CREPE: Cross-expertise Remote Early Prototyping Exchange

Anonymous Author(s)

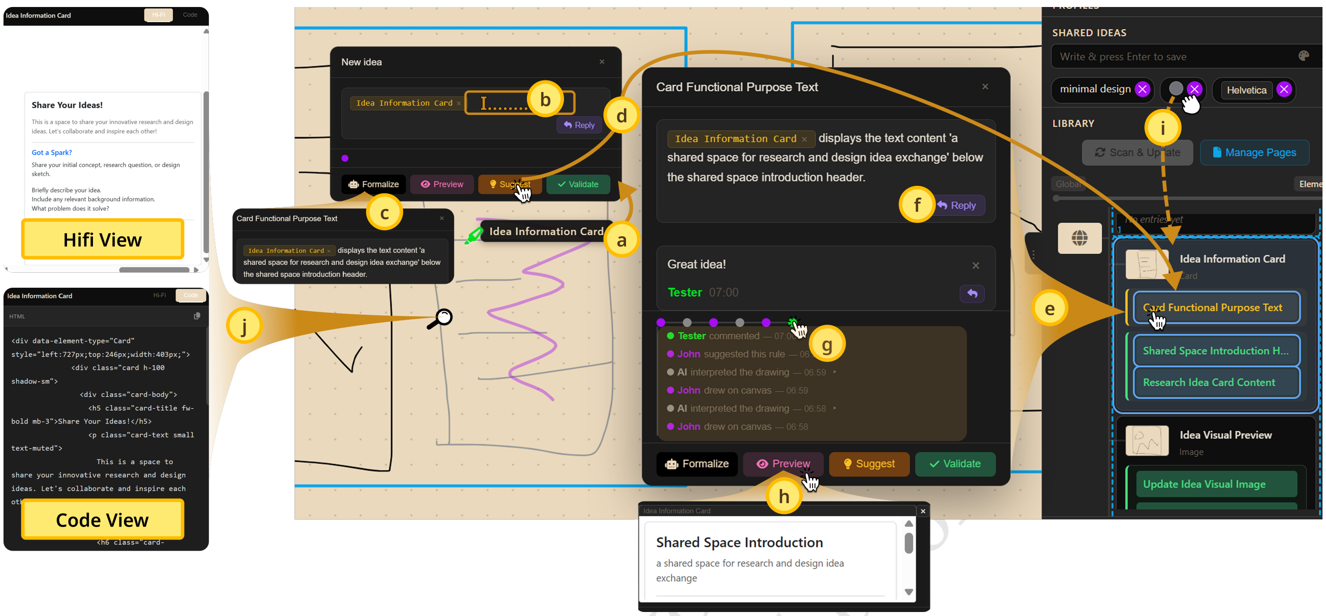

Figure 1: *CREPE* reframes collaborative prototyping as the construction of a shared, evolving artifact, where components and rules coordinate contributions across representations and expertise. Users select a named component on the canvas (a) and author a rule referencing it (b), optionally grounded in highlights and formalized by AI (c). The rule attaches to the component (d), persists in the library (e), and supports discussion (f) and creation history (g). Users can preview its HiFi effect (h) and drag shared ideas to specify component features and context (i). The same component can be viewed and acted upon as a sketch on the canvas, as a live interactive prototype (HiFi View), or as editable source code (Code View) (j).

Abstract

Interface design often involves remote collaboration among cross-expertise stakeholders. While AI tools make generating realistic interfaces easy, they do not replace the need for shared human reasoning, without which outputs are difficult to evaluate, question, or extend. We present CREPE, a system where stakeholders independently contribute to a shared artifact where elements and their specifications serve simultaneously as design commentary and executable parameters for a live preview. An AI mediator translates input into structured contributions, provides interpretation, and maintains a traceable history of decisions. A 5-day deployment with 6 client/designer pairs iterating on self-initiated websites and mobile applications shows that participants repurpose AI formalizations as communication scaffolds, fluidly combine sketch and high-fidelity views to surface misalignments, and progressively coordinate contributions without synchronous negotiation. We discuss how this approach reframes AI from artifact generation to cross-expertise mediation, and how it generalizes to other cross-expertise configurations and early prototyping contexts.

ACM Reference Format:

Anonymous Author(s). 2018. CREPE: Cross-expertise Remote Early Prototyping Exchange. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX).* ACM, New York, NY, USA, 15 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Building a user interface typically involves stakeholders with different backgrounds, each bringing a distinct lens to the same artifact and reading each other's contributions through different assumptions. Knowledge transfer between stakeholders is usually ad-hoc, scattered across meetings, email threads, and iterative feedback loops [18], and domain knowledge rarely comes in a form that other collaborators can directly operationalize [32].

A popular approach is to bring everyone together synchronously, whether in co-located workshops or remote sessions. AI-assisted tools have made these sessions increasingly effective at capturing discussion context and linking rationale to design elements [24].

But synchronous collaboration is not always possible or desirable. When collaborators must work asynchronously across expertise boundaries, the problem shifts from preserving shared rationale after a joint session to constructing mutual understanding from scratch over time. Without a synchronous exchange to fall back on, every contribution must speak for itself across an expertise boundary, and in current practice, one stakeholder absorbs the coordination burden of translating between representations, reconciling interpretations, and tracking decisions.

Meanwhile, AI tools now allow anyone to generate polished UI mockups from a text prompt in seconds[1], quickly producing interfaces, generating variations, and moving from prompt to prototype. However, a generated artifact reflects only the requester's understanding of the need: it appears finished yet embeds no design reasoning. In collaborative settings where interface design typically takes place, that reasoning is precisely what allows a collaborator to evaluate, question, or build upon a contribution. The challenge is therefore not producing interfaces faster, but enabling stakeholders who do not share a common design vocabulary to build something together without real-time communication.

We present *CREPE*, a system for asynchronous collaborative interface design where stakeholders with different expertise contribute through sketches, annotations, and rules that function simultaneously as human-readable design commentary and executable specifications for a live prototype. Rather than generating designs on behalf of a single user, the AI translates between contributors who think differently and work at different times, framing interface design as a human-to-human co-design process mediated by AI. A suggest/validate lifecycle supports gradual convergence, allowing open proposals to evolve into agreed-upon decisions while preserving a traceable decision history.

We ground our design in the theory of boundary objects [22, 33] and derive four design guidelines addressing the core challenges of asynchronous cross-expertise co-design: *translating* one's ideas into the shared artifact, *interpreting* the other's contributions, *aligning* on decisions without synchronous negotiation, and tracing the *provenance* of changes across turns. We contribute:

(1) Four design guidelines for asynchronous cross-expertise UI design, grounded in boundary object theory.
(2) *CREPE*, a system that instantiates these guidelines through an AI-mediated interaction model where sketches and rules serve as a shared language between participants with different expertise.
(3) Empirical findings from an in-situ deployment study with 6 cross-expertise pairs over 5 days on how clients and UI designers collaborate asynchronously to co-create an interface from scratch.

2 Related Work

2.1 Sketching and Early Stage Prototyping Interface

Sketching is a powerful medium in early-stage user interface design that enables designers to quickly externalize ideas, stay at an abstract level that fosters iteration, and avoid premature commitment to high-fidelity representations [29]. Much of its value lies in its ambiguity: different stakeholders can project different interpretations onto the same drawing [4, 11, 12], making it one of the few design activities that people with diverse expertise can all engage in. Building on these properties, prior systems have explored how to augment sketching with interactivity and structure. Early systems such as SILK [21] added interactivity and design memory to low-fidelity prototyping, while DENIM [26] enabled fluid transitions between abstraction levels while preserving the flexibility of sketches.

Subsequent work focused on enriching sketches with semantic structure and annotations. In DEMAIS [1], sketch strokes combined with annotations transform static storyboards into working multimedia prototypes. Similarly, Lin et al. [27] use visual language annotations to indicate reusable components and interaction transitions. Designers' Outpost [20] interprets arrows between page labels to construct navigation structures, while texSketch [34] uses structured marks to express causal links in diagrams. These systems highlight how sketches can be incrementally formalized into structured artifacts while retaining some flexibility. However, they focus on individual or co-located work. Calico [29] supports collaborative sketching but in synchronous, co-located settings between people with similar expertise. Juxtapose [15] supports parallel authoring of interface alternatives, but within a single-user workflow rather than across expertise boundaries.

More recently, generative interface prototyping has leveraged AI to produce polished interfaces directly from sketches or natural language descriptions. Eve [35] supports generation across levels of fidelity, enabling users to move from low- to high-fidelity representations. Sketch2Code [25] and Athena [2] incorporate conversational interaction to iteratively refine generation or scaffold application structure, while Misty [28] explores direct manipulation techniques for blending example interfaces with user-specified intent. Google Stitch[2] exemplifies this trend at product scale, generating polished UIs from a single user's prompt in seconds.

A key tension remains across these two lines of work. Sketch-based systems preserve flexibility but treat annotations as incremental transformations toward more structured representations, and remain focused on individual or co-located use. Generative systems produce richer outputs but treat sketches as static inputs to a single-user pipeline, losing the connection to the underlying reasoning. As a result, there is limited support for sketches as evolving, shared artifacts that accumulate meaning across contributors over time.

2.2 Asynchronous Remote Collaboration

Asynchronous collaboration tools enable users to contribute on their own schedules without requiring co-presence [10, 17, 31]. A core design concern in such systems is ensuring that participants can understand what others have done in their absence [9, 13]. Early systems such as Quilt [23] supported collaborative authoring through annotation, role-based access, and audit trail recording, while PREP [30] organized early-stage writing into parallel columns linking plans, drafts, and collaborator comments. PREP is notable for explicitly acknowledging that roles in natural collaboration are

[1]https://stitch.withgoogle.com/

[2]https://stitch.withgoogle.com/

fluid rather than fixed, even as the system still imposed them, a tension that remains largely unaddressed in subsequent asynchronous tools.

Recent efforts to embed AI into collaborative systems have focused predominantly on synchronous settings. CrossTalk [39] and EchoMind [7] transform live speech into structured representations in real time, while DesignMemo [24] connects meeting transcripts to visual design artifacts by annotating elements with semantic context. In all these systems, AI serves as a persistent, visible facilitator mediating between co-present participants.

A smaller body of work bridges synchronous and asynchronous modes. Meeting Bridges [37] carries structured outputs from meetings into downstream asynchronous work, and Vanukuru et al. [36] support continuity across recurring meeting sessions. However, these approaches still treat the synchronous meeting as the anchoring event from which asynchronous work extends. Fully asynchronous systems such as Winder [19] do support coordination without any synchronous anchor, but rely on pre-existing artifacts such as design files or recorded sessions. Across this landscape, existing work either centers collaboration on real-time interaction or assumes a stable artifact already exists. Few systems address asynchronous co-creation before the shared artifact has stabilized, where collaborators must build shared understanding and shared output simultaneously from scratch.

2.3 Boundary Objects in HCI

Boundary objects are defined as artifacts that are "plastic enough to adapt to local needs... yet robust enough to maintain a common identity across sites" [33]. Henderson [16] showed that engineering sketches function both as "conscription devices" that enlist participation and as boundary objects that support multiple interpretations across subcultures. Carlile [6] further showed that knowledge boundaries in product development require boundary objects capable not only of representing but also of transforming knowledge across functions, a capacity that becomes critical when the artifact itself is under construction.

In HCI, boundary object theory has been applied to various collaboration contexts: onboarding materials mediate between ML engineers and domain experts [5], intermediate visual outputs support negotiation between artists and commissioners [8], annotated prototypes enable coordination between designers and developers [14], and automated feedback bridges designer intent and user experience [38].

Across these works, boundary objects are treated as relatively stable artifacts around which collaboration occurs. However, in early-stage asynchronous design, the artifact does not yet exist; it must be collaboratively constructed across expertise boundaries without synchronous interaction. Based on these gaps in early-stage prototyping and asynchronous collaboration, we derive four design guidelines for supporting cross-expertise collaboration through shared, evolving artifacts:

- **DG1: Translation.** Since users cannot fully express their design intent without coding, the system should decompose contributions into discrete elements with clearly defined roles rather than pursuing high-fidelity representations. Keeping the fidelity low and leveraging multimodal input encourages rapid iteration over premature commitment.
- **DG2: Interpretation.** The system should support users in refining their contributions before handing off, reducing the interpretation gap. Rather than providing a separate interpretive feedback channel, this can be achieved through formalization itself: rewriting a contribution in standardized terms clarifies it simultaneously for the author, the absent collaborator, and the system, making translation and interpretation part of the same step.
- **DG3: Alignment.** The system should provide explicit mechanisms for asynchronous decision-making, enabling participants to distinguish between open suggestions and validated decisions, and to discuss specific elements through threaded comments.
- **DG4: Provenance.** The system should maintain a structured history of contributions and feedback, allowing each participant to review what happened during the other's turn, trace changes back to specific elements and rules, and use this information to refine their next contribution.

3 Scenario

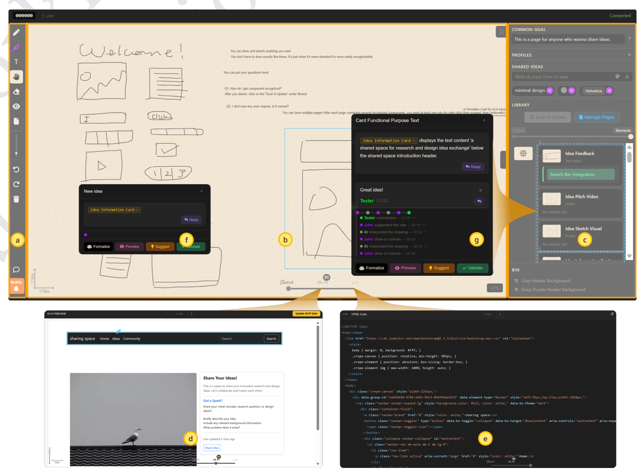

Figure 2: *CREPE* whole interface overview including (a) tool bar (b) default sketch view with infinite canvas (c) side panels including profile, common goals, shared ideas and component library (d) Hifi view (e) Code view.

Danna (designer) and Ed (end-user) are using *CREPE* collaborating on a bakery ordering page's initial prototype using *CREPE*.

Ed's first turn. Ed fills in his profile and shared ideas (Fig.2.c), then sketches a rough layout on the canvas — a header, a row of product cards, and an order button (Fig.2.b). He clicks Scan & Update on the side panel (Fig.2.c); the AI decomposes his sketch into named components: Header, Product Card, Order Button. These appear in the Library with thumbnails (Fig.2.c). Ed right-clicks Product Card and writes a rule (Fig.2.f): "each card should show a photo, name, and price." He clicks Formalize — the AI rewrites it into precise, standardized language. He reviews the formalization, clicks Suggest, and the rule enters the Library as a pending yellow entry (Fig.2.c).

Danna's first turn. Danna opens the workspace and sees Ed's sketch and pending suggestions in the Library. She reads the formalized rules, hovers over provenance dots to see Ed's original wording and the AI's interpretation (Fig.2.g), and clicks Validate on the ones she agrees with. Validated rules turn green. She tries the Hifi version of the page (Fig.2.d) then contributes her own work: using the highlighter to annotate the sketch, she adds a layout rule grouping the product cards horizontally. She clicks Formalize, then Preview to see the effect on the component inline, and clicks Suggest. She activates the Magic Lens (Fig.2.a) on the sketch canvas to inspect individual components' Hifi rendering without leaving the sketch view. She switches to Hifi view to check the overall prototype — only validated rules are reflected. She switches to Code view, selects an element, and adds a styling rule through the code editor. All rules, regardless of entry point, reference the same components and appear in the same Library.

Ed's second turn. Ed returns and sees Danna's suggestions. He validates the layout rule then goes to Hifi to update (Fig.2.d), then the layout updates immediately with the horizontal arrangement. On another suggestion, he clicks to revise, edits the rule text, formalizes his revision, and suggests it back. A thread (Fig.2.g) now shows the negotiation history: Ed's original, Danna's proposal, Ed's revision. He sketches a new element, scans it into the Library as Daily Special, and suggests a rule for it.

Danna's second turn. Danna reads Ed's revision thread, validates the revised rule, and builds on his new component with additional styling rules. The Hifi prototype now reflects a complete page — negotiated through four async turns, with every decision traceable through the Library's provenance dots.

4 Design

CREPE supports asynchronous cross-expertise collaborative prototyping by enabling stakeholders to construct, interpret, and refine a shared interface artifact without real-time coordination.

Grounded in our four design goals, *CREPE* organizes collaboration around components and rules as the central representation: components anchor contributions in a shared structure, while rules capture intent, behavior, and decisions as persistent, actionable artifacts. The artifact can be viewed and evolved across multiple representations, supporting alignment and preserving decision history over time.

The following sections describe how *CREPE* grounds contributions in a shared language of components and rules (DG1), supports interpretation across multiple representations (DG2), and structures alignment and provenance through the suggest/validate lifecycle (DG3, DG4).

4.1 Components and Rules as Shared Language

To support collaboration across expertise boundaries, contributions must be translated into a shared and interpretable structure (DG1). *CREPE* achieves this in two layers: sketches are decomposed into discrete named components, and rules attached to those components capture design intent as persistent, actionable artifacts.

4.1.1 Component-Centric Grounding. Before sketching, stakeholders establish shared context through two mechanisms: a common goal that describes the project's purpose, and shared ideas that capture features, constraints, or preferences as persistent chips visible to both participants. These serve as grounding context for the AI and function as a lightweight ideation channel, allowing stakeholders to align on intent before committing to spatial arrangements.

CREPE captures sketches and decomposes them into discrete named components as the central representational unit. After a stakeholder sketches, the system interprets the drawing into discrete components (e.g., header, navigation bar, search field) with naming that draws on the common goal and shared ideas (e.g., a sketched card is named "idea information card" rather than "card"). The components persist in a shared library, where users can revisit and refine their associated sketches, update their classifications, and reinterpret them (Fig. 3).

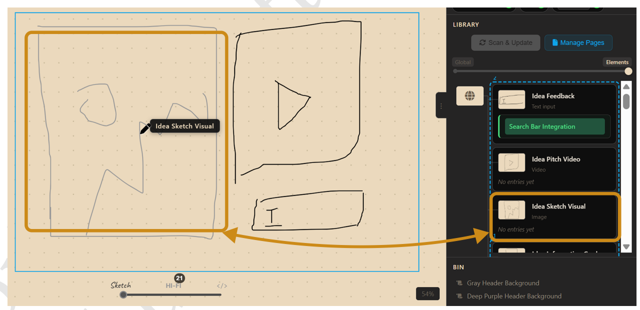

Figure 3: Scanned contextualized component in sketch and in library.

4.1.2 Rules as Annotation, Specification, and Interface State. Building on the shared component grounding, *CREPE* centers collaboration around rules. Users attach rules to recognized components to specify their behavior, functionality, or design intent, while also using them as lightweight annotations. They can be used independently or together with highlight strokes to provide contextual references. Rules function simultaneously as annotation, executable specification, and interface state.

Users select a component (Fig. 1.a) and author a rule (Fig. 1.b), optionally grounding it with sketch highlights. Users can then formalize the rules (Fig. 1.c), translating informal inputs into structured rules while standardizing how intent is expressed, enabling stakeholders with different expertise to understand and build on each other's work. Users can also preview the Hifi rendering of a component with its applied rules (Fig. 1.h). Once suggested (Fig. 1.d) or validated, the rule persists as part of the shared artifact and can be revisited and modified (Fig. 1.e) across asynchronous turns in the component library.

4.1.3 AI as Cross-Expertise Translator. In *CREPE*, the AI does not generate designs on behalf of a single user; instead, it operates at the boundary between contributors. When a user authors a rule, the formalization step rewrites it into standardized design language calibrated for the absent collaborator: the prompt receives the profiles of both stakeholders and the shared project context, distinguishes the contributing user (author) from the receiving user (audience), and produces a structured statement that the audience can interpret

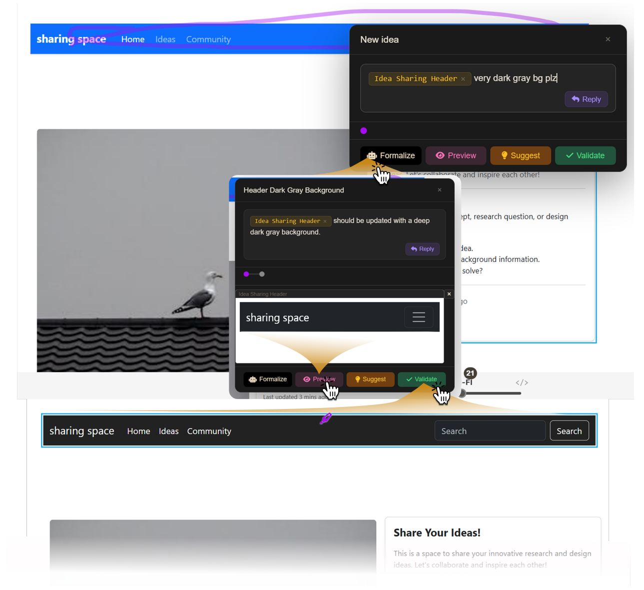

Figure 4: Interacting in the Hifi view. Highlighting elements and validating a rule triggers partial re-rendering of the affected component while other components remain untouched.

and act on without additional clarification. This positions AI as a translator between local vocabularies rather than a generator of finished artifacts. The same mechanism serves the author as well: by seeing their informal input rewritten in design terms, contributors can verify whether their intent survived the translation before committing it to the shared artifact. Formalization thus operates as a bidirectional calibration step, clarifying the contribution for the audience, the system, and the author simultaneously.

4.2 One Structure, Multiple Projections

Rather than treating sketch, Hifi view, and code as separate stages of one generation pipeline, *CREPE* maintains a single underlying structure grounded in components and rules, from which multiple representations are derived (DG2). The Hifi interface is incrementally constructed from sketches and validated rules, ensuring consistency across views. In the Hifi view, each component remains mapped to the component library and individually manipulable, allowing users to directly move and edit it.

Users define page boundaries in the sketch by selecting regions of components, which can then be rendered as separate Hifi views. This organizes the continuous design space into discrete interface outputs.

4.2.1 Expertise-Specific Entry Points Produce the Same Result. In *CREPE*, the prototyping entry point is expertise-specific but the output is shared: a designer highlighting something in sketch and writing a rule "group these together", a developer selecting <nav> in code and writing "wrap in flex container" (Fig. 5), and an end-user right-clicking in HiFi and writing "make this sticky" (Fig. 4) all produce the same data structure: a rule with {{component}}

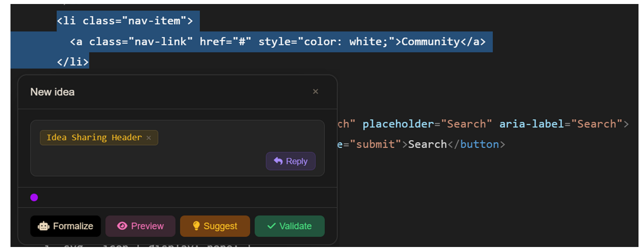

Figure 5: Interacting in the code view. Selecting a code snippet maps it to the corresponding component, enabling the same rule editing workflow as in sketch and Hifi views.

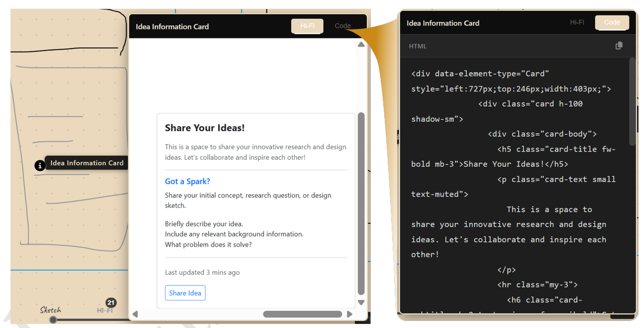

Figure 6: Users can activate the Magic Lens to inspect a component's Hifi rendering and corresponding code. Edits made in the code view are immediately reflected in the Hifi lens.

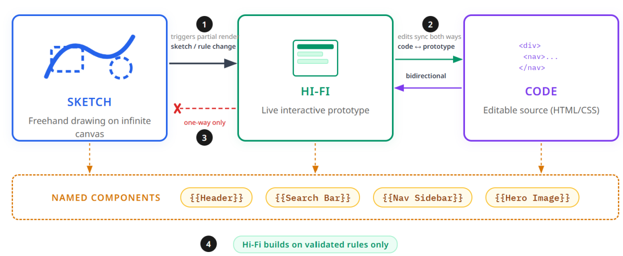

Figure 7: *CREPE* three-view architecture: (1) component-level partial rendering; (2) shared components link HiFi and code bidirectionally; (3) sketch provides one-way structure to Hifi; (4) only validated rules affect Hifi.

chips, stored in the same library and visible to everyone. This enables each stakeholder to interpret and manipulate the artifact in their preferred representation while remaining aligned on a shared structure.

4.2.2 Component-Level Preview and Magic Lens. To support local inspection and iteration, each rule editor provides an inline preview of its effect on the interface (Fig. 1.h). Inspired by see-through interfaces [3], the Magic Lens enables users to inspect component-level projections by hovering over components in sketch mode, revealing their current Hifi rendering and corresponding code based on the latest validated rules (Fig. 6).

4.3 Alignment and Provenance through Suggest/Validate Lifecycle

Asynchronous collaboration requires participants to make decisions, interpret others' contributions, and continue work across turns with shared context (DG3, DG4). *CREPE* structures this around the suggest/validate lifecycle of rules.

4.3.1 Temporally Stacked Rules for Each Component. Rules begin as open suggestions reflecting individual interpretations, and can be progressively refined and validated into agreed-upon decisions. Rules are organized per component as a temporally ordered sequence that distinguishes open suggestions from validated decisions. Suggestions (yellow) appear above validated rules (green), reflecting their provisional status, while validated rules form the stable specification below (Fig. 1.e). The lifecycle is not strictly linear. Users can open a validated rule to author a new suggestion on top of it, effectively reopening a settled decision for renegotiation. They can also respond to a suggestion with a counter-suggestion rather than validating or rejecting it. When a user does not want to commit to a suggestion or validation, they can leave comments within a rule card to express questions or uncertainties without advancing the rule's state. This allows negotiation to proceed at varying levels of commitment within the same structure.

4.3.2 Discussion and Creation History Thread for Each Rule. Each rule maintains an attached creation history thread that records how the rule was proposed, interpreted, and revised (Fig. 1.g). Stakeholders can discuss and refine rules through threaded comments (Fig. 1.f), localizing negotiation to specific components and rules.

By coupling each rule with its construction history, the system makes the evolution of decisions explicit and supports continuity across asynchronous turns, allowing stakeholders to build on prior contributions without losing context.

5 Implementation

CREPE is built around a shared component model that synchronizes sketch, Hi-Fi, and code views (Fig.7). Sketches serve as the persistent source of spatial intent, while Hi-Fi and code are generated from validated rules attached to components. Updates are propagated selectively: changes to a component trigger partial re-rendering in Hi-Fi, while Hi-Fi and code remain bidirectionally linked. Only validated rules affect the prototype, allowing the system to distinguish between tentative suggestions and committed decisions. *CREPE* is built with Flask-SocketIO and a JavaScript frontend; workspace state is synchronized via WebSockets and persisted as JSON files on disk.

The AI pipeline uses the Gemini API (`gemini-2.0-flash` for lightweight text tasks, `gemini-3-flash-preview` for vision and structured generation) across four stages. *Identify* receives a canvas snapshot and stroke metadata, returning bounding boxes and classification labels for detected UI components while using existing names to avoid duplication. *Interpret* takes the canvas image, component context, user profiles, shared ideas, and gestural annotations, then generates a structured rule statement for the absent collaborator. *Formalize* rewrites informal rule text into standardized design language while preserving `{{component}}` chip references. *Preview* generates or incrementally updates Bootstrap 5.3 HTML for individual components via Gemini function calling with a structured schema that guarantees each rendered element carries a `data-group-id` attribute linking it to its named component.

Rendering operates at component level rather than regenerating entire pages. A structured HTML builder assembles Bootstrap components from function-call parameters, and an HTML patcher applies constrained edit operations targeted by `data-group-id`, ensuring patches only modify allowed components. Pages are defined by selecting a region of components on the sketch canvas, each maintaining independent high-fidelity HTML; validating a rule triggers re-rendering only for pages containing that component. Rules are the central data structure, storing text with `{{component}}` references, author, action type (suggest or validate), linked annotations, optional high-fidelity previews, and a threaded conversation history. They are organized per component in the Library, separating suggestions from validated decisions, and can be created from sketch, high-fidelity, or code views, all producing the same underlying representation.

6 Study Methodology

We conducted a 5-day deployment study to evaluate *CREPE*. We investigate:

- How do participants translate their ideas into sketches and rules when collaborating asynchronously across expertise boundaries?
- How do designers and clients interpret and act on each other's contributions without direct, synchronous communication, and how does the AI mediator shape this process?
- How do collaborators reach agreement on design decisions through asynchronous turn-taking?
- How do participants use rule provenance (creation, changes, and discussion threads) to recalibrate design intent?

6.1 Participants

We recruited 12 participants (6 self-identified females, 6 self-identified males), organized into 6 randomly assigned pairs. Each pair comprised one UI designer and one end-user with domain expertise. UI designers had experience with interface design tools (e.g., Figma), while end-users were recruited for their domain knowledge rather than design skills. We followed our ethics board's guidelines for designing the study and presenting it to participants. All participants provided informed consent and agreed to voice and screen recordings. We anonymized the data in compliance with European privacy regulations (GDPR).

6.2 Procedure

We conducted a 5-day in-situ remote deployment with iterative researcher intervention, followed by a semi-structured interview with each participant. Each pair collaborated on an early-stage UI design for a website or application based on the client's expertise or personal interests. Pairs received a unique workspace access code and were asked to engage with the system at their convenience for at least 15 minutes per day, first reviewing their partner's contributions and then adding, modifying, validating, or commenting on elements. A shared onboarding workspace with an interactive demo, examples, and Q&A was accessible to all participants and authors throughout the study. Separate guidelines for designers and clients were iteratively refined alongside minor system adjustments based on participant feedback; updates were communicated daily by email. We maintain a complete changelog of all modifications (Appendix Table 3). The core interaction model and mechanisms under evaluation (sketching, rule authoring, suggest/validate/comment, and Hifi generation) remained unaltered; changes were limited to bug fixes, clarifications, and minor UI adjustments. Following the remote study, we conducted individual semi-structured interviews (approximately 40 minutes) either remotely or in person.

6.3 Data Collection and Analysis

We collected interaction logs (timestamps, sketch traces, actions, components, comments, and decisions), audio-recorded and transcribed all interviews, and captured screenshots of design artifacts produced by each pair.

We analyzed interview data from 11 participants[3] using reflexive thematic analysis, combining deductive and inductive approaches. Two authors independently coded transcripts based on the research questions, then reconciled discrepancies and aligned on a shared analytic lens. Codes were collaboratively refined and grouped, resulting in four themes: *Shared Artifact as Coordination Space, Expressing Stance through Suggest/Validate, Expertise Reconfiguration*, and *Fidelity Transitions as Interpretive Medium.* Quantitative interaction logs complemented the qualitative findings.

7 Results

Projects ranged from a musician portfolio website (P1) to an AI fitness coaching app (P2), a vegetable and seed marketplace (P3), a visual artist portfolio (P4), a wedding planning site (P5), and a theater archive (P6). P6's original designer dropped out during the study; C6's project was reassigned to one researcher.[4][5]

Our interaction logs show that both designers and clients engaged actively but differently (Table 1). Designers produced more component recognition scans (1,319 vs. 685), while clients authored more rules (211 vs. 76); both sketched substantially (designers 992 strokes, clients 798). Engagement varied across pairs in both iteration depth and coordination patterns. P3 completed three successive iterations on a single page, each building on the previous, and was the most active pair with 2,296 actions, 98 components, and 29 rules. C4 alone produced 1,126 actions over 9 days; C4 and D4 were both highly active, although mutual exchange remained secondary to individual output. C1's late onboarding left D1 designing alone for the first two days, while C5 and D5 completed a single round of exchange.

7.1 Shared Artifact as Coordination Space

Sketch as primary communication channel. Across all pairs, the sketch canvas served as the primary communication channel, supporting both design contributions and intent signaling. C4 found that "drawing a sketch felt more direct for communicating to the designer than editing their version," while D5 treated everything as implicitly proposed through sketches ("I thought that everything is kind of a suggestion"). C1 likened the process to musical collaboration, where participants work directly on the artifact — "cross it out and write something else", rather than exchanging feedback externally. However, rather than editing each other's work, over half the participants created parallel versions and selectively combined elements. P1 each maintained their own canvas zone; P3 iterated through three successive layouts placed adjacently; C2 reproduced his designer's header on his own pages rather than editing hers. C4 articulated the underlying social norm: "Modifying something on my own sketch doesn't feel like any offense at all. But if I have to modify someone else's sketch, there's a role reversal that makes me uncomfortable."

The missing attention provenance. The most requested feature was not better change tracking but knowing whether a partner had *seen* one's contributions. C5 wished for "a 'seen' or an 'OK' "; C3 wanted to know "if D3 has read the suggestion or not." This gap had design consequences: D3 interpreted silence on his "My Garden" feature as disinterest and removed it. C4 noticed a co-presence awareness without attention awareness when two users were online and "got a little nervous".

Improvised communication channels. Participants appropriated system features to support communication in context. D2 kept using shared ideas to communicate with the client because she found them "more straightforward" at the ideation stage (17 total shared ideas for P2, Table 2). P3 discovered text on the canvas as a primary communication channel, writing 66 text annotations (134 total across pairs, Table 2). C5 repurposed the highlighter as a "personal annotation layer", marking what was "not part of the site". Some pairs developed lightweight coordination strategies: D3 created a new page per session, C3 recombined elements across pages, and both used deletion to signal resolution.

Phase-dependent communication needs. Participants' communication protocols evolved over time, shifting from global to local: early sessions focused on project vision and aesthetic direction, mid-phase on page-level layout alternatives, and later on component-level refinements such as colors and interactions. P3 illustrates this progression: C3 began with shared ideas and an initial sketch, D3 responded with alternative layouts, and C3 selectively combined elements. By the final days, both shifted to component-level refinement. P1 showed a similar transition: C1 initially drove the design through sketches, then shifted to validating D1's proposals: "I do feel like I drove more [at first]... in a second phase, all the suggestions were more him driving."

In contrast, misalignment occurred when coordination operated at mismatched levels. P4 needed global discussion but lacked a space to "discuss about the global idea of the needs", while P2 diverged in the opposite direction, with C2 moving to component-level iteration as D2 remained in "ideation of functions ".

7.2 Expressing Stance through Suggest/Validate

Asymmetric appropriation across roles. Clients and designers appropriated suggest/validate in opposite directions. Two clients (C1, C5) validated nearly everything their designer proposed, interpreting suggestions as expert recommendations. C1 thought "that he knew better than me too. His proposals are probably good, because

[3]One designer dropped out of the study.
[4]We use C to denote client, D designer, and P pair.
[5]All quotes from non-English-speaking participants are translated by the authors.

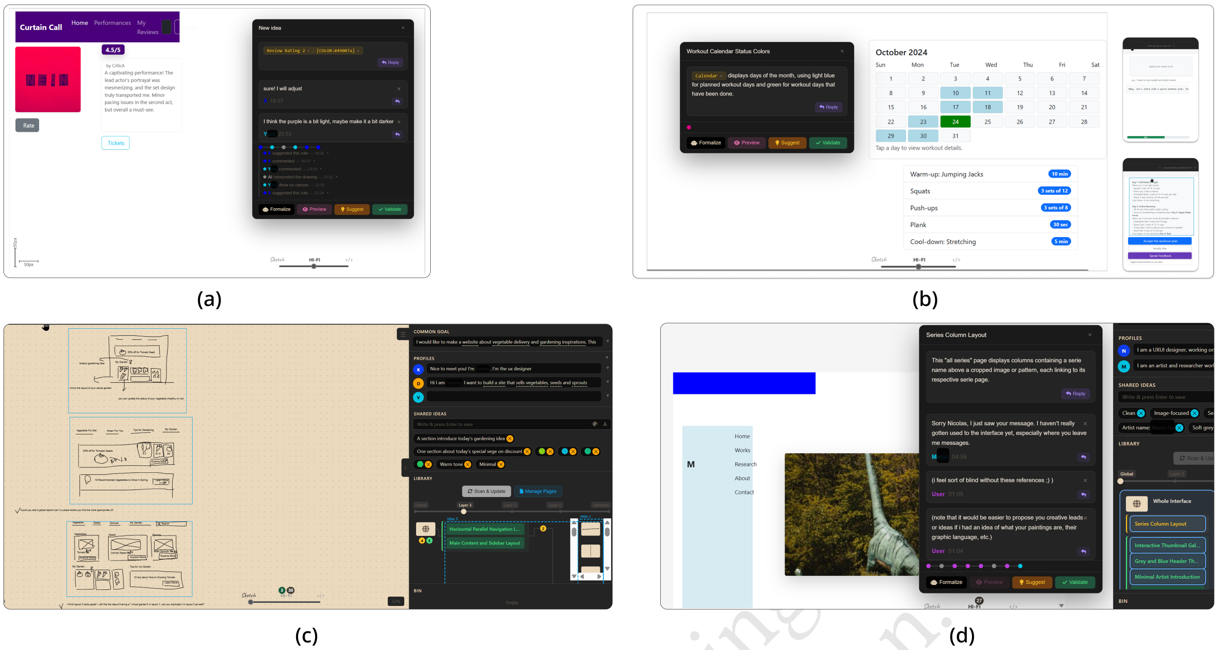

Figure 8: Participant workspace screenshots. (a) C6 and the researcher discuss a color adjustment within a rule thread. (b) C2 uses rules to specify detailed Hifi UI behavior and constructs a three-page app prototype. (c) P3 iterates three times in sketch mode, using shared ideas, sketches, text, and suggest/validate to communicate. (d) C4 and D4 engage in multiple discussions, with C4 driving implementation and refinement in Hifi.

that's what he does," while still feeling that he "drove more" while trusting the implementation. Designers, conversely, defaulted to suggesting rather than validating, framing their role as advisory. D1 noted, "I wasn't sure if it's something I could validate myself." D4 adopted a consistently deferential stance, "always click[ing] on suggest, to put it to [C4] for feedback," and at times feeling like he was "walking on eggshells." This reflects a productive tension: designers hold greater expertise but defer authority, while clients retain decision power despite less expertise.

Validation for commitment, suggestion for support. C2 and C4 both actively sketched and explored designs independently, repurposing suggest/validate as part of their individual workflows rather than as a coordination tool. In both cases, validate was primarily used to commit their own decisions rather than to signal collective agreement. C4 "mostly just validated directly" (Table 1), while C2 progressed quickly without designer feedback ("I'd just validate validate... I didn't think about it, honestly"). In contrast, suggest was interpreted by C2 as a request for assistance: "should have put suggest, 'help me,' " using it to signal uncertainty and invite input.

Negotiation beyond suggest/validate. P3 came closest to the intended lifecycle: C3 primarily suggested, while D3 validated after preview-checking. However, key negotiation occurred around features beyond the initial requirements. D3 initially proposed a virtual "My Garden" feature based on her own reasoning ("if someone wants to buy vegetable seeds, they probably have a garden"), but she removed it without feedback from C3. In a later iteration, C3 explicitly requested it via canvas text and chat; D3 then re-added it and marked it directly on the canvas. This illustrates a full negotiation cycle that unfolded both inside and outside the designed features: proposal, ambiguity, withdrawal, explicit request, implementation, confirmation.

7.3 Expertise Reconfiguration

Formalization as vocabulary bridge. Formalization was the most consciously valued feature, particularly by clients with ideas but limited design vocabulary. C1: "It puts the element terms, terms I absolutely don't have." C5: "A really good tool for making sure you're in sync between what you have in your head and what you draw." D3 named it his "favorite." The bridge was asymmetric: clients gained vocabulary they lacked, while designers used it mainly for precision. D1 found "it didn't change much; I was specific enough." However, formalization struggled with complex descriptions: C2 found it "removed three quarters of what I had asked for," and C3 wanted both versions side by side because "it's hard to see which part has changed."

Clients who gained design agency. The sketch, formalize, preview pipeline gave clients direct access to design iteration. D2 observed that *CREPE* creates "a very equalized feedback loop," unlike tools like Figma that do not actively engage non-designers. C4 pushed toward independence from designers, noting she could "be fine just working with the AI" if it understood her intent. Others exercised more selective control: C3 set high-level parameters (e.g., colors, "minimalistic" style) and observed that the system "magically picked up the color," while C6 kept input "very high level... I don't want to be micromanaging the designer." These behaviors span a spectrum from direct authorship to guided delegation.

Designers who learned client intent. Designers used *CREPE*'s artifacts to progressively infer client intent. D3 learned iteratively: early versions missed requirements ("At first I didn't see 'seeds' and 'sprouts' "), but later iterations aligned more closely. In the "My Garden" cycle, D3 removed a feature after no feedback, then restored it when explicitly requested, learning through the suggest/validate process which elements mattered. As D4 noted, the key challenge remained identifying "what the client is okay to change and what are their hard limits."

7.4 Fidelity Transitions as Interpretive Medium

Hifi as verification mirror. Participants used the Hifi view primarily to check whether the system had correctly interpreted their sketches. C2 "would draw something and right after go to hi-fi," while C3 checked it "every day... to see if it met my expectations." This verification loop also served as self-calibration: C4 realized her sketch lacked color after seeing a black-and-white output and "went back and added a light blue," while D1 used Hifi to identify missing elements.

Cross-fidelity reading of a partner's intent. Comparing sketch and Hifi helped participants reinterpret their partner's contributions, and in some cases revealed alignment that sketches alone obscured. C1 initially read D1's sketches as "very classic website, lots of buttons, a bit old-school," but the Hifi narrowed the gap: "The sketch indicates more potential differences. And when you see the Hifi, you realize it's relatively aligned." C2 noted that Hifi revealed a level of polish not visible in sketches: "Her page, when I went to Hifi, was so clean compared to mine." For D5, the Hifi was an aid for understanding the client's intent: "the Hifi view would explain some things a bit better because there is this layer of AI interpretation that is already doing some sketch interpretation for you."

Fidelity gaps as design inspiration. The gap between sketch and Hifi was not merely an error to correct; in several cases it generated new design ideas. C5 discovered accordion components she had not conceived: "I hadn't thought of dropdown menus at all. It was rather good that the AI reminded me they existed." C2 specified a calendar as a simple image; the Hifi produced an interactive calendar with clickable days, which he kept. D3 noted that even when the preview "doesn't go the way I like, sometimes I never thought about that layout before."

8 Discussion

8.1 Artifact-Centered Collaboration without Direct Communication

Participants appropriated the shared workspace as a communication medium: they wrote large text on the canvas, repurposed the highlighter as a personal annotation layer, created parallel sketches instead of editing each other's work, and used deletion to signal resolution. These locally invented practices carried communicative intent without being explicitly designed as communication channels, demonstrating how coordination emerged through the artifact itself. This aligns with Dourish and Bellotti's notion of shared feedback [9], where awareness is conveyed passively through the workspace rather than explicit messaging. Our findings extend this concept to fully asynchronous, cross-expertise collaboration, where the artifact must communicate not only what was done, but also why and for whom.

However, artifact-centered coordination exposed a key limitation: visibility of contributions does not guarantee their acknowledgment. While participants could see changes, they often lacked confirmation that their work had been noticed or understood, leading to ambiguous interpretations of silence (e.g., requests for "seen" indicators, uncertainty about removed features). This points toward a design space for richer awareness support in asynchronous systems. Our observations surface at least three distinct gaps beyond change visibility: whether a contribution was seen (C5's request for a "seen" indicator), how it was interpreted (D3 reading silence as disinterest), and whether it was accepted or rejected (C3 wanting to know if a suggestion had been read). Future systems could address these gaps explicitly, though our study does not provide a complete taxonomy.

In addition, coordination needs shifted over time as work moved from global intent to component-level refinement. Collaboration progressed when interaction matched this granularity (e.g., evolving from shared ideas to detailed refinement), and broke down when it did not (e.g., global discussions constrained by component-level tools). These findings suggest that while shared artifacts can effectively support communication without direct interaction, they benefit from lightweight acknowledgment mechanisms and support for phase-appropriate, multi-level coordination.

8.2 Coordination through Lightweight Cues

The suggest–validate mechanism, designed as a rule state transition, was consistently appropriated as a social signaling system. Clients used validation to assert decisions, while designers used suggestion to propose and defer. The same binary action conveyed trust, independence, uncertainty, or caution depending on context (e.g., C1 validating "absolutely everything," C2 "validate validate validate," D4 "walking on eggshells," and C2 using suggest as "help me"). At the same time, coordination can embed multiple cues combining different system features. In P3's "My Garden" cycle (proposal, silence, withdrawal, explicit request, and re-implementation) negotiation unfolded across suggest/validate, sketching, chat, and deletion.

These findings suggest that in asynchronous cross-expertise settings where direct communication is absent, artifact state transitions become the primary channel through which social dynamics (trust, deference, authority, uncertainty) are expressed. This reframes the design challenge: the question is not how to add communication features to an asynchronous workspace, but how to recognize that every mechanism acting on the shared artifact is already a communication feature. Designing for this means accepting that coordination will be assembled by users from whatever stateful cues the system provides, and that the system's role is to keep those cues legible across expertise boundaries.

8.3 Dynamic Redistribution of Expertise and Authority

The *CREPE* study was designed with fixed roles, designer and client, but participants dynamically reconfigured who held design authority and how expertise flowed. Some clients (e.g., C6) kept input at a high level and relied on their designer for implementation, while others (e.g., C2, C4) actively sketched, formalized, and validated their own work, effectively taking on designer-like roles. D2 described the dynamic as "equalized" compared to collaborative design tools that do not actively encourage non-designers to act.

By allowing clients to sketch loosely, formalize into design vocabulary they did not previously possess, and preview the result immediately, the system gave non-experts a path into design iteration that did not require learning a professional tool. Formalization added to this value since it lent non-designers design language temporarily at the moment of use.

Within this setting, expertise and authority became decoupled. Designers continued to contribute technical and interpretive expertise, for example D3 learning iteratively which elements mattered through the "My Garden" negotiation cycle, yet often deferred decision authority to clients. As seen in D1 and D4's hesitation to validate and C4's strong authorial vision, this dynamic was not a breakdown but a structured tension between expertise and authority. The suggest–validate mechanism made this asymmetry visible and negotiable, enabling participants to adjust not only the design but also their roles over time.

8.4 Diagnostic Fidelity Transitions

Participants used fidelity transitions not to produce final outputs, but to diagnose and refine their understanding. The HiFi view functioned as a shared interpretive layer, helping participants identify missing details (C4), clarify alignment beyond sketches (C1), and better interpret intent through AI-mediated rendering (D5).

Rather than serving as the end of a generation pipeline, the prototype acted as a third perspective, an AI-mediated reading of the shared artifact against which participants could check their assumptions. This made it possible to resolve interpretation gaps that annotation or discussion alone could not address. For example, C1 only recognized that his version and the designer's were "relatively aligned" after viewing the Hifi.

Critically, the value of generation lay in what it revealed (missing details, unstated assumptions, and misalignment) in early-stage design, not in what it produced. This distinguishes *CREPE* from tools such as Google Stitch[6], where AI generates polished interfaces from a single user's prompt. In *CREPE*, generation serves cross-expertise mediation: the output is not a deliverable but a diagnostic surface that both stakeholders can read, question, and act on from their own perspective.

8.5 Limitations and Future Work

Our study prioritized ecological validity over experimental control: rather than comparing *CREPE* with existing tools under fixed conditions, we conducted an in-situ deployment that captures coordination practices as they emerge. A comparative evaluation could complement these findings by isolating the effect of specific mechanisms (e.g., suggest/validate, formalization) against established workflows.

The five-day deployment captures the emergence of coordination strategies in early-stage design but not their longer-term evolution. A longitudinal deployment is needed to understand how artifact-centered coordination develops over time.

The study is limited to pairs, which led to relatively stable interaction patterns (e.g., one suggesting and the other validating, or both using suggest/validate independently). While this setup isolates the designer-client boundary central to our design goals, real-world projects involve multiple stakeholders with overlapping concerns. Other configurations (e.g., designer-developer) may introduce different dynamics that we have not explored. Extending the coordination model to multi-stakeholder settings is a direction for future work, where rules are validated by relevant roles and status reflects approved perspectives.

9 Conclusion

This paper presents a design approach for asynchronous cross-expertise collaboration in which sketches and rules function simultaneously as expressive design actions and persistent coordination mechanisms, enabling collaborators to construct mutual understanding without synchronous interaction. We contribute: (1) four design guidelines for asynchronous cross-expertise collaboration, grounded in boundary object theory, that address translation, interpretation, alignment, and provenance; (2) *CREPE*, a system that instantiates these guidelines through an AI-mediated interaction model where sketches and rules serve as a shared, executable language across collaborators; (3) empirical findings from a 5-day deployment with 6 client/designer pairs showing how participants coordinate through the shared artifact, appropriate system features as social signals, dynamically redistribute expertise and authority, and use fidelity transitions as diagnostic tools rather than generation endpoints. This work highlights the potential of reframing AI in prototyping systems from a generator serving individual users to a mediator supporting human-human collaboration.

References

[1] Brian P. Bailey, Joseph A. Konstan, and John V. Carlis. 2001. DEMAIS: designing multimedia applications with interactive storyboards. In *Proceedings of the Ninth ACM International Conference on Multimedia* (Ottawa, Canada) *(MULTIMEDIA '01)*. Association for Computing Machinery, New York, NY, USA, 241–250. doi:10.1145/500141.500179

[2] Jon-Tait Beason, Ruijia Cheng, Eldon Schoop, and Jeffrey Nichols. 2026. Athena: Intermediate Representations for Iterative Scaffolded App Generation with an LLM. In *Proceedings of the 31st International Conference on Intelligent User Interfaces (IUI '26)*. Association for Computing Machinery, New York, NY, USA, 284–306. doi:10.1145/3742413.3789133

[3] Eric A. Bier, Maureen C. Stone, Ken Pier, William Buxton, and Tony D. DeRose. 1993. Toolglass and magic lenses: the see-through interface. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques* (Anaheim, CA) *(SIGGRAPH '93)*. Association for Computing Machinery, New York, NY, USA, 73–80. doi:10.1145/166117.166126

[4] Bill Buxton. 2007. *Sketching User Experiences: Getting the Design Right and the Right Design.* Morgan Kaufmann.

[5] Carrie J Cai, Samantha Winter, David Steiner, Lauren Wilcox, and Michael Terry. 2021. Onboarding Materials as Cross-functional Boundary Objects for Developing AI Assistants. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI EA '21)*. Association for Computing Machinery, New York, NY, USA, Article 43, 7 pages. doi:10.1145/3411763.3443435

[6] Paul R. Carlile. 2002. A Pragmatic View of Knowledge and Boundaries: Boundary Objects in New Product Development. *Organization Science* 13, 4 (2002), 442–455.

[6] https://stitch.withgoogle.com/

doi:10.1287/orsc.13.4.442.2953
[7] Weihao Chen, Chun Yu, Yukun Wang, Meizhu Chen, Yipeng Xu, and Yuanchun Shi. 2025. EchoMind: Supporting Real-time Complex Problem Discussions through Human-AI Collaborative Facilitation. *Proc. ACM Hum.-Comput. Interact.* 9, 7, Article CSCW406 (Oct. 2025), 38 pages. doi:10.1145/3757587
[8] John Joon Young Chung and Eytan Adar. 2023. Artinter: AI-powered Boundary Objects for Commissioning Visual Arts. In *Proceedings of the 2023 ACM Designing Interactive Systems Conference* (Pittsburgh, PA, USA) *(DIS '23)*. Association for Computing Machinery, New York, NY, USA, 1997–2018. doi:10.1145/3563657.3595961
[9] Paul Dourish and Victoria Bellotti. 1992. Awareness and coordination in shared workspaces. In *Proceedings of the 1992 ACM Conference on Computer-Supported Cooperative Work* (Toronto, Ontario, Canada) *(CSCW '92)*. Association for Computing Machinery, New York, NY, USA, 107–114. doi:10.1145/143457.143468
[10] W. Keith Edwards, Elizabeth D. Mynatt, Karin Petersen, Mike J. Spreitzer, Douglas B. Terry, and Marvin M. Theimer. 1997. Designing and implementing asynchronous collaborative applications with Bayou. In *Proceedings of the 10th Annual ACM Symposium on User Interface Software and Technology* (Banff, Alberta, Canada) *(UIST '97)*. Association for Computing Machinery, New York, NY, USA, 119–128. doi:10.1145/263407.263530
[11] William W. Gaver, Jacob Beaver, and Steve Benford. 2003. Ambiguity as a Resource for Design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '03)*. ACM, 233–240. doi:10.1145/642611.642653
[12] Vinod Goel. 1995. *Sketches of Thought.* MIT Press. doi:10.7551/mitpress/6270.001.0001
[13] Carl Gutwin and Saul Greenberg. 2002. A Descriptive Framework of Workspace Awareness for Real-Time Groupware. *Computer Supported Cooperative Work (CSCW)* 11, 3 (2002), 411–446. doi:10.1023/A:1021271517844
[14] Björn Hartmann, Sean Follmer, Antonio Ricciardi, Timothy Cardenas, and Scott R. Klemmer. 2010. d.note: revising user interfaces through change tracking, annotations, and alternatives. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. Association for Computing Machinery, New York, NY, USA, 493–502. doi:10.1145/1753326.1753400
[15] Björn Hartmann, Loren Yu, Abel Allison, Yeonsoo Yang, and Scott R. Klemmer. 2008. Design as Exploration: Creating Interface Alternatives through Parallel Authoring and Runtime Tuning. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology (UIST '08)*. ACM, 91–100. doi:10.1145/1449715.1449732
[16] Kathryn Henderson. 1991. Flexible Sketches and Inflexible Data Bases: Visual Communication, Conscription Devices, and Boundary Objects in Design Engineering. *Science, Technology, & Human Values* 16, 4 (1991), 448–473. doi:10.1177/016224399101600402
[17] Kasper Hornbæk, Per Ola Kristensson, and Antti Oulasvirta. 2025. Collaboration. In *Introduction to Human-Computer Interaction.* Oxford University Press. arXiv:https://academic.oup.com/book/0/chapter/528999722/chapter-pdf/64021510/oso-9780192864543-chapter-8.pdf doi:10.1093/oso/9780192864543.003.0008
[18] Victoria Jackson, André van der Hoek, and Rafael Prikladnicki. 2022. Collaboration tool choices and use in remote software teams: emerging results from an ongoing study. In *Proceedings of the 15th International Conference on Cooperative and Human Aspects of Software Engineering* (Pittsburgh, Pennsylvania) *(CHASE '22)*. Association for Computing Machinery, New York, NY, USA, 76–80. doi:10.1145/3528579.3529171
[19] Tae Soo Kim, Seungsu Kim, Yoonseo Choi, and Juho Kim. 2021. Winder: Linking Speech and Visual Objects to Support Communication in Asynchronous Collaboration. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 453, 17 pages. doi:10.1145/3411764.3445686
[20] Scott R. Klemmer, Mark W. Newman, Ryan Farrell, Mark Bilezikjian, and James A. Landay. 2001. The designers' outpost: a tangible interface for collaborative web site. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology* (Orlando, Florida) *(UIST '01)*. Association for Computing Machinery, New York, NY, USA, 1–10. doi:10.1145/502348.502350
[21] James A. Landay and Brad A. Myers. 1995. Interactive Sketching for the Early Stages of User Interface Design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '95)*. ACM, New York, NY, USA, 43–50. doi:10.1145/223904.223910
[22] Charlotte P. Lee. 2007. Boundary Negotiating Artifacts: Unbinding the Routine of Boundary Objects and Embracing Chaos in Collaborative Work. *Computer Supported Cooperative Work (CSCW)* 16, 3 (2007), 307–339. doi:10.1007/s10606-007-9044-5
[23] Mary D. P. Leland, Robert S. Fish, and Robert E. Kraut. 1988. Collaborative document production using quilt. In *Proceedings of the 1988 ACM Conference on Computer-Supported Cooperative Work* (Portland, Oregon, USA) *(CSCW '88)*. Association for Computing Machinery, New York, NY, USA, 206–215. doi:10.1145/62266.62282
[24] Boyu Li, Linjie Qiu, Duotun Wang, Qianxi Liu, Ryo Suzuki, Mingming Fan, and Zeyu Wang. 2025. DesignMemo: Integrating Discussion Context into Online Collaboration with Enhanced Design Rationale Tracking. *Proc. ACM Hum.-Comput. Interact.* 9, 7, Article CSCW398 (Oct. 2025), 32 pages. doi:10.1145/3757579
[25] Ryan Li, Yanzhe Zhang, and Diyi Yang. 2024. Sketch2Code: Evaluating Vision-Language Models for Interactive Web Design Prototyping. arXiv:2410.16232 [cs.CL] https://arxiv.org/abs/2410.16232
[26] James Lin, Mark W. Newman, Jason I. Hong, and James A. Landay. 2000. DENIM: Finding a Tighter Fit Between Tools and Practice for Web Site Design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '00)*. ACM, New York, NY, USA, 510–517. doi:10.1145/332040.332486
[27] James Lin, Michael Thomsen, and James A. Landay. 2002. A visual language for sketching large and complex interactive designs. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Minneapolis, Minnesota, USA) *(CHI '02)*. Association for Computing Machinery, New York, NY, USA, 307–314. doi:10.1145/503376.503431
[28] Yuwen Lu, Alan Leung, Amanda Swearngin, Jeffrey Nichols, and Titus Barik. 2025. Misty: UI Prototyping Through Interactive Conceptual Blending. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. Association for Computing Machinery, New York, NY, USA, Article 1108, 17 pages. doi:10.1145/3706598.3713924
[29] Nicolas Mangano, Alex Baker, Mitch Dempsey, Emily Navarro, and André van der Hoek. 2010. Software Design Sketching with Calico. In *Proceedings of the 25th IEEE/ACM International Conference on Automated Software Engineering (ASE '10)*. ACM, New York, NY, USA, 23–32. doi:10.1145/1858996.1859003
[30] Christine M. Neuwirth, David S. Kaufer, Ravinder Chandhok, and James H. Morris. 1990. Issues in the design of computer support for co-authoring and commenting. In *Proceedings of the 1990 ACM Conference on Computer-Supported Cooperative Work* (Los Angeles, California, USA) *(CSCW '90)*. Association for Computing Machinery, New York, NY, USA, 183–195. doi:10.1145/99332.99354
[31] Gary M. Olson and Judith S. Olson. 2000. Distance Matters. *Human-Computer Interaction* 15, 2-3 (2000), 139–178. doi:10.1207/S15327051HCI1523_4
[32] Soya Park, April Yi Wang, Ban Kawas, Q. Vera Liao, David Piorkowski, and Marina Danilevsky. 2021. Facilitating Knowledge Sharing from Domain Experts to Data Scientists for Building NLP Models. In *Proceedings of the 26th International Conference on Intelligent User Interfaces* (College Station, TX, USA) *(IUI '21)*. Association for Computing Machinery, New York, NY, USA, 585–596. doi:10.1145/3397481.3450637
[33] Susan Leigh Star and James R. Griesemer. 1989. Institutional Ecology, 'Translations' and Boundary Objects: Amateurs and Professionals in Berkeley's Museum of Vertebrate Zoology, 1907–39. *Social Studies of Science* 19, 3 (1989), 387–420. doi:10.1177/030631289019003001
[34] Hariharan Subramonyam, Colleen Seifert, Priti Shah, and Eytan Adar. 2020. texSketch: Active Diagramming through Pen-and-Ink Annotations. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–13. doi:10.1145/3313831.3376155
[35] Sarah Suleri, Vinoth Pandian Sermuga Pandian, Svetlana Shishkovets, and Matthias Jarke. 2019. Eve: A Sketch-based Software Prototyping Workbench. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI EA '19)*. Association for Computing Machinery, New York, NY, USA, 1–6. doi:10.1145/3290607.3312994
[36] Rishi Vanukuru, Payod Panda, Xinyue Chen, Ava Elizabeth Scott, Lev Tankelevitch, and Sean Rintel. 2025. Designing Interfaces that Support Temporal Work Across Meetings with Generative AI. In *Proceedings of the 2025 ACM Designing Interactive Systems Conference (DIS '25)*. Association for Computing Machinery, New York, NY, USA, 3600–3620. doi:10.1145/3715336.3735833
[37] Ruotong Wang, Lin Qiu, Justin Cranshaw, and Amy X. Zhang. 2024. Meeting Bridges: Designing Information Artifacts that Bridge from Synchronous Meetings to Asynchronous Collaboration. *Proc. ACM Hum.-Comput. Interact.* 8, CSCW1, Article 35 (April 2024), 29 pages. doi:10.1145/3637312
[38] Jason Wu, Yi-Hao Peng, Xin Yue Amanda Li, Amanda Swearngin, Jeffrey P Bigham, and Jeffrey Nichols. 2024. UIClip: A Data-driven Model for Assessing User Interface Design. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology (UIST '24)*. Association for Computing Machinery, New York, NY, USA, 1–16. doi:10.1145/3654777.3676408
[39] Haijun Xia, Tony Wang, Aditya Gunturu, Peiling Jiang, William Duan, and Xiaoshuo Yao. 2023. CrossTalk: Intelligent Substrates for Language-Oriented Interaction in Video-Based Communication and Collaboration. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology* (San Francisco, CA, USA) *(UIST '23)*. Association for Computing Machinery, New York, NY, USA, Article 60, 16 pages. doi:10.1145/3586183.3606773

A Appendix

A.1 Supplementary Images

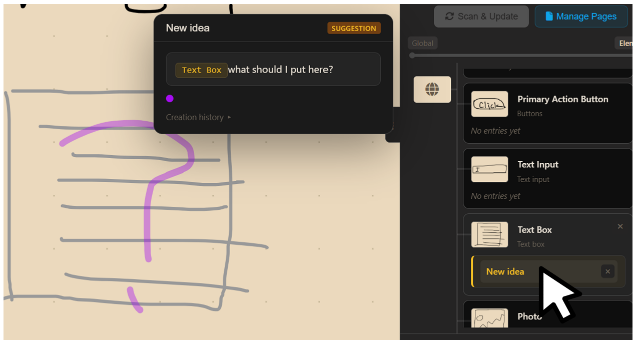

Figure 9: Hovering over a rule in the library traces its associated component, highlights its annotation, and displays a brief summary of the rule.

A.2 Quantitative Interaction Log Data from Study

Table 1: Collaboration activity metrics across 6 pairs over the 5-day deployment. Active span indicates the number of calendar days between a pair's first and last logged action; the study protocol spanned 5 days, but some pairs started late or extended slightly beyond.

Pair	Project	Active Span (Days)	Components	Rules	Pages	Role	Actions	Suggested	Validated
P1	Musician Portfolio	7	37	10	2	D1	664	5	0
						C1	134	0	5
P2	AI Fitness App	6	42	38	7	D2	176	0	1
						C2	963	2	31
P3	Vegetable Marketplace	6	98	29	3	D3	1615	3	14
						C3	681	8	4
P4	Artist Portfolio	9	62	28	7	D4	200	2	0
						C4	1126	1	23
P5	Wedding Planning	5	31	8	3	D5	383	0	2
						C5	40	3	2
P6	Theater Archive	6	12	10	2	D6	390	2	7
						C6	125	0	1
Total							**6497**	**26**	**90**

Table 2: Additional interaction metrics across pairs.

Pair	Sessions	Temporal Overlap (min)	Canvas Texts	Component Renames	Shared Ideas	Revisions of actions(%)
P1	55	119	13	18	3	14
P2	17	0	11	18	17	31
P3	113	0.4	66	10	8	15
P4	39	0	27	13	7	20
P5	15	0	17	4	7	20
P6	47	42	0	8	7	10

A.3 System Modification Logs During Study

Table 3: System modifications during the deployment period.

Day	Change	Category
D2	Hi-Fi rendering with user input images	Bug fix
D2	Chat feature added	Feature addition
D2	Page deletion with content cleanup	UI adjustment
D2	Chat notification fix	Bug fix
D3	Image persistence fix, code-to-HiFi sync	Bug fix
D3	Improved formalization and enrichment prompts	Refinement
D3	Page sync fix	Bug fix

A.4 Study Demographics Data

Table 4: Participant demographics. Each pair consisted of one designer and one client with domain expertise.

UI Designers				Clients			
ID	**Gender**	**Age**	**Profession**	**ID**	**Gender**	**Age**	**Profession**
D1	F	20–29	UX Designer	C1	M	30–39	Musician
D2	F	30–39	UX Designer, Researcher	C2	M	20–29	Engineer
D3	F	20–29	UX Designer, Researcher	C3	M	20–29	ASIC Engineer, Musician
D4	M	40–49	UX/UI Designer	C4	F	30–39	Artist
D5	M	20–29	3D & UX Designer	C5	F	20–29	Cultural Coordinator
D6	M	20–29	UX Designer	C6	F	20–29	ML Engineer