Agent-Based Auditing of Online Platforms for Algorithmic Manipulative Personalization

Nikos Lazaridis
Utrecht University
Utrecht, Netherlands
n.lazaridis@students.uu.nl

Pınar Yolum
Utrecht University
Utrecht, Netherlands
p.yolum@uu.nl

ABSTRACT

Manipulative user interface designs, known as dark patterns, and algorithmic price discrimination are increasingly used to exploit consumers in digital environments. While often studied separately, these practices represent interconnected forms of algorithmic harm that leverage user data to personalize manipulative experiences, a phenomenon this study terms Algorithmic Manipulative Personalization (AMP). Existing detection methods for these harms are often limited, with dark pattern detectors focusing on static, single-page analysis and price discrimination research relying on rigid, non-GUI-based web crawlers, leaving dynamic, behavior-driven manipulations largely unaddressed. Accordingly, this study addresses the critical need for a unified and dynamic auditing methodology to expose these sophisticated, personalized manipulations. We develop and evaluate a novel, agent-based methodology where autonomous, LLM-powered agents simulate diverse user personas to audit websites for both dark patterns and price discrimination. Our findings on real-world airline websites demonstrate the viability of this approach, with agents successfully completing complex booking tasks and discovering consistent evidence of price and content personalization. The proposed methodology serves as a significant step towards improving transparency and fairness in online platforms by providing the tools to detect and analyze algorithmic manipulative personalization. This work provides a robust framework for researchers and regulators to audit algorithmic systems, enabling effective oversight of online platforms and bringing them closer to being ethical and accountable.

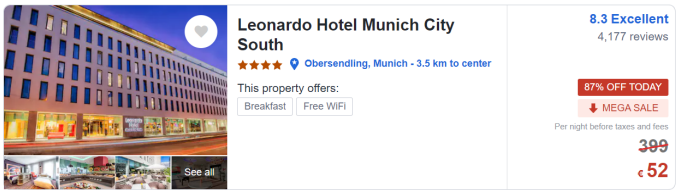
KEYWORDS

Dark patterns, Price discrimination, Algorithmic Auditing, LLM agents

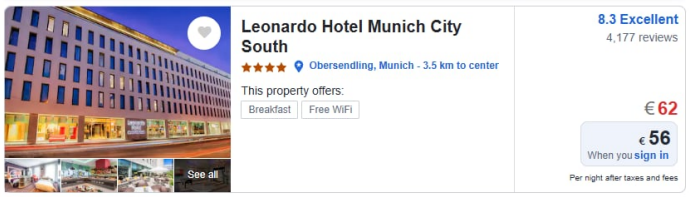
1 INTRODUCTION

Digital platforms have evolved sophisticated mechanisms to influence user behavior through algorithmic personalization. Two particularly concerning practices have emerged: *algorithmic price discrimination* [1], which leverages user data to charge different prices for identical services, and *dark patterns* [8], which employ deceptive interface designs to coerce users into unintended actions. While traditionally studied as separate phenomena, these practices increasingly converge in what we term **algorithmic manipulative personalization (AMP)**—the coordinated deployment of discriminatory pricing and personalized manipulative designs to exploit individual psychological and economic vulnerabilities.

The scale and sophistication of AMP pose significant challenges to consumer protection and market fairness [10, 14]. Unlike static manipulative designs that affect all users uniformly, AMP adapts its tactics to individual user profiles, making detection substantially more difficult. A platform might show urgent scarcity messages only to price-sensitive users while presenting luxury framing to affluent visitors, or deploy different pricing strategies based on inferred willingness to pay. This personalized manipulation creates an invisible tier system where users receive fundamentally different experiences without their knowledge. Figure 1 illustrates such a case: the same hotel listing on a major booking platform was accessed simultaneously with identical search parameters but from different residential IP proxies. One user was shown a dramatic "87% OFF TODAY" *Mega Sale* banner with a fictitious anchor price, while the other user saw the same offer without urgency cues. Although the underlying deal was identical, the manipulative interface selectively deployed deceptive urgency and reference pricing, making the manipulation invisible to anyone encountering only one version.



(a) Location A (Pakistan proxy) with urgency and fictitious reference price.



(b) Location B (Greece proxy) without urgency cues.

Figure 1: Illustrative example of an algorithmic dark pattern in a booking platform (Leonardo Hotel Munich). Urgency cues and inflated reference pricing are selectively deployed based on user location.

Existing detection methods cannot effectively address AMP. Price discrimination studies rely on rigid web crawlers unable to interact with dynamic, JavaScript-driven interfaces or complete the multi-step journeys that trigger personalization [11, 19, 20]. Dark pattern detection, in turn, focuses on static screenshots and misses manipulations that unfold over time or across user interactions [12, 15, 25].

Both fail to capture algorithms that adapt deceptive tactics to user-specific profiles, making manipulations visible only to particular personas.

We address these gaps with an autonomous, agent-based auditing framework. LLM-powered GUI agents embody distinct user personas and navigate realistic, multi-step workflows while recording complete interaction traces for post-hoc analysis. Our contributions are threefold: (1) we formalize algorithmic manipulative personalization as a unified framework combining price discrimination and dark patterns; (2) we introduce a methodology for systematic, persona-based auditing of dynamic web interfaces; and (3) we develop a temporal dark-pattern detection model that captures sequential manipulations invisible to static methods.

Our evaluation across six major airline platforms demonstrates both the feasibility and effectiveness of agent-based algorithmic auditing. Agents successfully completed complex booking workflows with a 65.8% success rate across 79 experimental runs, while systematic persona-based comparisons revealed significant evidence of price discrimination (up to 65.8% price differences) and content personalization. Notably, while we found extensive price and content manipulation, dark patterns were uniformly applied across personas rather than selectively deployed, suggesting platforms may avoid discriminatory application of overtly coercive designs while readily engaging in price-based manipulation.

The remainder of this paper is structured as follows. Section 2 reviews related work in algorithmic auditing and automated detection. Section 3 formalizes the problem of algorithmic manipulative personalization and outlines detection requirements. Section 4 details our agent-based framework architecture. Section 5 presents our experimental evaluation across three phases: agent viability testing, detection model validation, and systematic personalization auditing. Section 6 discusses the implications and limitations of our findings and provides pointers for future work.

2 RELATED WORK AND LIMITATIONS

Dark pattern detection and algorithmic price discrimination both have been studied for some time but in isolation.

Dark Pattern Detection: The automated detection of dark patterns has evolved rapidly from manual identification to sophisticated machine learning approaches. Early work relied on community-driven reporting platforms such as Reddit's `r/assholedesign` [22] and crowdsourced collections, but these approaches lacked systematic detection capabilities and were limited by user participation. Recent advances have employed deep learning for automated recognition. Mansur et al. [15] introduced *AIDUI*, which uses computer vision to analyze UI screenshots and detect ten categories of dark patterns through visual cue detection and spatial analysis. *UI-Guard* [2] expanded this approach using knowledge-driven systems that leverage six essential UI properties to identify patterns without requiring application-specific rules. These systems achieved high accuracy on static pattern recognition but were fundamentally limited to single-screen analysis.

A critical advancement came with *AppRay* [3], which introduced task-oriented app exploration combined with automated detection to identify both single-UI and multiple-UI dark patterns. By employing LLM-guided exploration with traditional automated tools, AppRay could detect dynamic patterns that require analyzing transitions between UI states. However, AppRay's analysis was limited to pre-recorded app exploration sessions and could not adapt to personalized content variations. Most recently, *DeceptiLens* [13] leveraged multimodal large language models with retrieval-augmented generation (RAG) [7] to provide explainable dark pattern detection. While achieving comparable accuracy to human experts, DeceptiLens analyzed individual UI screenshots in isolation and could not detect patterns that emerge only through specific user behavioral triggers. We use this as our base model in Section 4.4.

Algorithmic Price Discrimination Detection: Early work by Mikians et al. [19] established the fundamental methodology to uncover differential pricing: deploying multiple vantage points with different user profiles to query the same products and comparing results for systematic price differences. Their distributed measurement system used trained "persona" profiles (affluent vs. budget-conscious) and geographic diversity to detect price gaps up to 166% for digital goods. Hannak et al. [11] advanced this methodology by introducing formal experimental controls and large-scale automation. Their framework combined real users recruited through Amazon Mechanical Turk with synthetic browser-based agents configured with different consumer profiles. Their comprehensive analysis of 16 e-commerce sites revealed multiple forms of discrimination, including lower hotel prices for logged-in users and device-based product ranking variations.

Machine learning approaches have also focused on identifying algorithmic pricing strategies through pattern analysis. Chen et al. [4] studied Amazon Marketplace by collecting four months of price data and applying time-series analysis to detect algorithmic behavior patterns. Their work revealed that algorithmic sellers often achieved higher sales despite not offering the lowest prices, and helped reverse-engineer Amazon's Buy Box algorithm.

Limitations of Existing Approaches: Current detection methodologies exhibit several critical limitations that prevent effective AMP detection. First, price discrimination research overwhelmingly relies on rigid web crawlers that are programmed for specific queries and websites, making generalization difficult and complex user journey simulation nearly impossible. These tools operate at the source code level without "seeing" rendered GUIs, making them incapable of detecting visually manifested dark patterns or navigating dynamic interfaces where content is generated responsively.

Second, dark pattern detection approaches remain largely confined to static analysis of individual screenshots. While recent work has begun addressing multi-UI patterns, these methods cannot detect personalized patterns that appear only for specific user profiles or vary dynamically during sessions based on behavioral inputs.

Third, existing methodologies exhibit blind spots regarding user behavior and personalization triggers. Crowdsourcing approaches lack controlled experimental conditions, making it difficult to isolate variables and attribute differences to specific platform manipulations. Machine learning approaches for price discrimination analyze market dynamics rather than user-specific personalization, failing to model the precise behavioral triggers that cause discriminatory outcomes for individual users.

Finally, no existing work addresses the convergence of price discrimination and dark patterns into coordinated manipulative

strategies. Research has treated these as independent phenomena, missing the possibility that platforms may deploy them in concert to create more sophisticated exploitation mechanisms.

3 PROBLEM DEFINITION

We first formalize the algorithmic manipulative personalization problem and then provide the requirements that an auditing methodology needs to capture when detecting instances of the problem.

3.1 Algorithmic Manipulative Personalization

We define *algorithmic manipulative personalization (AMP)* as the systematic deployment of **(i)** price discrimination, **(ii)** content personalization, or **(iii)** dark patterns, based on algorithmically inferred user characteristics, with the intent to exploit rather than serve user interests.

Before formalizing AMP, we briefly clarify the three component constructs as used in this paper. *Price discrimination* refers to the practice of charging different prices to different users for the identical service or product based on inferred user attributes (e.g., location, device, or browsing history) rather than on objective service characteristics [1]. *Content personalization* refers to differential presentation of the same offering—through ranking, framing, available options, or accompanying incentives—such that distinct users see materially different versions of the same item. *Dark patterns* are user interface design choices that exploit cognitive biases to steer users toward decisions that benefit the platform at the user's expense, such as fabricating urgency, hiding costs, obstructing opt-outs, or pre-selecting unwanted options [8, 9]. We adopt the 18-pattern taxonomy of Gray et al. [9], enumerated and used operationally in Section 4.4. The novelty of AMP is not the existence of any of these mechanisms in isolation, but their algorithmic conditioning on user-specific signals: the same platform interaction can yield systematically different prices, different presentations, and different manipulative cues for different users.

Let U be a user with profile vector $p = (d, b, t)$, where d represents demographic attributes (e.g., age, gender, location), b represents behavioral patterns (e.g., browsing or purchasing history), and t represents the technical fingerprint (e.g., device, browser, connection metadata). A platform P implements AMP when

$$\begin{aligned} \exists U, U' : p \neq p' \wedge \\ \big(& price_P(U, s) \neq price_P(U', s) \\ & \vee\ content_P(U) \neq content_P(U') \\ & \vee\ darkpattern_P(U) \neq darkpattern_P(U') \big) \end{aligned} \tag{1}$$

where s denotes an identical service or product, and the differential treatment is designed to maximize platform utility rather than improve user experience.

AMP encompasses two primary manifestations. The first operates at the moment of decision-making: the system monitors behavioural cues such as hesitation or price sensitivity and responds with targeted interventions (e.g., scarcity messages, anchored reference prices) that exploit cognitive uncertainty at vulnerable decision points. The second operates through sustained influence over many sessions, gradually reshaping user expectations—e.g., recommendation streams that drift toward higher price points or reinforcement schedules tuned to individual engagement patterns. Where the first reacts to existing vulnerability, the second cumulatively creates it.

3.2 Detection Requirements

Effective AMP detection requires satisfying several technical and methodological criteria:

Interactive Capability: The system must interact with fully rendered graphical user interfaces rather than relying solely on static HTML parsing, since modern personalization is delivered through dynamic, JavaScript-driven interfaces that are invisible to source-level crawlers.

Behavioral Realism: Detection must simulate authentic user behavior (clicks, mouse movement, dwell times) that triggers personalization algorithms; simple query-response bots typically do not activate these systems.

Temporal Analysis: The system must capture and analyze sequences of UI changes over time, as many manipulative patterns only emerge through multi-step user journeys.

Controlled Experimentation: The methodology must maintain experimental rigor with controlled variables, isolating the persona attributes under test in order to attribute differential treatment to platform algorithms rather than to user-side noise.

Persona Differentiation: The system must project distinct, persistent user identities that algorithms can reliably categorize and respond to differentially, since algorithmic manipulation is predicated on user segmentation.

Scalability: The approach must be generalizable across platforms and sustainable for systematic auditing without requiring platform-specific engineering, so that researchers, regulators, and auditors can apply the same methodology across many platforms with minimal reconfiguration.

4 AGENT-BASED AUDITING FRAMEWORK

Our framework employs autonomous GUI agents powered by multimodal large language models to systematically probe platforms for algorithmic personalization. The architecture follows a modular design with three primary components operating in a perception-action cycle, as illustrated in Figure 2.

The *Planner* serves as the cognitive core, processing multimodal environmental inputs and formulating action plans to achieve assigned goals. Each planner is configured with a fixed system prompt that defines its operational constraints, response format, and behavioral guidelines. This prompt acts as an architectural blueprint ensuring reproducible behavior independent of stochastic model variations. The full text of this system prompt is provided in the Supplementary Material.

The *Controller* functions as an action mediator, translating the planner's abstract plans into concrete browser commands through a dynamic tool registry. This registry maintains a comprehensive action space including navigation (`go_to_url`, `search_google`), element interaction (`click_element`, `input_text`, content manipulation (`extract_content`, `scroll_down`), tab management (`open_tab`,

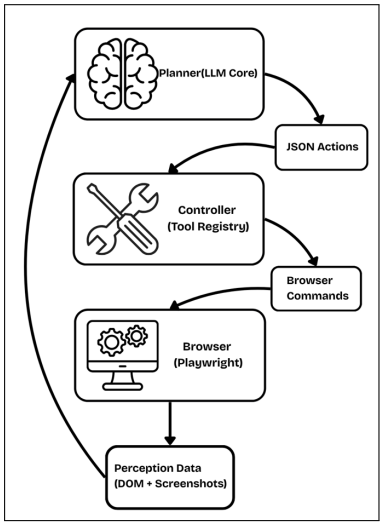


Figure 2: Conceptual architecture of the AMP detection framework.

`switch_to_tab`), and specialized tools (calendar navigation, Gmail API integration for verification codes).

The *Browser* component manages the physical web interaction through Playwright-controlled Chromium instances. Each agent operates within an isolated `BrowserContext` that maintains independent cookies, session storage, and fingerprint configurations, ensuring persona-specific experiences without cross-contamination.

4.1 Multimodal Perception System

Planner perception integrates four complementary information streams: a *topological state* (page hierarchy, visited states, available transitions), a *DOM abstraction* (cleaned textual DOM with indexed interactive elements), a *visual stream* (full-page screenshots with numerically labelled bounding boxes), and *execution feedback* (structured action-outcome reports that close the perception-action loop).

4.2 Persona Design and Identity Persistence

Systematic persona construction is crucial for triggering algorithmic personalization. We design personas that vary along both technical and behavioral dimensions to create distinct profiles.

Technical Fingerprinting: Each persona maintains a unique and internally consistent technical profile including user-agent strings (indicating different device/OS combinations), viewport resolutions and color schemes, geographic coordinates and timezone settings, HTTP header suites including Accept-Language and Client-Hints, and residential proxy IP addresses for geographic authenticity.

Behavioral Differentiation: Personas exhibit distinct interaction patterns designed to trigger algorithmic categorization, varying along five axes: *interaction frequency* (frequent vs. rare connector), *search consistency* (repeated identical queries vs. diverse exploration), *price sensitivity* (low-price sort + discount filters vs. newest/best-seller + luxury filters), *engagement depth* (long dwell + high scroll depth vs. rapid skimming), and *purchase intent* (decisive buyer vs. window shopper with high cart-abandonment).

Identity Persistence: Maintaining consistent digital identity across sessions requires sophisticated state management. Each persona operates within an isolated BrowserContext with a dedicated cookie jar stored persistently between sessions, similar to [5]. Furthermore, to jumpstart the personalization process and establish a clear baseline identity, cookie jars can be prepopulated with curated sets of cookies indicative of a target persona's browsing history. Browser fingerprints are configured consistently to enable platform recognition across visits. Session continuity is maintained through careful proxy IP consistency and behavioral pattern adherence.

4.3 Stealth and Anti-Detection Mechanisms

Commercial platforms deploy bot-detection systems that could contaminate experimental results by serving non-personalized content to detected automation. Our framework therefore combines three evasion layers. *Browser-level evasion* alters automation indicators (setting `navigator.webdriver` to undefined, spoofing realistic `navigator.plugins` and `navigator.languages`, and tuning Chromium launch arguments) and routes all traffic through residential proxies. *Action-level humanization* replaces deterministic input with Bézier-curved mouse trajectories, character-by-character typing with variable delays, and randomized inter-action pauses. *Advanced evasion* adds JavaScript-injected overrides of common fingerprinting probes, canvas-fingerprint spoofing, and WebRTC configuration that keeps the apparent geography consistent with the proxy.

4.4 Dark Pattern Detection Models

Traditional dark pattern detection analyzes static screenshots, missing dynamic patterns that unfold across user interactions. We develop a novel post-hoc sequential analysis framework to identify a target set of 18 dark patterns: Comparison Prevention (hiding alternatives), Confirmshaming (guilting users into consent), Disguised Ads (ads masked as content), Fake Scarcity (illusory stock limits), Fake Social Proof (fabricated popularity cues), Fake Urgency (false time pressure), False Hierarchy (misleading emphasis through layout), Forced Action (requiring unrelated steps), Hidden Costs (undisclosed charges), Interface Interference (designs impeding choices), Misdirection (drawing attention away from crucial options), Nagging (repetitive prompts to comply), Obstruction (making opt-outs difficult), Preselection (default consent options), Privacy Zuckering (tricking users into data sharing), Sneak into Basket (adding items without consent), Trick Wording (ambiguous or deceptive phrasing), and Visual Interference (obscuring information through design).

These specific patterns were chosen based on established taxonomies in human-computer interaction literature, and their detailed definitions can be found in the work of Gray et al. [9].

Our general detection framework operates by processing user interface (UI) screenshots as its primary input. For each screenshot, the models retrieve relevant context from a specialized knowledge base containing scholarly definitions and examples of dark patterns. By comparing the visual and textual elements of the UI against this

retrieved information, the model provides a final prediction, a confidence score, and a text-based explanation justifying its decision. To execute this, we designed five increasingly sophisticated models.

Model 1 (Baseline – DeceptiLens) : This model serves as the baseline and is a direct reimplementation of the DeceptiLens architecture [13], based on the publicly available implementation on GitHub. This model uses a multimodal LLM with a Retrieval-Augmented Generation (RAG) process to classify dark patterns and provide structured, reference-backed explanations. Each dark pattern category is evaluated independently through separate prompt executions as done in the original paper to ensure focused reasoning while scaling the number of API calls with the number of categories. For the RAG component, a domain-specific knowledge base was constructed from all academic papers cited in this paper (i.e. [9, 12, 16–18, 21, 23, 24]). The corpus was segmented into coherent passages, embedded with the all-MiniLM-L6-v2 model, and indexed in FAISS [6] to enable retrieval of contextually relevant scholarly material during classification.

Model 2 & 3 (Annotator-Auditor) : Models 2 and 3 introduce the "annotator–auditor" dual-agent framework. The effectiveness of the dual-agent framework hinges on meticulous prompt engineering. The Annotator (Model 2) is instructed to act as an HCI expert to identify any plausible dark pattern occurrences, even when the evidence is ambiguous. The Auditor (Model 3), by contrast, applies a strict "no false positives" rule and evaluates each of the Annotator's findings with the sole possible outcomes of agree or disagree. This framing is crucial for the review mechanism. The pattern is accepted under two circumstances: (i) the Annotator marks a pattern as present and the Auditor agrees; or (ii) the Annotator marks a pattern as absent but the Auditor disagrees, in which case the Auditor's stance overrides. In all other cases, the pattern is rejected.

Model 4 (Dual-Agent Holistic) : Model 4 preserves the dual-agent review mechanism but restructures the analysis so that all 18 dark pattern categories are evaluated within a single prompt–response cycle. This architectural change reduces API calls by over 90% per image without compromising the rigor of dual-agent validation.

Model 5 (Ensemble Refinement) : Model 5 is a post-processing and refinement layer applied to outputs from Models 1 and 4. Rather than relying on dataset-specific rules, it applies broadly applicable analytical checks to improve classification accuracy. These include semantic disambiguation between visually or linguistically similar categories (e.g., distinguishing "Fake Scarcity" from "Fake Urgency" via OCR-detected numerical or temporal cues), verification that any text cited in an explanation is actually present in the corresponding screenshot, and filtering of common, benign marketing phrases that may otherwise trigger false positives. Confidence thresholds are applied to balance precision and recall, and a cap on the maximum number of distinct detections per image mitigates over-classification.

4.5 Temporal Analysis of Data

For patterns whose manipulative nature is inherently sequential (Obstruction, Sneak into Basket, Hidden Costs, Misdirection), Models 4 and 5 analyze sequences of four consecutive UI frames rather than individual screenshots, following a four-stage protocol: (i) *temporal feature detection*—identifying UI element changes across frames (appearance and disappearance of elements, content modification, price or option variations); (ii) *pattern assessment*—scoring asymmetric interaction flows (e.g., easy subscription, hard cancellation), progressive restriction, and delayed revelation of critical information; (iii) *journey progression analysis*—mapping how the interface guides or constrains user choices end-to-end; and (iv) *conclusive synthesis*—integrating these observations into a final pattern-presence judgement with justification.

5 EXPERIMENTAL EVALUATION

We conducted a comprehensive three-phase experimental evaluation to validate each component of our framework[1].

5.1 Phase 1: Agent Viability Assessment

To establish the baseline operational capability of our agents, we selected six airlines representing diverse business models and market segments: United Airlines and Turkish Airlines (global market leaders), Ryanair and EasyJet (low-cost carriers), Emirates (premium carrier), and LATAM Airlines Brasil (geographic specialist).

For each platform we designed multi-step tasks based on manual interface inspection, covering *core flight booking* (full search-through-payment workflow), *ancillary services* (hotel and car-rental booking through the airline platform), and *exploratory tasks* (promotional deal browsing and destination exploration).

A standardized agent persona executed 18 distinct tasks with 3-5 repetitions each, totaling 79 experimental runs. Each run was configured with identical system prompts and behavioral parameters to isolate platform-specific performance variations.

5.1.1 Performance Metrics. We defined four metrics beyond binary success/failure. *Progress* (range 0–1) measures the proportion of optimal task steps completed, capturing how deep into the funnel the agent advanced even when failing. *Action Efficiency* is the ratio of minimum-required to actual actions for successful runs. *Stepwise Success Rate* is the proportion of individual actions the agent self-assesses as successful: after each action it compares the new UI state against the intended outcome and records the judgement in the `evaluation_previous_goal` field of its structured JSON response. *Failure Categorization* classifies error modes into seven types: *Unhandled Edge Case*, *Capability-Blocking Mechanism*, *Plan Incompletion Failure*, *State Management Failure*, *Systematic Tool Use Failure*, *Navigational Derailment*, and *Maximum Steps Limit*.

5.1.2 Results and Analysis. Performance varied significantly across task types and platforms; Tables 1 and 2 report the results. Notably, progress scores remained high across all settings (0.73–0.96) even when success rates dropped, indicating that most failures are late-stage rather than early collapses—agents typically reach advanced steps before erroring, with bottlenecks concentrated in final decision points and edge-case handling.

[1]The full text of the system prompts of each model as well as an example prompt for each task category are provided in the Supplementary Material. The agent implementations, detection models, and the annotated screenshot corpus used in Phase 2 are publicly released at https://github.com/nlazaridi/Personalised-Dark-Pattern-Detection to support reproducibility.

Table 1: Task Performance

Task Type	Success Rate	Action Efficiency	Progress
Exploration	92.86%	0.894	0.96
Car Rental	60.00%	0.767	0.81
Hotel Booking	60.00%	0.584	0.85
Flight Booking	60.00%	0.543	0.85

Table 2: Platform Performance

Platform	Success Rate	Action Efficiency	Progress
LATAM Airlines Brasil	93.33%	0.632	0.95
Turkish Airlines	73.33%	0.710	0.88
United Airlines	63.64%	0.729	0.84
Emirates	56.25%	0.670	0.84
Ryanair	54.55%	0.560	0.91
EasyJet	45.45%	0.834	0.73

Error Analysis: Figure 3 shows the frequency distribution of error categories, with *Systematic Tool Use Failure* and *State Management Failure* dominating. Tool failures typically arose when custom UI components were incompatible with standard interaction methods designed for semantic HTML elements, and state-management failures arose from agent confusion after unexpected page transitions or popups. A more nuanced view comes from breaking errors down by success status (Figure 4), which shows that both dominant error types appear in substantial numbers of successful runs as well; agents frequently overcome intermediate breakdowns through persistent interaction without explicit corrective instruction. Successful runs also exhibit a strong correlation between Step-wise Success Rate and Action Efficiency ($r = 0.56$), suggesting that the agent's internal self-assessment aligns with objective performance.

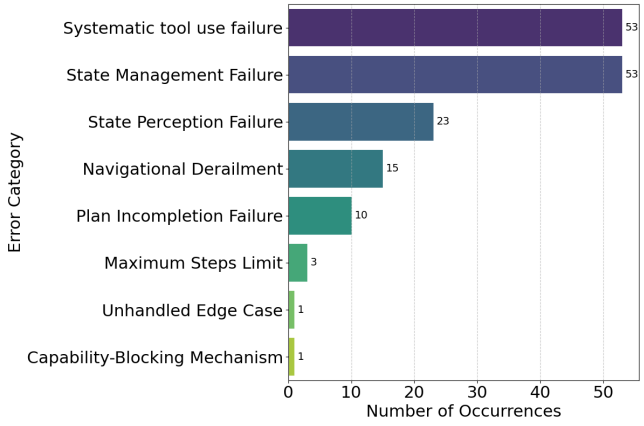


Figure 3: Distribution of error types.

5.2 Phase 2: Dark Pattern Detection

The second phase studies to what extent the agents can detect dark patterns.

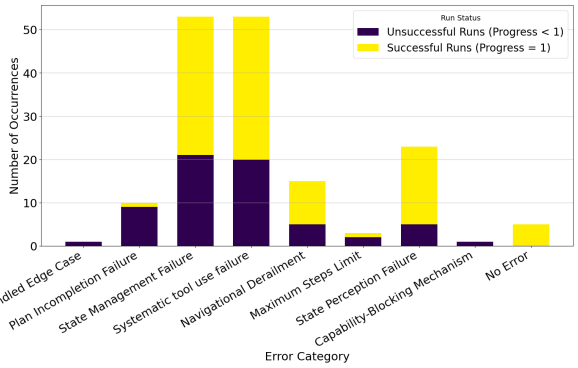


Figure 4: Distribution of error types by success status.

5.2.1 Dataset Creation. Using the 214 unique UI screenshots collected during Phase 1, we built a ground-truth dataset for dark-pattern detection. Two custom annotation tools supported the work: a detailed inspection interface for drawing precise bounding boxes, and a model-assisted labelling interface displaying screenshots alongside automated outputs. Ground-truth labelling was performed by a single trained annotator (one of the authors) following the dark-pattern definitions of Gray et al. [9]. This prioritises definitional consistency over inter-annotator validation; the implications for generalizability are discussed in Section 6.

5.2.2 Model Performance. The five detection models explained in Section 4.4 were evaluated on the annotated dataset using precision, recall, and F1-score metrics appropriate for the inherently imbalanced data. Table 3 presents the comparative performance of each model, showing clear performance hierarchy. The baseline Model 1 achieved extremely high recall (0.965) but suffered from poor precision (0.319), confirming the need for more sophisticated architectures. Models 2 and 3 showed even lower precision while maintaining high recall.

Table 3: Dark Pattern Detection Results

Model	Accuracy	Precision	Recall	F1	API Calls
Model 1 (DeceptiLens)	0.865	0.319	0.965	0.480	18
Model 2 (Annotator)	0.634	0.147	0.969	0.255	18
Model 3 (Auditor)	0.730	0.179	0.885	0.298	18
Model 4 (Dual-Agent)	0.974	0.823	0.761	0.791	1
Model 5 (Ensemble)	0.977	0.869	0.765	0.814	1

The largest gain came with Model 4's holistic approach, which achieved balanced performance (0.823 precision, 0.761 recall) while requiring 95% fewer API calls. By evaluating all 18 patterns within a single prompt, the model is forced into comparative reasoning—selecting the best-fit category rather than making 18 independent yes/no decisions. This matters because several dark pattern categories exhibit substantial semantic overlap: an "Only 2 seats left" message could plausibly be classified as scarcity, urgency, or social proof, and per-pattern detectors lack a mechanism to choose between them. Forcing comparative judgement is the primary driver of Model 4's precision gain.

Model 5's ensemble refinement achieved the highest overall performance (0.869 precision, 0.814 F1). It is not a generative agent but a programmatic post-processing layer over the outputs of Models 1 and 4, applying (a) evidence grounding—verifying that text cited in an explanation actually appears in the corresponding screenshot; (b) semantic disambiguation between closely related categories; and (c) confidence-threshold filtering. Because Model 5 depends on the upstream outputs, it should be read as a refinement step rather than a standalone detector.

5.2.3 Temporal Analysis. The introduction of temporal analysis for dynamic patterns yielded dramatic improvements in specific categories. These four patterns are inherently sequential and cannot be reliably identified from a single screenshot. For instance, *Sneak into Basket* requires observing an item being added to a cart across two states without an explicit user action. Likewise, *Misdirection* and *Obstruction* are defined by the manipulative user journey itself, which involves confusing navigation or a large number of steps. Finally, detecting *Hidden Costs* is only possible by comparing the price presented at the beginning of a checkout flow to the inflated final price on the payment page. Table 4 shows that this sequential analysis significantly enhanced detection performance. These results validate that sequential context is crucial for identifying multi-step manipulative flows, which are invisible in static analysis.

Table 4: Impact of temporal analysis on dark pattern detection performance.

Pattern	Model 1 F1	Model 5 F1	Improvement
Sneak into Basket	0.25	1.00	4.0× improvement
Misdirection	0.49	0.95	1.9× improvement
Obstruction	0.12	0.67	5.6× improvement
Hidden Costs	0.74	0.85	15% improvement

5.2.4 Comparison with Prior Approaches. Because Model 1 is a faithful reimplementation of DeceptiLens [13]—the most recent published static-screenshot dark pattern detector—Tables 3 and 4 can be read directly as head-to-head comparisons between our approach and the state of the art in static analysis. On the same screenshot corpus, the static baseline (Model 1) attains an F1 of 0.480, whereas our holistic detector (Model 4) and ensemble refinement (Model 5) reach 0.791 and 0.814 respectively, driven primarily by precision gains (0.319 → 0.869). The temporal extension is even more decisive: on the four sequential patterns, the static baseline reaches F1 between 0.12 and 0.49 for *Sneak into Basket*, *Misdirection*, and *Obstruction*, and only our temporal extension lifts them to 0.67–1.00, with similar gains for *Hidden Costs*. For algorithmic price discrimination, the relevant prior baseline is the family of HTTP-level web crawlers used by Mikians et al. [19] and Hannak et al. [11]; all of the personalization findings reported in Section 5.3.2 arise from interactions that require executing client-side JavaScript, completing multi-step booking flows, and persisting cookies across sessions, none of which a non-GUI crawler can reliably perform. The findings therefore extend, rather than overlap with, the coverage of prior auditing tools.

5.3 Phase 3: Personalization Auditing

The final phase studies how the finalized agent framework can be used to audit websites.

5.3.1 Experimental Design. We conducted a three-week audit to investigate algorithmic personalization with two personas. **Persona A (Frequent/Consistent)** runs Chrome on Windows, US IP and locale, twice-daily sessions, and repeats identical-route searches. **Persona B (Rare/Diverse)** runs Safari on Mac, Canadian IP and locale, twice-weekly sessions, and varies destinations and dates. Both were deployed on the three most stable Phase 1 platforms (LATAM, Turkish, United), running shared tasks for direct comparison and unique tasks to reinforce behavioural profiles.

5.3.2 Personalization Findings. **Price Discrimination Evidence:** Systematic price differences were observed across platforms, with Turkish Airlines showing the most pronounced discrimination: €1048 vs. €783 for identical flights (33.8% difference), €2369 vs. €1429 average hotel room prices (65.8%), and 8–10% consistent gaps on promotional fares for "trending destinations".

Content and Interface Personalization: Platforms varied presentation beyond pricing in four ways. *Inventory manipulation*: different numbers of available options were shown (5 vs. 10 rental cars on United Airlines), with some of the cheapest cars initially hidden and electric vehicles selectively shown to only one persona. *Descriptive framing*: identical hotel rooms were described as "includes high-speed internet & parking" vs. "highly-rated luxurious stay" depending on persona. *Reward differentiation*: one persona was offered 5× more loyalty miles for identical bookings. *Option steering*: suggested destinations emphasized more expensive choices for specific personas.

Dark Pattern Analysis: Comprehensive analysis of captured user journeys revealed no evidence of persona-targeted dark patterns; standard manipulative designs (urgency, scarcity, social proof) were uniformly applied across both personas with identical frequency and presentation.

Platform Detection and Blocking: United Airlines blocked both agents after approximately one week. The agents could still search for flights, but every itinerary returned with no available tickets, while ancillary services remained accessible. This blocking occurred without explicit CAPTCHA challenges or error messages, suggesting behavioural pattern recognition.

6 DISCUSSION AND DIRECTIONS

Agent Effectiveness: The 65.8% success rate establishes the basic viability of LLM-powered agents for algorithmic auditing, and the correlation between Step-wise Success Rate and Action Efficiency suggests reasonably reliable self-assessment. The error analysis points to two architectural bottlenecks: *Systematic Tool Use Failure* (custom UI components built from non-semantic HTML) and *State Management Failure* (recovery from unexpected state transitions). Both must be addressed before production-scale auditing is realistic.

Dark Pattern Detection: The temporal-analysis gains validate the core hypothesis that many manipulative designs are inherently sequential. Patterns like *Sneak into Basket* (0.25 → 1.00 F1) and *Misdirection* (0.49 → 0.95) are weakly handled by static methods and become reliably detectable once user journeys are modelled.

Persistent low performance on *Comparison Prevention*, *False Social Proof*, and *Privacy Zuckering* indicates that some patterns remain resistant to automated detection and likely require richer contextual modelling or domain-specific training data.

Algorithmic Personalization Evidence: The price differences observed—65.8% on hotels and 33.8% on flights for Turkish Airlines—are compelling evidence of sophisticated personalization in real-world e-commerce. Neither persona consistently received lower prices, suggesting optimization beyond demographic targeting. The compensatory reward offering (5× more miles for higher prices) further suggests platforms adjust the whole value proposition, not price alone.

Absence of Personalized Dark Patterns: The absence of persona-targeted dark patterns is arguably as informative as the presence of price discrimination. While platforms readily engage in price-based manipulation, they appear to avoid discriminatory application of overtly coercive interface designs. This points to a strategic asymmetry, where price personalization is treated as standard business practice and is widely tolerated, whereas personalized manipulative interfaces likely carry higher legal or reputational risk and are therefore deployed uniformly when deployed at all.

Limitations and Scope: This is a feasibility study with four explicit limitations. The evaluation spans a single industry (airline booking), six platforms, two personas, and 79 Phase 1 runs. The 65.8% task success rate reflects the current state of LLM-based GUI agents rather than the methodology, and agents that fail late still yield informative traces. Ground-truth labelling was done by a single trained annotator, prioritising consistency over inter-annotator validation. The absence of dark-pattern personalization under our two-persona contrast does not rule out that finer-grained segmentation or other industries would surface patterns our design does not trigger.

Adversarial Auditing and Policy Directions: Platforms adapt their detection in response to auditing, and the stealth measures of Section 4.3 will not be enough once detection draws on server-side telemetry and longitudinal behavioural profiling. Useful responses include scaling the persona pool well beyond the two used here, running agents from many residential exit IPs with rotating accounts so no single signature is fingerprinted, having agents occasionally complete real bookings rather than always stopping at payment, and cross-checking each agent's view against an independent control session on a different network so that silent denials, like the covert United Airlines block where every itinerary returned no tickets, are not mistaken for null findings. None of this is sufficient on its own, and technical work must be paired with regulation guaranteeing auditor access.

REFERENCES

[1] Emilio Calvano, Giacomo Calzolari, Vincenzo Denicolo, and Sergio Pastorello. 2020. Artificial intelligence, algorithmic pricing, and collusion. *American Economic Review* 110, 10 (2020), 3267–3297.

[2] Jieshan Chen, Jiamou Sun, Sidong Feng, Zhenchang Xing, Qinghua Lu, Xiwei Xu, and Chunyang Chen. 2023. Unveiling the tricks: Automated detection of dark patterns in mobile applications. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. 1–20.

[3] Jieshan Chen, Zhen Wang, Jiamou Sun, Wenbo Zou, Zhenchang Xing, Qinghua Lu, Qing Huang, and Xiwei Xu. 2024. From Exploration to Revelation: Detecting Dark Patterns in Mobile Apps. *arXiv preprint arXiv:2411.18084* (2024).

[4] Le Chen, Alan Mislove, and Christo Wilson. 2016. An empirical analysis of algorithmic pricing on amazon marketplace. In *Proceedings of the 25th international conference on World Wide Web*. 1339–1349.

[5] Nurullah Demir, Daniel Theis, Tobias Urban, and Norbert Pohlmann. 2022. Towards understanding first-party cookie tracking in the field. *arXiv preprint arXiv:2202.01498* (2022).

[6] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The faiss library. *arXiv preprint arXiv:2401.08281* (2024).

[7] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yixin Dai, Jiawei Sun, Haofen Wang, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997* 2, 1 (2023).

[8] Colin M Gray, Yubo Kou, Bryan Battles, Joseph Hoggatt, and Austin L Toombs. 2018. The dark (patterns) side of UX design. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–14.

[9] Colin M Gray, Cristiana Teixeira Santos, Nataliia Bielova, and Thomas Mildner. 2024. An ontology of dark patterns knowledge: Foundations, definitions, and a pathway for shared knowledge-building. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. 1–22.

[10] Mateusz Grochowski, Agnieszka Jabłonowska, Francesca Lagioia, Giovanni Sartor, et al. 2022. Algorithmic Price Discrimination and Consumer Protection. *Technology and Regulation* 2022, Special Issue: Should Data Drive Private Law? (2022), 36–47.

[11] Aniko Hannak, Gary Soeller, David Lazer, Alan Mislove, and Christo Wilson. 2014. Measuring price discrimination and steering on e-commerce web sites. In *Proceedings of the 2014 conference on internet measurement conference*. 305–318.

[12] Daniel Kirkman, Kami Vaniea, and Daniel W Woods. 2023. DarkDialogs: Automated detection of 10 dark patterns on cookie dialogs. In *2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)*. IEEE, 847–867.

[13] Emre Kocyigit, Arianna Rossi, Anastasia Sergeeva, Claudia Negri Ribalta, Ali Farjami, and Gabriele Lenzini. 2025. DeceptiLens: an Approach supporting Transparency in Deceptive Pattern Detection based on a Multimodal Large Language Model. In *Proceedings of the 2025 ACM Conference on Fairness, Accountability, and Transparency*. 1942–1959.

[14] Mark Leiser and Cristiana Santos. 2023. Dark patterns, enforcement, and the emerging digital design acquis: Manipulation beneath the interface. (2023).

[15] SM Hasan Mansur, Sabiha Salma, Damilola Awofisayo, and Kevin Moran. 2023. Aidui: Toward automated recognition of dark patterns in user interfaces. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*. IEEE, 1958–1970.

[16] Arunesh Mathur. 2021. Identifying and measuring manipulative user interfaces at scale on the web. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–5.

[17] Arunesh Mathur, Gunes Acar, Michael J Friedman, Eli Lucherini, Jonathan Mayer, Marshini Chetty, and Arvind Narayanan. 2019. Dark patterns at scale: Findings from a crawl of 11K shopping websites. *Proceedings of the ACM on human-computer interaction* 3, CSCW (2019), 1–32.

[18] Arunesh Mathur, Mihir Kshirsagar, and Jonathan Mayer. 2021. What makes a dark pattern... dark? Design attributes, normative considerations, and measurement methods. In *Proceedings of the 2021 CHI conference on human factors in computing systems*. 1–18.

[19] Jakub Mikians, László Gyarmati, Vijay Erramilli, and Nikolaos Laoutaris. 2012. Detecting price and search discrimination on the internet. In *Proceedings of the 11th ACM workshop on hot topics in networks*. 79–84.

[20] Jakub Mikians, László Gyarmati, Vijay Erramilli, and Nikolaos Laoutaris. 2013. Crowd-assisted search for price discrimination in e-commerce: First results. In *Proceedings of the ninth ACM conference on Emerging networking experiments and technologies*. 1–6.

[21] Arya Ramteke, Sankalp Tembhurne, Gunesh Sonawane, and Ratnmala N Bhimanpallewar. 2024. Detecting Deceptive Dark Patterns in E-commerce Platforms. *arXiv preprint arXiv:2406.01608* (2024).

[22] Asshole Design Reddit. 2014. *Reddit: When Assholes Design Things.* https://www.reddit.com/r/assholedesign/

[23] Arianna Rossi, Rachele Carli, Marietjie W Botes, Angelica Fernandez, Anastasia Sergeeva, and Lorena Sánchez Chamorro. 2024. Who is vulnerable to deceptive design patterns? A transdisciplinary perspective on the multi-dimensional nature of digital vulnerability. *Computer Law & Security Review* 55 (2024), 106031.

[24] Cristiana Santos, Viktorija Morozovaite, and Silvia De Conca. 2025. No harm no foul: how harms caused by dark patterns are conceptualised and tackled under EU data protection, consumer and competition laws. *Information & Communications Technology Law* (2025), 1–47.

[25] Than Htut Soe, Cristiana Teixeira Santos, and Marija Slavkovik. 2022. Automated detection of dark patterns in cookie banners: how to do it poorly and why it is hard to do it any other way. *arXiv preprint arXiv:2204.11836* (2022).