

A Modality-Adaptive Software Architecture for Explainable Clinical Decision Support

Danila Lissovoy¹, Assylbek Khalyk¹, Akylbek Oralov¹, Rakhat Mustafin¹, Alisher Kenzhagaliyev¹, and Hashim Ali^{1*}

¹ Department of Computer Science, School of Computing and Artificial Intelligence, Nazarbayev University, Astana, Kazakhstan

*Corresponding author: hashim.ali@nu.edu.kz

ABSTRACT

Artificial intelligence (AI)-enabled clinical decision support systems (CDSSs) increasingly combine structured clinical data and medical images, but their explanations differ in cost, format, storage, and failure behavior. This study presents a modality-adaptive software architecture for a browser-based CDSS supporting 10-year coronary heart disease (CHD) prediction, 30-day hospital readmission prediction, and chest-radiograph classification with 14 independent output scores. The architecture treats explanation selection as a request-time service policy: Shapley additive explanations (SHAP) accompany tabular predictions; local interpretable model-agnostic explanations (LIME) are requested separately; gradient-weighted class activation mapping (Grad-CAM) and its generalized variant (Grad-CAM++) are computed only for threshold-positive classes within the five highest-ranked scores; and large language model (LLM) narratives are attempted when configured. React, FastAPI, PostgreSQL, and a seven-day Redis cache implement the design. Seven synthetic fixture families satisfied the specified selection conditions. In a backend experiment involving 7,818 radiographs, heatmap generation averaged 140.2 ms per radiograph, or 78.2% of the 179.3-ms mean total. A separate in-process service-layer test completed all 3,600 requests successfully using deterministic prediction and explanation substitutes and SQLite persistence. The architecture specifies bounded explanation work and separate retention paths. Full-stack performance, clinical-model calibration, explanation quality, and clinician benefit remain subjects for future evaluation.

Keywords: clinical decision support; software architecture; explainable artificial intelligence; medical imaging; health information systems.

1. INTRODUCTION

A clinical prediction reaches a user through software that validates inputs, applies preprocessing, invokes a model, presents explanations and warnings, and stores enough context for later review. The quality of an artificial intelligence (AI)-enabled clinical decision support system (CDSS) therefore depends on the information system around the model, as well as on discrimination or calibration. The FUTURE-AI guideline places traceability, usability, robustness, and explainability alongside fairness and universality [1]. DECIDE-AI asks for early clinical evaluations to describe workflow, human factors, safety, and changes introduced by the system [2], while TRIPOD+AI strengthens the reporting of data, predictors, validation, and model performance [3]. Together, these sources make software architecture part of the clinical argument rather than an implementation detail.

Medical applications often use the word "multimodal" for a single model that fuses images, text, signals, and structured variables [4,5]. A working information system faces a second form of multimodality: it may host several independent clinical tasks whose inputs, models, explanations, and response payloads differ. A cardiovascular risk model needs signed feature contributions. A readmission model may operate on sparse encounter variables. A chest-radiograph classifier needs class-specific spatial maps. Consequently, explanation delivery must account for the type of data and clinical context. Applying an undifferentiated explanation operation to all three tasks would ignore those differences.

Most clinical explainable artificial intelligence (XAI) research evaluates an attribution method, a predictive model, or an explanation interface. Less attention has been paid to the request-time decisions that connect them: whether an explanation should run automatically or on demand, how many image classes should receive heatmaps, whether generated text may fail without blocking the prediction, and how long heavy visual responses should remain available. These decisions affect latency, storage, failure isolation, and the amount of information shown to a clinician. They are therefore architectural decisions with direct consequences for system quality.

This paper develops and evaluates a modality-adaptive explanation architecture for one CDSS containing three clinical workflows. Research question (RQ) 1 asks how the architecture can deliver modality-appropriate explanations through a consistent interaction and history model. RQ2 asks what computational bounds and measured costs accompany eager, on-demand, selective, and optional explanation strategies. RQ3 asks which capabilities remain available when optional services fail and what information is needed to trace a prediction and its explanation. The contribution is a software architecture and an evaluation of its implemented request paths and component-level behavior. The clinical tasks provide integration cases; model development, calibration, decision analysis, and clinical utility are reserved for subsequent validation.

2. RELATED WORK

2.1 Explanation methods and their limits

The Shapley additive explanations (SHAP) method expresses a prediction as additive feature contributions and satisfies local accuracy and consistency properties under its formal assumptions [6]. The local interpretable model-agnostic explanations (LIME) method samples the neighborhood of an instance and fits an interpretable local surrogate [7]. The methods can rank features differently because they use different reference distributions, perturbations, and approximations. The agreement is useful for comparison but does not establish causal validity or fidelity of the explanation.

For convolutional neural networks, gradient-weighted class activation mapping (Grad-CAM) combines class gradients in a convolutional layer to produce a coarse localization map [8]. Grad-CAM++ modifies the gradient weighting to improve localization in some multi-instance settings [9]. A heatmap shows where model activations influence a score. It is not a lesion boundary, a causal account, or a record of the radiologist's attention. Sanity checks and adversarial analyses further show that plausible-looking attributions can remain insensitive to model parameters or be manipulated [10,11].

Clinical critiques have therefore cautioned against treating post-hoc explanations as a substitute for model validity, fairness, or evidence of benefit [12,13]. The causal structure may also be built into model development rather than added only after

prediction; for example, causal-pathway-guided distillation can transfer temporal representations into an interpretable tree model for intensive care tasks [14]. The present work addresses a complementary systems question: how different explanation forms are invoked, returned, and retained within a clinical application.

2.2 Human-centered explanation interfaces

A systematic review of explainable medical-imaging systems found limited formative research with intended users and little empirical evaluation of interface design [15]. Reviews of CDSS design and explainable interfaces likewise identify workflow fit, layered information, interaction, and contestability as separate design concerns [16,17]. Healthcare professionals tend to prefer concise contextual information with access to detail when needed [18]. These findings motivate progressive disclosure in this application; its effect on clinician understanding and appropriate reliance has not been tested.

Explanation format can also affect reliance on a model. One clinical vignette study found that clinically framed explanations improved usability and acceptance compared with SHAP alone [19], but text can increase acceptance of an incorrect recommendation. In a chest-radiograph study, combining text with saliency maps reduced over-reliance compared with text alone [20]. Saliency maps themselves require caution: a benchmark against radiologist annotations found that Grad-CAM performed relatively well among the tested methods, but every method remained below the human localization reference [21]. The evaluation should therefore separate understandability, clinical relevance, faithfulness, plausibility, efficiency, and decision effects [22–24].

2.3 Software architecture and provenance

The International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), and Institute of Electrical and Electronics Engineers (IEEE) standard ISO/IEC/IEEE 42010 treats architecture descriptions as views addressing stakeholder concerns [25]. ISO/IEC 25010:2023 identifies functional suitability, performance efficiency, compatibility, interaction capability, reliability, security, maintainability, flexibility, and safety as product-quality concerns [26]. ISO/IEC 25059 adds AI-specific quality concerns [27]. The National Institute of Standards and Technology (NIST) AI Risk Management Framework emphasizes documentation, measurement, and oversight [28]. These sources inform the architecture analysis; no formal compliance claim is made.

Recent platforms show the value of modular service boundaries. MLXOps4Medic separates machine-learning and explainability operations for medical-imaging development and records workflow provenance [29]. The clinical decision support deployment platform described by Yoo et al. [30] adapts Fast Healthcare Interoperability Resources (FHIR) inputs, calls R or Python reasoning services, and returns results to an electronic health record (EHR) interface. FHIR Provenance provides a standards-based representation of the agents, activities, and entities that create a clinical resource [31]. Their primary emphasis is lifecycle orchestration or interoperability; the present architecture explicitly specifies request-time selection among explanation modes.

2.4 Research gap

The literature offers mature attribution methods, multimodal modeling frameworks, interface guidance, and deployment platforms. The reviewed literature leaves an architecture question unresolved: How can one application support independent



tabular and imaging workflows, select explanations according to modality and user action, separate compact history from large visual responses, and report explanation cost separately from prediction cost? This paper examines that information-systems question. It does not introduce a new prediction or attribution algorithm.

3. MATERIALS AND METHODS

3.1 Study design and research questions

The study used an architecture-focused case evaluation of the lifecycle of a prediction request. The unit of analysis was the path from validated input through inference, explanation selection, response assembly, presentation, and persistence. Request-path analysis, policy conformance checks, imaging-stage timing, and service-layer load testing addressed different execution boundaries. Table 1 links each research question with its method and the interpretation allowed by that method.

Table 1. Research questions and evaluation methods.

Question	Focus	Methods	Interpretation
RQ1	Modality-specific explanation delivery	Architecture views, request-path analysis, and interface mapping	Implemented logical relationships and information flow
RQ2	Explanation workload and request cost	Synthetic policy checks, analytical bounds, imaging timing, and service-load testing	Policy conformance and measurements within distinct execution boundaries
RQ3	Dependency behavior and provenance	Request-path and persistence-field analysis	Configured fallback paths and provenance requirements; no fault injection

3.2 Clinical tasks and data

The application contains two structured-data tasks and one image task. The coronary heart disease (CHD) workflow accepts 13 clinical predictors derived from the Framingham risk setting, including age, sex, cholesterol, systolic blood pressure, smoking, diabetes, body mass index, heart rate, glucose, antihypertensive medication, prevalent hypertension, cigarettes per day, and pulse pressure [32]. The implemented adapter is configured for a random-forest artifact.

The readmission workflow uses demographic, encounter, utilization, laboratory, diagnosis-category, and medication variables derived from the Diabetes 130-US Hospitals dataset [33]. A Light Gradient Boosting Machine (LightGBM) classifier is configured to operate on 207 processed features encoded from 39 raw request fields. The radiograph workflow uses a DenseNet-121 architecture with a 14-class output configured for the National Institutes of Health (NIH) ChestX-ray14 labels [34] and the CheXNet model family [35]. The three tasks remain independent; the application shares authentication, response patterns, history, and explanation controls rather than combining inputs into one predictive model.

3.3 Architectural drivers

The design translates the concerns of the clinician, developer, operator, and assurance into seven architectural drivers. Clinicians need a consistent result structure and control over optional details. Developers need separable task adapters and explanation

strategies. Operators need predictable computational demand, observable failures, and proportional storage. Assurance teams need enough provenance to connect a response to its computational lineage. Table 2 maps these drivers to the mechanisms implemented and the evidence still required.

Table 2. Architectural drivers and implementation mechanisms.

Driver	Quality concern	Implementation mechanism	Further evaluation
Modality-appropriate explanation	Functional suitability and interaction	SHAP and LIME for tabular data; Grad-CAM variants for images	Clinician assessment of relevance and comprehension
Controlled explanation workload	Performance efficiency	On-demand LIME; threshold and top-five image selection; optional narrative	Per-method latency, resource use, and cache-hit measurement
Progressive disclosure	Interaction and controllability	Primary SHAP view, separate LIME action, selectable image findings	Task-based usability and calibrated-reliance study
Failure isolation	Reliability and robustness	Caught narrative errors return null; Redis write errors are logged	Controlled fault injection and recovery tests
Traceability	Transparency	Identifier, user, timestamp, inputs, and compact outputs are stored	Model, data, threshold, preprocessing, explainer, and environment versions
Storage proportionality	Performance and maintainability	Compact PostgreSQL record plus seven-day Redis response cache	Retention, privacy, payload-size, and cache-effect study
Boundary protection	Security and safety	JSON Web Token (JWT) authentication, input schemas, file checks, and explicit model-availability error	Threat modeling, access-control tests, and operational monitoring

3.4 System architecture

Figure 1 presents the application architecture. A React and TypeScript client sends requests using JavaScript Object Notation (JSON) or multipart uploads to the FastAPI application programming interface (API). JWT authentication protects the prediction and history routes. Route handlers validate each request and delegate to a task-specific prediction service. Explanation orchestration is distributed across the route and service layers: it selects the explanation policy, attempts narrative generation when configured, assembles the response, and writes history. PostgreSQL stores users and compact longitudinal results. Redis caches image-rich radiograph responses for a seven-day time to live (TTL). Appendix B distinguishes the stored history fields from the proposed versioned provenance extension.

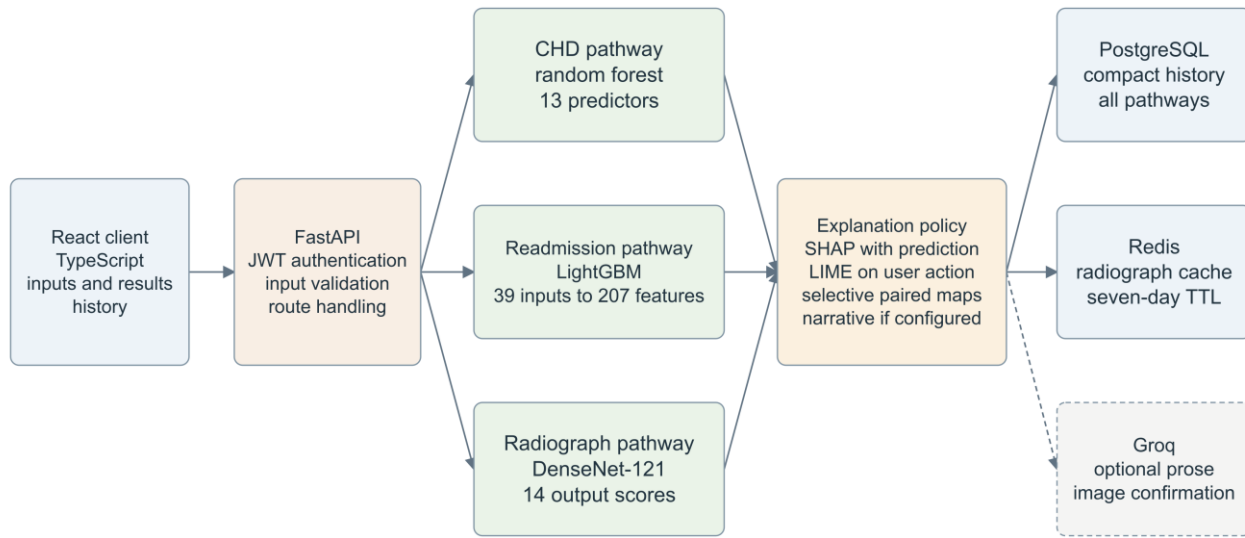


Figure 1. Logical components of the implemented architecture. FastAPI coordinates three independent pathways. PostgreSQL stores compact history; Redis caches radiograph images and maps for seven days. Groq supplies optional text and secondary image confirmation.

The separation between prediction and explanation is logical rather than a network boundary. Each task service owns its preprocessing and model adapter, while the route determines which explanatory output to request and how to present it. This arrangement keeps task-specific code close to the model but preserves a common contract for authentication, response status, history identifiers, and optional narrative text.

3.5 Explanation orchestration policy

The policy uses four execution modes. SHAP is eager for both tabular workflows because it supplies the default feature-contribution view. LIME is on demand and runs only after a user requests a complementary local surrogate. For image input, the service first ranks all 14 scores, retains the first five ranks, and then applies each retained class threshold. For an input image x , $p_i(x)$ denotes the score of output i , τ_i its configured threshold, $R_5(x)$ the indices of the five highest scores, and $H(x)$ the subset selected for maps. The policy is defined by the following expression.

$$H(x) = \{i \in R_5(x) : p_i(x) \geq \tau_i\}$$

Each selected class receives both Grad-CAM and Grad-CAM++, yielding zero to five pairs, or zero to ten individual maps. An eligible class at rank six receives no map. The route returns all 14 scores even when no leading class reaches its threshold. Narrative calls are attempted during prediction when credentials are configured; exceptions or absent credentials yield a null narrative. Figure 2 shows the request paths, and Table 3 summarizes the task-specific behavior. Appendix A gives the thresholds and the separate overall-status rule.

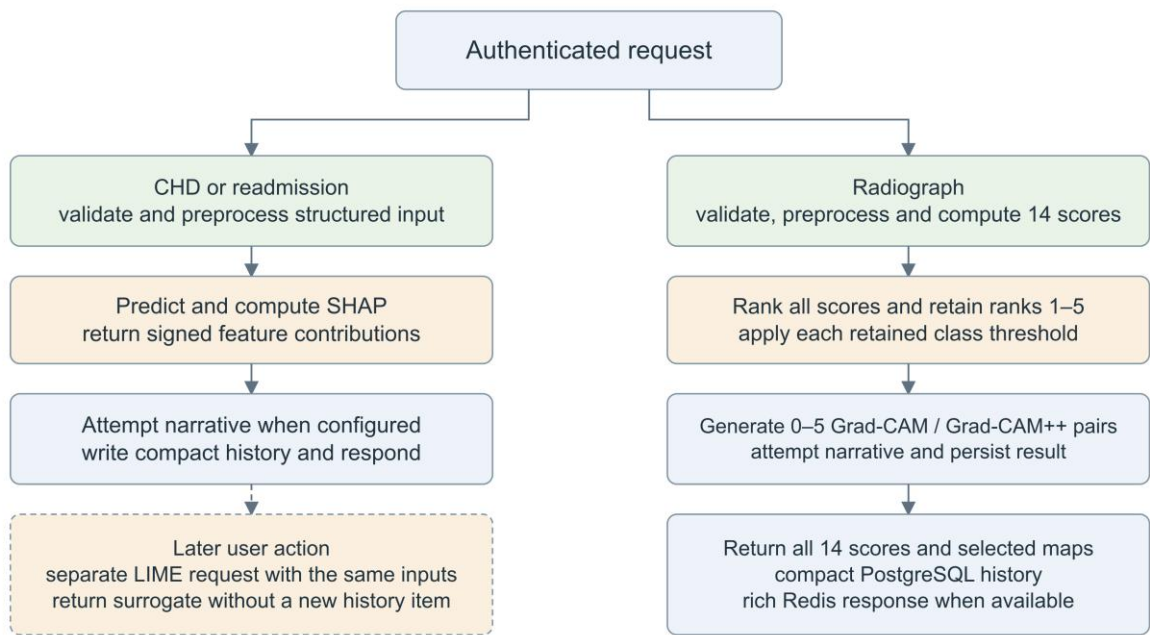


Figure 2. Request-time explanation policy. SHAP accompanies structured-data predictions; a later user action sends the input again for LIME without creating a new history item. Radiographs are ranked before thresholds are applied within ranks 1–5, producing at most five pairs of maps. Narrative generation depends on provider configuration. The rich radiograph response is cached when Redis is available.

Table 3. Task-specific prediction, explanation, and persistence behavior.

Task	Prediction output	Default explanation	Optional or selective output	Persistence
10-year risk of coronary heart disease	Model score and configured category	Signed SHAP contributions	LIME after user action; optional narrative	Compact PostgreSQL record
30-day readmission risk	Model score and configured category	Signed SHAP contributions	LIME after user action; optional narrative	Compact PostgreSQL record
Chest radiograph	Fourteen scores, detected flags, and configured overall status	Paired maps for eligible classes within the five leading ranks	Narrative when configured; secondary image-confirmation gate	Compact PostgreSQL record plus a seven-day Redis response

3.6 Application interface and persistence

The interface uses the same sequence for all tasks: input, prediction status, primary explanation, optional detail, and history. Structured-data pages show a model score, configured category, patient context, signed SHAP drivers, an optional narrative, and a distinct control for LIME. The radiograph page shows a configured overall status, all 14 findings ranked by probability, the original image, and selectable Grad-CAM and Grad-CAM++ views. Figure 3 illustrates this information architecture. The composition is schematic and has not been evaluated with clinicians.

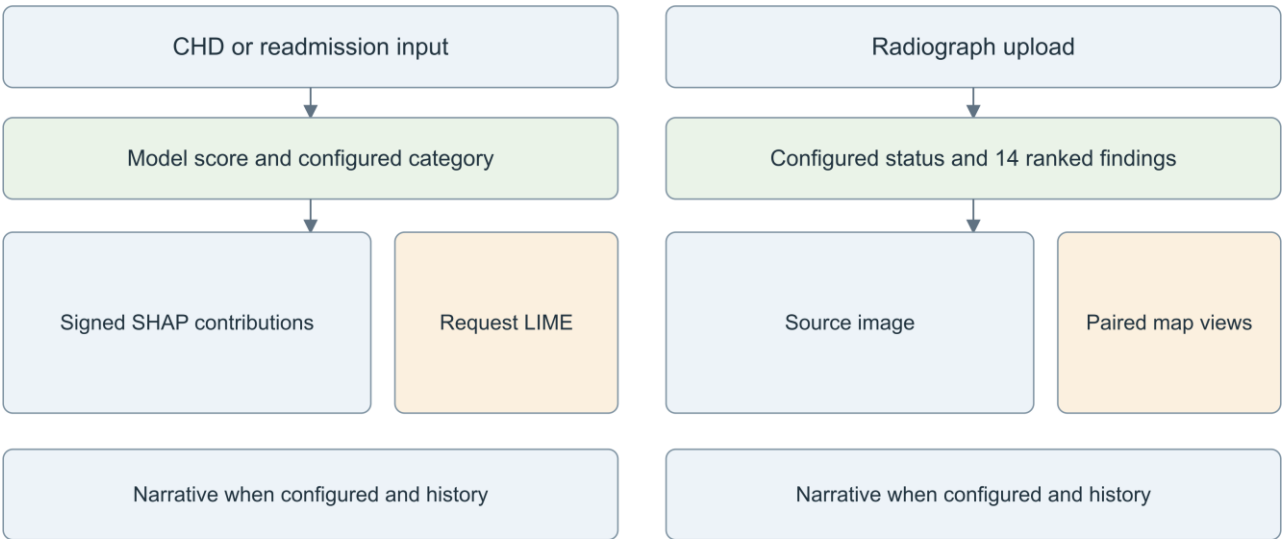


Figure 3. Screen composition in the implemented application. Structured-data views place the model score and configured category alongside SHAP, with a separate LIME action. The imaging view presents all 14 findings, the source image, and paired maps for selected classes. These arrangements describe the interface; comprehension, usability, and clinical effects remain unmeasured.

Each history item contains a universally unique identifier (UUID), user association, timestamp, task type, structured inputs or a radiograph filename, and compact output. Rich radiograph content, including encoded source images and heatmaps, is stored separately in Redis with a seven-day time to live (TTL). The split avoids placing large response bodies in the relational history table while retaining a relational index of completed predictions. Model digests, threshold versions, and explainer configurations are not bound to these records. The displayed categories are application settings, not validated treatment thresholds.

3.7 Evaluation procedures

Evaluation focused on architecture and software behavior. Seven synthetic fixture families checked the image-selection logic without running a model: all-below and all-above cases, threshold equality, exclusion of an eligible sixth-ranked class, class-specific thresholds, equal-score ties, and changes in the rank cap. A separate fixed synthetic vector was evaluated with threshold offsets of -0.05, 0, and +0.05. These controlled inputs test policy conformance and map-invocation counts; they do not estimate selection frequencies in a clinical cohort or execution time. Appendix A provides the fixtures and results, and Supplementary File S1 contains the executable selector snapshot and checks.

The imaging timing experiment decomposed backend work into preprocessing, inference, paired-heatmap generation, and total processing time across 7,818 radiographs. Upload, network transfer, database access, external image confirmation, narrative calls, and browser rendering were excluded. Aggregate stage times are retained, but per-image traces, warm-up conditions, and the workstation hardware configuration were not recorded with the timing summary; these measurements therefore provide descriptive context rather than a portable performance benchmark.

The service-layer load test used ten warm-up calls per route followed by three replicates of 100 measured requests at concurrency levels 1, 4, 8, and 16 for authenticated identity, CHD, and readmission routes, giving 3,600 measured requests. Route handling, JWT authentication, input validation, Structured Query Language (SQL) persistence, and JSON serialization remained active. Predictors, explainers, and narrative functions were replaced by deterministic substitutes; an isolated SQLite database and in-process Asynchronous Server Gateway Interface (ASGI) transport were used. Throughput was averaged across replicates; latency percentiles pooled the 300 requests in each route–concurrency condition. The 95th percentile is denoted p95. Supplementary File S1 includes all request traces, summaries, workload metadata, and the harness. No full-stack or controlled dependency-failure experiment was performed.

Single-case SHAP and LIME comparisons and five LIME repetitions on one CHD input were retained as qualitative illustrations of method disagreement. They do not estimate population stability or determine the interface design. A subsequent multi-case evaluation should assess local-surrogate fidelity, rank and sign stability, and perturbation sensitivity [10,11].

4. RESULTS

4.1 Implemented request paths

All three workflows follow the same application-level contract while maintaining modality-specific internal steps. The two tabular routes validate structured input, apply their task preprocessor, compute a model score, compute SHAP, attempt an optional narrative, persist the compact response, and return the result. LIME remains a separate request tied to the same input supplied again. The image route validates the file, scores 14 findings, selects classes for paired heatmaps, attempts optional text, writes a compact history item, and caches the full response. These request paths address RQ1 by showing that a shared interaction and history model can coexist with distinct explanation strategies.

The shared record structure supports user-scoped longitudinal retrieval, while computational provenance remains incomplete. The identifier, time, task, structured inputs or radiograph filename, and compact output are available. The model digest, training-data release, preprocessing version, threshold set, explainer parameters, random seed, and runtime environment are not bound to every response. Section 5.4 and Appendix B specify the resulting provenance requirements.

4.2 Selection policy and computational bounds

Table 4 compares the analytical map-invocation bounds for four policies. Mapping all 14 outputs requires 28 individual calls; a fixed top-five policy requires ten. The implemented rank-then-threshold rule requires zero to ten calls. All seven synthetic fixture families satisfied their specified conditions. With the all-above fixture, rank caps of 1, 3, 5, and 14 selected 1, 3, 5, and 14 classes, respectively. For the separate threshold-offset fixture, offsets of -0.05 , 0 , and $+0.05$ selected five, four, and zero classes,



corresponding to ten, eight, and zero individual map invocations. Appendix A reports the controlled cases. These results establish bounded behavior on specified inputs; comparative clinical selection frequencies and latency remain to be measured.

Table 4. Analytical bounds on individual heatmap invocations per radiograph.

Policy	Selection rule	Individual map calls	Interpretation
All outputs	Every one of 14 outputs	28	Fixed work across all labels
Fixed top five	Ranks 1–5, without thresholds	10	Fixed rank-based workload
Threshold only	All outputs reaching their class threshold	0–28	No rank cap
Implemented	Threshold-positive outputs within ranks 1–5	0–10	Rank cap plus class-specific gating

4.3 Backend timing and service layer performance

For the radiograph pipeline, mean preprocessing, inference, and heatmap times were 6.0, 33.0, and 140.2 ms, respectively. Mean total backend processing time was 179.3 ms, the median was 193.3 ms, and the maximum was 798.0 ms. Heatmaps accounted for 78.2% of the mean total. Component and total means are independently rounded to one decimal place. The dominance of map generation motivates a workload cap, but these measurements do not establish an optimal cap or a measured improvement over alternative policies.

The service-layer test completed all 3,600 measured requests with successful Hypertext Transfer Protocol (HTTP) responses. At concurrency 16, CHD and readmission routes achieved mean throughputs of 550 and 612 requests/s and pooled p95 latencies of 116.4 and 109.1 ms, respectively. Their pooled median latencies were 3.7 and 3.5 ms. These values characterize deterministic substitutes, SQLite persistence, and in-process transport. Figure 4 displays the two experiments separately; Table 5 states their execution boundaries.

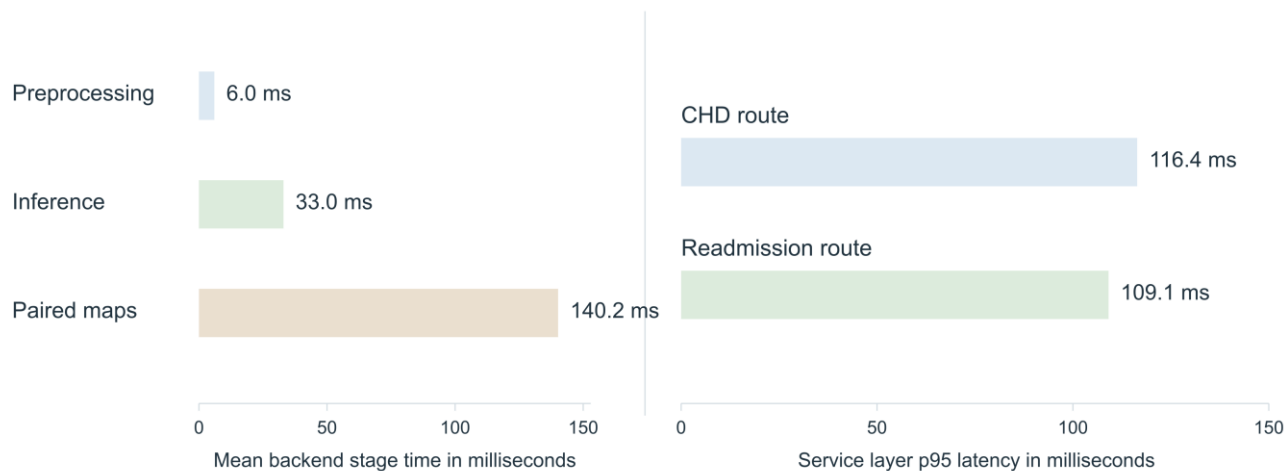


Figure 4. Timing measurements from separate experiments. Left: mean backend stage time over 7,818 radiographs; heatmaps account for 140.2 of 179.3 ms, with component and total means rounded independently. Right: pooled p95 route latency at concurrency 16 using deterministic substitutes, SQLite, and in-process transport. Upload, real-model service-load performance, operational databases, external providers, and browser rendering are not represented by these route measurements.

Table 5. Operational measurements and experimental boundaries.

Experiment	Execution boundary	Result	Excluded components
Radiograph timing	7,818 images; preprocessing, inference, paired maps	Mean total 179.3 ms; maps 140.2 ms; median total 193.3 ms; maximum 798.0 ms	Upload, persistence, network, provider calls, browser
CHD route	Deterministic substitutes; SQLite; in-process; concurrency 16	Mean 550 requests/s; pooled median 3.7 ms; p95 116.4 ms	Real model and explanations, PostgreSQL, Redis, sockets, browser
Readmission route	Deterministic substitutes; SQLite; in-process; concurrency 16	Mean 612 requests/s; pooled median 3.5 ms; p95 109.1 ms	Real model and explanations, PostgreSQL, Redis, sockets, browser

4.4 Illustrative explanation observations

The single-case examples produced different feature rankings under SHAP and LIME. Such differences are technically plausible because the methods use different reference distributions, perturbations, and approximations. Agreement describes local concordance rather than clinical correctness; disagreement calls for inspection without establishing that the underlying prediction is wrong.

Across five LIME repetitions on one CHD input, some coefficients varied substantially, particularly when their mean contribution was close to zero. A coefficient of variation is unstable in that setting and is not used here as evidence of explanation reliability. LIME also depends on discretization, neighborhood sampling, sample count, and surrogate fit. These illustrations justify caution in interpreting post hoc attributions; they do not support a population-level stability claim or a conclusion about clinician preferences.



4.5 Dependency behavior

The request paths define different behavior for optional and required dependencies. A caught narrative failure returns null text while preserving the primary prediction and attribution. A Redis write failure is logged, and compact history can still be committed. Missing radiograph weights produce HTTP 503. The external image-confirmation check fails open after the grayscale heuristic when its provider is unavailable. A PostgreSQL write failure can terminate a request after computation. Table 6 summarizes these implementation paths and their untested conditions; no availability or recovery rate is inferred.

Table 6. Dependency behavior and remaining evaluation.

Condition	Implemented path	Capability retained	Remaining evaluation
Narrative disabled or call fails	Returns null narrative	Prediction and primary explanation	Grounding, latency, failure injection, timeout controls
Redis write fails	Logs warning; relational write proceeds	Immediate response and compact history if commit succeeds	Recovery and cache consistency
Radiograph weights absent	HTTP 503 before inference	Other routes subject to their dependencies	Readiness and artifact integrity checks
Image confirmation unavailable	Grayscale image may pass	Image-processing path remains available	Safety of fail-open admission
PostgreSQL write fails	Request may fail after computation	No committed-history fallback demonstrated	Transactions, recovery, retry

Note. The table describes application paths rather than controlled fault-injection results.

5. DISCUSSION

5.1 Architectural contribution

The main contribution is an explicit mapping from request type to explanation method, invocation trigger, computational limit, retention rule, and failure behavior. This mapping allows each clinical workflow to use its designated explanation without imposing one explanation format or latency budget on the entire application. It also makes the policy testable: each decision can be evaluated for cost, stability, usability, and recovery under its own operating conditions.

This pattern also separates the user-facing contract from the internal model diversity. The shared contract covers identity, input validation, result status, history, and access to optional detail. Task adapters retain their own features, preprocessing, thresholds, and explanations. That separation supports a common extension point; the effort needed to add a pathway has not been measured.

5.2 Cost-aware explanation and persistence

The imaging experiment identifies explanation generation as the largest measured backend stage, while the policy checks establish the map-count bound. They answer different questions: the timing summary describes one configured pipeline, whereas controlled fixtures verify selection behavior. The top-five cap and class thresholds remain policy choices. A high score does not guarantee an informative map, and an eligible class below rank five may be omitted. A subsequent workload study should retain

per-image scores and compare selection frequency, latency, and payload size across caps and thresholds on a locked evaluation sample.

The persistence design applies the same proportionality principle to storage. Compact longitudinal records reside in PostgreSQL, while encoded images and heatmaps expire after seven days. This avoids storing encoded images in each relational record and allows a separate retention policy for visual data. The tradeoff must be visible to users because an older history item may retain its prediction but no longer reproduce the original heatmaps. Storage policy, cache-hit rate, payload size, and privacy controls should be measured in a deployment study.

5.3 Human factors and explanation quality

The interface uses prior literature as a design rationale by using a concise initial result and optional detail [16,18]. No clinician or patient participated, so the effect on comprehension, appropriate reliance, or workflow remains unknown. Users may read signed SHAP values as causal effects, treat SHAP and LIME disagreement as model error, or interpret a Grad-CAM hotspot as a lesion outline. The generated narrative may affect both readability and over-reliance. Formative studies should therefore measure comprehension and decision behavior along with satisfaction.

Explanation evaluation should distinguish agreement from fidelity. The illustrations in Section 4.4 provide no basis for estimating the reliability of either method. A larger study should repeat explanations across prespecified strata and measure local-surrogate fidelity, rank stability, sign consistency, perturbation robustness, and subgroup behavior. Image explanations require separate tests of faithfulness to model behavior and plausibility against expert regions [22,23].

5.4 Reliability and provenance requirements

Failure isolation should be stated per capability. It is reasonable to return a prediction without optional prose if the interface makes the omission clear. Returning a compact history after a cache entry expires is acceptable only when the retention rule is explicit. Falling back from an external image-confirmation service to a local heuristic carries a different safety risk. Operational readiness should therefore expose a capability manifest that reports model load, explainer initialization, database and cache connectivity, and external-service status. The Groq client has no application-level timeout or retry configuration; effective defaults depend on the installed software development kit (SDK). Appendix C records prompt constraints, failure behavior, and these operational limits.

A proposed extension would give each prediction a versioned provenance record. At minimum, it should bind a pseudonymous request and user identifier, input checksum, timestamp, preprocessing version, model name and digest, training-data release, threshold-set version, explanation method and parameters, random seed, where applicable, application version, runtime digest, and narrative-provider status. A future FHIR representation could express the result as an Observation or RiskAssessment and link its generation activity through Provenance [31]. This mapping would support EHR integration, but versioned provenance was not implemented on the evaluated paths. Appendix B separates existing fields from the proposed schema.

5.5 Implications for information systems

The central information system problem is coordination between models, services, stores, and user views. The case therefore illustrates a broader design lesson: explainability should be managed as a service with latency, state, access control, retention, and failure semantics. Treating it this way provides explicit configuration and testing points for additional tasks and infrastructure settings. The current experiments do not compare hospitals or countries, so claims about transfer across settings would be premature.

5.6 Limitations

The evaluation has several limitations. The service-load test used deterministic substitutes, SQLite, and in-process transport; operational full-stack performance and dependency recovery were not measured. Imaging results are aggregate stage timings without a retained hardware record or per-image traces. Synthetic selection checks test software conditions rather than clinical workload distributions. Threshold development used retrospective data without a documented independent split, and output-to-label ordering must be checked against a trusted checkpoint manifest. The study reports no calibration or decision analysis of the configured tabular adapters, no population-level explanation validation, and no clinician evaluation. The scores and categories therefore have no established clinical actionability. Versioned computational provenance also remains a requirement.

The next evaluation should freeze and checksum the model artifacts, verify the output-class and feature manifests, retain stage traces and scores on an independent sample, and compare policy cost under alternative caps and thresholds. The complete application should then be tested with PostgreSQL, Redis, real explainers, network transport, and controlled dependency failures. Calibration, subgroup assessment, decision analysis, and prospective clinician evaluation are subsequent model and clinical validation tasks; they are not inferred from the architecture tests.

6. CONCLUSION

A clinical decision support application can deliver modality-specific explanations through one coherent architecture when explanation selection is treated as a request-time policy. The implementation combines eager SHAP, user-triggered LIME, rank-then-threshold image mapping, and provider-dependent narrative text. Synthetic checks verify the configured selection conditions, while backend timing identifies paired maps as the largest measured imaging stage. The service-load results apply to the stated in-process test. The architecture provides explicit invocation, retention, and failure rules for further systems evaluation; clinical-model validity and clinician benefit require separate investigation.

Ethics Statement

The study used aggregate results from public, de-identified research datasets and evaluated software behavior. It involved no participant recruitment, intervention, access to identifiable health information, or live clinical use. Institutional review board approval and informed consent were therefore not applicable.

Funding Statement

This research received no external funding.



Author Contributions

Danila Lissovoy: Data curation, Methodology, Investigation, Software, Writing – original draft. Assylbek Khalyk: Data curation, Methodology, Investigation, Software, Writing – original draft. Akylbek Oralov: Software. Rakhat Mustafin: Software. Alisher Kenzhegaliyev: Methodology, Software. Hashim Ali: Conceptualization, Validation, Supervision, Writing – review & editing.

Conflicts of Interest

The authors declare no conflict of interest.

Data Availability Statement

The Framingham-derived cardiovascular data, Diabetes 130-US Hospitals data, and NIH ChestX-ray14 data are subject to the access conditions of the sources cited in Section 3.2. Supplementary File S1 contains the architecture test scripts, synthetic fixtures, threshold configuration, request schemas, archived service-load traces, and aggregation procedure. No participant records, trained model artifacts, or per-image radiograph scores and timing traces are distributed. Appendices A–C specify the policies and replication boundaries.

Acknowledgments

None.

APPENDIX A RADIOGRAPH POLICY AND CONFORMANCE CHECKS

A.1 Preprocessing and configured thresholds

The image pathway resizes the shortest side to 256 pixels, center-crops to 224 × 224 pixels, converts to red, green, and blue (RGB) channels, and applies ImageNet normalization. DenseNet-121 produces 14 logits, transformed by a sigmoid. The Grad-CAM target is the final convolution of the final dense layer in the fourth dense block (features.denseblock4.denselayer16.conv2). Table A1 lists the configured thresholds and output order. This order documents the route configuration; its correspondence with a trained checkpoint must be verified from a trusted class manifest before deployment.

Table A1. Configured radiograph output order and class thresholds.

Index	Output label	Threshold
0	Atelectasis	0.460
1	Cardiomegaly	0.484
2	Effusion	0.466
3	Infiltration	0.525
4	Mass	0.524
5	Nodule	0.496
6	Pneumonia	0.345
7	Pleural Thickening	0.291

Index	Output label	Threshold
8	Consolidation	0.521
9	Edema	0.452
10	Emphysema	0.649
11	Fibrosis	0.698
12	Pneumothorax	0.475
13	Hernia	0.786

The thresholds were iteratively blended 30% toward Youden-derived cutoffs using data designated for testing. A threshold-search script and split manifest are not available; independent threshold validation is therefore not established. These values specify operational gating and do not represent validated diagnostic cutoffs.

A.2 Map selection and overall status

Map selection follows the rule in Section 3.5: rank all scores, inspect ranks 1–5, and retain each class whose score reaches its own threshold. Detected flags apply that threshold to all 14 outputs. Overall status is a third rule: ABNORMAL is returned if any of ten trigger classes reaches its threshold plus 0.11; Atelectasis, Nodule, and Consolidation instead require a margin of 0.14. Infiltration, Pneumonia, Pleural Thickening, and Pneumothorax do not trigger overall status. Thus the literal interface label NORMAL can coexist with a threshold-positive finding. These status labels must not be interpreted as verified clinical normality.

A.3 Synthetic conformance and controlled sensitivity

Table A2 reports seven hand-constructed fixture families. The executable checks extract the two ranking-and-selection assignments and literal configuration from the route using the Python abstract syntax tree, avoiding model or service imports. A portable snapshot is included in Supplementary File S1. The checks use NumPy 2.3.5 and double-precision synthetic scores; no patient data or model inference is involved.

Table A2. Results of the seven synthetic selection fixtures.

Fixture	Selected classes	Individual maps
All scores below thresholds	0	0
All scores above thresholds, cap five	5	10
One score exactly equals its threshold	1	2
Six outputs eligible, sixth rank excluded	5	10
Four higher scores fail their own thresholds; fifth qualifies	1	2
Equal-score ties, cap five	5	10
Cap changed to 1 / 3 / 5 / 14 on all-above scores	1 / 3 / 5 / 14	2 / 6 / 10 / 28

All seven families met their expected conditions. Equal-score ordering repeated over ten calls in the same runtime, but the implementation has no explicit stable tie-breaker; ordering is not guaranteed across NumPy versions. For an additional fixed



vector, the five leading scores were 0.80 (Hernia), 0.70 (Fibrosis), 0.65 (Emphysema), 0.54 (Infiltration), and 0.52 (Mass); all other scores were zero. Adding -0.05 , 0 , or $+0.05$ to every threshold selected five, four, or zero classes, respectively, giving ten, eight, or zero map invocations. These controlled results describe count sensitivity on one specified vector. Cohort selection frequencies, diagnostic effects, and actual map execution times require separate evaluation.

APPENDIX B PROVENANCE REQUIREMENTS

B.1 Existing history and proposed extension

Table B1 distinguishes the existing record from the proposed extension. A history entry contains a prediction identifier, user association, timestamp, task, and compact inputs and outputs. The radiograph response also names its model and returns thresholds. The evaluated paths do not bind these outputs to artifact digests, explainer configuration, or a versioned runtime.

Table B1. Existing history fields and proposed provenance requirements.

Field group	Existing record	Proposed extension
Identity and time	Prediction identifier, user, timestamp	Stable request identifier and clock semantics
Input	Structured input or compact imaging output	Authorized source identifier and input digest
Model	Human-readable radiograph model label	Model checksum, data release, feature/class manifest, preprocessing version
Policy	Radiograph thresholds returned in response	Threshold-set version, rank cap, status margins
Explanation	SHAP contributions or selected maps	Method and library versions, background sample, target layer, parameters, applicable seed
Execution	No runtime bound to each result	Application commit, environment digest, provider status

B.2 Persistence and access

A future provenance entry should be committed with the compact prediction record and inherit its user-scoped authorization. A schema version, retention policy, and migration plan are needed before the record can support replay. A FHIR Provenance resource could connect the producing activity to an Observation or RiskAssessment [31]. This is a requirements specification; no implemented provenance endpoint, versioned ledger, or FHIR exchange is claimed. An expired Redis image cannot be recovered from a compact history record.

APPENDIX C REQUEST CONTRACTS AND REPRODUCIBILITY

C.1 Routes and input schemas

Table C1 lists the implemented route contract. Prediction, explanation, and history requests require JWT authentication; login issues the token. LIME endpoints accept the structured input again and do not accept an existing prediction identifier.

Supplementary File S1 includes the complete machine-readable request-field definitions extracted from the structured-data schemas.

Table C1. Implemented routes and principal responses.

Method and path	Request	Principal response
POST /auth/login	Credentials	Bearer token
GET /auth/me	Bearer token	Authenticated user
POST /chd/predict	13-field CHD JSON	Score, category, SHAP, narrative, prediction identifier
POST /chd/explain-lime	CHD JSON	Recomputed score and LIME explanation
POST /readmission/predict	39-field encounter JSON	Score, category, SHAP, narrative, prediction identifier
POST /readmission/explain-lime	Encounter JSON	Recomputed score and LIME explanation
POST /predict	Multipart radiograph	14 findings, overall status, source image, selected maps, narrative, identifier
GET /history	Bearer token	Up to 50 compact user-scoped records
GET /history/{prediction_id}	Bearer token	Cached rich response when available, otherwise compact record

The CHD fields are age, sex, total_cholesterol, systolic_bp, smoking, diabetes, bmi, heart_rate, glucose, bp_meds, prevalent_hypertension, cigs_per_day, and pulse_pressure. Here bmi denotes body mass index, bp blood pressure, and meds medication; these strings are API identifiers. Sex, smoking, diabetes, medication use, and prevalent hypertension are numeric indicators. The configured category boundaries are 0.15 and 0.439 for CHD and 0.102 and 0.30 for readmission. Their selection has no documented calibration or decision-analytic justification; they are display settings for a research application. SHAP and LIME return up to ten contributions. LIME requests specify 1,000 perturbation samples for CHD and 500 for readmission; request functions do not set a random seed.

C.2 Image and failure semantics

The image endpoint accepts Joint Photographic Experts Group (JPEG; .jpg and .jpeg), Portable Network Graphics (PNG), bitmap (.bmp), Tagged Image File Format (TIFF; .tif and .tiff), Digital Imaging and Communications in Medicine (DICOM; .dcm), WebP (.webp), and Graphics Interchange Format (GIF) files. It checks readability, applies a grayscale heuristic, and requests external image confirmation when available. DICOM decoding requires an optional package. No application file-size limit is configured. Unsupported extensions produce HTTP 400, a rejected non-radiograph produces HTTP 422, and absent model weights produce HTTP 503. Redis failures are logged; the response has no cache-warning field. A relational commit failure may terminate the request after computation.

C.3 Narrative configuration

Groq provides both text generation and secondary image confirmation. The defaults are llama-3.1-8b-instant for narratives and meta-llama/llama-4-scout-17b-16e-instruct for image confirmation. The text prompt requests three to four sentences grounded



in supplied scores and SHAP contributions, prohibits treatment recommendations, and ends with a screening-not-diagnosis statement; output is limited to 300 tokens. The image prompt requests a yes/no medical-scan decision and allows five tokens. The software development kit (SDK) client has no application-level timeout or retry settings. Calls are attempted when credentials are configured; caught text failures yield null. An image-confirmation failure allows a grayscale image to proceed. Prompt constraints provide intended grounding, not a measured guarantee against hallucination. No content-safety, provider-latency, or fault-injection result is claimed.

C.4 Supplementary files and replication boundaries

Table C2 describes Supplementary File S1, including JSON and comma-separated values (CSV) files. Service-load traces contain 3,600 measured requests and no clinical records. The recorded host used Python 3.12.14 on an arm64 macOS platform with 15 logical processors; the exact processor model and memory capacity were not recorded. Core recorded versions were FastAPI 0.121.3, HTTPX 0.28.1, SQLAlchemy 2.0.44, Pydantic 2.13.5, NumPy 2.3.5, and pandas 2.2.3. Throughput is the arithmetic mean of three replicate rates; p95 pools 300 latencies per route and concurrency condition.

Table C2. Contents and scope of Supplementary File S1.

Directory or file	Contents	Replication scope
policy/	Selector snapshot, synthetic checks, JSON and CSV outcomes	Selection conditions and count sensitivity
load/	Original request traces, replicate and aggregate summaries, workload metadata	Recalculation of service-load results
scripts/	Relocated load harness and aggregation script	Repeatable procedure with installed dependencies; host-dependent timings
contracts/	Request schemas and threshold/status configuration	Exact interface fields and policy settings
source/	Selected application source without credentials or model files	Inspection and service-layer harness support
MANIFEST.sha256	Secure Hash Algorithm with 256-bit digests (SHA-256)	Integrity of packaged files

The package preserves the original load traces and provides a current source snapshot with checksums; the historical application commit was not recorded. Re-running the harness may change timings. The full clinical stack additionally requires the random-forest artifact, LightGBM model and preprocessing objects, DenseNet-121 checkpoint, trusted feature and class manifests, and a complete environment lock. These are not included. The aggregate radiograph timings cannot be independently reconstructed without their original per-image traces and execution environment.

REFERENCES

1. Lekadir, K., Frangi, A. F., Porras, A. R., Glocker, B., Cintas, C., Langlotz, C. P., Weicken, E., Asselbergs, F. W., Prior, F., Collins, G. S., Kaissis, G., Tsakou, G., Buvat, I., Kalpathy-Cramer, J., Mongan, J., Schnabel, J. A., Kushibar, K., Riklund, K., Marias, K., ... Starmans, M. P. A. (2025). FUTURE-AI: International consensus guideline for trustworthy and deployable artificial intelligence in healthcare. *BMJ*, 388, e081554. <https://doi.org/10.1136/bmj-2024-081554>

2. Vasey, B., Nagendran, M., Campbell, B., Clifton, D. A., Collins, G. S., Denaxas, S., Denniston, A. K., Faes, L., Geerts, B., Ibrahim, M., Liu, X., Mateen, B. A., Mathur, P., McCradden, M. D., Morgan, L., Ordish, J., Rogers, C., Saria, S., Ting, D. S. W., ... Perkins, Z. B. (2022). Reporting guideline for the early-stage clinical evaluation of decision support systems driven by artificial intelligence: DECIDE-AI. *Nature Medicine*, 28(5), 924–933. <https://doi.org/10.1038/s41591-022-01772-9>

3. Collins, G. S., Moons, K. G. M., Dhiman, P., Riley, R. D., Beam, A. L., Van Calster, B., Ghassemi, M., Liu, X., Reitsma, J. B., Van Smeden, M., Boulesteix, A.-L., Camaradou, J. C., Celi, L. A., Denaxas, S., Denniston, A. K., Glocker, B., Golub, R. M., Harvey, H., Heinze, G., ... Logullo, P. (2024). TRIPOD+AI statement: Updated guidance for reporting clinical prediction models that use regression or machine learning methods. *BMJ*, 385, e078378. <https://doi.org/10.1136/bmj-2023-078378>
4. Acosta, J. N., Falcone, G. J., Rajpurkar, P., & Topol, E. J. (2022). Multimodal Biomedical AI. *Nature Medicine*, 28(9), 1773–1784. <https://doi.org/10.1038/s41591-022-01981-2>
5. Soenksen, L. R., Ma, Y., Zeng, C., Boussioux, L., Villalobos Carballo, K., Na, L., Wiberg, H. M., Li, M. L., Fuentes, I., & Bertsimas, D. (2022). Integrated Multimodal Artificial Intelligence Framework for Healthcare Applications. *npj Digital Medicine*, 5, 149. <https://doi.org/10.1038/s41746-022-00689-4>
6. Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. *Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17*, 4768–4777. <https://dl.acm.org/doi/10.5555/3295222.3295230>
7. Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). “Why Should I Trust You?”: Explaining the Predictions of Any Classifier. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1135–1144. <https://doi.org/10.1145/2939672.2939778>
8. Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. *2017 IEEE International Conference on Computer Vision (ICCV)*, 618–626. <https://doi.org/10.1109/ICCV.2017.74>
9. Chattopadhyay, A., Sarkar, A., Howlader, P., & Balasubramanian, V. N. (2018). Grad-CAM++: Generalized Gradient-Based Visual Explanations for Deep Convolutional Networks. *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 839–847. <https://doi.org/10.1109/WACV.2018.00097>
10. Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., & Kim, B. (2018). Sanity checks for saliency maps. *Advances in Neural Information Processing Systems* 31, 9505–9515.
11. Slack, D., Hilgard, S., Jia, E., Singh, S., & Lakkaraju, H. (2020). Fooling LIME and SHAP: Adversarial Attacks on Post hoc Explanation Methods. *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, 180–186. <https://doi.org/10.1145/3375627.3375830>
12. Babic, B., Gerke, S., Evgeniou, T., & Cohen, I. G. (2021). Beware Explanations from AI in Health Care. *Science*, 373(6552), 284–286. <https://doi.org/10.1126/science.abg1834>
13. Ghassemi, M., Oakden-Rayner, L., & Beam, A. L. (2021). The false hope of current approaches to explainable artificial intelligence in health care. *The Lancet Digital Health*, 3(11), e745–e750. [https://doi.org/10.1016/S2589-7500\(21\)00208-9](https://doi.org/10.1016/S2589-7500(21)00208-9)
14. Ali, H. (2026). Causal-Pathway-Guided DNN-GBDT Distillation for Interpretable Artificial Intelligence in Intensive Care Units. *Informatics*, 13(8), 124. <https://doi.org/10.3390/informatics13080124>
15. Chen, H., Gomez, C., Huang, C.-M., & Unberath, M. (2022). Explainable medical imaging AI needs human-centered design: Guidelines and evidence from a systematic review. *npj Digital Medicine*, 5, 156. <https://doi.org/10.1038/s41746-022-00699-2>
16. Bayor, A. A., Li, J., Yang, I. A., & Varnfield, M. (2025). Designing Clinical Decision Support Systems (CDSS)—A User-Centered Lens of the Design Characteristics, Challenges, and Implications: Systematic Review. *Journal of Medical Internet Research*, 27, e63733. <https://doi.org/10.2196/63733>
17. Calem, J., Dias, J., & Nunes, N. J. (2024). Intelligent Systems in Healthcare: A Systematic Survey of Explainable User Interfaces. *Computers in Biology and Medicine*, 180, 108908. <https://doi.org/10.1016/j.combiomed.2024.108908>
18. Van Dort, B. A., Engelsma, T., Sneekes, P., Peute, L., & Medlock, S. (2026). Explainable AI in Hospital Clinical Decision Support Systems: A Scoping Review of Healthcare Professionals’ Perspectives. *PLOS Digital Health*, 5(5), e0001417. <https://doi.org/10.1371/journal.pdig.0001417>
19. Hur, S., Lee, Y., Park, J., Jeon, Y. J., Cho, J. H., Cho, D., Lim, D., Hwang, W., Cha, W. C., & Yoo, J. (2025). Comparison of SHAP and Clinician Friendly Explanations Reveals Effects on Clinical Decision Behaviour. *npj Digital Medicine*, 8, 578. <https://doi.org/10.1038/s41746-025-01958-8>
20. Kayser, M., Menzat, B., Emde, C., Bercean, B., Novak, A., Espinosa, A., Papiez, B. W., Gaube, S., Lukasiewicz, T., & Camburu, O.-M. (2024). Fool Me Once? Contrasting Textual and Visual Explanations in a Clinical Decision-Support Setting. *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 18891–18919. <https://doi.org/10.18653/v1/2024.emnlp-main.1051>
21. Saporta, A., Gui, X., Agrawal, A., Pareek, A., Truong, S. Q. H., Nguyen, C. D. T., Ngo, V.-D., Seekins, J., Blankenberg, F. G., Ng, A. Y., Lungren, M. P., & Rajpurkar, P. (2022). Benchmarking Saliency Methods for Chest X-Ray Interpretation. *Nature Machine Intelligence*, 4(10), 867–878. <https://doi.org/10.1038/s42256-022-00536-x>
22. Brankovic, A., Cook, D., Rahman, J., Delaforce, A., Li, J., Magrabi, F., Cabitza, F., Coiera, E., & Bradford, D. (2025). Clinician-informed XAI evaluation checklist with metrics (CLIX-M) for AI-powered clinical decision support systems. *npj Digital Medicine*, 8(1), 364. <https://doi.org/10.1038/s41746-025-01764-2>
23. Jin, W., Li, X., Fatehi, M., & Hamarneh, G. (2023). Guidelines and Evaluation of Clinical Explainable AI in Medical Image Analysis. *Medical Image Analysis*, 84, 102684. <https://doi.org/10.1016/j.media.2022.102684>
24. Rosenbacke, R., Melhus, Å., McKee, M., & Stuckler, D. (2024). How Explainable Artificial Intelligence Can Increase or Decrease Clinicians’ Trust in AI Applications in Health Care: Systematic Review. *JMIR AI*, 3, e53207. <https://doi.org/10.2196/53207>
25. International Organization for Standardization. (2022). ISO/IEC/IEEE 42010:2022: Software, Systems and Enterprise—Architecture Description (ISO/IEC/IEEE 42010:2022). International Organization for Standardization. <https://www.iso.org/standard/74393.html>
26. International Organization for Standardization. (2023). ISO/IEC 25010:2023: Systems and software engineering—Systems and software Quality Requirements and Evaluation (SQuaRE)—Product quality model. <https://www.iso.org/standard/78176.html>
27. International Organization for Standardization. (2023). ISO/IEC 25059:2023: Software Engineering—Systems and Software Quality Requirements and Evaluation (SQuaRE)—Quality Model for AI Systems (ISO/IEC 25059:2023). International Organization for Standardization. <https://www.iso.org/standard/80655.html>
28. Tabassi, E. (2023). Artificial Intelligence Risk Management Framework (AI RMF 1.0) (NIST AI 100-1). National Institute of Standards and Technology. <https://doi.org/10.6028/NIST.AI.100-1>
29. Huang, J., & Liu, Y. (2025). MLXOps4Medic: A Service Framework for Machine Learning and Explainability Operations in Medical Imaging AI Development. *IEEE Access*, 13, 158149–158169. <https://doi.org/10.1109/ACCESS.2025.3606838>

30. Yoo, J., Lee, J., Min, J. Y., Choi, S. W., Kwon, J.-M., Cho, I., Lim, C., Choi, M. Y., & Cha, W. C. (2022). Development of an Interoperable and Easily Transferable Clinical Decision Support System Deployment Platform: System Design and Development Study. *Journal of Medical Internet Research*, 24(7), e37928. <https://doi.org/10.2196/37928>
31. HL7 International. (2023). FHIR Release 5: Provenance Resource. <https://hl7.org/fhir/R5/provenance.html>
32. D'Agostino, R. B., Sr., Vasan, R. S., Pencina, M. J., Wolf, P. A., Cobain, M., Massaro, J. M., & Kannel, W. B. (2008). General Cardiovascular Risk Profile for Use in Primary Care: The Framingham Heart Study. *Circulation*, 117(6), 743–753. <https://doi.org/10.1161/CIRCULATIONAHA.107.699579>
33. Strack, B., DeShazo, J. P., Gennings, C., Olmo, J. L., Ventura, S., Cios, K. J., & Clore, J. N. (2014). Impact of HbA1c Measurement on Hospital Readmission Rates: Analysis of 70,000 Clinical Database Patient Records. *BioMed Research International*, 2014, 781670. <https://doi.org/10.1155/2014/781670>
34. Wang, X., Peng, Y., Lu, L., Lu, Z., Bagheri, M., & Summers, R. M. (2017). ChestX-ray8: Hospital-Scale Chest X-Ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases. *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition*, 3462–3471. <https://doi.org/10.1109/CVPR.2017.369>
35. Rajpurkar, P., Irvin, J., Zhu, K., Yang, B., Mehta, H., Duan, T., Ding, D., Bagul, A., Langlotz, C., Shpanskaya, K., Lungren, M. P., & Ng, A. Y. (2017). CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning (arXiv:1711.05225). *arXiv*. <https://doi.org/10.48550/arXiv.1711.05225>