iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

AGENTIC UX: AN END-TO-END AI FRAMEWORK FOR RAPID UX DEVELOPMENT

B. Karthikeya, M. Rajeshwari, G. Divyanjali
Final Year Students, Department of Computer Science Engineering (Data Science),
J.B. Institute of Engineering and Technology (UGC Autonomous), Hyderabad, Telangana, India.

ABSTRACT
What happens when words become interfaces? A smart chain of eight AI helpers turns plain descriptions into working app screens. Not one piece gets lost along the way. Product thoughts start as talk, then move step by step - shaped by separate minds built for specific jobs. One figures out who will use it. Another maps how people navigate. Someone writes detailed plans while another sketches rough layouts. Then colors, buttons, spacing appear. Every stage locks in information so nothing vanishes between steps. While teams usually pass files across different apps - losing meaning each time - this flow keeps everything together. The result travels smoothly from concept to ready-made design. Behind it all, clean data passes forward like notes in a relay race. No gaps. No repeats. Just progress. Live previews show React and Tailwind CSS parts right inside a frame on screen. Instead of starting from scratch, users can upload their own PRD files to skip early steps. Every output gets bundled - specs, sketches, visuals, and working code - all ready to download later. A suggested add-on lets people view designs through phone cameras using WebXR tech around them. Checks based on ten standard usability rules happen automatically every time something builds. No failures appeared during twenty-nine separate checks covering speed, function, teamwork, and real-user tests. Even with nine smart tools passing data back and forth, responses come under half a minute always.

Keywords:
Multi Agent AI, UX Automation, React Code Generation, Prompt Engineering, PRD Generation, Persona Generation, Wireframe Generation, UX Validation, WebXR, Augmented Reality, NextJS, TailwindCSS, LLM, Product Development, AI Design.

INTRODUCTION

Building a digital product usually means bringing together different people - product managers work alongside researchers, designers team up with coders. One step follows another: needs get pinned down first, then comes studying users, shaping how things feel, building screens, testing it all. These tasks rely on separate tools, handled one after another by distinct experts working in sequence. Even if parts of it function well, this method brings complications - shifting focus across phases slows momentum, effort piles up, timelines stretch out, skilled people become essential. Progress often hinges heavily on who is available and how smoothly they sync. Limited resources make it especially tough for new ventures, solo builders, compact dev teams. Fast progress in large language models, tools that create content, also systems with multiple working parts has opened doors to automate pieces of how products are built. Right now, available software can handle separate tasks in designing, like writing code or building screens. Yet a single tool that manages every part of the user experience journey does not exist yet. To address this gap, we present Agentic UX - a complete system using coordinated teams of smart components to run the full UX process from start to finish. It uses eight specialized helpers, each focused on one phase: understanding goals, creating user profiles, structuring experiences, defining features, sketching layouts, making interfaces, checking ease of use, then delivering final files. Working step by step keeps agents on track, their inputs shaped, results clear. Automation ties pieces together while smooth handoffs help coders move fast. What stands out is how well these teams of AIs handle real UX tasks without stumbling. Outputs fit neatly into workflows, built for people who build software. The rhythm between actions matters just as much as the final result.

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

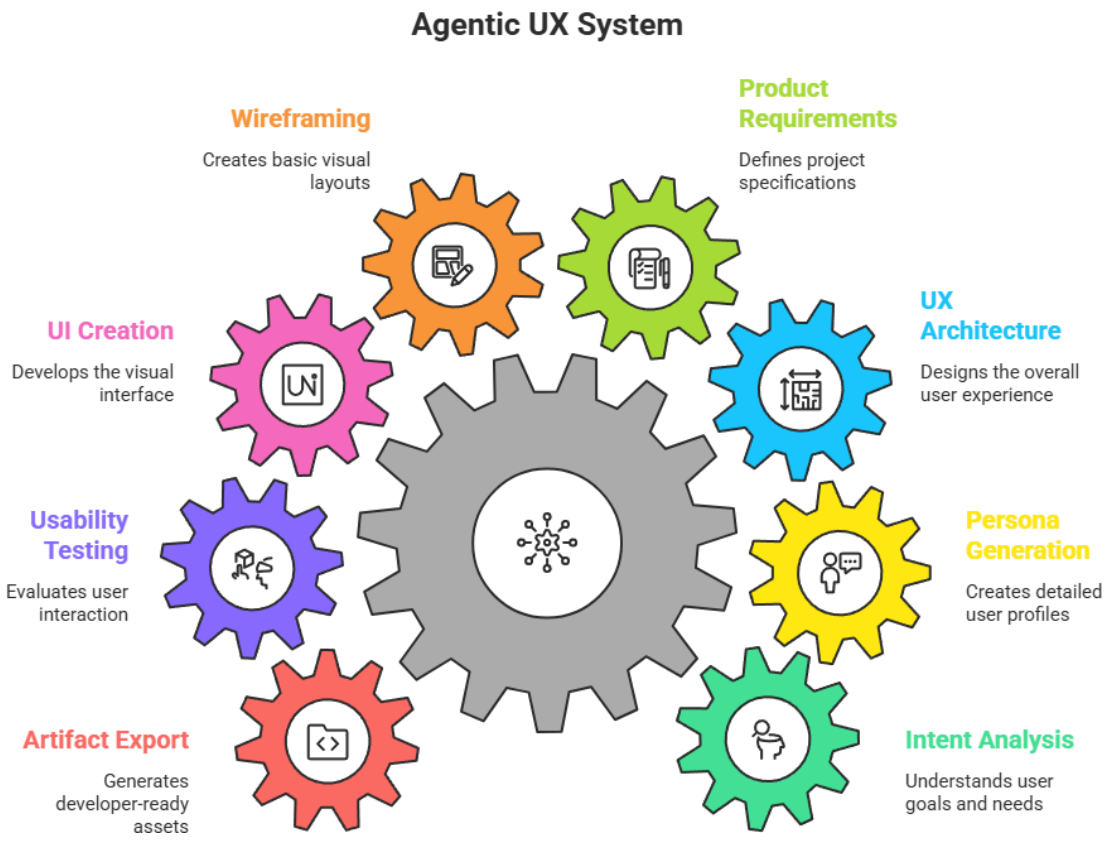


Figure 1: Agentic UX System Overview

OBJECTIVES

This study aims to build a complete system capable of turning a product concept - written in everyday words - into a working developer interface without manual steps. Starting from an idea shared through simple language, the framework moves forward by handling every stage on its own. One piece follows another until design becomes function. Through automated layers, thoughts take shape as tools. Each phase connects smoothly so nothing breaks the flow. What begins as speech ends as code.

Specific goals include the following:

- A fresh team of specialists takes shape when one agent digs into what the product truly aims to do. Another steps in, sketching who will actually use it, shaped by real behaviors. Structure emerges through a separate mind mapping how pieces fit together. One voice writes down every needed detail so nothing slips. Visual blueprints appear, drawn out step by step. Screens form next, built with clarity in mind. A different pair of eyes checks whether things feel right to users. Finally, each output gets packaged, ready for the next phase.
- From start to finish, keeping the story intact matters most when data moves across systems. Shaped formats hold details firmly so every agent understands exactly what is meant. Meaning stays clear because structure prevents drift during handoffs between parts.
- One way to kick things off: toss in product thoughts using everyday words. Another path opens when you bring a ready-made PRD document on board instead.
- A fresh layout begins with React pieces snapping together. Each piece wears its look through Tailwind classes baked right in. While you shape the code, changes show up instantly nearby. The screen updates without clicking or waiting. Design flows straight into view as elements shift on their own.

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

- Start by checking how well things work from the user's view, using basic experience guidelines. Feedback comes next - clear, organized thoughts that point out what needs fixing. Suggestions follow a logical path, shaped by real interaction issues found during review. Each step ties back to actual use, not assumptions.
- One way to move forward: pull together everything built during a project - PRDs, sketches, interface layouts, working code - and pack it into downloadable bundles. What happens next? All pieces leave in one go, pulled out neatly when needed. Files stack up behind the scenes, ready to shift locations without hassle. Picture this: every element stored along the path of development gets bundled on demand. No gaps show up later because nothing stays behind. Everything created - from early notes to final builds - gets included automatically.
- Trying out whether an AR preview tool works when built with WebXR.

METHODOLOGY

One after another, every phase in the Agentic UX flow sends the user's message to eight distinct agents, each built for a particular role. Following along, artificial intelligence units take what came before - packed neatly into a JSON format - and study it carefully before handing off their own organized result. Only at the PRD point does someone step in directly; everywhere else, machines pass insights forward without pause.

A. Intent Analysis Agent

Right off the bat, a loose product thought lands in plain words. From there, it gets shaped into clear pieces: what it's called, what issue it tackles. Though fuzzy bits might remain, they stay untouched on purpose. One piece at a time, the area it fits into comes next - no guessing. Then out pops how many parts make up its layout. Built around real input, nothing added, just sorted. What you get back? A clean, usable structure in standard data form. Even if things are unclear, they're kept as-is. Features show up based on what was said, not assumed. Complexity level appears only when mentioned directly. Where it will run matters too - that detail surfaces here. No extras sneak in during this sorting stage. At the very end stands a proper JSON bundle ready to pass along.

B. Persona Generation Agent

A fresh set of user portraits takes shape once the Intent Analysis result arrives. Two or three distinct profiles emerge, shaped by what came before. Each one carries traits such as age range, goals, frustrations, habits, a spoken line they might say, along with a five-part path showing how they discover, start using, apply daily, face hurdles, then reach their goal. These characters shift depending on who they represent and the product area involved.

C. UX Architecture Agent

Out of the insights from user goals and profile details, a structural plan takes shape. This map outlines how pages stack in levels. Navigation gets shaped with guidance on paths users follow. Journey patterns link up with key actions they take along the way. Layout choices appear alongside notes on interface pieces. Tone suggestions fit within the visual framework. Each part connects through purpose, not just placement.

D. PRD Generation Agent and Approval Gate

From earlier results, the PRD Generation Agent drafts a Product Requirements Document containing an overview at the top, followed by what needs fixing. Each feature appears with clear conditions written as Given/When/Then rules, along with how users move through them. The layout of screens shows up in brief form. Technical notes come after that. Someone reads it once done. Work halts here until confirmation arrives. Sections may rebuild separately if changes pop up later.

E. Wireframe Generation Agent

Inside the app interface, previews take shape as HTML. Starting from the sitemap, each screen gets mapped using just twenty allowed components. The Wireframe Generation Agent pulls details from both the PRD and UX structure. Instead of freeform design, strict limits keep output readable later. Because parsing matters down the line, boundaries guide every layout choice. Each tree of elements appears in JSON format before showing up visually.

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

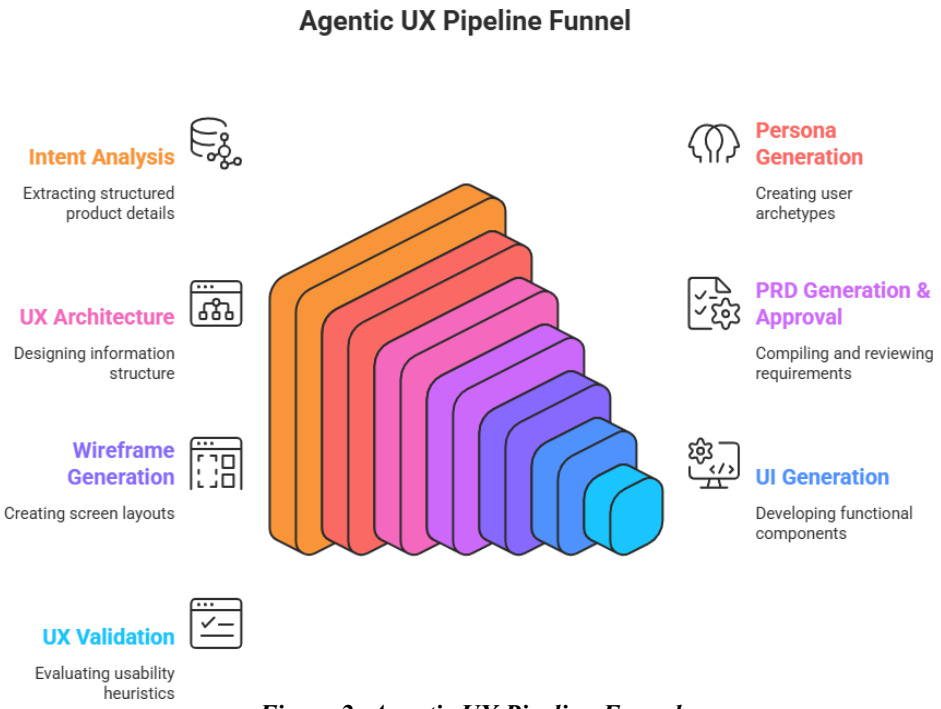


Figure 2: Agentic UX Pipeline Funnel

F. UI Generation Agent

From sketches to code, the system builds React parts styled with Tailwind straight from layout outlines. Each piece ships with a standard export so it slots into projects without fuss. Responsiveness comes built in through Tailwind's breakpoint logic. Design values stay uniform no matter how many screens get made. What pops out runs live inside a secure preview frame tucked right into view.

G. UX Validation Agent

Every so often, a flaw shows up when the UX Validation Agent checks React code against Nielsen's ten rules for usability. Found one? Then comes a clear write-up detailing where things went off track. On which screen did it happen - that detail lands right at the top. Tied to each issue is the specific heuristic broken, named by number and label. How bad is it? The report ranks seriousness plainly. Next, a straightforward explanation says why the rule was bent. Pinpoints in the actual code reveal the exact spot needing attention. Alongside sits guidance - practical steps to set it right.

H. Export Module

Ready to go. Files get packed up once you hit export. Picture the PRD showing up as either a PDF or DOCX - your pick. Wireframes come through as crisp PNGs, one per screen. That clean UI layout? Also lands as a single PNG snapshot. Then there is the React bundle, zipped tight. Inside: each component living in its own file, an App.js calling the shots with useState for navigation, plus a README laid out plainly for developers

The System Runs On These Technologies:

On top sits the Next.js interface - managing chat, previews, and exports. Below that runs a chain of eight specialized agents, each playing its own role. These pieces feed into a final stage where files take shape and get stored securely. Sitting beside them, an orchestration system holds shared memory across steps. It checks every result fits the expected format. When APIs stumble, it tries again quietly. Each part moves only when the one before settles. Nothing jumps ahead. Structure keeps things moving without noise. Decisions stay close to where they matter most.

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

Table 1: Technology Stack of Agentic UX

Component	Technology
Frontend Framework	Next.js 14 with React 18
Styling	Tailwind CSS 3
AI API	Anthropic Claude API (claude-sonnet-4)
Document Parsing	pdf-parse, mammoth.js
File Export	jsPDF, docx.js, jszip
Database / Auth	Supabase (PostgreSQL + Auth)
AR Preview (conditional)	WebXR Device API, MediaPipe Hands
Runtime	Node.js 18+

RESULTS AND DISCUSSION

Twenty-nine trials made up the check of the Agentic UX setup, split across four batches. Every single one finished without fault, hitting full marks in outcome. Testing individual pieces covered a dozen scenarios - things like login handling, talking to the AI service, helper tools, reading uploaded PRDs, and exporting data. That twelfth run stood out because it showed how the software deals with broken responses from the language model, fixing structure issues quietly in the browser instead of crashing straight away.

Midway through the checks, eight core connections across modules got a run-through during integration testing. When it came time for test IT4, the system faced the PRD approval checkpoint - there, the workflow engine paused exactly where expected for manual feedback instead of barreling forward. Only once someone gave consent did operations pick up again, skipping nothing that had already passed. Later on, IT5 turned its attention to how iframes behaved inside isolated frames, viewed live in Chrome, then cross-checked against Firefox and Safari just to be sure.

A time of 1.2 seconds appeared for LCP - under the two-second threshold. Finishing fast, the pipeline with nine agents cleared its work in 18 seconds, beating the 30-second cap. In exactly 3.4 seconds, the tool split apart a 12-page DOCX file. Less than one second was needed for iframe rendering: only 0.9. Clocking in at 6.2 seconds, the full export assembled four items into a single zip

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

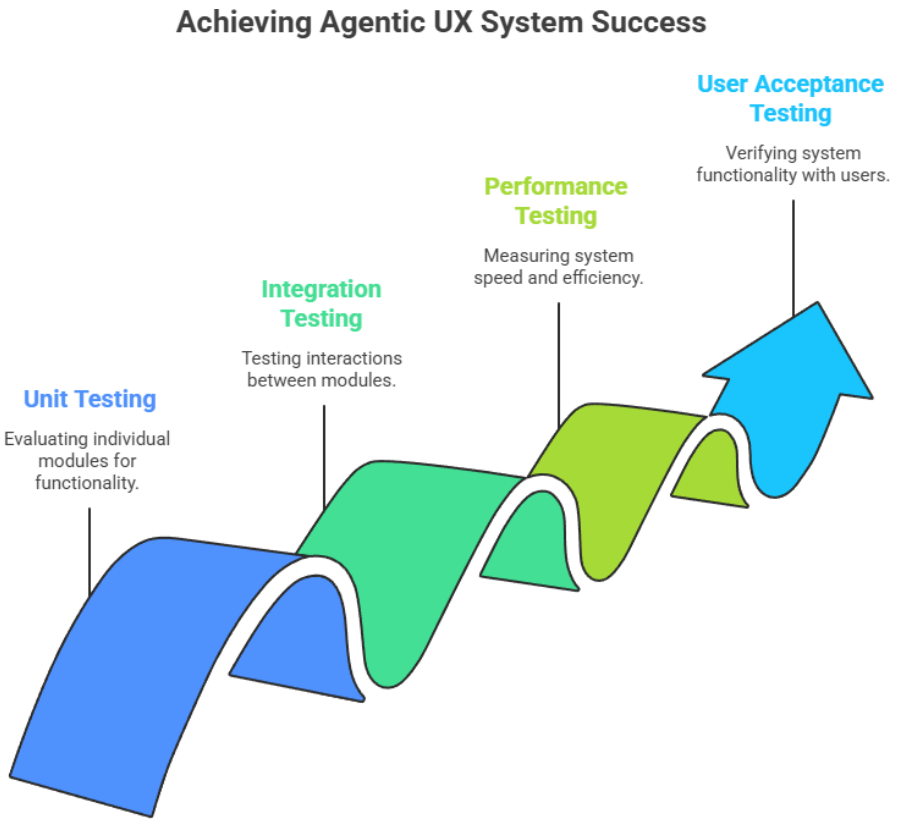


Figure 3: Achieving Agentic UX System Success

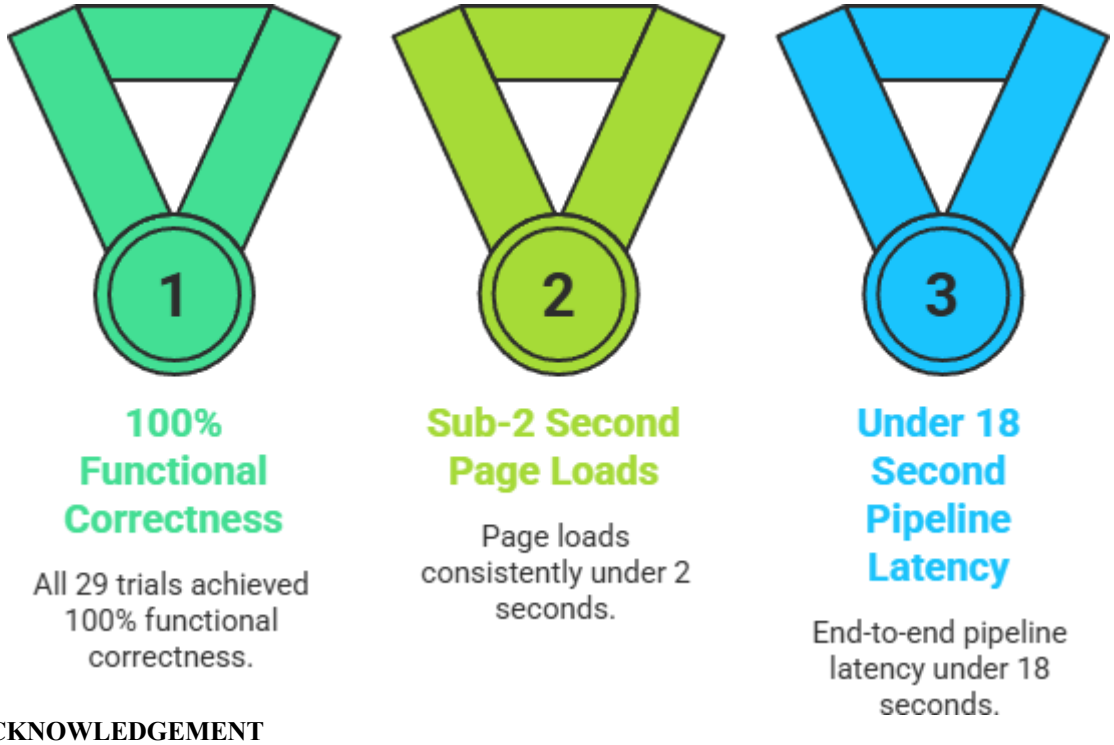
After testing, the team confirmed both key features worked as intended. First round checks made sure AR previews ran fully - spotting WebXR surfaces, asking camera access, recognizing gestures through MediaPipe. During the second phase, they checked if React-built code dropped straight into projects without needing changes.

Table 2: Test Case Summary

Category	Total	Passed	Failed	Rate
Unit Testing	12	12	0	100%
Integration	8	8	0	100%
Performance	5	5	0	100%
UAT	4	4	0	100%
TOTAL	**29**	**29**	**0**	**100%**

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

Figure 4 : Results of Agentic UX



ACKNOWLEDGEMENT

Thanks go to the department staff for helping out along the way while this work took shape. Biggest thanks land on the project mentor and academic lead, whose know-how shaped key parts of the design. They also opened doors to lab tools needed for testing - no small thing. Shouts rise for every tester who showed up, tried features, shared real reactions when using the system. Their time fed directly into how things changed down the line. Appreciation settles there, quiet but solid.

CONCLUSION

Start to finish, one system handles every stage - no gaps, no switches. Picture words turning into interfaces without dropping meaning along the way. From first idea to final layout, each piece flows into the next. Instead of juggling separate tools, everything stays connected. Eight steps fold into a single process. Design grows step by step but never loses its thread. What begins as conversation becomes structure, then visuals, then function. Testing happens where it fits, not after the fact. Code appears when ready, pulled from working models. Nothing gets lost between handoffs because there are no handoffs.

Twenty-nine tests ran perfectly every time. Not one failed to work right across browsers when embedded. Pages came up fast - always in less than two seconds. The whole process from start to finish took fewer than eighteen seconds. Each result stayed beneath our required limits. User testing confirmed everything functions as intended. Both standout features passed clearly. The WebXR augmented reality view worked without issue. Code produced can go straight into development use.

One big step forward? Making product creation easier for everyone. Solo builders, startup founders, even tiny crews can now design full products on their own - no need to master UX, coding, or product planning first. Coming down the line: smarter teamwork between automated helpers, visual analysis tools that give live reactions, richer hand movements in augmented reality demos, along with extra ways to structure project docs. Things shift when tech opens doors once locked.

REFERENCES

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

[1] Brown, T., Mann, B., et al. (2020). Language Models are Few-Shot Learners. Advances in Neural Information Processing Systems, 33, 1877–1901.
[2] Park, J.S., O'Brien, J.C., et al. (2023). Generative Agents: Interactive Simulacra of Human Behavior. Proceedings of UIST 2023, ACM.
[3] Nielsen, J. (1994). 10 Usability Heuristics for User Interface Design. Nielsen Norman Group. https://www.nngroup.com/articles/ten-usability-heuristics/
[4] Chen, M., Tworek, J., et al. (2021). Evaluating Large Language Models Trained on Code. arXiv:2107.03374.
[5] Speicher, M., Hall, B.D., & Nebeling, M. (2019). What is Mixed Reality? Proceedings of CHI 2019, ACM, Paper 537.
[6] W3C. (2023). WebXR Device API. World Wide Web Consortium. https://www.w3.org/TR/webxr/
[7] Luger, E., & Sellen, A. (2016). Like Having a Really Bad PA. Proceedings of CHI 2016, ACM.
[8] Next.js Documentation — Version 14. Vercel. https://nextjs.org/docs
[9] Next.js Documentation — Version 14. Vercel. https://nextjs.org/docs
[10] Anthropic. (2024). Claude API Documentation. https://docs.anthropic.com/
[11] Google. (2023). MediaPipe Hands. https://developers.google.com/mediapipe/solutions/vision/hand_landmarker
[12] Supabase Documentation. https://supabase.com/docs