Chapter 4

Development of Smart-Doc System

The previous chapter highlighted both the promise and the limitations of AI-powered virtual patients: while feasible and increasingly sophisticated, open questions remain regarding their capacity to foster complex reasoning, support metacognition, and address cognitive biases in authentic learning contexts.

In response to these gaps, the **SmartDoc** platform was developed to operationalize the theoretical and empirical foundations outlined in Chapters 2 and 3. SmartDoc is designed as a bias-aware clinical simulation environment that enables medical students to practice diagnostic reasoning in natural language while receiving targeted metacognitive feedback. The system integrates the principles of dual-process reasoning, bias taxonomy, and debiasing strategies (Chapter 2) with state-of-the-art advances in AI-powered virtual patients (Chapter 3).

The architecture prioritizes three pedagogical goals:

1. **uthenticity** — Realistic patient interactions through conversational interfaces that mimic real anamnesis, bridging the theory–practice gap.

2. **Bias wareness** — Detection of reasoning patterns associated with anchoring, confirmation, or premature closure, with just-in-time prompts to encourage reflection.

3. **Structured Reflection** — Embedded checkpoints that support deliberate reflection and consolidation of diagnostic reasoning skills.

In this chapter, we describe the design of the SmartDoc platform. Section 4.1 introduces the conceptual system architecture and core components. Section 4.2.2 explains how language models are integrated to balance realism and reliability. Section 4.2.6 details the novel intent-driven case design, grounded in Mull et al.'s clinical case of cognitive error. Finally, Section 4.5 illustrates a typical workflow from the learner's perspective.

4.1 System Design

4.1.1 Conceptual Overview

SmartDoc implements a modular simulation architecture built on the principle of *progressive disclosure* (see Section 2.2.2) and informed by experiential learning theory. The system separates responsibilities into three layers:

- **Domain logic** — the reasoning engine and pedagogical rules,
- **PI layer** — orchestrating communication and state management,
- **Presentation layer** — the learner-facing interface for interacting with the virtual patient.

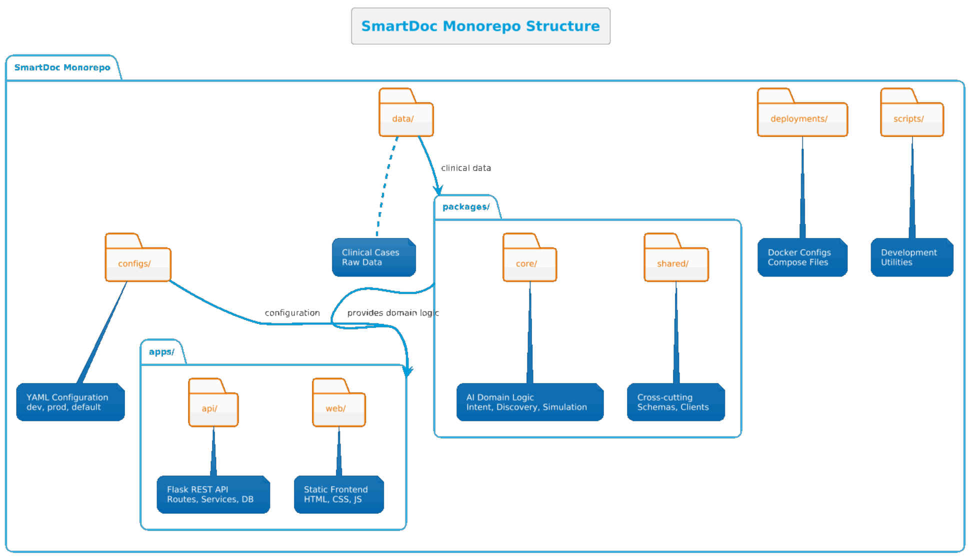

Figure 4.1: High-level architecture of SmartDoc showing the interaction between core modules.

At the core of the system lies the *Intent-Driven Disclosure Manager*, which orchestrates three essential processes:

1. **Understanding learner input** via an LLM-powered *Intent Classifier*,
2. **Controlling information flow** through a *Discovery Processor* that releases case data incrementally,
3. **Monitoring reasoning patterns** through a *Bias Analyzer* that detects bias-prone behaviors in real time.

This architecture ensures that the simulation reflects authentic diagnostic practice (as emphasized in Chapter 3) while embedding cognitive safeguards to mitigate common reasoning errors (Chapter 2).

4.1.2 Core Components

Intent Classification. Learner utterances are mapped to a predefined set of diagnostic intents (e.g., request history, order test, suggest diagnosis). Rather than relying on rigid question trees, SmartDoc leverages natural language classification, allowing students to interact freely. This module operationalizes *System 1 pattern recognition* while keeping pathways open for *System 2 monitoring*.

Discovery Processing. Clinical information is released progressively, linked to learner queries. This implements the principle of **progressive disclosure**, ensuring that data emerges as a result of active inquiry rather than passive reception. Such design aligns with constructivist learning theory, promoting deep engagement over rote recall.

Simulation Engine. This orchestrates the entire learning loop:

```
Student asks   Intent classified   Relevant information disclosed
  Bias analysis applied   Response and feedback generated
```

It integrates reasoning traces with pedagogical scaffolds, ensuring each step is both educational and technically coherent.

Bias nalyzer. Bias detection is achieved through a combination of rule-based heuristics (e.g., repeated focus on one hypothesis) and LLM reasoning analysis. The module is directly grounded in the bias taxonomy (Section 2.2.2), with real-time interventions such as:

```
IF focus > 70% on same hypothesis THEN prompt "What else could
explain these findings?"
```

This transforms abstract debiasing strategies into actionable, in-the-moment coaching.

Session Management. All interactions are logged with bias timestamps and reflection responses. This provides a *reasoning trace* for both formative feedback and summative assessment, enabling empirical evaluation of diagnostic reasoning skills (see Chapter 5).

Together, these components implement a learning cycle where action, feedback, and reflection are tightly coupled, addressing the shortcomings of both traditional scripted simulations and unguided clinical encounters.

4.2 Technical Implementation

The SmartDoc system was implemented as a modular pipeline that translates free-text learner inputs into pedagogically meaningful interactions. Its design reflects three principles identified in earlier chapters: (i) *progressive disclosure* of clinical information (Chapter 2), (ii) *real-time bias detection* and metacognitive scaffolding (Chapter 2), and (iii) *reliability and reproducibility* across contexts, a key limitation noted in recent studies (Chapter 3).

4.2.1 rchitecture & Execution Pipeline

The pipeline follows a sequence of phases (Figure **??**) that mirror the reasoning process of a medical encounter. Each phase is encapsulated in a dedicated module, enabling independent testing and future extension.

1. Query Initiation and Routing. When a learner submits a question (e.g., *"What are the patient's vital signs?"*), the system captures both the input and its context (session, diagnostic phase). The request is passed to the central orchestrator, the *Intent-Driven Disclosure Manager*.

2. Intent Classification. The system determines what the learner is trying to do (e.g., gather history, request labs, propose diagnosis). This operationalizes **System 1 pattern recognition** while maintaining hooks for **System 2 monitoring**. If the confidence of classification is low, the system applies fallbacks to preserve robustness.

3. Discovery Processing. Once the intent is identified, the system decides what information to reveal. Information is disclosed progressively, ensuring that students must *seek* evidence rather than receive it passively. This implements the pedagogical principle of *progressive disclosure*, encouraging active inquiry and preventing premature closure.

4. Response Generation. Contextually appropriate responses are generated to maintain immersion:

- anamnesis phase: family member dialogue,
- physical exam: descriptive findings,
- investigations: lab or imaging reports.

This provides authenticity while aligning with experiential learning theory.

5. Bias Detection and Session Logging. In parallel, the *Bias Analyzer* monitors interaction patterns for signs of anchoring, confirmation bias, or premature closure. Simple heuristics and LLM-based analysis are combined, for example:

```
if focus_on_single_hypothesis > 70%:
    trigger_bias("anchoring")
    prompt_reflection("What else could explain these findings?")
```

Detected events are logged with timestamps, creating a traceable record of reasoning for both feedback and later research analysis.

6. ssembly and Delivery. The system compiles the patient reply, any new discoveries, bias warnings, and progress indicators into a structured response. This is returned to the learner's interface and persisted for evaluation.

This modular execution pipeline ensures controlled information flow (supporting debiasing strategies), authentic immersion, and reproducibility—directly addressing both pedagogical requirements and gaps identified in Chapter 3.

4.2.2 LLM Integration

Given the variability of large language models, SmartDoc uses an abstraction architecture to remain model-agnostic, ensuring both flexibility and reproducibility.

4.2.2.1 Provider bstraction

All models are accessed through a standard `LLMProvider` interface with a single `generate()` function. This design:

- decouples pedagogical logic from vendor-specific implementations,
- allows deployment in diverse environments (local models for research, cloud-based models for production),
- supports substitution as models evolve, a key concern raised in the literature about sustainability.

4.2.2.2 Dependency Injection

Rather than hardcoding model use, providers are injected into each component (e.g., intent classifier, bias analyzer). This enables:

- isolated testing with mock providers,
- configuration tailored to educational scenarios (fast models for classification, more capable models for reasoning),
- consistent software engineering practices that ensure reliability.

4.2.2.3 Prompt Engineering

Prompts are modular and configurable, allowing experimentation without changing the underlying code. Each prompt defines:

- *role*: e.g., "You are a patient describing symptoms,"
- *context*: e.g., "current diagnostic phase = anamnesis,"

- *constraints*: e.g., “reply in JSON format with labels.”

This structure supports A/B testing of pedagogical strategies (e.g., different ways of framing reflection questions), linking directly to the literature’s call for rigorous evaluation of design choices.

4.2.2.4 Testing and Validation

Robustness is ensured by simulating diverse model behaviours:

- normal responses,
- timeouts or unavailable services,
- unexpected outputs.

Fallback mechanisms (e.g., switching to deterministic mappings) maintain availability, ensuring that educational functionality is not compromised by model instability. This directly addresses concerns in Chapter 3 regarding reproducibility of AI-VP systems.

4.2.3 Database & State Management

To support both responsive simulation and robust research analytics, SmartDoc adopts a dual-layer state management architecture:

1. **In-memory session state** — ensures real-time responsiveness during the learner’s interaction with the virtual patient.
2. **Persistent database storage** — captures full reasoning traces, bias events, and reflection data for subsequent analysis.

This separation allows the system to deliver immediate educational feedback while also creating durable records for evaluation and research.

4.2.3.1 Conceptual Schema

The underlying schema is organized around *educational workflows* rather than technical implementation. Key entities include:

- `Conversations`: the main unit of analysis, linking all messages and events.
- `Messages`: full history of learner queries and system responses.
- `DiscoveryEvents`: when and how clinical information was revealed (operationalizing *progressive disclosure*).
- `BiasWarnings`: logged when patterns of anchoring, confirmation, or premature closure are detected (direct link to Chapter 2).

- `ReflectionResponses`: learner's answers to structured prompts, supporting deliberate reflection research.
- `DiagnosisSubmissions`: final diagnostic hypotheses with justification, enabling outcome evaluation.

A simplified view of the schema is illustrated in Figure 4.2. This design ensures that each learner session produces a rich, analyzable dataset that links behaviour (intents, queries) with outcomes (diagnosis accuracy, bias awareness).

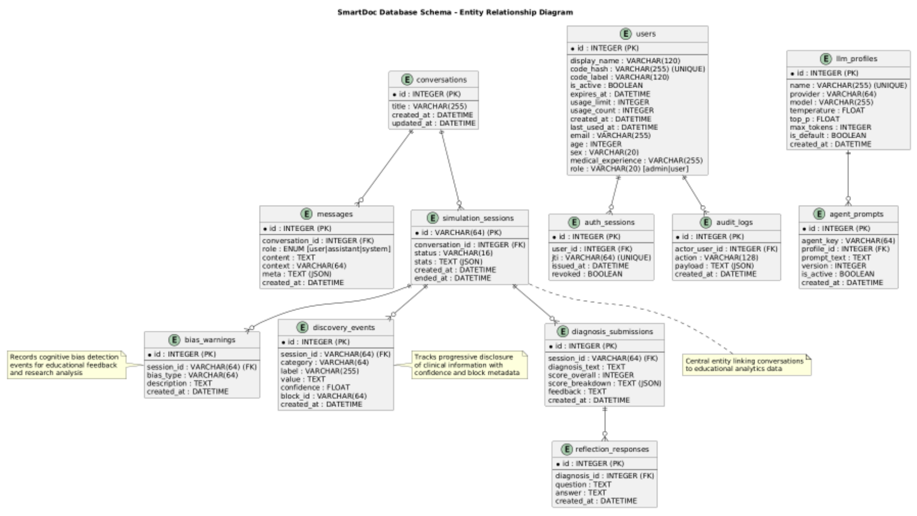

Figure 4.2: Conceptual schema: capturing reasoning traces, bias events, and reflection data.

4.2.3.2 State Management rchitecture

During a simulation, the *Progressive Disclosure Store* maintains session-level state in memory (revealed information blocks, queries, working hypotheses). Hooks such as `on_reveal` or `on_interaction` trigger updates to the database when key pedagogical events occur. This allows the system to both:

- deliver instant responses to the learner, and
- log structured traces for later analysis.

Pseudo-code sketch:

```
on_reveal(info_block):
    update_session_state(info_block)
    log_to_database(event="discovery", timestamp=now)

on_bias_detected(bias_type):
    issue_warning(bias_type)
    log_to_database(event="bias", bias=bias_type, timestamp=now)
```

This event-driven design ensures that cognitive bias detection (Chapter 2) and reflection prompts are not only experienced by the learner but also captured for empirical study (Chapter 3).

4.2.3.3 Choice of Database Technology

For research purposes, portability and reproducibility were prioritised. A lightweight single-file database (*SQLite*) was chosen because it:

- allows easy sharing of complete simulation datasets for replication studies,
- eliminates external dependencies for participants in multi-site trials,
- guarantees data integrity even in abrupt session terminations.

Although SQLite does not support massive concurrency, SmartDoc's use case involves individual or small-cohort learning sessions, making this a pragmatic and effective choice. The abstraction layer (SQLAlchemy) ensures that the system can migrate to enterprise-grade databases (e.g. PostgreSQL) if future large-scale deployments require it.

4.2.3.4 Logging and Reproducibility

SmartDoc implements structured logging for both technical and pedagogical events. For each session, the following are captured:

- learner queries and classified intents,
- information blocks revealed and their sequence,
- bias warnings triggered,
- reflection responses submitted.

These logs create a complete *reasoning trace*, enabling:

1. **Formative feedback** — personalised debriefs for learners.
2. **Summative evaluation** — assessment of diagnostic accuracy and bias awareness.
3. **Research analytics** — large-scale studies of interaction patterns, replicating the calls in Chapter 3 for more transparent evaluation of AI-VPs.

By embedding reproducibility at the data layer, SmartDoc ensures that each learning session doubles as a research opportunity.

4.2.4 Deployment

A critical requirement for SmartDoc is that it can be reliably deployed in both research and educational settings. The deployment architecture was therefore designed with three guiding principles:

1. **Reproducibility** — ensuring that the same configuration can be reproduced across different sites and machines.
2. **Scalability** — enabling the platform to support cohorts of learners by scaling up model inference capacity when needed.
3. **ccessibility** — minimizing technical barriers for institutions, favouring lightweight setups that do not require specialised infrastructure.

4.2.4.1 Containerisation and Reproducibility

To guarantee consistent behaviour across environments, the system is distributed as containerised services (e.g., backend API, database, language model interface). This encapsulation ensures that all dependencies are versioned and portable, allowing educational institutions or research collaborators to reproduce experiments without lengthy installation steps. Containerisation also aligns with recent calls in the literature for reproducible AI-based educational systems (see Chapter 3).

4.2.4.2 Model Hosting and Flexibility

SmartDoc supports two main deployment modes:

- *Local hosting*, using lightweight models for research replication and settings with limited internet access.
- *Cloud hosting*, connecting to high-accuracy commercial models for large-scale or production use.

This dual-mode design reflects a balance between *cost*, *accuracy*, and *data governance*, allowing institutions to select the configuration best suited to their pedagogical and ethical context.

4.2.4.3 Scalability Considerations

Although individual learners typically interact with the system in isolation, classroom or multi-cohort settings require concurrent sessions. The architecture therefore supports horizontal scaling: additional inference services can be launched in parallel, each serving multiple SmartDoc instances. This ensures that response times remain low, preserving immersion and educational value even with larger groups.

4.2.4.4 Security and Data Integrity

Given that learner interactions include reasoning traces and reflection data, deployment must also ensure *data integrity and privacy*. The platform implements:

- role-based access (distinguishing learners, educators, administrators),
- encrypted storage of sensitive configuration parameters,
- automated backups of educational data.

These safeguards align with ethical considerations around learner data use raised in Chapter 3.

4.2.4.5 Educational Impact of Deployment Design

By prioritising reproducibility and portability, SmartDoc enables institutions to adopt the platform with minimal setup, making bias-aware simulation accessible beyond well-resourced centres. The scalability features ensure that the system remains responsive in group teaching contexts, while the emphasis on security reinforces trust in its use for formal training. Together, these choices demonstrate that deployment is not only a technical concern but also an *educational enabler*, supporting widespread adoption and rigorous evaluation.

4.2.5 User Interfaces

The SmartDoc platform provides a web-based interface designed explicitly for clinical education. Its design prioritises usability, immersion, and alignment with pedagogical principles such as progressive disclosure, scaffolded learning, and metacognitive reflection.

4.2.5.1 Interface rchitecture

Three complementary interfaces support different roles:

- **Simulation Interface** — the learner-facing environment for conducting clinical interviews and reasoning through cases.
- **dministrative Dashboard** — tools for educators and researchers to configure cases, manage learners, and monitor outcomes.
- **uthentication Portal** — secure entry point with role-based access for learners and instructors.

This separation of concerns reflects the dual needs of *educational delivery* and *research oversight*.

4.2.5.2 Medical Simulation Interface

The simulation interface is organised around the phases of diagnostic reasoning, each represented as a tab or section (see Figure 4.3):

1. **Patient Information** — static demographic and background data.
2. **Clinical Interview (namnesis)** — free-text dialogue with the virtual patient or family member.
3. **Physical Examination** — simulated physical findings provided upon request.
4. **Investigations** — laboratory and imaging results, revealed progressively.
5. **Results & Reflection** — final diagnosis entry, performance summary, and metacognitive checkpoint.

`figures/diagrams/ui-wireframe.png`

Figure 4.3: Wireframe of the learner-facing interface. Tabs reflect the phases of diagnostic reasoning, supporting progressive disclosure.

This phased design implements *progressive disclosure*, requiring learners to actively request information rather than passively receiving it. It mirrors the iterative nature of real diagnostic encounters.

4.2.5.3 Bias wareness and Reflection Prompts

Two features embed the debiasing strategies discussed in Chapter 2:

- **Bias Warnings** — triggered when the Bias Analyzer detects patterns of anchoring, confirmation, or premature closure. Warnings are displayed non-intrusively, e.g., *"You seem focused on a single hypothesis. Consider alternative explanations."*
- **Metacognitive Checkpoints** — at key points (e.g., before finalising a diagnosis), learners must reflect on their reasoning:

```
- What evidence supports your current diagnosis?
- What evidence contradicts it?
- What alternative diagnoses should be considered?
```

These features operationalise *cognitive forcing strategies* and *deliberate reflection*, turning abstract debiasing methods into practical, in-the-moment educational interventions.

4.2.5.4 dministrative Dashboard

The educator-facing dashboard provides case configuration and oversight tools. Key functions include:

- adding or modifying cases and bias triggers,
- managing cohorts of learners,
- reviewing aggregated analytics (bias frequency, diagnostic accuracy, reflection quality).

This supports both curriculum integration and educational research, enabling instructors to monitor not only outcomes but also *processes of reasoning*.

4.2.5.5 Educational Design Principles

The UI design is grounded in the following principles:

- **Cognitive load management** — phased workflows prevent overwhelming learners with information.
- **Immediate feedback** — bias warnings appear in real time, guiding learners without derailing immersion.
- **Scaffolded reflection** — structured prompts encourage deliberate reflection at moments of diagnostic closure.

- **Traceability** — interaction histories are logged, creating reasoning traces for personalised feedback and research.

Together, these interface elements embody the alignment between technology and pedagogy: SmartDoc is not only a functional application, but an *educational instrument* deliberately crafted to foster bias awareness, metacognition, and diagnostic expertise.

4.2.6 Case Design: Intent-Driven Simulation

The SmartDoc platform introduces a novel approach to clinical case design that embeds educational triggers and bias detection directly into the case structure itself. This intent-driven simulation model represents a significant innovation in medical education technology, moving beyond traditional linear case presentations to create dynamic, conversational learning experiences.

4.2.6.1 Case Schema rchitecture

Each SmartDoc case is defined using a structured schema that integrates **progressive disclosure**, **bias triggers**, and **reflection notes** directly into the case content. This ensures that diagnostic reasoning, bias awareness, and metacognition are not peripheral features but are embedded in the simulation itself.

A minimal case definition includes four core elements:

1. **Initial Presentation** — chief complaint and basic context.
2. **Information Blocks** — modular clinical facts (history, exam, labs, imaging, medications), progressively revealed through learner queries.
3. **Bias Triggers** — metadata that defines when and how anchoring, confirmation, or framing biases are likely to occur, along with corrective prompts.
4. **Educational Notes** — learning objectives, clinical pearls, and structured reflection questions linked to the case.

```
{
  "caseId": "mull_case",
  "caseTitle": " n elderly woman with 'heart failure'",
  "initialPresentation": {
    "chiefComplaint": "Worsening shortness of breath",
    "historyOfPresentIllness": "Elderly Spanish-speaking woman, history provided by
         son..."
  },
  "informationBlocks": [...],
  "biasTriggers": {...},
  "educationalNotes": {...}
}
```

By embedding these components into a single schema, SmartDoc avoids the limitations of scripted, branching cases. Instead, the case itself defines when information is revealed, where bias prompts are activated, and how reflection is scaffolded. This modular structure serves as the foundation for the progressive disclosure and bias-aware design principles described in the following sections.

4.2.6.2 Information Blocks and Progressive Disclosure

Information is stored in modular *information blocks*, each annotated with:

- **level** — depth of inquiry required (e.g., Level 1 = basic, Level 2 = deeper follow-up),
- **reveal policy** — when the block is disclosed (`immediate`, `escalate`, `conditional`),
- **prerequisites** — other blocks that must be revealed first,
- **isCritical** — whether the block is essential to reach the correct diagnosis.

Example block.

```
{
  "blockId": "critical_infliximab",
  "blockType": "Medications",
  "content": "Records reveal the patient has been receiving infliximab for
      rheumatoid arthritis.",
  "isCritical": true,
  "intentTriggers": ["meds_full_reconciliation_query"],
  "level": 2,
  "prerequisites": ["meds_ra_uncertainty"],
  "revealPolicy": "escalate"
}
```

In this example, the infliximab clue (key to diagnosing tuberculosis) is only revealed after learners probe medication history in detail — resisting *premature closure*. This mirrors real-world reasoning and forces learners to engage in deeper inquiry.

Progressive disclosure in practice.

- **Level 1:** basic data (vital signs, initial medication list) disclosed immediately.
- **Level 1 (uncertainty):** queries about rheumatoid arthritis drugs return only partial information, requiring persistence.
- **Level 2 (critical):** infliximab use is revealed only after escalation or prerequisite discovery.

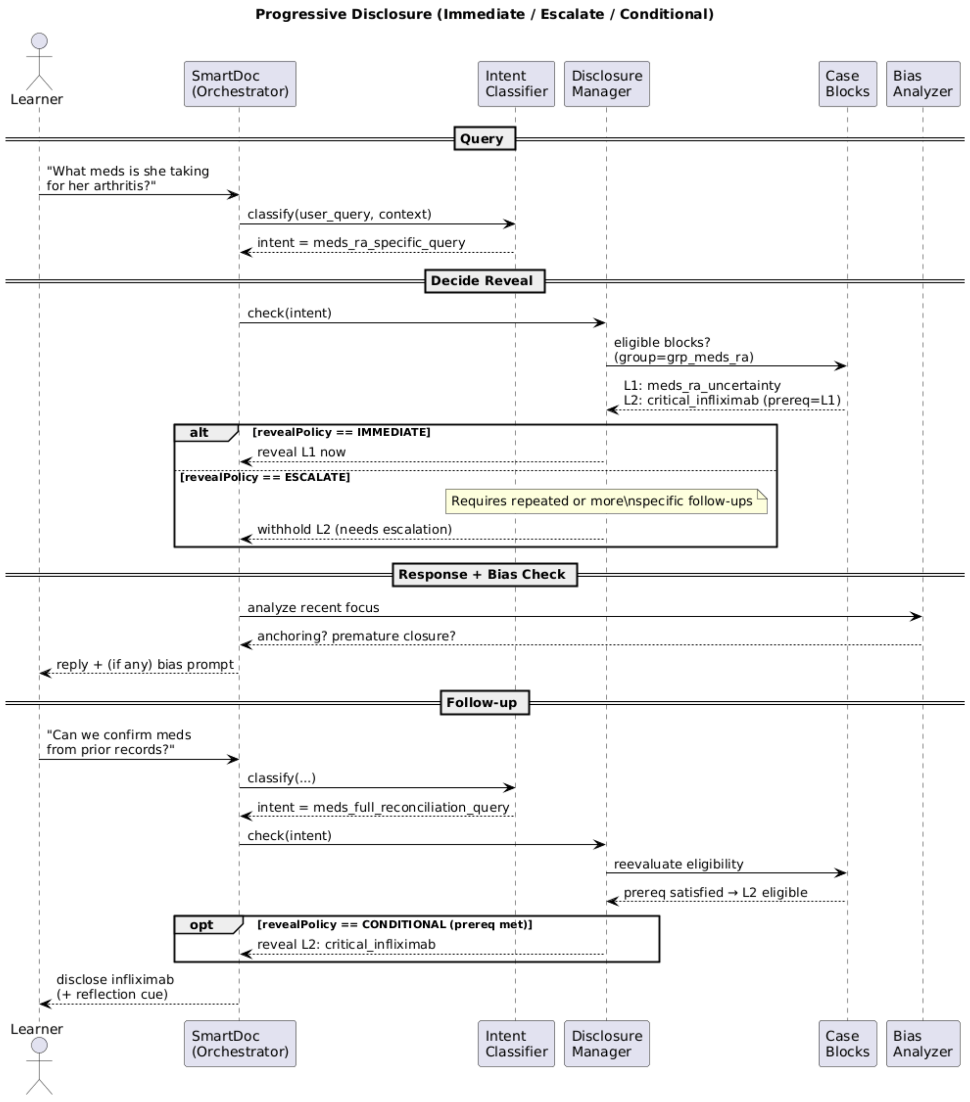
Progressive Disclosure (Immediate / Escalate / Conditional)
Learner
SmartDoc (Orchestrator)
Intent Classifier
Disclosure Manager
Case Blocks
Bias Analyzer
Query
"What meds is she taking for her arthritis?"
classify(user_query, context)
intent = meds_ra_specific_query
Decide Reveal
check(intent)
eligible blocks? (group=grp_meds_ra)
L1: meds_ra_uncertainty
L2: critical_infliximab (prereq=L1)
alt [revealPolicy == IMMEDIATE]
reveal L1 now
[revealPolicy == ESCALATE]
Requires repeated or more\nspecific follow-ups
withhold L2 (needs escalation)
Response + Bias Check
analyze recent focus
anchoring? premature closure?
reply + (if any) bias prompt
Follow-up
"Can we confirm meds from prior records?"
classify(...)
intent = meds_full_reconciliation_query
check(intent)
reevaluate eligibility
prereq satisfied → L2 eligible
opt [revealPolicy == CONDITIONAL (prereq met)]
reveal L2: critical_infliximab
disclose infliximab (+ reflection cue)
Learner
SmartDoc (Orchestrator)
Intent Classifier
Disclosure Manager
Case Blocks
Bias Analyzer

Figure 4.4: Progressive disclosure ladder: Level 1 = surface data; Level 2+ = deeper information, often bias-resistant and critical for correct diagnosis.

Educational link. This mechanism operationalises:

- **nchoring bias** — learners who stop at the BNP or first chest X-ray risk missing contradictory findings.
- **Premature closure** — learners who fail to escalate queries miss the infliximab clue.
- **Framing bias** — the initial "elderly woman with heart failure" label influences which blocks are pursued.

4.2.6.3 Embedded Bias Triggers

Bias triggers are encoded directly in case metadata, allowing real-time monitoring of learner reasoning. Example (anchoring):

```
"biasTriggers": {
  "anchoring": {
    "anchorInfoId": "imaging_cxr_preliminary",
    "contradictoryInfoId": "critical_echo",
    "description": " nchoring on preliminary CXR delays recognition of normal echo.
        "
  }
}
```

Pseudo-code for bias detection.

```
IF recent queries focus >70% on "heart/cardiac":
    Trigger(" nchoring Bias Warning")
    Prompt("What else could explain dyspnoea?")
```

This transforms abstract debiasing strategies (e.g., cognitive forcing) into concrete, in-the-moment interventions.

4.2.6.4 Educational Notes and Reflection Support

Each case embeds learning objectives, clinical pearls, and reflection prompts. Example:

```
"educationalNotes": {
  "learningPoints": ["Medication reconciliation is crucial in multimorbidity"],
  "clinicalPearls": ["TNF-alpha inhibitors like infliximab increase TB risk"],
  "bias wareness": [" nchoring on initial impressions delays correct diagnosis"]
}
```

At diagnostic closure, structured prompts require learners to reflect:

```
- What evidence supports your diagnosis?
- What evidence contradicts it?
- What alternative explanations remain plausible?
```

This operationalises **deliberate reflection** (Chapter 2) and ensures that reflection is not optional but embedded in the workflow.

4.2.6.5 pplication: The Mull Case

The prototype case adapts Mull et al. [32], in which an elderly woman with dyspnoea was repeatedly misdiagnosed with heart failure due to anchoring, confirmation, and framing biases. Smart-Doc encodes this case by:

- framing the presentation as “elderly patient with heart failure”,

- embedding anchoring triggers around chest X-ray and BNP results,
- requiring detailed medication reconciliation to reveal infliximab use.

4.2.6.6 Educational Impact and Innovation Summary

The intent-driven design provides several innovations:

1. **Conversational learning** — natural dialogue replaces scripted question trees.
2. **Embedded bias education** — bias triggers built into case metadata.
3. **Progressive disclosure** — learners earn critical clues by resisting bias-prone shortcuts.
4. **Scaffolded reflection** — structured checkpoints operationalise metacognition.
5. **Research-ready design** — interaction logs capture reasoning traces for analysis.

In this way, SmartDoc transforms case-based learning from passive fact recall to active, bias-aware clinical reasoning practice.

4.2.7 Example Workflow

TODO

4.3 Summary

This chapter described the development of SmartDoc, highlighting how technical decisions were guided by pedagogical principles and empirical evidence. The platform was designed not merely as a simulation engine, but as a bias-aware educational tool aligned with the theoretical foundations of dual-process reasoning, cognitive bias taxonomy, and deliberate reflection (Chapter 2), and addressing limitations identified in the literature on AI-powered virtual patients (Chapter 3).

4.3.1 Key Design Principles

Three principles informed the system architecture:

1. **Progressive Disclosure** — ensuring learners actively request information, reducing premature closure and supporting authentic reasoning.
2. **Bias- ware Design** — embedding triggers for anchoring, confirmation, and framing biases directly in case metadata, allowing real-time monitoring and corrective prompts.
3. **Metacognitive Scaffolding** — integrating reflection checkpoints and educational notes to foster deliberate reflection on diagnostic reasoning.

4.3.2 Core Technical Contributions

SmartDoc advances the state of AI-powered simulation through:

- an intent-driven disclosure engine that supports natural conversational learning rather than scripted branching,
- a bias analyzer that transforms cognitive forcing strategies into just-in-time prompts,
- a case schema that unifies clinical content with educational metadata, enabling both learner guidance and research analytics.

Together, these contributions address two persistent gaps: (i) the lack of simulation systems that make reasoning *visible and improvable*, and (ii) the need for reproducibility and transparency in AI-based educational tools.

4.3.3 Experimental Configuration

While SmartDoc is capable of providing *live bias feedback* during learner interactions, this feature was intentionally disabled in the evaluation study with clinical apprentices (see Chapter 5). Instead, participants completed the interview without intervention and only after submitting a final diagnosis were they presented with structured reflection prompts and LLM-generated feedback on:

- diagnostic accuracy,
- relevant learning points,
- possible biases exhibited,
- and mitigation strategies for future reasoning.

This configuration ensured that learners' reasoning processes were not artificially altered during the simulation, while still allowing SmartDoc to capture bias-prone behaviours for later analysis.

4.3.4 From Design to Evaluation

By embedding theory-driven design into a robust technical framework, SmartDoc demonstrates how cognitive psychology, bias taxonomy, and simulation principles can be operationalised in an AI-powered educational system. The platform introduces several innovations: intent-driven case design, embedded bias triggers, and structured reflection checkpoints, all aimed at making clinical reasoning both visible and improvable.

Having established the rationale and technical architecture of SmartDoc, the next chapter evaluates how these design decisions perform in practice. Chapter 5 presents the results of pilot testing with clinical interns, examining the system's usability, realism, and educational impact, and assessing whether SmartDoc can effectively raise learners' awareness of cognitive bias in diagnostic reasoning.