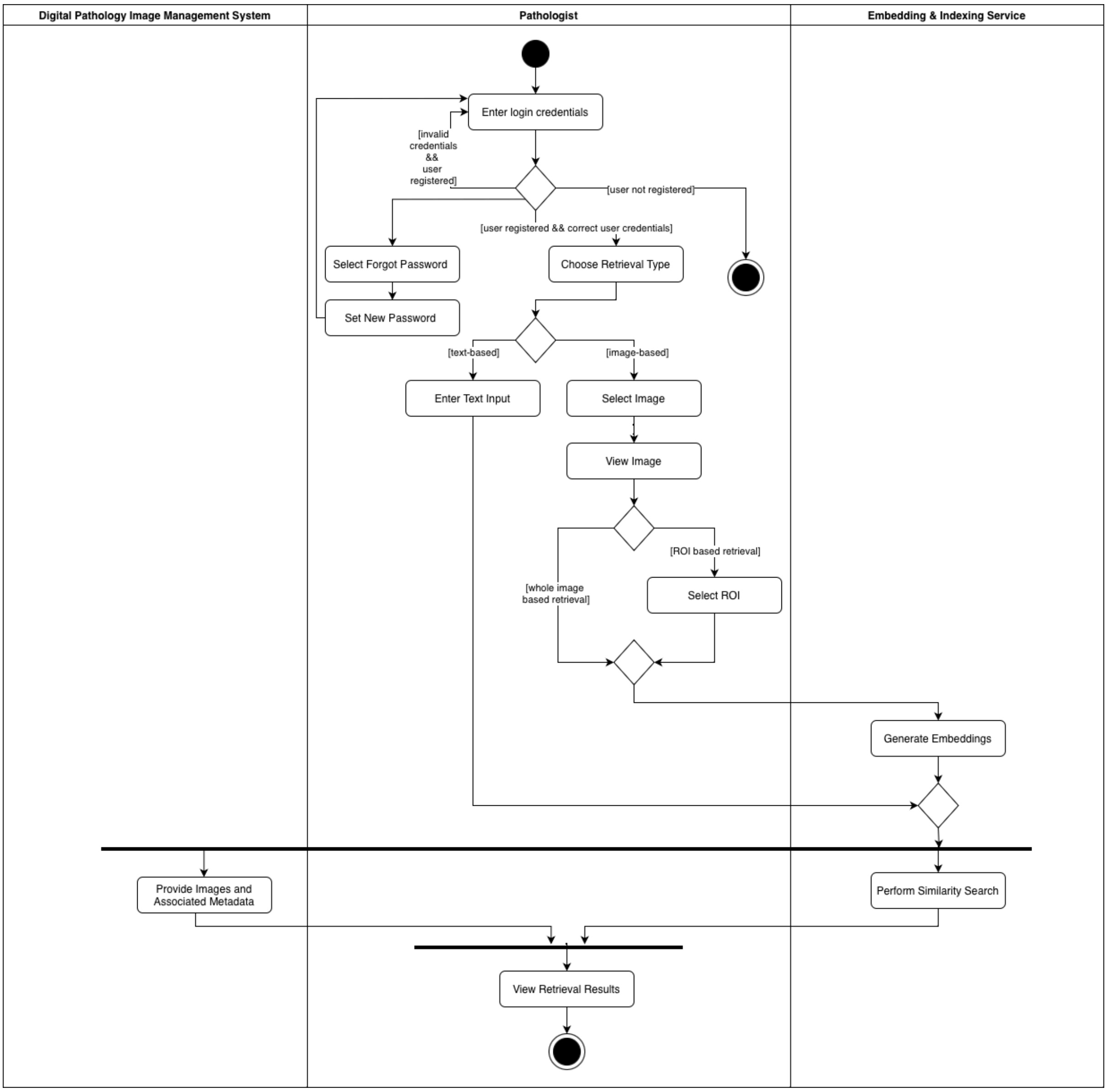

Figure 4. Image Retrieval Activity Diagram

3.5.5 User Interface - Navigational Paths and Screen Mock-ups

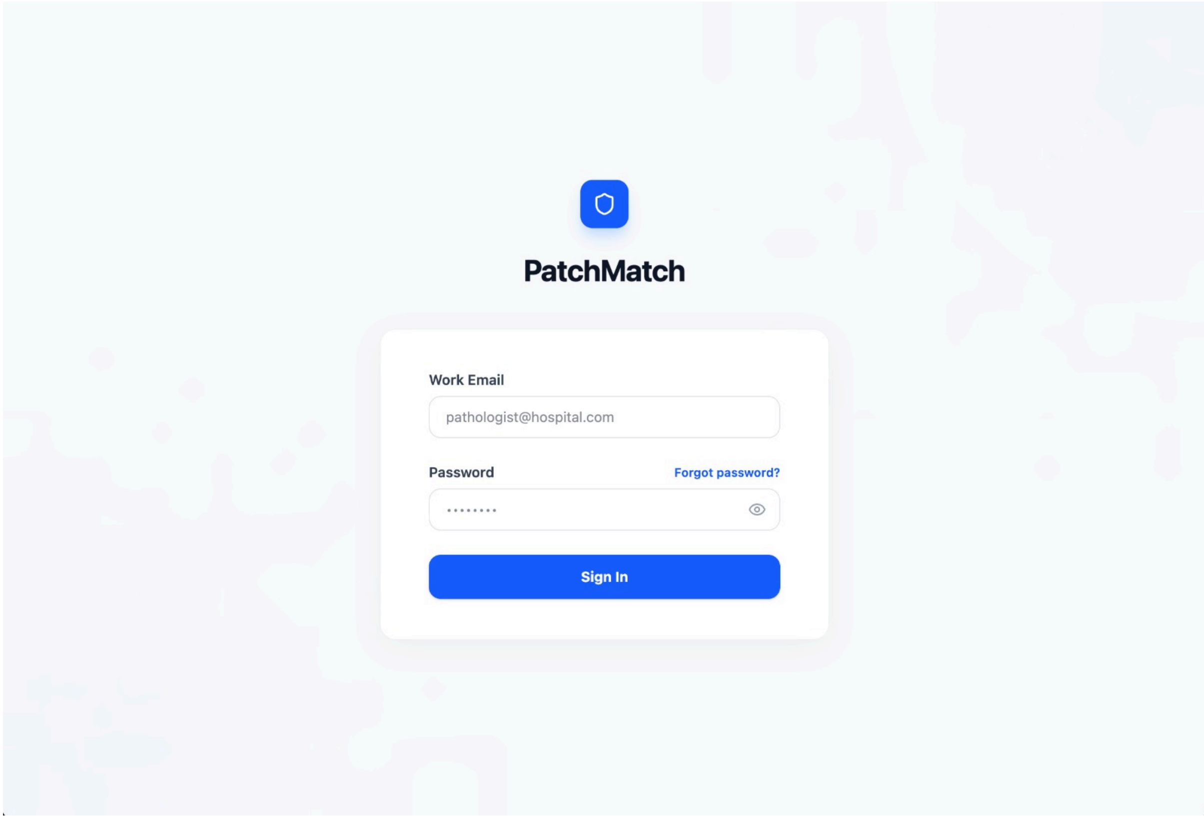

Figure 5. Login Page

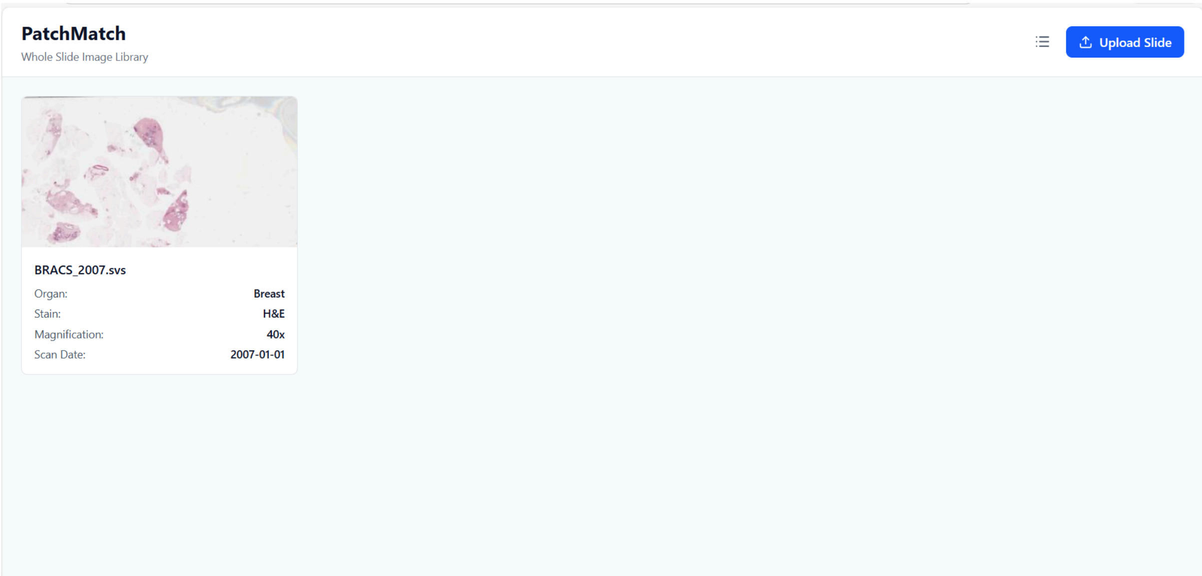

Figure 6. Home Page (Grid View)

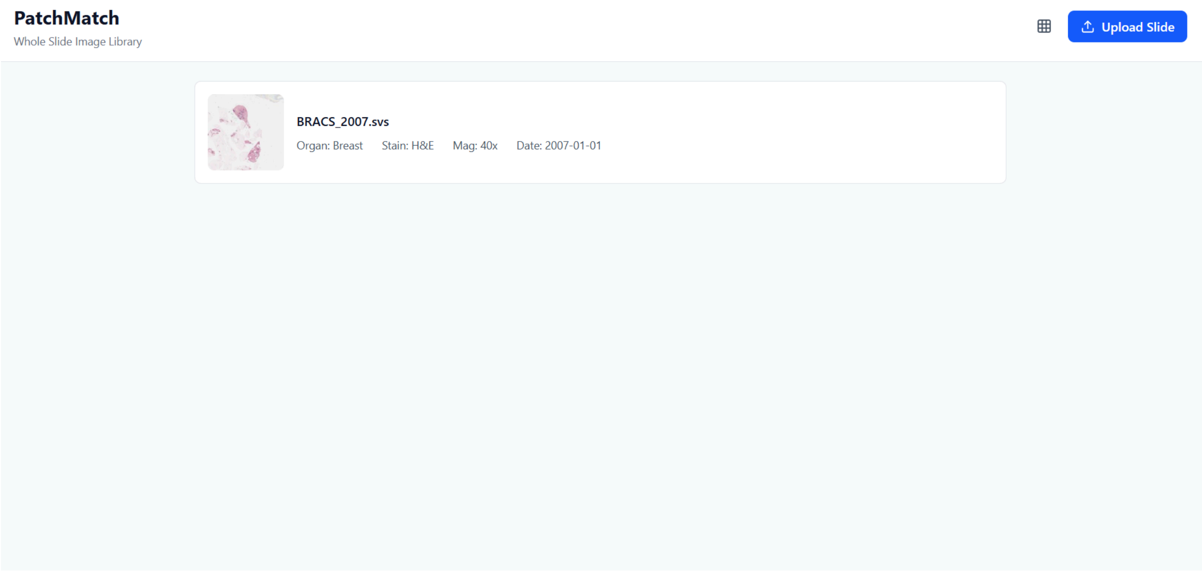

Figure 7. Home Page (List View)

Figure 8. Upload WSI Pop-Up

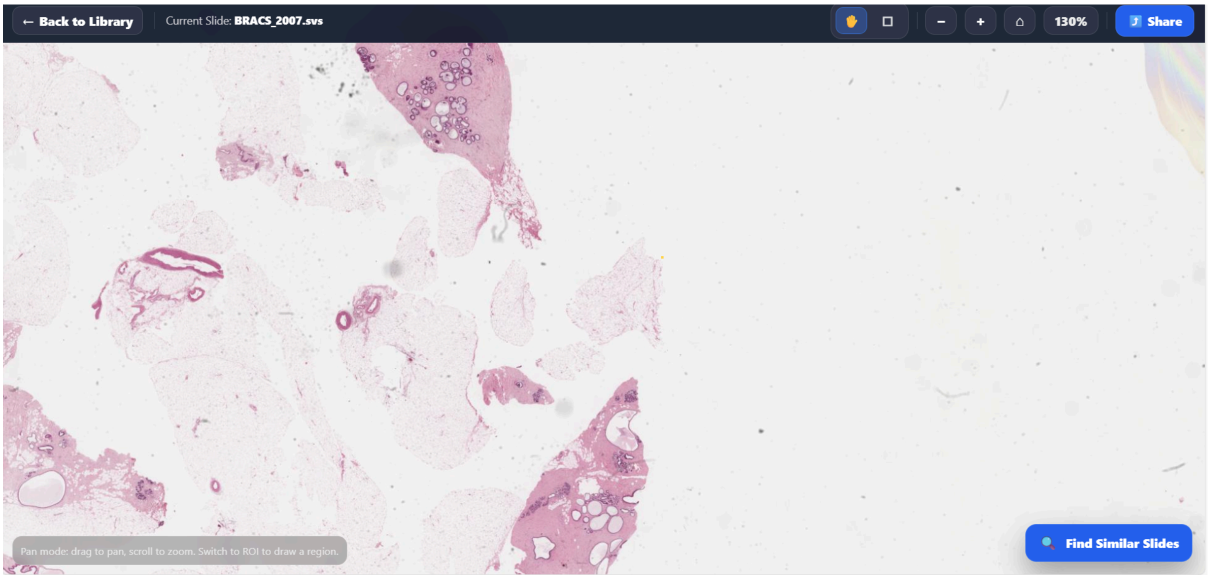

Figure 9. WSI Viewer

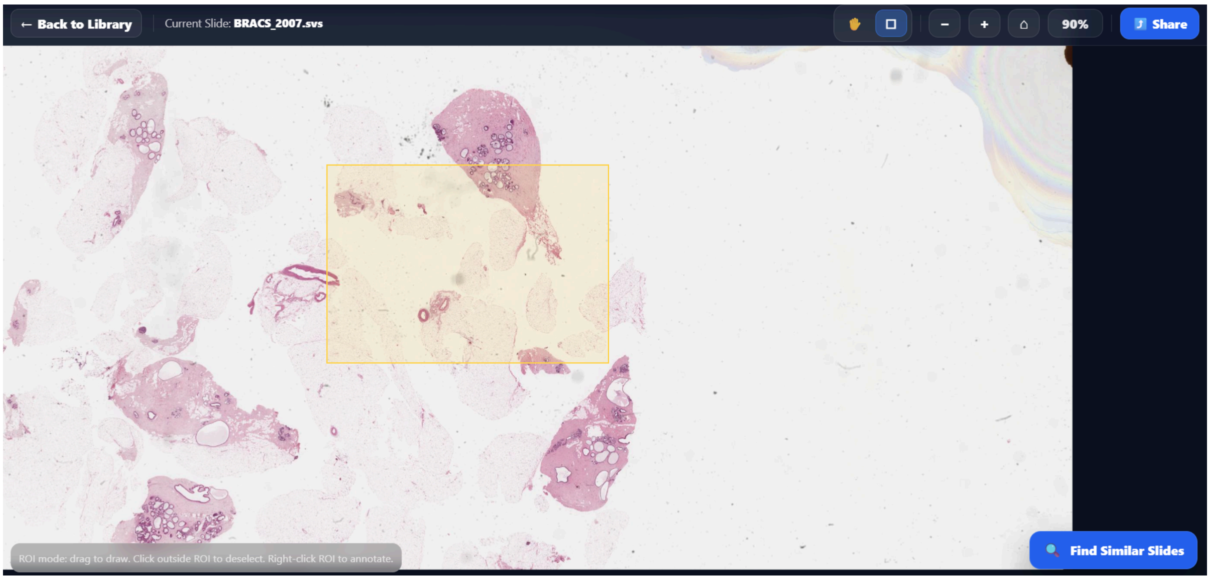

Figure 10. ROI Selection Mode

Figure 11. Content-Based Image Retrieval with Similarity Score

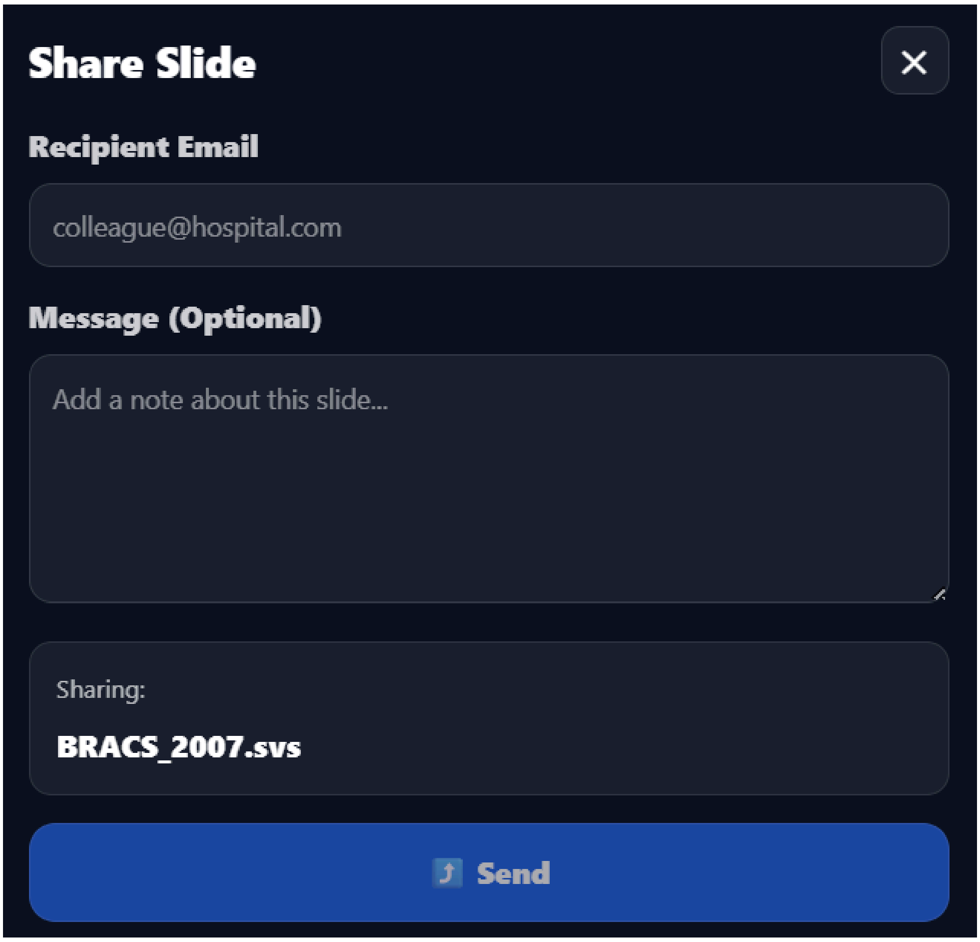

Figure 12. Share Slide Pop-Up

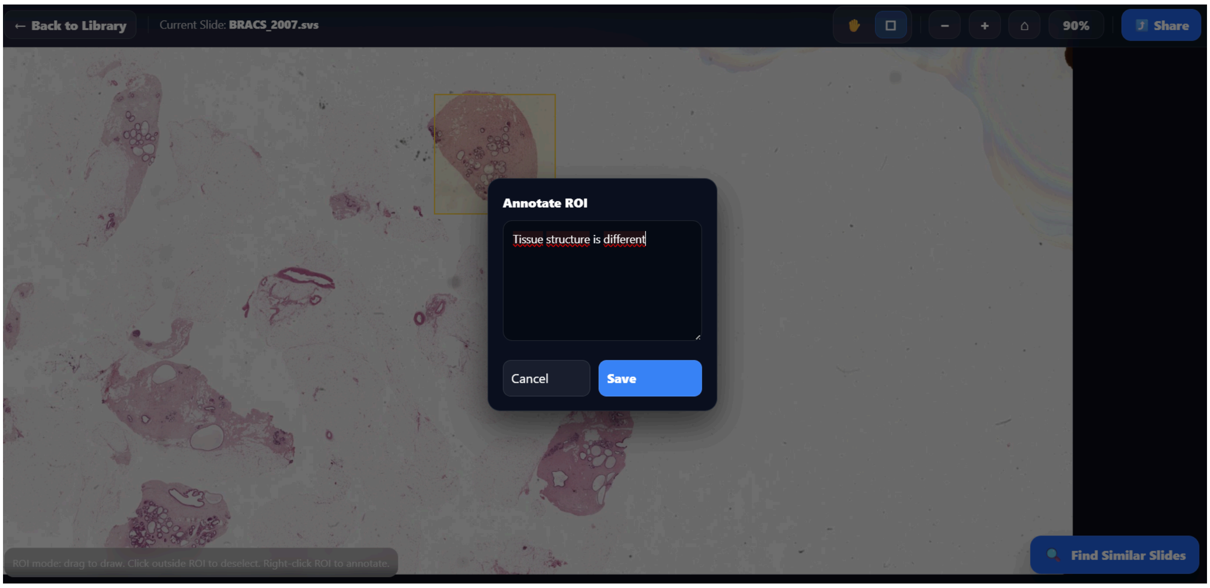

Figure 13. Annotate Region of Interest

4. Other Analysis Elements

4.1. Consideration of Various Factors in Engineering Design

The PatchMatch system is designed by considering multiple technical, ethical, social, and contextual factors arising from its use in digital pathology. As the system works with medical image data and supports clinical and research workflows, these factors play an important role in shaping analysis decisions and guiding future design choices.

Healthcare and Clinical Context

PatchMatch is designed for use in real-world clinical and healthcare settings. That's why accuracy, reliability, and responsible use play an important role in the design of the system. As the system aims to support clinicians by providing similarity-based cases, the system should focus on explainable retrieval results, avoid making automatic diagnoses by leaving final decisions entirely to clinicians.

Patient Safety and Risk Awareness

As PatchMatch is used on patient pathology images, patient safety must be carefully considered. Incorrect interpretation of results or overreliance on the system could pose risks in clinical settings. For this reason, the system is analyzed and designed as a support tool that assists clinicians rather than making diagnoses itself. Emphasis on contextual presentation of results, and clear communication of system limitations helps reduce potential misuse.

Ethical Use of Medical Data

Ethical handling of sensitive medical data is a prior factor in the system analysis. PatchMatch must ensure that patient data is accessed only by authorized users. Images and reports must be handled in compliance with ethical guidelines. This affects how the system controls access, anonymizes data, and tracks user actions.

Data Privacy and Legal Compliance

The system must be complied with data protection regulations. Decisions regarding data storage, encryption, access logging, and user authentication are crucial.

Social and Professional Impact

PatchMatch influences how pathologists, researchers, and interns work with digital pathology data. The system aims to support professional collaboration and knowledge exchange with annotation sharing, and explainable similarity results.

Educational and Research Considerations

PatchMatch is intended to support educational and research workflows beyond its clinical use. Therefore, features such as regional-level comparison, annotation, and multimodal exploration are planned to be developed for this use. The system is designed to make complex pathology data easier to understand for trainees and researchers. This supports learning without requiring advanced technical expertise.

Economic and Resource Considerations

There is a huge computational and storage demands of WSIs and deep learning–based feature extraction. The analysis accounts for the need to balance performance and scalability with infrastructure costs. Design choices such as reusing stored embeddings and supporting scalable indexing mechanisms aim to reduce redundant computation and resource usage over time.

Table 1: Effect Levels of Considerations of Engineering Design Factors

Factor	Short Description	Effect Level (0–10)
Healthcare and Clinical Context	Clinical accuracy, reliability, and responsible usage	9
Patient Safety and Risk Awareness	Prevention of misuse and clinical risk	9
Ethical Use of Medical Data	Authorized access and ethical data handling	8
Data Privacy and Legal Compliance	Regulatory compliance, encryption, and access control	10
Social and Professional Impact	Collaboration and professional workflow support	7
Educational and Research Considerations	Support for training and research use cases	6
Economic and Resource Considerations	Computational cost, storage efficiency, and scalability	7

4.1.1 Constraints

The development and deployment of PatchMatch in digital pathology involve several constraints, including technical challenges, economic considerations, and ethical responsibilities associated with large-scale medical imaging data.

Implementation Constraints

- **Whole-Slide Image Size:** WSIs can reach sizes of up to 50,000 × 50,000 pixels. This makes it impossible to load and operate them entirely. That's why tiling and multi-resolution viewing mechanisms should be applied.
- **Computational Requirements:** Feature extraction and embedding of WSIs require GPU acceleration for efficient performance. Processing a large number of WSIs and indexing their patches causes significant computational and time constraints.
- **Scalable Similarity Search:** The retrieval database must efficiently compare large numbers of patch embeddings and maintain acceptable query performance as the dataset grows.
- **File Format and Integration Limitations:** WSIs are generally stored in the .svs, and .ndpi format which requires tools like OpenSlide. Although standards such as DICOM [9] and HL7 [10] are crucial for hospital integration, they are not yet widely adopted in digital pathology, which limits interoperability.
- **Data Availability and Completeness:** The effectiveness and interpretability of retrieval results depend on the availability of associated pathology reports and

metadata for the images stored in the system. If such information is inaccessible, contextual understanding of retrieved cases may be limited.

Economic Constraints

- **Infrastructure Costs:** PatchMatch requires servers equipped with GPUs, sufficient storage and reliable backend infrastructure.
- **Development Resources:** Developing a medical-focused information retrieval system involves costs for data collection and system validation.

Ethical Constraints

- **Patient Privacy:** WSIs and associated metadata and reports contain sensitive patient information. All of them must be anonymized before use, and the system must comply with regulations such as KVKK [11] and GDPR [12].
- **Bias and Dataset Limitations:** If certain tissue types or patient populations are underrepresented in the datasets used, retrieval performance may be biased. Dataset composition must be documented, and limitations must be clearly acknowledged.
- **Interpretation of Results:** The system makes it clear that results are based on case similarity, not diagnoses. Explanations are provided to help users interpret the results accordingly.

User Experience Constraints

- **Workflow Integration and Complexity:** The system must support the interaction of features and integrate them into existing pathology workflows.

- **Interpretability and Responsiveness:** Retrieved results and explanations must be presented clearly. Interface or performance limitations may affect user understanding and adoption.

4.1.2 Standards

Medical Software Development Standards

- The system follows IEC 62304 [13] standards 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 [14] , which focuses on usability, validation, and release practices for health software.

Medical Imaging and Data Exchange Standards

- The system considers standards such as DICOM [9] to support future interoperability with hospital PACS systems.
- The system is designed to be compatible with commonly used digital pathology formats, such as open standards and tooling. This enables integration with existing clinical infrastructures.
- HL7 [10] standards are considered for the exchange of clinical metadata and pathology reports when integrating with hospital information systems.

Quality and Risk Management Standards

- ISO 13485 [15] standards are considered for development practices as it emphasizes quality management, documentation, and controlled change processes in medical software.
- ISO 14971 [16] guides about managing risks such as incorrect retrieval results, and misunderstanding similarity explanations.

Regulatory and Compliance Considerations

- The system is reviewed in line with the EU Medical Device Regulation [17], which covers clinical decision-support software.
- The EU Artificial Intelligence Act [18] is considered to promote transparent and responsible use of AI in medical settings.
- Regulatory pathways in other regions, such as the FDA Class II (510(k)) [19] process in the United States, are taken into account for future deployment.

Data Protection and Privacy Standards

- PatchMatch follows GDPR [12] and KVKK [11] requirements to process medical images and patient data lawfully.
- To protect patient data, the system uses access controls, encryption, audit logs, and anonymization when data is used for research.

Documentation and Modeling Standards

- UML 2.5.1 [20] is used for modeling system architecture and workflows.
- IEEE 830 [21] principles help keep the software requirements well structured, clear, and easy to verify.

4.2. Risks and Alternatives

Data Access and Privacy

Risk: Obtaining large whole slide image datasets is difficult because of the strict data protection regulations. In Türkiye, KVKK [11] governs the collection, storage, and processing of personal data, which includes WSIs and pathology reports. Similar constraints exist under GDPR [12] in the European Union. Compliance requires data use agreements, patient consent management, and anonymization procedure.

Alternative: Establishing formal collaborations with hospitals and research institutions through data use agreements, applying robust anonymization and de-identification techniques can be a solution. In addition, using public datasets where possible can also help reduce legal and administrative delays.

Regulatory Requirements for Clinical Deployment

Risk: Since PatchMatch is designed to support clinical diagnosis, it is subject to medical device regulations. Some regulatory requirements introduce additional testing, documentation, and quality management overhead, slowing down development and deployment.

Alternative: PatchMatch should be a clinical decision support or research tool rather than a primary diagnostic system. Also early regulatory gap analyses can be conducted, medical software standards can be followed, and phased regulatory compliance to gradually transition toward clinical deployment may be planned.

Heterogeneity of Images

Risk: Whole slide images differ significantly in terms of resolution, staining protocols, scanner types, laboratory workflows, and patient populations. This heterogeneity makes accurate similarity matching and retrieval difficult and may result in inconsistent outputs and limited generalization across hospitals.

Alternative: Applying normalization techniques for staining and resolution, incorporating robust feature extraction methods, and training models on diverse datasets could be solutions. Also, evaluating performance across heterogeneous sources can help improve robustness and reliability.

Hospital Integration

Risk: Hospitals operate within tightly controlled IT environments. Integrating external software requires compliance with cybersecurity policies, firewall restrictions, data access controls, and internal security reviews. Additionally, slow procurement procedures and legacy infrastructure can significantly delay deployment and user adoption.

Alternative: The system could be modular and secure that can be deployed within hospital approved environments. We can use standardized APIs, follow healthcare IT security standards, and provide clear documentation on data flow and security to simplify hospital approval and integration processes.

Data Availability

Risk: Paired datasets that include both whole slide images and their corresponding diagnostic reports are limited. Even when image data is available, it may not be provided in standard WSI formats such as SVS, but instead as images which lack multi-resolution and metadata information required for detailed analysis.

Alternative: We can consider using medical image–text datasets that do not involve whole slide images. We may obtain compact visual embeddings from those datasets. Another option is to omit the report-based retrieval component.

Table 2: Risks and Alternative Solutions

Risk	**Likelihood**	**Effect on the project**	**B Plan Summary**
Data Access and Privacy	High	Medium-High	Use anonymized, publicly available datasets only.
Regulatory Requirements for Clinical Deployment	Medium	High	Make the system as a research or decision-support tool.
Heterogeneity of Images	Low-Medium	Medium	Apply normalization and test on diverse datasets.

Hospital Integration	Low	Medium	Deploy the system as a standalone, modular platform.
Data Availability	Medium-High	High	Focus on image-only retrieval or non-WSI datasets.

4.3. Project Plan

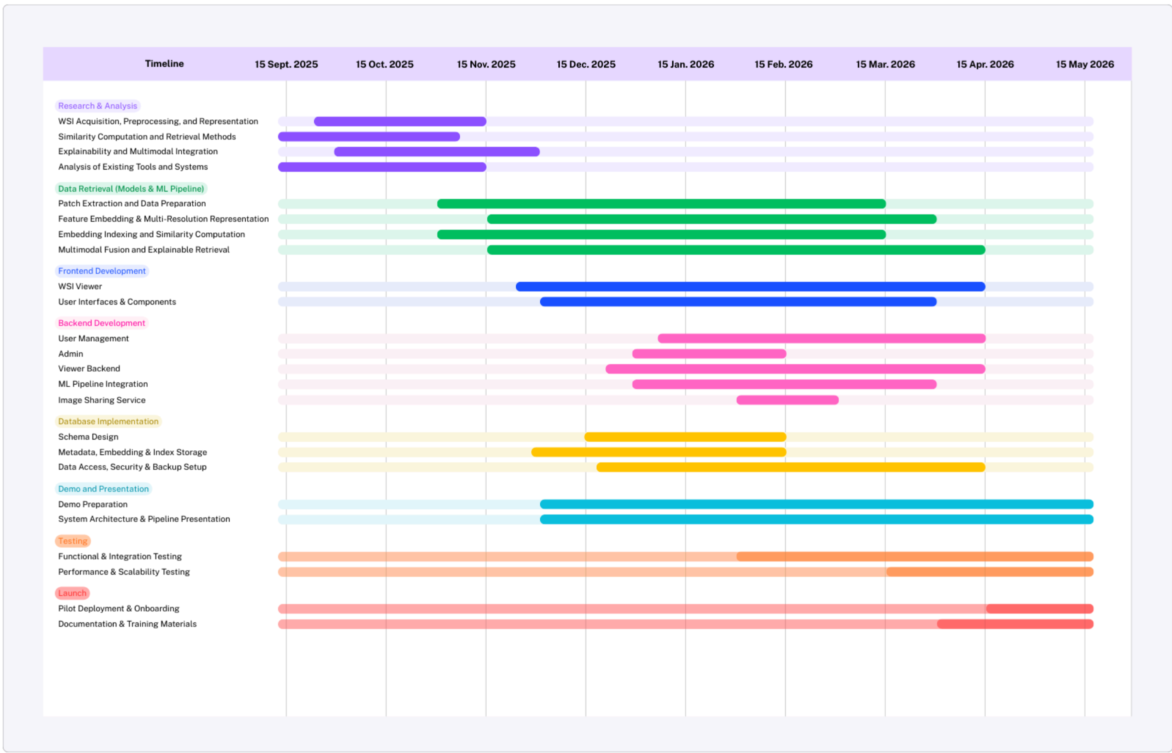

Figure 14. Gantt Chart of the Project
Note: The full resolution of the diagram is available at:
https://www.canva.com/design/DAG74frB7N8/YFMHYrTOUC1hvZn56GuTww/edit?utm_content=DAG74frB7N8&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton

Table 3: List of Work Packages

WP#	**Work Package Title**	**Leader**	**Members Involved**	**Timeline**
WP1	Research & Analysis	Ekin Köylü	All Members	15 Sept. 2025 - 1 Dec. 2025
WP2	Data Retrieval (Models & ML Pipeline)	Ekin Köylü	All Members	1 Nov. 2025 - 15 Apr. 2026

WP3	Frontend Development	Elif Lara Oğuzhan	Emre Yazıcıoğlu Ekin Köylü	25 Nov. 2025 - 15 Apr. 2026
WP4	Backend Development	Emre Yazıcıoğlu	Elif Lara Oğuzhan İlke Latifoğlu Bertan Uran	25 Dec. 2025 - 15 Apr. 2026
WP5	Database Implementation	Bertan Uran	İlke Latifoğlu	1 Dec. 2025 - 15 Apr. 2026
WP6	Demo and Presentation	İlke Latifoğlu	All Members	1 Dec. 2025 - 15 May 2026
WP7	Testing	İlke Latifoğlu	Emre Yazıcıoğlu	1 Feb. 2026 - 15 May 2026
WP8	Project Launch	Bertan Uran	All Members	1 Apr. 2026 - 15 May 2026

WP1: Research & Analysis			
Start date: 15 Sept. 2025 **End date:** 1 Dec. 2025			
Leader:	Ekin Köylü	**Members involved:**	All Members

Objectives: Conduct a comprehensive academic analysis of existing methods related to whole-slide image (WSI) retrieval in digital pathology. This includes studying WSI acquisition and preprocessing techniques, deep learning based feature extraction methods, similarity computation approaches, explainability mechanisms, and multimodal integration strategies. The insights gained from this analysis will guide the design decisions of the PatchMatch system and ensure alignment with state-of-the-art.

Tasks:

Task 1.1 WSI Acquisition, Preprocessing, and Representation: Review academic studies on WSI digitization, tiling strategies, tissue region extraction, data augmentation, and feature representation methods to understand how large WSIs are prepared and represented for analysis and retrieval.

Task 1.2 Similarity Computation and Retrieval Methods: Study distance-based and learning-based similarity computation approaches

used in WSI retrieval, including patch-level and slide-level similarity modeling, with a focus on scalability and effectiveness. **Task 1.3 Explainability and Multimodal Integration:** Explore explainability techniques in digital pathology and multimodal retrieval methods to combine image features with pathology report text to make similarity results more transparent and easier to explore. **Task 1.4 Analysis of Existing Tools and Systems:** Investigate existing digital pathology platforms and research prototypes to understand their functionalities, limitations, and design choices, and to identify gaps that motivate the proposed PatchMatch system.			
Deliverables **D1.1:** Comprehensive literature review included in the project specification document. **D1.2:** Presentation to domain experts demonstrating existing digital pathology tools and systems, used to gather feedback and refine functional and non-functional requirements for PatchMatch.			
WP2: Data Retrieval (Models & ML Pipeline)			
Start date: 1 Nov. 2025 **End date:** 15 Apr. 2026			
Leader:	Ekin Köylü	**Members involved:**	All Members
Objectives: Design and implement the complete ml pipeline to enable similarity-based retrieval in PatchMatch. This includes patch extraction, feature embedding generation, multi-resolution representation, similarity computation, multimodal fusion with reports, and explainable retrieval mechanisms.			
Tasks: **Task 2.1 Patch Extraction and Data Preparation:** Extract image patches from WSIs using tiling and tissue filtering strategies, and prepare visual and textual data required for downstream embedding generation. **Task 2.2 Feature Embedding and Multi-Resolution Representation:** Deep learning models are implemented to generate embeddings at both patch and slide levels. These embeddings use multi-resolution			

representations to capture information from individual cells as well as broader tissue structures. **Task 2.3 – Embedding Indexing and Similarity Computation:** Design and implement similarity computation pipelines by indexing visual and textual embeddings in a vector database to support efficient slide-level and ROI-level comparison. **Task 2.4 – Multimodal Fusion and Explainable Retrieval:** Image and text embeddings are integrated to support text-to-image and combined image–text similarity search. Explainability features, such as heatmaps and feature-level explanations, are added to help justify and interpret the similarity results.
Deliverables ***D2.1:*** Patch Extraction and Embedding Generation Pipeline ***D2.2:*** Similarity Computation and Vector Indexing Module ***D2.3:*** Multimodal Fusion and Explainable Retrieval Component

WP3: Frontend Development			
Start date: 25 Nov. 2025 **End date:** 15 Apr. 2026			
Leader:	Elif Lara Oğuzhan	**Members involved:**	Emre Yazıcıoğlu, Ekin Köylü
Objectives: To construct the frontend components of the application			
Tasks: **Task 1.1 WSI Viewer:** This task focuses on developing an interactive whole-slide image viewer that allows users to open WSIs, zoom at different magnifications, pan across the slide, and select regions of interest. It ensures smooth navigation without loading the entire image into memory. **Task 1.2 User Interfaces of The Pages & Components:** It involves designing and implementing the user interfaces of all pages and UI components, such as search, retrieval results, annotation, and sharing pages.			

Deliverables **D1.1:** Viewer Implementation **D1.2:** User Interface Design			
WP4: Backend Development			
Start date: 25 Dec. 2025 **End date:** 15 Apr. 2026			
Leader:	Emre Yazıcıoğlu	**Members involved:**	Elif Lara Oğuzhan, İlke Latifoğlu, Bertan Uran
Objectives: To build the backend system of the application			
Tasks: **Task 2.1 User Management:** It handles user authentication, authorization, and profile management. It ensures that only authorized users can access WSIs, annotations, and retrieval results based on their roles and permissions. **Task 2.2 Admin:** This task provides administrative control over the system, such as managing users, monitoring system usage, and handling configuration settings. It supports maintenance, security, and overall system reliability. **Task 2.3 Viewer Backend:** It enables efficient streaming of whole-slide images to the frontend by serving multi-resolution tiles. It ensures smooth zooming, panning, and navigation without loading entire WSIs into memory. **Task 2.4 ML Pipeline Integration:** It integrates preprocessing, patch extraction, embedding generation, and retrieval models into the system. It allows WSIs, ROIs, and text queries to be processed and compared using machine learning pipelines. **Task 2.5 Image Sharing Service:** It allows users to share WSIs, retrieval results, and annotations with other authorized users. Thus, it supports collaboration.			

Deliverables **D2.1:** User Management Module Implementation **D2.2:** Viewer Backend and Image Streaming Service **D2.3:** Integrated ML Retrieval and Sharing Pipeline **D2.4:** Administrative Control and System Configuration Module

WP5: Database Implementation			
Start date: 1 Dec. 2025 **End date:** 15 Apr. 2026			
Leader:	Bertan Uran	**Members involved:**	İlke Latifoğlu
Objectives: The objective of the Database phase is to design and implement a robust, scalable, and secure data layer that supports all core system functionalities. This phase focuses on defining clear data structures for slides, patches, users, and reports while ensuring data integrity and consistency across the platform. It also enables efficient storage and retrieval of metadata and embeddings to support advanced search and analytics features. Additionally, this phase establishes secure data access controls and reliable backup mechanisms to protect data and ensure system continuity.			
Tasks: **Task 1.1 Schema Design (Slides, Patches, Users, Reports):** The objective of this task is to design a normalized and extensible database schema that accurately represents core entities and their relationships. The schema ensures data integrity, supports efficient querying, and allows for future system enhancements. **Task 1.2 Metadata, Embedding & Index Storage:** The objective of this task is to implement efficient storage for metadata, semantic embeddings, and indexes to enable fast retrieval and advanced search capabilities. The design prioritizes scalability and performance for large and growing datasets.			

Task 1.3 Data Access, Security & Backup Setup: The objective of this task is to establish secure and controlled data access while protecting sensitive information. This includes implementing role-based permissions, enforcing security best practices, and configuring automated backup and recovery processes.

Deliverables

D1.1: Entity–Relationship Diagram (ERD)

D1.2: Metadata schema and storage design

D1.3: Embedding storage solution (e.g., vector tables or vector database integration)

D1.4: Data access layer (APIs, queries, or ORM configurations)

WP6: Demo and Presentation

Start date: 1 Dec. 2025 **End date:** 15 May 2026

Leader:	İlke Latifoğlu	**Members involved:**	All members

Objectives: The objective of the Demo and Presentation phase is to clearly communicate the system's functionality, architecture, and value to stakeholders. This phase focuses on preparing a stable and well-structured demonstration that showcases key features and workflows. It also aims to present the overall system architecture and data pipeline in a clear and understandable manner. The phase ensures that technical and non-technical audiences can easily grasp the system design, capabilities, and outcomes.

Tasks:

Task 2.1 Demo Preparation: The objective of this task is to prepare a reliable and polished system demonstration that highlights core functionalities and use cases. This includes validating system stability, preparing demo data, and ensuring smooth execution during the presentation.

Task 2.2 System Architecture & Pipeline Presentation: The objective of this task is to clearly present the system architecture and data pipeline, explaining how components interact and data flows through the system.

The presentation aims to communicate design decisions, scalability considerations, and technical trade-offs effectively.
Deliverables **D2.1:** Configured and validated demo environment **D2.2:** Prepared demo data and user scenarios **D2.3:** Technical presentation slides

WP7: Testing			
Start date: 1 Feb. 2026 **End date:** 15 May 2026			
Leader:	İlke Latifoğlu	**Members involved:**	Emre Yazıcıoğlu
Objectives: The objective of the Testing phase is to validate that the system functions correctly, performs reliably, and meets quality standards before deployment. This phase focuses on verifying functional correctness across all components and ensuring smooth integration between system modules. It also evaluates system performance, scalability, and stability under expected and peak workloads. The phase helps identify and mitigate risks early, ensuring the system is production-ready.			
Tasks: **Task 1.1 Functional & Integration Testing :** The objective of this task is to verify that all system features function as intended and that integrated components interact correctly. This ensures end-to-end workflows operate reliably without errors or data inconsistencies. **Task 1.2 Performance & Scalability Testing :** The objective of this task is to assess system performance under normal and peak loads and evaluate its ability to scale. This helps ensure acceptable response times, resource utilization, and system stability as usage grows.			

Deliverables **D1.1:** Functional test cases and test scripts **D1.2:** Performance and load test plans			
WP8: Project Launch			
Start date: 1 Apr. 2026 **End date:** 15 May 2026			
Leader:	Bertan Uran	**Members involved:**	All members
Objectives: The objective of the Launch phase is to deploy the system to a real user environment and ensure a smooth transition from development to operational use. This phase focuses on onboarding initial users, validating deployment processes, and providing clear documentation and training materials. It ensures that users can effectively adopt the system and that operational procedures are well-defined. The phase also collects early feedback to support stabilization and future improvements.			
Tasks: **Task 2.1 Pilot Deployment & Onboarding:** The objective of this task is to deploy the system in a controlled pilot environment and onboard initial users. This allows validation of real-world usage while ensuring users are supported during adoption. **Task 2.2 Documentation & Training Materials:** The objective of this task is to create clear and accessible documentation and training resources for users and administrators. These materials ensure consistent system usage, maintenance, and long-term sustainability.			
Deliverables **D2.1:** Pilot deployment plan and execution report **D2.2:** User onboarding materials **D2.3:** User documentation and user guides			

4.4. Ensuring Proper Teamwork

Building the PatchMatch system relies heavily on strong teamwork. It brings together many interconnected parts such as image processing, AI pipelines, backend services, and user interfaces that must work seamlessly together.

Division of Responsibilities

- Tasks are divided based on individual expertise and project needs. Work is shared across backend, frontend, AI pipelines, and WSI processing and storage.
- Each team member is responsible for specific parts of the system and works closely with others during integration to ensure everything fits and works well together.

Tools for Collaboration

- **GitHub:** Used for source code management, version control, and code reviews.
- **ClickUp:** Used for task management, sprint planning, and progress tracking to ensure clear assignment of responsibilities and deadlines.
- **Google Drive:** Used to share documents, meeting notes, and report drafts.
- **WhatsApp:** Used for day-to-day communication and coordination among team members.
- **Zoom:** Used for online meetings, discussions, and supervisor check-ins when in-person meetings are not feasible.

Meetings and Communication

- Regular weekly team meetings are held to discuss progress, challenges, and upcoming tasks.
- Meetings with the project supervisor are held approximately every 2–3 weeks to receive feedback, and address design decisions.
- Meetings with domain experts, such as pathologists and innovation experts, are conducted when needed to gather clinical insights, validate assumptions, and guide design decisions

Tracking Contributions

- Individual contributions are tracked through GitHub commit history, pull requests, and code reviews.
- Task completion and workload distribution are monitored using ClickUp.
- Documentation contributions and design discussions are recorded in shared documents to ensure visibility of non-coding contributions.

4.5. 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 [12] and KVKK [11] 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.

4.6. Planning for New Knowledge and Learning Strategies

PatchMatch requires a significant amount of technical knowledge, with the main pillars being a basic understanding of digital histopathology, machine learning, and full-stack software development, so that every team member can make an effective contribution to the project. Below is a more detailed list of topics that the team should study and practice.

Required Technical Knowledge

- **Digital Pathology Fundamentals**: Understanding WSI formats (.svs, .ndpi), tissue structures, and how pathologists actually use these images in practice.
- **Deep Learning for Medical Images**: Learning feature extraction methods, embedding generation, and similarity search techniques that can benefit histopathology.
- **Vector Databases and Similarity Search**: Working with FAISS or similar tools to efficiently index and retrieve similar patches from large collections.
- **Medical Image Processing**: Using libraries like OpenSlide to handle gigapixel images, tiling methods, and multi-resolution viewing.

- **Web-based Medical Image Viewers**: Building responsive interfaces that can smoothly display and navigate large pathology images.
- **Explainable AI**: To show users why certain results were retrieved (heatmaps, attention mechanisms).
- **Medical Data Standards**: Understanding DICOM [9], HL7 [10], and medical device regulations .

Learning Strategies

- **Hands-on Experimentation**: Setting up small test environments to try patch extraction, feature embedding, and similarity search before building the full system.
- **Consulting to Mentors and Domain Experts:** Working with pathologists to understand their workflow and needs, and with our mentor to discuss the implementation strategy.
- **Large Language Models and Online Resources:** Using documentation from libraries and LLMs to fill the knowledge gap and set a personalized learning pace.
- **Literature Review:** Reviewing papers on the topic, such as Yottixel, SMILY, and HIPT, to catch up with the latest research.

5. Glossary

CBIR (Content-Based Image Retrieval): A method of finding images based on their visual content rather than text labels. PatchMatch uses CBIR to find similar tissue patterns by comparing the actual image features.

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.

6. References

1. Sectra, “Sectra Digital Pathology Solution,” Sectra Medical Systems. [Online]. Available: https://medical.sectra.com/product/sectra-digital-pathology-solution/
2. Iron Mountain, “Digital Pathology Solutions,” Iron Mountain. [Online]. Available: https://www.ironmountain.com/industries/healthcare-services/pathology-solutions
3. Huron Digital Pathology, “Lagotto — content-based image retrieval solution for pathology,” Huron Digital Pathology. [Online]. Available: https://www.hurondigitalpathology.com/lagotto/
4. N. Hegde et al., “Similar image search for histopathology: SMILY,” NPJ Digital Medicine, vol. 2, no. 1, p. 56, 2019.
5. S. Kalra, H. R. Tizhoosh, C. Choi, S. Shah, P. Diamandis, C. J. V. Campbell, and L. Pantanowitz, “Yottixel – An Image Search Engine for Large Archives of Histopathology Whole Slide Images,” Medical Image Analysis, vol. 65, p. 101757, 2020. doi: 10.1016/j.media.2020.101757.
6. A. Pedersen, M. Valla, A. M. Bofin, J. P. de Frutos, I. Reinertsen, and E. Smistad, “FastPathology: An open-source platform for deep learning-based research and decision support in digital pathology,” arXiv preprint arXiv:2011.06033, 2020.
7. Ibex Medical Analytics (Ibex), “Home – Ibex Medical Analytics,” Ibex AI. [Online]. Available: https://ibex-ai.com/
8. draw.io (diagrams.net), “Online diagram editor and flowchart tool,” 2025. [Online]. Available: https://www.draw.io/
9. DICOM Standards Committee, DICOM Supplement 145: Whole Slide Microscopic Image IOD and SOP Classes, 2010.
10. Health Level Seven International, “HL7 Standards.” [Online]. Available: https://www.hl7.org/
11. Kişisel Verileri Koruma Kanunu (KVKK), Law No. 6698, 2016. [Online]. Available: https://www.kvkk.gov.tr/

12. European Parliament and Council, Regulation (EU) 2016/679 (General Data Protection Regulation – GDPR), 2016.
13. IEC 62304:2006/AMD1:2015, Medical device software — Software life cycle processes, IEC, 2015.
14. IEC 82304-1:2016, Health software — Part 1: General requirements for product safety, IEC, 2016.
15. ISO 13485:2016, Medical devices — Quality management systems — Requirements for regulatory purposes, ISO, 2016.
16. ISO 14971:2019, Medical devices — Application of risk management to medical devices, ISO, 2019.
17. European Parliament and Council, Regulation (EU) 2017/745 (Medical Device Regulation – MDR), 2017.
18. European Parliament and Council, Regulation (EU) 2024/1689 (Artificial Intelligence Act), 2024.
19. U.S. Food and Drug Administration (FDA), “Premarket Notification 510(k).” [Online]. Available: https://www.fda.gov/medical-devices/premarket-submissions
20. Object Management Group, OMG Unified Modeling Language (UML), Version 2.5.1. OMG, Dec. 2017. [Online]. Available: https://www.omg.org/spec/UML/2.5.1/
21. IEEE, IEEE Recommended Practice for Software Requirements Specifications (IEEE Std 830-1998). IEEE Computer Society, 1998. [Online]. Available: https://standards.ieee.org/standard/830-1998.html