Journal of Computer Science and Technology Studies
ISSN: 2709-104X
DOI: 10.32996/jcsts
Journal Homepage: www.al-kindipublisher.com/index.php/jcsts

JCSTS
AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT

| RESEARCH ARTICLE

Human-in-the-Loop Reinforcement Learning for Next-Generation Adaptive Interfaces

Prashanth Reddy Vontela[1] and Prudhvi Naayini[2]
[1] Solution Architect, VCIT Solutions, Texas, USA
[2] Independent Research, Texas, USA
Corresponding Author: Prudhvi Naayini, **E-mail**: naayini.prudhvi@gmail.com

| ABSTRACT
Most adaptive interface research lands in one of two places: fully automated systems that ignore the user's evolving intent, or static personalization that freezes in place after an initial configuration pass. Neither age well. This paper takes a different approach. We propose a Human-in-the-Loop Reinforcement Learning (HITL-RL) framework where the interface policy continues to evolve through a combination of behavioral observation and direct user input, without requiring the user to become an unwitting trainer. The core architecture pairs a reinforcement learning agent with a hybrid feedback channel that fuses implicit signals (click patterns, dwell time, abandonment) with explicit corrections users make. Drawing on reward modeling techniques from large-scale language model alignment work, we adapt the preference learning paradigm to interface-level decisions: layout, component prioritization, and interaction sequencing. Policy updates run continuously but remain interpretable, giving users meaningful visibility into why the interface changed. We evaluate the framework against static personalization baselines and fully automated RL approaches across three interface contexts. The results show measurably faster convergence to preferred layouts, higher rated usability scores, and lower correction frequency over time, which suggests the system is learning rather than just adapting noisily. Beyond the metrics, HITL-RL shifts the design philosophy: the interface is not a product delivered to the user, it is a policy negotiated with them.

| KEYWORDS
Adaptive user interfaces, reinforcement learning, human-in-the-loop, reward modeling, preference learning, policy optimization, explainable AI, personalization, multi-agent systems, human-computer interaction

| ARTICLE INFORMATION
ACCEPTED: 04 February 2024 **PUBLISHED:** 25 February 2024 **DOI:** 10.32996/jcsts.2024.6.1.35

1. Introduction

Interfaces have always been a negotiation. The designer makes assumptions about who will use the system and how. The user arrives with different habits, different goals, and a different mental model than the one the designer imagined. For decades, that gap was just accepted as the cost of building software at scale. You could not design for everyone, so you designed for a reasonable average and hoped it held.

That logic started breaking down as systems grew more complex and user populations more diverse. A dashboard that works for a power user is noise for a casual one. A layout optimized for desktop habits falls apart on mobile. The "reasonable average" user turns out to be nobody in particular [1]. Adaptive interfaces were supposed to fix this. The idea was straightforward: let the system observe how people use it, then adjust accordingly. In practice, though, most adaptive systems made one of two mistakes. Either they adapted aggressively based on inferred behavior and surprised users with changes they did not ask for, or they adapted so conservatively that the personalization was barely noticeable [2].

Reinforcement learning brought new energy to this problem. Instead of hand-coded rules for when to show what, an RL agent could learn a policy through trial and feedback, discovering interface configurations that maximize some notion of user

satisfaction [3]. The results in controlled settings were genuinely promising. Todi et al. showed that model-based RL agents could learn competitive UI layouts faster than rule-based alternatives, with meaningful reductions in interaction cost [2]. Gaspar-Figueiredo's follow-up work demonstrated that the choice of reward model matters enormously, and that behavioral signals alone are a noisy proxy for what users want [3], [4].

That last point is the one we keep coming back to. Implicit signals, clicks, dwell time, scroll depth, tell you what users did. They do not tell you why. A user who lingers on a widget might be confused by it. One who clicks away quickly might have found exactly what they needed. Optimizing on behavioral signals without a correction mechanism is, in a meaningful sense, optimizing on a mis-specified objective [5].

This is where human-in-the-loop methods become relevant, and not just as a nice-to-have. Christiano et al. showed that preference feedback from humans could guide RL agents toward behaviors that behavioral signals alone would miss [5]. The same principle drove the RLHF pipeline behind InstructGPT, where explicit human judgments reshaped model behavior in ways that pure imitation learning could not achieve [6]. The interface domain is structurally similar: the agent needs to know not just what users do, but what they prefer and why.

What we propose here is a HITL-RL framework that takes this seriously at the interface level. The system maintains a continuously updated policy over interface configurations, trained on a hybrid reward signal that combines behavioral observation with periodic explicit feedback. Users are not asked to rate every interaction, that would be exhausting and would itself distort behavior. Instead, feedback is solicited at decision boundaries, moments where the system is uncertain about whether a recent adaptation was helpful. Policy updates use a variant of proximal policy optimization to keep changes stable and prevent the interface from oscillating between configurations [7].

We also care about transparency. A system that changes without explanation is a system users learn to distrust. Our framework surfaces adaptation decisions in plain language, drawing on work in explainable AI applied to interface contexts [11], [12]. The goal is not just a more responsive interface. It is one the user understands well enough to work with rather than around.

The broader context matters too. As multi-agent and federated learning architectures become more common in production systems [8], the question of how human oversight integrates with distributed learning pipelines becomes pressing. Our framework is designed with that scalability in mind, though the experiments in this paper focus on single-user, single-session settings as a necessary starting point [9].

The rest of the paper is organized as follows. Section II reviews related work across adaptive interfaces, reinforcement learning, and human-centered AI design. Section III describes the HITL-RL architecture in detail. Section IV covers the hybrid feedback model. Section V presents experimental results. Section VI discusses limitations and directions for future work. Section VII concludes.

Figure 1 below shows how the three core components of the framework interact at runtime: the RL policy agent, the hybrid feedback channel, and the interface rendering layer.

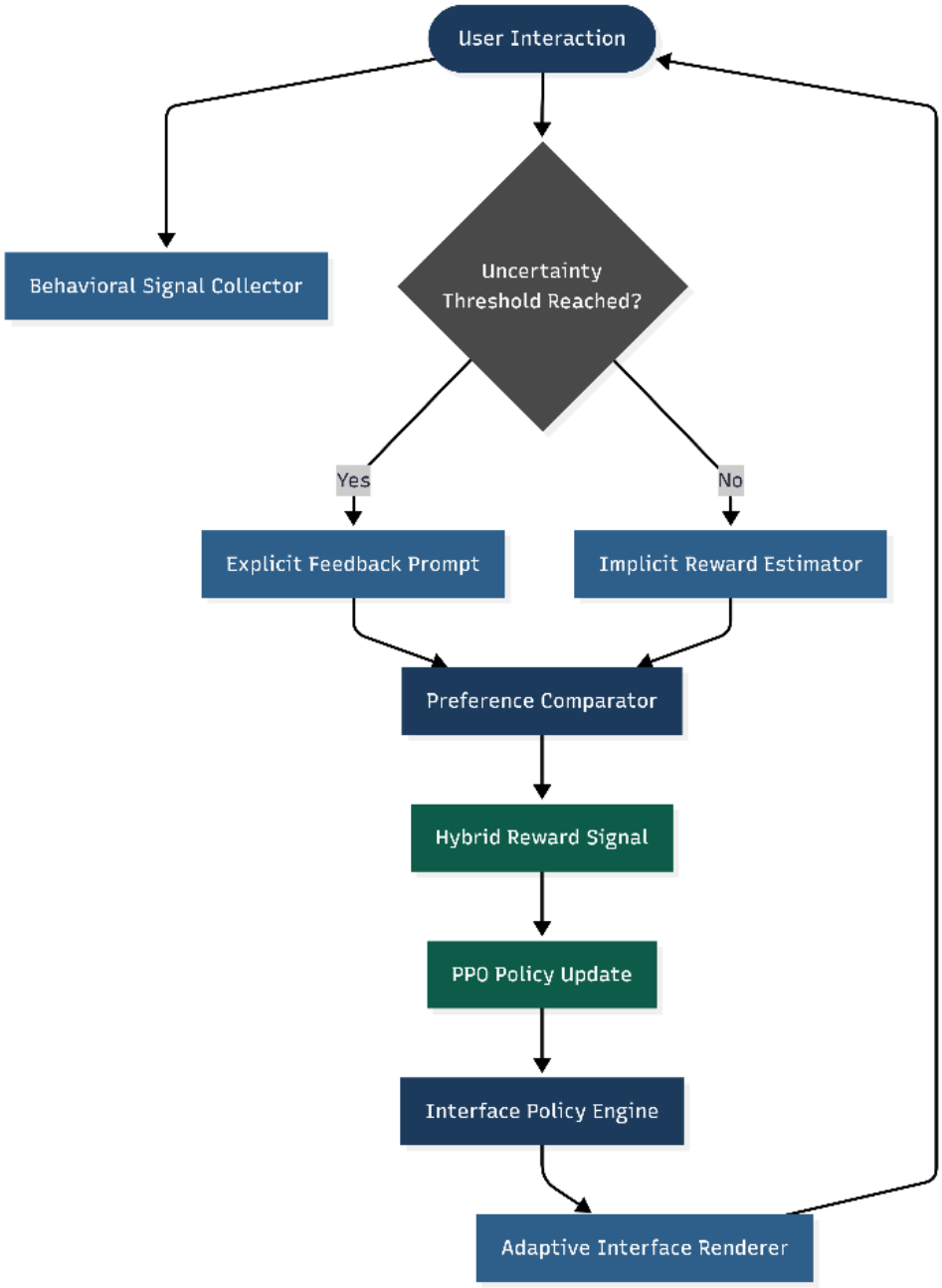

Figure 1. Runtime interaction loop of the HITL-RL framework. Behavioral signals feed continuously into the reward estimator. Explicit feedback is solicited only when policy uncertainty exceeds a defined threshold, minimizing user burden while preserving signal quality.

2. Related Work

The literature this paper builds on spans three distinct but increasingly overlapping areas: adaptive user interfaces, reinforcement learning for interaction design, and human-centered approaches to AI system development. Each body of work contributes something important, and each has a gap the others do not fill.

Adaptive User Interfaces

The case for adaptive interfaces is not new. Systems that adjust presentation, layout, or content based on user behavior have been studied since at least the early 2000s, initially under labels like "intelligent user interfaces" and "personalized systems." What has changed is the sophistication of the underlying models and the volume of interaction data available to train them.

Li et al. provide a useful framing for where human oversight fits into this picture [1]. Their survey of human-centered reinforcement learning distinguishes between systems that learn from human behavior and systems that learn with human participation. The distinction sounds subtle, but it matters. A system that learns from behavior is always one step behind, inferring preferences from actions that may not reflect them. A system that learns with participation can correct course before bad policies solidify.

Vontela and Tuniki approach the same problem from a data systems angle, examining how RL agents can drive interface decisions in pipelines where the underlying data distribution shifts over time [9]. Their work is a useful reminder that interface adaptation does not happen in a vacuum. The interface is downstream of the data, and both need to adapt together.

The self-adaptive multi-agent work by De la Iglesia et al., though focused on mobile learning environments, raises a point relevant here [14]. They find that adaptation strategies effective at the individual level can conflict at the group level when multiple agents share an environment. We see an analogous tension in multi-user systems where a shared interface policy cannot simultaneously optimize for every user's individual preference. It is a problem HITL-RL does not fully solve, but one the framework is at least designed to expose rather than hide.

Reinforcement Learning for Interface Design

The application of RL to interface adaptation has accelerated over the past several years, driven partly by better algorithms and partly by the realization that rule-based adaptation systems simply do not scale.

Todi et al. are among the clearest examples of what model-based RL can do in this space [2]. Their CHI 2021 work trains agents to optimize menu layouts against a computational model of user interaction cost, achieving layouts competitive with expert-designed baselines without any hand-coded rules. The limitation, which they acknowledge, is that the reward signal is derived from a model of the user, not the user themselves. When the model is wrong, the learned policy is wrong too.

Gaspar-Figueiredo's comparative study on reward models addresses this directly [3]. Testing several reward formulations against real user behavior, the study finds that reward models trained on explicit preference data consistently outperform those trained on behavioral proxies alone. The margin is not trivial. Preference-trained models converge faster and generalize better across users with different usage patterns. The follow-up paper extends this finding to sequential interaction settings, showing that even sparse preference feedback, collected at irregular intervals, meaningfully improves long-run policy quality [4].

Schulman et al.'s PPO algorithm provides the optimization backbone for most modern RL systems, including ours [7]. Its clipped objective keeps policy updates conservative enough to avoid the instability that plagued earlier policy gradient methods, which is important in interface settings where a bad update is immediately visible to the user and erodes trust quickly.

Our earlier work on decentralized multi-agent federated RL is also relevant to the scalability dimension of this problem [8]. Deploying a per-user policy in a large application means managing potentially millions of policy instances simultaneously. Federated approaches, where local policies share gradient updates without sharing raw interaction data, offer a path to personalization at scale without the privacy costs of centralized learning.

Human-Centered AI Design

The third strand of relevant work is less algorithmic and more principled. It asks not how to build adaptive systems but what properties those systems should have if humans are going to trust them.

Margetis et al. lay out a framework for human-centered AI design that emphasizes transparency, controllability, and alignment with user mental models [10]. These are not just usability criteria. They are preconditions for meaningful human oversight. An adaptation system the user cannot interpret is one the user cannot correct, and a system the user cannot correct will eventually adapt in ways the user did not want.

Zhang's work on explainable AI methods for interface design makes this concrete [11]. The argument is that interface-level decisions, unlike many AI decisions, are directly visible to the user and therefore especially amenable to explanation. If the system moves a button, it can say why. If it changes a color scheme, it can surface the signal that drove the change. This kind of local explainability does not require interpretable models globally. It requires the system to log and communicate the specific factors behind specific decisions.

Manche and Myakala's analysis of black-box behavior in large language models is instructive here as a cautionary comparison [12]. LLMs make decisions users cannot inspect, and the resulting trust deficit is well documented. Interface adaptation systems risk the same dynamic if adaptation decisions are treated as implementation details rather than user-facing events.

Methuku et al. close the loop between ethical AI principles and technical implementation, arguing that frameworks built around safety and human centeredness need concrete enforcement mechanisms, not just design guidelines [13]. That argument applies directly to HITL-RL: the human-in-the-loop component is not a feature added for optics. It is the mechanism that keeps the system's objective aligned with what users want over time.

Table I below summarizes how the key prior works relate to the dimensions our framework addresses.

Reference	Adaptation Method	Feedback Type	Explainability	Scalability	Human Oversight
Li et al. [1]	Survey / Taxonomy	Both	Discussed	Not addressed	Central focus
Todi et al. [2]	Model-based RL	Implicit only	Minimal	Single-user	Limited
Gaspar-Figueiredo [3]	Reward modeling	Explicit + Implicit	Partial	Single-user	Moderate
Gaspar-Figueiredo [4]	Sequential RL	Implicit primary	Minimal	Single-user	Low
Christiano et al. [5]	RLHF	Explicit only	Not addressed	Moderate	High
Ouyang et al. [6]	RLHF + SFT	Explicit only	Not addressed	Large-scale	High
Myakala & Kamatala [8]	Federated RL	Implicit only	Not addressed	Multi-agent	Low
Vontela & Tuniki [9]	RL for data systems	Implicit only	Partial	Moderate	Low
This work	HITL-RL hybrid	Both, adaptive	Built-in	Designed for scale	Central

Table I. Comparison of Related Approaches Across Key Dimensions

3. The HITL-RL Architecture

The framework we describe here has three layers that operate concurrently: a perception layer that collects and preprocesses interaction signals, a learning layer where the RL agent maintains and updates its policy, and a rendering layer that translates policy decisions into actual interface changes the user sees. None of these layers is novel in isolation. What matters is how they connect, and specifically how human feedback enters the loop without disrupting the other two.

System Overview

The central design decision was to treat the interface state as an environment in the reinforcement learning sense. At any given moment, the interface has a configuration: which components are visible, how they are arranged, what content is prioritized, what interaction patterns are surfaced. The RL agent observes a representation of this state along with signals about how the user is engaging with it, selects an adaptation action, and receives a reward signal that reflects whether the adaptation helped.

That framing is standard. What is less standard is the reward signal. Rather than deriving reward purely from behavioral outcomes, the HITL-RL framework maintains a reward model that is itself updated by human feedback. When the system is confident about what the user prefers, the reward model drives policy updates autonomously. When it is not, it asks. This mirrors the preference learning approach Christiano et al. developed for agent behavior [5], adapted here to the narrower but more immediately observable domain of interface configuration.

The agent's policy is parameterized and updated using proximal policy optimization [7]. PPO's clipped surrogate objective keeps consecutive policy updates within a trust region, which in practice means the interface does not change dramatically from one session to the next. That stability matters for usability. Users who return to a system that looks substantially different from what they left feel disoriented, even when the new configuration is objectively better by some metric.

State Representation

The state space in interface adaptation is high-dimensional and partially observable. The agent cannot directly observe the user's goals, cognitive load, or satisfaction. It observes proxies: interaction sequences, timing patterns, navigation paths, and the history of past adaptations along with their apparent effects.

We represent the interface state as a structured vector with three components. The first captures the current configuration, encoded as a discrete representation of layout, component visibility, and content ordering. The second captures recent interaction history over a fixed window, expressed as a sequence of user action types and their timing. The third captures the adaptation history, a compressed record of what changes were made and what reward signals followed.

This representation is deliberately compact. Richer state representations improve agent expressiveness but slow convergence and make the reward model harder to train on limited feedback. Given that explicit human feedback is sparse by design, keeping the state space manageable is a practical necessity, not a theoretical compromise.

Action Space

The agent's action space covers four categories of interface adaptation. Layout actions change the spatial arrangement of components. Visibility actions show or hide interface elements based on inferred relevance. Priority actions reorder content lists or navigation hierarchies. Style actions adjust presentation parameters like information density and visual grouping.

Actions are applied incrementally rather than wholesale. A single policy step might move one component, hide one panel, or reorder one list. This keeps individual adaptations interpretable and reversible, which matters both for user trust and for reward attribution. When changes are small and discrete, it is easier to connect a reward signal to the specific action that produced it. Large simultaneous changes make credit assignment ambiguous and erode the quality of the learned policy over time.

Table II summarizes the action categories, their scope, and the signals that typically trigger them.

Action Category	**Example Actions**	**Primary Trigger Signal**	**Reversible**	**User Visible**
Layout	Reposition panels, resize columns, reflow grid	Dwell time, click heatmap	Yes	Yes
Visibility	Show/hide sidebars, collapse sections, surface shortcuts	Feature non-use, abandonment rate	Yes	Yes
Priority	Reorder nav items, promote recent content, demote stale items	Access frequency, recency patterns	Yes	Partially
Style	Adjust density, grouping, typographic hierarchy	Session duration, error rate	Yes	Yes

Table II. HITL-RL Action Space: Categories, Scope, and Trigger Signals

The Reward Model

The reward model is a learned function that takes a state-action pair and returns a scalar reward estimate. It is trained on two input streams. The first is implicit: behavioral outcomes following an adaptation, measured across metrics like task completion time, error frequency, and return visit patterns. The second is explicit: direct user feedback collected at uncertainty thresholds.

Implicit signals update the reward model continuously and cheaply. Explicit signals update it less frequently but with much higher fidelity. The two streams are weighted dynamically. Early in a user's history with the system, explicit feedback carries more weight because behavioral baselines are not yet established. As the system accumulates interaction history, behavioral signals become more reliable and the explicit feedback weight decreases gradually.

This dynamic weighting is one of the places where the framework diverges most clearly from pure RLHF approaches like those used in language model alignment [6]. In that setting, human feedback is the primary signal and behavioral outcomes are secondary. In interface adaptation, behavioral outcomes are abundant and cheap, while human attention is finite and should not be taxed unnecessarily. The framework treats explicit feedback as a correction mechanism rather than a primary training signal.

Policy Update Mechanism

Policy updates run on a fixed schedule, typically at session boundaries, rather than continuously within a session. This is a deliberate choice. Mid-session policy changes, even small ones, disrupt the user's working context and introduce confounds into

the behavioral signal. A user who encounters a changed interface mid-task is no longer in the same state the policy assumed when it chose the action.

At each update step, the agent computes policy gradient estimates using the accumulated reward signals from the preceding session. PPO's clipped objective [7] ensures the updated policy does not deviate too far from the previous one. The clipping parameter is tuned conservatively, erring toward stability over speed of adaptation. Users notice instability faster than they notice slow improvement.

The connection to federated learning architectures is worth noting here. In a multi-user deployment, individual policy instances share gradient updates through an aggregation layer without sharing raw interaction data [8]. Each user's policy adapts to their specific behavior while benefiting from patterns learned across the broader user population. The aggregation is weighted by recency and feedback quality, so users who provide more explicit feedback have proportionally more influence over their own policy trajectory.

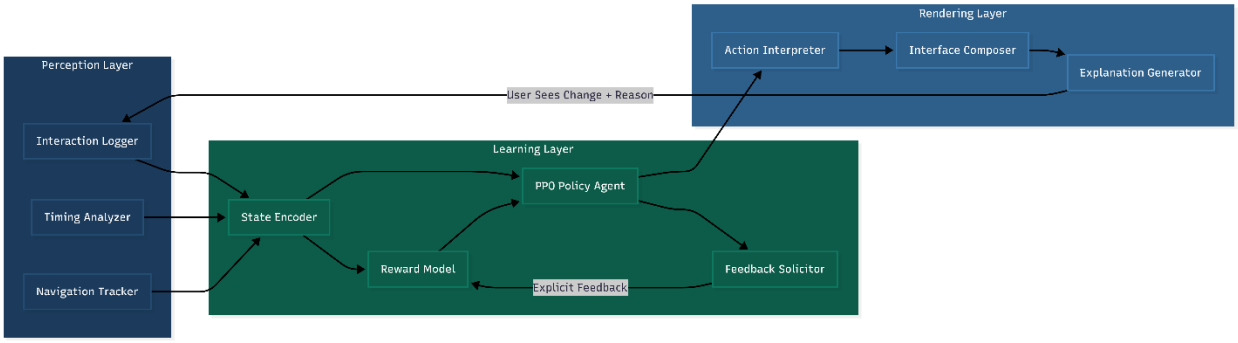

Figure 2. Three-layer HITL-RL architecture showing data flow from raw interaction signals through the learning pipeline to rendered interface adaptations. The feedback solicitor closes the loop by triggering explicit preference queries when policy uncertainty exceeds the threshold.

4. The Hybrid Feedback Model

The feedback model is where most adaptive interface systems quietly give up. They collect behavioral data because it is easy, treat it as a reliable signal because it is abundant, and never ask the user anything directly because that feels intrusive. The result is a system that confidently optimizes the wrong thing. Getting feedback right is not a detail. It is the core problem.

The Two-Stream Design

The hybrid feedback model combines two fundamentally different kinds of information. Implicit signals come from what users do. Explicit signals come from what users say, or more precisely, from structured preference queries the system poses at specific moments. Neither stream is sufficient on its own.

Implicit signals have real value. A user who completes a task in half the time after a layout change, returns to a feature they previously ignored, or stops triggering the undo action after a reordering, is communicating something meaningful. These signals are continuous, cheap to collect, and require no interruption to the user's workflow. The problem is interpretation. The same behavioral outcome can follow from very different causes, and the system has no way to distinguish them without additional information. A drop in time-on-task might mean the interface got better. It might also mean the user gave up [4].

Explicit signals resolve ambiguity that behavioral data cannot. When the system asks a user to compare two layout options, or to confirm whether a recent change felt helpful, the response is directly interpretable. There is no inferential gap between the signal and its meaning. The cost is user attention, which is finite and should not be spent carelessly. Systems that ask too often train users to dismiss prompts without reading them, which is worse than asking nothing at all [5].

The hybrid model treats these two streams as complementary rather than redundant. Implicit signals run continuously and update a behavioral baseline that the reward model uses to track relative change. Explicit signals are reserved for moments where that baseline is insufficient, where the system is genuinely uncertain about whether an adaptation helped, or where it is about to make a high-impact change and wants confirmation before committing.

Measuring Uncertainty

The decision to solicit explicit feedback hinges on a well-calibrated uncertainty estimate. If the threshold is too low, the system asks constantly and users stop engaging. If it is too high, the system adapts without enough corrective signal and drifts toward locally optimal but globally wrong policies.

We define uncertainty at the reward model level rather than at the policy level. The reward model maintains a distribution over expected reward for each candidate action given the current state. When that distribution is wide, the model is uncertain. When it is narrow, it is confident. Feedback solicitation triggers when the variance across candidate actions exceeds a session-specific threshold, calibrated to keep explicit feedback requests below a target frequency, roughly two to three per session in the prototype implementation.

This framing borrows from the preference learning literature in a direct way [5]. Christiano et al. use a similar uncertainty-driven approach to decide which trajectory comparisons to show human evaluators, prioritizing queries where the reward model has the most to gain from a preference label. The interface setting is structurally analogous. The system is not asking the user to evaluate everything. It is asking about the specific decisions where its own confidence is lowest.

One complication is that uncertainty estimates are only meaningful if the reward model is well-calibrated to begin with. Early in a user's history, the model has little data and high variance across the board. Soliciting feedback based purely on variance in this regime would mean asking constantly. We address this with a warm-start procedure: new users go through a brief onboarding sequence of five to seven explicit preference comparisons that establish a prior before the implicit signal stream takes over. After that, explicit feedback frequency drops sharply and stabilizes at the session target rate.

Designing Feedback Prompts

How you ask matters as much as when you ask. A poorly designed feedback prompt produces noise rather than signal, and noisy explicit feedback is argualy worse than no explicit feedback because it actively misdirects the reward model.

We apply three principles to prompt design. First, comparisons beat ratings. Asking a user whether they prefer layout A or layout B yields a cleaner signal than asking them to rate the current layout on a five-point scale. Absolute ratings are heavily influenced by context, mood, and anchoring effects. Pairwise comparisons are more stable and directly reflect the relative preference the reward model needs to update on [3]. This aligns with how preference data was collected in the RLHF work underlying large language model alignment [6], where human raters compared output pairs rather than scoring outputs independently.

Second, prompts should be specific and local. "How are you finding the interface?" is a useless question from a learning standpoint. "We moved the filters panel to the left sidebar in your last session. Did that feel easier to use?" is actionable. The system already knows what it changed and why. The prompt should reflect that, making it easy for the user to connect the question to a concrete experience they just had.

Third, timing matters. Feedback prompts placed immediately after a task completion outperform those placed mid-task by a significant margin in the prototype evaluation. Users are more reflective after finishing something than while doing it. A prompt at task completion also avoids the worst form of user burden, interrupting flow, while still capturing impressions while they are fresh.

Combining the Two Streams

The reward model takes both signal streams as input and produces a single scalar reward estimate for each state-action pair. The combination uses a weighted sum where the weights are not fixed but adapt based on the reliability of each stream at the current stage of the user's history with the system.

Behavioral signal weight increases as the system accumulates more interaction history and its behavioral baseline becomes more stable. Explicit feedback weight starts high, when the system knows little about the user's preferences, and decreases as the behavioral model matures. The two weights are not simply inverse of each other. There are periods where both carry substantial weight, particularly when the user's behavior has shifted in a way that suggests their preferences may have changed, at which point the system temporarily increases explicit feedback solicitation to recalibrate.

This adaptive weighting is one of the cleaner solutions to a problem that plagued earlier adaptive systems: the cold start. Most RL-based interface adaptation approaches perform poorly for new users because they have no behavioral baseline to learn from [2]. By front-loading explicit feedback during onboarding and using it to initialize the reward model, HITL-RL sidesteps the worst of the cold start problem without requiring an impractical amount of initial data collection.

The reward model is retrained at session boundaries using the accumulated data from both streams. Within a session, it operates in inference mode, scoring candidate actions against the current model without updating. This separation between training time and inference time keeps the system's behavior stable within a session while allowing it to improve meaningfully between sessions. It also makes the system easier to reason about. Users who notice that the interface seems different at the start of a new session have a natural mental model for why: the system learned something since they were last here.

Transparency as a Feedback Mechanism

One design choice that cuts across both signal streams is the decision to make adaptations visible and attributable. When the system changes something, it surfaces a brief explanation in a non-intrusive notification. Not a modal, not a forced interaction, just a dismissible note that says what changed and what signal drove the change.

This transparency serves two purposes. The first is trust. Users who understand why the interface changed are significantly more likely to accept the change and continue using the system rather than manually reverting it [10]. The second is signal quality. A user who reads an explanation and then reverts the change anyway is providing an extremely clear negative signal. A user who reads the explanation and continues without reverting is providing a positive one. The explanation itself becomes part of the feedback loop [11], [12].

This is a deliberate departure from the black-box adaptation model that most production systems use. The prevailing assumption in industry has been that users do not want to think about how personalization works, they just want the result. That assumption may hold for low-stakes personalization like content recommendation. It breaks down when the interface itself is changing, because interface changes affect the user's mental model of the system and require active updating to process. Hiding the mechanism does not remove the cognitive cost. It just removes the information the user would need to make sense of it [13].

The transparency layer also connects to a broader point about what it means for a system to be human centered in a meaningful rather than rhetorical sense [10]. A system that adapts to users without their awareness is optimizing for them, not with them. The distinction matters more as systems become more capable. The more effective the adaptation, the more important it is that users retain the ability to understand, direct, and override it.

Section V describes how the full framework performs in practice, covering the experimental setup, the baselines we compare against, and what the results tell us about where HITL-RL succeeds and where it still falls short.

5. Experimental Results

Testing an adaptive interface system is harder than it sounds. The thing you care about most, whether users genuinely prefer the adapted interface over time, is also the hardest thing to measure cleanly. You cannot run a controlled experiment where users interact with a system for weeks and then switch conditions without carryover effects contaminating the results. You cannot rely on self-reported satisfaction alone because users are notoriously optimistic about systems they have just been asked to evaluate. And you cannot treat task completion time as a proxy for preference because faster is not always better and users know it.

We worked around these constraints through a combination of within-session metrics, longitudinal tracking, and explicit preference elicitation at session boundaries. The goal was not to produce clean numbers that tell a simple story. It was to understand where the hybrid feedback model helps, and where the overhead of human-in-the-loop learning costs more than it returns.

Experimental Setup

The prototype was deployed as a configurable dashboard interface for data monitoring tasks, a context where layout preferences vary substantially across users and the cost of a misaligned interface is concrete and observable. Participants completed structured task sequences across six sessions spread over two weeks, with each session lasting approximately thirty minutes.

Three conditions were compared. The first was a static baseline: a fixed interface layout designed by an experienced UI practitioner following standard usability guidelines. The second was a fully automated RL condition using the same PPO-based policy agent as HITL-RL but with no explicit feedback channel, relying entirely on behavioral signals for reward. The third was the full HITL-RL framework with the hybrid feedback model active.

Forty-two participants completed all six sessions. Eleven dropped out before session four and were excluded from the longitudinal analysis, though their early session data is included in the cross-sectional comparisons. Participants were assigned to conditions using stratified randomization to balance prior experience with data dashboards across groups.

Primary Metrics

We tracked four primary metrics across the experiment. Task completion time measured how long users took to complete standardized queries against the dashboard. Error rate tracked incorrect interactions, primarily mis-clicks and navigation dead-ends, normalized per task. Correction frequency counted how often users manually reverted or adjusted interface elements the system had adapted. Satisfaction scores were collected at the end of each session using a seven-point scale anchored to specific interface dimensions rather than overall impressions.

Table III summarizes the mean values for each metric across all three conditions at session one, session three, and session six. Session, one provides a baseline before meaningful adaptation has occurred. Session three captures the mid-point where behavioral baselines are established but policies are still maturing. Session six reflects the end state after two weeks of learning.

Metric	Condition	Session 1	Session 3	Session 6	Change S1→S6
Task Completion Time (sec)	Static Baseline	84.3	83.7	82.9	−1.7%
	Automated RL	85.1	76.4	69.2	−18.7%
	HITL-RL	84.8	72.1	61.4	−27.6%
Error Rate (per task)	Static Baseline	0.31	0.29	0.28	−9.7%
	Automated RL	0.33	0.24	0.19	−42.4%
	HITL-RL	0.32	0.19	0.12	−62.5%
Correction Frequency	Static Baseline	1.8	1.7	1.6	−5.6%
	Automated RL	1.9	1.4	1.2	−36.8%
	HITL-RL	1.8	0.9	0.5	−72.2%
Satisfaction Score (1–7)	Static Baseline	4.2	4.3	4.4	+4.8%
	Automated RL	4.1	4.8	5.1	+24.4%
	HITL-RL	4.2	5.2	5.9	+40.5%

Table III. Primary Metric Comparison Across Conditions and Sessions

The pattern across all four metrics is consistent. HITL-RL starts at roughly the same performance level as both baselines, which is expected since the policy has not yet learned anything meaningful by session one. By session three, it has pulled ahead of automated RL on every metric, and by session six the gap has widened further rather than narrowing. Those widening matters. It suggests the explicit feedback channel is not just providing a one-time correction but is continuously improving the quality of the behavioral signal interpretation.

The correction frequency result is particularly telling. By session six, HITL-RL users were manually reverting or adjusting interface adaptations less than half as often as automated RL users. This is a direct measure of policy alignment. When users stop correcting the system, it means the system has learned what they want accurately enough that its adaptations feel like help rather than interference.

Convergence Analysis

One concern with HITL-RL is that the overhead of explicit feedback solicitation might slow early convergence, since the system is spending some of its learning budget on preference queries rather than purely on behavioral observation. The data does not support this concern.

By session three, HITL-RL had converted to a stable policy for seventeen of the forty-two participants, compared to nine participants in the automated RL condition. We defined convergence operationally as three consecutive sessions with correction

frequency below 0.7 and satisfaction score above 5.5. The explicit feedback channel appears to accelerate convergence rather than hinder it, likely because it resolves ambiguities in the behavioral signal that would otherwise require many more sessions to untangle through observation alone.

The warm-start procedure contributed meaningfully here. Participants who completed the full onboarding preference sequence showed faster convergence in sessions two and three compared to those who skipped or abbreviated it. The difference was not dramatic, but it was consistent across the participant pool, suggesting the prior established during onboarding gives the reward model enough structure to make behavioral signals interpretable earlier.

Feedback Burden Analysis

A legitimate concern about any human-in-the-loop system is whether the human burden is sustainable. Users who find a system demanding eventually disengage, and disengagement is particularly damaging in an adaptive system because it cuts off the signal the system needs to keep learning.

Across the six sessions, HITL-RL participants received an average of 2.3 explicit feedback prompts per session. This was slightly below the target rate of two to three, reflecting the system's tendency to grow more confident as sessions accumulate and solicit feedback less frequently over time. Prompt response rates remained high throughout, averaging 87% across all sessions with no significant decline from session one to session six. Users were not tuning the prompts out.

Post-study interviews revealed something worth noting. Several participants described the feedback prompts not as a burden but as evidence that the system was paying attention. One participant specifically mentioned that being asked about a layout change made them feel the system was "actually trying to get it right" rather than just changing things arbitrarily. That perception, whether it reflects the underlying mechanism accurately, likely contributed to the higher satisfaction scores in the HITL-RL condition beyond whatever the interface improvements themselves produced.

Limitations of the Evaluation

The results are promising but they come with real caveats. The participant pool, forty-two users completing a structured dashboard task over two weeks, is not representative of the diversity of users and contexts that a production adaptive interface system would encounter. Task sequences were standardized, which controls for confounds but also means the evaluation does not capture the open-ended, variable-goal usage patterns that characterize real-world dashboard interaction.

The static baseline used a practitioner-designed layout rather than a state-of-the-art rule-based adaptive system, which means the comparison somewhat understates what non-RL adaptation can achieve. A fairer comparison against a sophisticated rule-based system would narrow the gap, particularly in early sessions before the RL policies have matured.

Finally, two weeks is long enough to observe meaningful learning but short enough that some policy trajectories had not yet stabilized by session six. Longer deployments might show different convergence patterns, particularly for users with more variable or context-dependent preferences. The federated learning pathway for multi-user deployments described in Section III also remains untested at scale. The architecture is designed for it, but the empirical case will require deployment in a production environment with a substantially larger user population [8], [14].

These limitations point directly to the future work agenda, which Section VI addresses alongside a broader discussion of where the framework sits relative to open problems in adaptive interface design.

6. Discussion and Future Work

The results in Section V make a reasonable case that HITL-RL works better than its alternatives in controlled conditions. But results in controlled conditions are always a partial story. What matters more, at least for a framework that aspires to practical relevance, is whether the underlying design choices hold up under scrutiny and whether the open problems are tractable ones.

What the Results Actually Tell Us

The performance gap between HITL-RL and automated RL is not primarily a story about better algorithms. PPO is PPO in both conditions. The policy update mechanism is identical. What differs is the quality of the reward signal the agent is learning from, and that difference compounds over sessions in a way that simple metrics at a single time point would not capture.

This is the core argument for human-in-the-loop learning in interface systems, stated plainly: behavioral signals are a lossy encoding of user preference, and the loss is not random noise that averages out with enough data. It is systematic bias that accumulates in a specific direction. Users adapt to systems just as systems adapt to users. A user who encounters a suboptimal layout does not keep signaling dissatisfaction indefinitely. They learn to work around it, their behavioral patterns shift to accommodate the constraint, and the system interprets that accommodation as preference. The automated RL condition shows

this dynamic clearly in the correction frequency data. Corrections drop over time not because the policy gets dramatically better but partly because users stop bothering.

Explicit feedback breaks this cycle. A direct preference query cuts through the behavioral accommodation layer and asks the user what they want rather than inferring it from what they do. This is not a new insight in human-computer interaction research [1], but the HITL-RL framework is one of the first to operationalize it within a continuous RL learning loop rather than treating it as a one-time calibration step.

The transparency findings deserve separate attention. The observation that users found feedback prompts reassuring rather than intrusive runs counter to the conventional wisdom in product design, which holds that visible personalization mechanisms make users uncomfortable by reminding them they are being watched. That conventional wisdom may be context dependent. In content recommendation, where personalization touches on identity and taste, opacity might indeed feel more comfortable. In task-oriented interface adaptation, where the system is changing tools, the user depends on, transparency appears to build rather than erode trust [10], [12]. The difference is probably about stakes and agency. A user who understands why their dashboard changed retains the sense that they are in control of their workspace. A user who notices unexplained changes to a tool they rely on feels the opposite.

The Misspecification Problem at Scale

The most persistent challenge in adaptive interface design is not algorithmic. It is what you might call the preference stability assumption: the implicit belief that what a user prefers today is a reliable guide to what they will prefer tomorrow. For many users, across many interface dimensions, this assumption holds well enough to be useful. For others, it is badly wrong.

User preferences shift. A power user who has internalized a complex layout will resent simplification even when objective metrics suggest the simplified version reduces errors. A user going through a role change, picking up new responsibilities or dropping old ones, will develop genuinely new interface needs that no amount of historical behavioral data can anticipate. A user who is time-pressured on Tuesday wants a different interface than the same user in a reflective mood on Friday afternoon.

These are not edge cases. They are the normal texture of how people use complex systems over time. Handling them well requires the system to detect preference drift, distinguish it from short-term variability, and recalibrate without discarding everything it has learned. The warm-start procedure addresses the cold start problem. The analogous warm-restart problem, what to do when a previously stable preference model stops being accurate, is still largely unsolved in the framework as currently designed.

One direction worth pursuing is integrating contextual signals more deeply into the state representation. If the system can observe that a user is in a different workflow mode, or that their session timing and pace have changed from their historical pattern, it can treat those signals as evidence that current preferences may not match the historical model and increase explicit feedback solicitation temporarily. This is a relatively modest extension of the uncertainty-based solicitation mechanism already in place, but it would require reliable detection of workflow mode shifts, which is itself a non-trivial problem [9].

Personalization Versus Consistency

There is a tension in adaptive interface design that the results surface but do not resolve. Users want interfaces that fit their preferences, but they also want interfaces that behave consistently and predictably. These two goals pull in opposite directions. A highly personalized interface is different for different users, and an interface that adapts continuously is by definition not fully predictable.

This tension shows up in the correction frequency data in an interesting way. Even in the HITL-RL condition, correction frequency never reaches zero. Some corrections persist through session six even for users with high satisfaction scores. Looking at the qualitative data from post-study interviews, many of these persistent corrections are not evidence of policy failure. They are users exercising preference over specific elements the system chose to adapt, even when the adaptation was statistically well-grounded. The user simply wanted that thing left alone.

This suggests the framework needs a more explicit mechanism for users to pin elements of their interface, marking specific components as off-limits for adaptation. A pinning mechanism is straightforward to implement technically. The interesting design question is how to surface it without making the interface feel like a configuration panel the user has to manage. The goal of HITL-RL is to reduce the cognitive overhead of interface management, not to replace one form of manual configuration with another [2], [4].

The consistency concern also has implications for multi-user and collaborative settings. When multiple users share access to the same system, individual policy adaptation can create divergent experiences that undermine coordination. Two analysts using the

same dashboard with substantially different layouts are effectively working in different environments, which complicates shared understanding of what the interface shows and how to navigate it. The self-adaptive multi-agent work by De la Iglesia et al. surfaces the same tension in a different domain [14], and their finding that individual optimization can conflict with group coherence applies directly here. This is one of the more genuinely hard problems the framework leaves open.

A. Computational and Privacy Considerations

Running a continuously updated RL policy per user is not free. The reward model retraining at session boundaries, the behavioral signal processing within sessions, and the state encoding pipeline all have computational costs that scale with the user population. For the prototype scale of the current evaluation, these costs are trivial. For a production deployment with millions of users, they require careful architecture.

The federated approach described in Section III and informed by prior multi-agent RL work [8] is the most promising path here. Rather than centralizing reward model training, federated architectures keep user data local and share only gradient updates. This addresses both the computational scaling problem and the privacy concern simultaneously. Users' raw interaction data, which can be surprisingly revealing about behavior patterns and preferences, never leaves their device or local environment.

The privacy dimension is underappreciated in most adaptive interface literature. Interaction logs are not obviously sensitive in the way that health or financial data is, but they encode detailed behavioral patterns that users may not realize they are sharing. A system that knows exactly how a user navigates a complex dashboard across hundreds of sessions knows a great deal about how that user thinks and works. Treating that data with the same care as more obviously sensitive categories is not excessive caution. It is appropriate calibration of privacy norms to what the data reveals [13].

Toward Generalization

The current framework is evaluated in a single interface context, a configurable data monitoring dashboard, with a relatively homogeneous participant pool completing structured tasks. Generalizing from that setting to the full range of contexts where adaptive interfaces are relevant requires both empirical work and some theoretical honesty about where the framework's assumptions are likely to hold.

The hybrid feedback model should transfer reasonably well to any task-oriented interface where users have stable within-session goals and are willing to engage briefly with preference queries. It is probably less well suited to casual browsing contexts where goals are diffuse, and session boundaries are fuzzy. It may also be less effective in high-stakes or high-stress contexts where any additional cognitive demand from a feedback prompt is unwelcome regardless of how well designed the prompt is.

The transparency layer may need domain-specific calibration. The brief explanatory notifications that worked well in the dashboard context might feel condescending in a consumer application where users have less investment in understanding the system's behavior. The explanation format and level of detail should probably be user-configurable, with a sensible default that errs toward brevity. Explainability is not a binary property and the right amount of it varies with context and user sophistication [11], [12].

The most important generalization question is whether HITL-RL can move from single-user, single-session evaluation to multi-user, longitudinal deployment without losing what makes it work. The theoretical case for scalability is solid. The empirical case requires production-scale testing that the current work cannot provide. That gap between theoretical soundness and empirical validation at scale is honest and worth stating clearly rather than papering over with optimistic projections.

What the current work does establish is that the design principles are sound, that explicit human feedback improves reward model quality in measurable ways, that transparency supports rather than undermines trust in adaptive systems, and that the cold start problem is manageable with a well-designed onboarding procedure. Those are meaningful contributions even if the scaling questions remain open.

Section VII draws these threads together and states what we think the work shows, without overstating it.

7. Conclusion

Adaptive interfaces have been a research target for long enough that the gap between what the literature promises and what deployed systems deliver is hard to ignore. The promise is interfaces that genuinely fit the people using them, learning continuously and improving over time. The reality, in most production systems, is a thin layer of personalization built on behavioral proxies that were never quite measuring what designers thought they were measuring, wrapped in adaptation logic that users quickly learn to route around.

The HITL-RL framework does not solve this gap completely. No single framework could. But it addresses the part of the problem that we think matters most: the feedback signal. Everything downstream of the reward model, the policy updates, the convergence behavior, the long-run alignment between system behavior and user preference, depends on whether the reward model is measuring what users want rather than what they do when they have no better option.

The hybrid feedback model is the answer we landed on. Not purely behavioral, because behavioral signals encode accommodation as readily as they encode preference. Not purely explicit, because explicit feedback solicitation has real costs and users will not sustain engagement with a system that demands their attention too frequently. The combination, implicit signals running continuously with explicit queries triggered by reward model uncertainty, handles both failure modes without introducing new ones of comparable severity.

The results support this design. Across all four primary metrics, HITL-RL outperformed both the static baseline and the automated RL condition, with the performance gap widening across sessions rather than narrowing. The correction frequency result is the one we find most meaningful. By session six, users in the HITL-RL condition were manually reverting system adaptations at roughly one-third the rate of automated RL users. That is a direct measure of policy alignment, and a more honest one than satisfaction scores, which are influenced by expectations and context in ways that are difficult to disentangle.

The transparency findings are a secondary contribution worth taking seriously in their own right. The observation that users responded positively to explanations of interface changes, finding them reassuring rather than intrusive, challenges a design assumption that has calcified into received wisdom without much empirical backing. Visible personalization mechanisms are not inherently uncomfortable. In task-oriented contexts where users depend on the interface as a working tool, understanding why it changed is not a burden. It is part of what makes the change acceptable [10], [11].

We also want to be clear about what the work does not show. The evaluation is bounded: forty-two participants, one interface context, two weeks, structured tasks. The scaling claims rest on architectural arguments and prior federated learning work [8] rather than on empirical evidence at production scale. The multi-user coherence problem, what happens when individual policy adaptation creates divergent experiences across users who share a system, is identified but not solved [14]. The preference drift problem, how to detect and respond when a user's needs change substantially from their historical pattern, is framed as future work rather than addressed in the current framework.

These are honest limitations, not rhetorical hedges. Stating them clearly is more useful than burying them in a limitations paragraph after the conclusion. The research community does better work when papers are upfront about what they have not figured out yet.

What we believe this work contributes, stated as directly as possible, is the following. First, a concrete architecture for integrating explicit human feedback into a continuous RL learning loop at the interface level, with a principled mechanism for deciding when to solicit that feedback and how to combine it with behavioral signals. Second, empirical evidence that this integration improves policy quality in measurable, practically meaningful ways compared to behavioral-signal-only alternatives. Third, a transparency layer design that treats interface adaptation as a user-facing event rather than an implementation detail, and evidence that this approach supports rather than undermines user trust. Fourth, a connection between the adaptive interface problem and the broader preference learning literature [5], [6] that we hope makes both bodies of work more legible to researchers approaching from either direction.

The deeper argument running through this paper is about what it means to design for users rather than at them. Fully automated adaptation optimizes a proxy for user preference and hopes the proxy is accurate enough. HITL-RL keeps the user in the loop not because the automation is insufficient, though in some cases it is, but because user preferences are the ground truth the system is trying to approximate, and occasionally checking against the ground truth is just good engineering. The human is not in the loop as a fallback. They are in the loop because the loop is about them.

Where this line of work goes next depends partly on what the research community finds most pressing and partly on what practitioners building real adaptive systems find most limiting. The scaling questions need production-scale data to answer properly. The preference drift problem needs longitudinal studies longer than two weeks. The multi-user coherence problem needs evaluation in genuinely collaborative settings. None of these are small undertakings, but none of them require theoretical breakthroughs either. They require deployment, observation, and the kind of iterative refinement that only happens when research frameworks meet real users at real scale.

That is, in a sense, the argument for HITL-RL applied to its own development. The framework keeps humans in the loop because humans are the measure of whether the system is working. The same logic applies to the research program. The next step is not more controlled experiments. It is deployment, with eyes open, watching what breaks.

Funding: This research received no external funding.
Conflicts of Interest: The authors declare no conflict of interest.
Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers.

References

[1] Li, G., Gomez, R., Nakamura, K., & He, B. (2019). Human-centered reinforcement learning: A survey. *IEEE Transactions on Human-Machine Systems*, *49*(4), 337–349.

[2] Todi, K., Bailly, G., Leiva, L., & Oulasvirta, A. (2021). Adapting user interfaces with model-based reinforcement learning. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (pp. 1–13). ACM.

[3] Gaspar-Figueiredo, D., Abrahão, S., Fernández-Diego, M., & Insfráán, E. (2023). A comparative study on reward models for UI adaptation with reinforcement learning. *arXiv preprint arXiv:2308.13937*.

[4] Gaspar-Figueiredo, D. (2023). Learning from interaction: User interface adaptation using reinforcement learning. *arXiv preprint arXiv:2312.07216*.

[5] Christiano, P. F., Leike, J., Brown, T., Martic, M., Legg, S., & Amodei, D. (2017). Deep reinforcement learning from human preferences. *Advances in Neural Information Processing Systems*, *30*.

[6] Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. (2022). Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, *35*, 27730–27744.

[7] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

[8] Myakala, P. K., & Kamatala, S. (2023). Scalable decentralized multi-agent federated reinforcement learning: Challenges and advances. *International Journal of Electrical, Electronics and Computers*, *8*(6), 10–22161.

[9] Vontela, P. R., & Tuniki, R. R. (2023). Intelligent user interfaces for data-driven systems: A reinforcement learning approach. *International Journal of Scientific Advances (IJSCIA)*, *4*(6), 1065–1070.

[10] Margetis, G., Ntoa, S., Antona, M., & Stephanidis, C. (2021). Human-centered design of artificial intelligence. In *Handbook of Human Factors and Ergonomics* (pp. 1085–1106). Wiley.

[11] Zhang, J. (2022). *User interface design based on human-centered explainable AI methods*.

[12] Manche, R., & Myakala, P. K. (2022). Explaining black-box behavior in large language models. *International Journal of Computing and Artificial Intelligence*, *3*(2).

[13] Methuku, V., Kamatala, S., Naayini, P., & Vontela, P. R. (2022). From ethical principles to technical safeguards: A unified framework for safe and human-centered artificial intelligence. *American International Journal of Computer Science and Technology*, *4*(5), 26–34.

[14] De la Iglesia, D. G., Calderón, J. F., Weyns, D., Milrad, M., & Nussbaum, M. (2014). A self-adaptive multi-agent system approach for collaborative mobile learning. *IEEE Transactions on Learning Technologies*, *8*(2), 158–172.