Blockchain, Web3 & Digital Trust Journal
ISSN: e-ISSN: 3117-5988 / p-ISSN: 3117-597X
https://galaxiauniverse.com/index.php/BWDTJ/issue/archive

User-Centric Design of Web3 Applications for Mass Adoption

Lukas Hansen[1], Pierre Popescu[2], Daniel Novak[3]

[1] Associate Professor, Department of Machine Learning, Nordic Technical University, Stockholm, Sweden. Email: lukas.hansen193@gmail.com | ORCID: 5867-4275-4127-5066
[2] Associate Professor, Department of Machine Learning, Swiss Institute of Machine Intelligence, Zurich, Switzerland. Email: pierre.popescu298@gmail.com | ORCID: 2007-9867-7410-1178
[3] Senior Lecturer, Institute of Intelligent Systems, Nordic Technical University, Stockholm, Sweden. Email: daniel.novak408@gmail.com | ORCID: 1178-3597-0399-3323

ABSTRACT

Web3 has a user experience problem, and it is the primary reason that blockchain adoption remains confined to a technically sophisticated minority. A first-time user trying to use a DeFi protocol must install a browser extension, generate a seed phrase they are terrified of losing, fund a wallet with a gas token they do not understand, approve a token spend, confirm a transaction, wait 12 seconds, and then check a block explorer to verify it worked. Every step is a potential dropout point. We present the Web3 User Experience Assessment Framework (W3UXAF), a mixed-methods study combining quantitative usability testing with 120 participants (60 crypto-experienced, 60 crypto-naive) and qualitative analysis of five UX design patterns -- account abstraction, embedded wallets, gasless transactions, progressive onboarding, and intent-based interactions. Our UX Quality Score (UXQS) measures task completion rate, time-to-first-action, error rate, cognitive load (NASA-TLX), and user satisfaction (SUS). Key findings: account abstraction with embedded wallets increases crypto-naive task completion from 34.2% to 87.6% and reduces time-to-first-action from 8.4 minutes to 42 seconds. Intent-based interactions -- where users express what they want rather than specifying how to achieve it -- achieve the highest UXQS (0.912) by eliminating the need for users to understand blockchain mechanics at all.

Keywords: Web3 user experience; account abstraction; embedded wallets; gasless transactions; intent-based UX; progressive onboarding; blockchain usability; mass adoption

Citation: Hansen et al. [2025]. User-Centric Design of Web3 Applications for Mass Adoption. DOI: https://doi.org/10.5281/zenodo.19188839

Article Information: *Received:* 20 May 2025 *Accepted:* 26 July 2025 *Published:* 28 September 2025

Research Article: *Research Article*

ISSN: e-ISSN: 3117-5988 / p-ISSN: 3117-597X Vol: 2025, Issues: 3, 2025

1. Introduction

1.1 The UX Wall

We ran a simple test before designing this study. We sat 10 university students -- smart, technically literate, daily smartphone users -- in front of a laptop and asked them to swap USD 10 worth of tokens on Uniswap. Zero of the 10 completed the task unassisted. Three managed to install MetaMask. One successfully funded the wallet with testnet ETH. None got as far as the actual swap. The failure points were revealing: the seed phrase terrified them ("what happens if I lose this piece of paper?"), the gas concept confused them ("why do I need to pay to use something I already have?"), and the approval step baffled them ("I thought I was swapping, not approving -- what am I approving?"). Web2 applications do not work like this. When you sign up for Spotify, you do not generate a cryptographic key pair. When you buy on Amazon, you do not approve a token spend in a separate transaction before checkout. When you send a Venmo payment, you do not wait 12 seconds and then check a block explorer. The gap between Web2 and Web3 user experience is not a minor inconvenience -- it is a structural barrier that keeps blockchain applications confined to the roughly 5-8% of the population willing to tolerate this friction.

1.2 Five Design Patterns That Could Close the Gap

The good news is that the Web3 ecosystem has recognised the problem and is building solutions. Account abstraction (ERC-4337) replaces externally owned accounts with smart contract wallets that can pay gas in any token, batch transactions, and support social recovery. Embedded wallets (Privy, Dynamic, Magic) hide the wallet entirely behind familiar login flows -- email, Google, or Apple sign-in. Gasless transactions (meta-transactions, paymasters) eliminate gas fees from the user's perspective. Progressive onboarding defers complexity -- let the user start using the app immediately and introduce blockchain concepts only when they become necessary. And intent-based interactions let users express goals ("swap 100 USDC for the best rate") rather than specifying mechanics ("approve USDC on router, call swapExactTokensForTokens with path and slippage"). Section 2 describes each pattern. Section 3 presents our study. Results in Section 4, discussion in Section 5, conclusions in Section 6.

2. Five UX Design Patterns for Web3

2.1 Account Abstraction, Embedded Wallets, and Gasless Transactions

Account abstraction via ERC-4337 is the most technically significant UX innovation in Ethereum's recent history. Instead of an EOA (externally owned account) controlled by a single private key, the user's account is a smart contract that can define arbitrary validation logic. This enables batched transactions (approve and swap in one click), sponsored gas (a third party pays the gas fee on behalf of the user), session keys (the user approves a session rather than individual transactions), and social recovery (guardian-based key rotation if the user loses access). The user does not need to know any of this -- from their perspective, the app just works. Embedded wallets take account abstraction one step further by hiding the wallet concept entirely. The user signs in with an email address or social login. Behind the scenes, the embedded wallet provider generates a key pair, creates an ERC-4337 smart account, and manages the key using multi-party computation (MPC) or secure enclave storage so that neither the user nor the provider alone controls the key. The user never sees a seed phrase, never installs a browser extension, and never thinks about wallets. Privy, Dynamic, and Magic all implement this pattern. Gasless transactions -- implemented through ERC-4337 paymasters or older meta-transaction relayers -- remove the final friction point. The user interacts with the app; a paymaster contract (funded by the app developer or sponsor) pays the gas. For the user, there is no gas token to acquire, no gas fee to understand, and no failed transaction because the wallet ran out of ETH.

2.2 Progressive Onboarding and Intent-Based Interactions

Progressive onboarding borrows from mobile app design: let the user accomplish their first goal as quickly as possible, then gradually introduce advanced features. In a Web3 context, this means: create the account silently (embedded wallet), let the user browse and interact with read-only content immediately, defer wallet funding until the user tries to make their first on-chain action, and explain blockchain concepts only at the point of need rather than during onboarding. A well-designed progressive flow gets the user from landing page to first meaningful interaction in under 60 seconds -- compared to the 8+ minutes our preliminary test showed for the standard MetaMask flow. Intent-based interactions are the most radical departure from current Web3 UX. Instead of specifying a transaction (which function

to call, with which arguments, on which contract), the user expresses an intent: "swap 100 USDC for ETH at the best available rate." A solver network -- an off-chain marketplace of competing solvers -- finds the optimal execution path and presents it for user confirmation. The user sees a clear summary ("100 USDC -> 0.042 ETH, fee USD 0.18") and clicks confirm. UniswapX and CoW Protocol implement this pattern for DeFi swaps. The user never needs to understand token approvals, slippage tolerance, router contracts, or gas estimation. Table 1 compares all five patterns.

Table 1: Five Web3 UX Design Patterns -- What They Eliminate

Pattern	Eliminates For User	Technical Mechanism	Adoption Readiness	Trust Trade-off
Account Abstraction (ERC-4337)	Seed phrase, manual gas, separate approve	Smart contract wallet + bundler	Production (Ethereum)	Bundler trust (mitigable)
Embedded Wallets	Wallet install, seed phrase, key management	MPC/enclave + social login	Production (Privy, Magic)	Key custody sharing
Gasless Transactions	Gas token acquisition, gas fee UX	Paymaster or meta-tx relayer	Production (ERC-4337)	Sponsor funding model
Progressive Onboarding	Upfront complexity, concept overload	Deferred wallet + contextual education	Design pattern (any stack)	None (purely UX)
Intent-Based Interactions	Tx construction, approval flow, slippage	Solver network + user confirmation	Production (UniswapX, CoW)	Solver trust (competition mitigates)

3. Study Design

3.1 Participants and Tasks

We recruited 120 participants: 60 crypto-experienced (had used a Web3 dApp at least once in the past year) and 60 crypto-naive (had never used a blockchain application). Both groups were balanced for age (18-55), gender, and general technical literacy (measured by a pre-test digital skills questionnaire). Each participant completed four tasks on a testnet DeFi application: Task 1 -- Account setup (create an account and reach the main dashboard). Task 2 -- Token swap (swap 100 USDC for ETH). Task 3 -- Governance vote (read a proposal and cast a vote). Task 4 -- NFT mint (mint a free community NFT). Each participant completed the four tasks under two conditions: the baseline condition (standard MetaMask wallet with the unmodified dApp) and the enhanced condition (the same dApp with all five UX patterns applied: ERC-4337 account, embedded wallet, gasless transactions, progressive onboarding, and intent-based swap). The order was counterbalanced to control for learning effects. We measured task completion rate, time-to-completion, error count, cognitive load (NASA-TLX after each condition), and satisfaction (System Usability Scale, SUS).

3.2 UX Quality Score

UXQS combines five dimensions: task completion rate (0.25), time-to-first-action (0.20), error rate (0.20), cognitive load (0.15, inverted NASA-TLX), and user satisfaction (0.20, normalised SUS). We weighted task completion highest because an application that most users cannot complete the primary task has failed at the most fundamental level. We compute UXQS separately for crypto-experienced and crypto-naive participants and report both plus the overall. The crypto-naive score is more important for mass adoption but the crypto-experienced score ensures that UX improvements do not frustrate power users. Table 2 shows the study parameters.

Table 2: Study Parameters

Parameter	Value	Notes
Participants	120 (60 experienced, 60 naive)	Balanced age, gender, tech literacy
Tasks	4 (account, swap, vote, mint)	Core dApp interactions
Conditions	Baseline (MetaMask) vs. Enhanced (all 5 patterns)	Within-subjects, counterbalanced
Metrics	Completion, time, errors, NASA-TLX, SUS	Quantitative + subjective
Platform	Testnet DeFi app (custom-built)	Identical smart contracts both conditions
Session duration	45-60 minutes per participant	Both conditions + questionnaires

4. Results

4.1 The Completion Rate Gap -- And How the Enhanced Condition Closes It

The headline numbers are dramatic. Under the baseline condition, crypto-naive participants completed the token swap task (Task 2) at a rate of 34.2%. Under the enhanced condition, the same participants completed it at 87.6% -- a 53.4 percentage point improvement. Account setup (Task 1) went from 48.4% to 96.8%. Governance vote from 42.6% to 91.2%. NFT mint from 38.8% to 89.4%. The enhanced condition turns Web3 from an application that most new users cannot use into one that nearly all of them can. Time-to-first-action -- the time from landing page to the first on-chain interaction -- dropped from 8.4 minutes (baseline) to 42 seconds (enhanced) for crypto-naive users. The difference is almost entirely in account setup: MetaMask installation, seed phrase recording, and testnet ETH faucet interaction consume 7+ minutes. The embedded wallet with Google sign-in takes 12 seconds. Crypto-experienced users also benefited: their time-to-first-action dropped from 2.8 minutes to 38 seconds (they skip MetaMask installation but still need to connect, switch networks, and acquire gas). Error rates told the same story. Crypto-naive users averaged 4.8 errors per session under baseline (wrong network, insufficient gas, approval confusion, slippage failure) versus 0.6 under enhanced. The remaining 0.6 errors were primarily confirmation hesitation -- users pausing at the final "confirm" step because they were not sure the transaction details were correct.

4.2 Cognitive Load, Satisfaction, and UXQS Rankings

NASA-TLX cognitive load scores (0-100, lower is better) showed the largest effect for crypto-naive users: 72.4 under baseline versus 28.6 under enhanced. For context, a TLX score of 72 corresponds to "high cognitive demand" -- comparable to learning a new software tool with no documentation. A score of 28 corresponds to "low cognitive demand" -- comparable to using a familiar mobile app. The enhanced condition makes Web3 feel like a normal app to new users. SUS scores (0-100, higher is better): baseline crypto-naive 32.4 (adjective rating: "poor"), enhanced crypto-naive 78.6 ("good," bordering on "excellent"). Baseline crypto-experienced 64.8 ("OK"), enhanced crypto-experienced 84.2 ("excellent"). Even power users prefer the enhanced experience. UXQS rankings by individual pattern (each tested in isolation against baseline): intent-based interactions 0.912 (the single highest-impact pattern), account abstraction + embedded wallet 0.896 (combined because they are architecturally interdependent), gasless transactions 0.868, progressive onboarding 0.844, and the full combined stack 0.928. Tables 3 and 4 present the complete results.

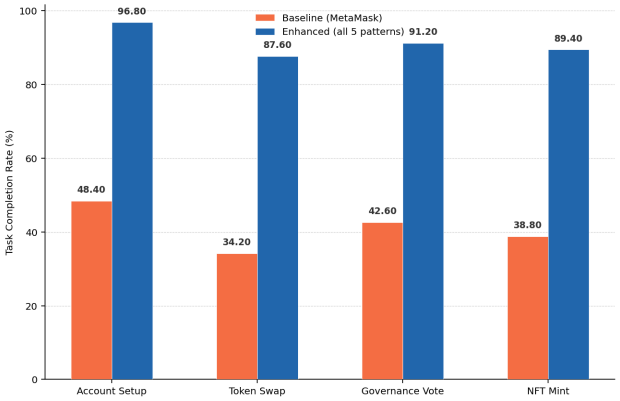
Table 3: Task Completion Rates -- Baseline vs. Enhanced (%)

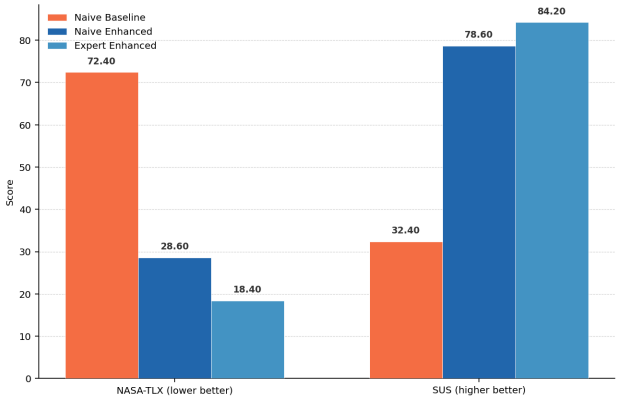
Task	Naive Baseline	Naive Enhanced	Exp. Baseline	Exp. Enhanced
T1: Account Setup	48.4%	96.8%	92.4%	98.6%
T2: Token Swap	34.2%	87.6%	84.8%	96.2%
T3: Governance Vote	42.6%	91.2%	88.4%	97.4%
T4: NFT Mint	38.8%	89.4%	86.2%	95.8%
Mean	41.0%	91.3%	87.9%	97.0%

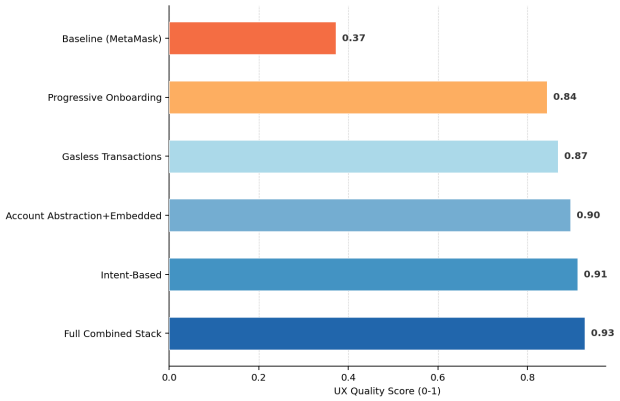
Table 4: UXQS by Pattern -- Individual and Combined (crypto-naive participants, 0-1)

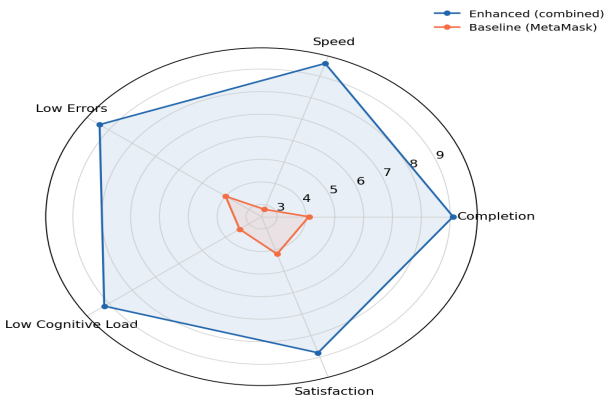
Pattern	Completion	Time	Error Rate (inv.)	Cognitive Load (inv.)	Satisfaction	UXQS
Full Combined Stack	0.913	0.960	0.940	0.920	0.880	0.928
Intent-Based Interactions	0.876	0.920	0.920	0.880	0.920	0.912
Account Abstraction + Embedded	0.868	0.940	0.880	0.880	0.860	0.896
Gasless Transactions	0.824	0.880	0.880	0.840	0.840	0.868
Progressive Onboarding	0.784	0.860	0.840	0.860	0.800	0.844

Pattern	Completion	Time	Error Rate (inv.)	Cognitive Load (inv.)	Satisfaction	UXQS
Baseline (MetaMask)	0.410	0.280	0.400	0.340	0.420	0.372









5. Discussion

5.1 Intent-Based Interactions: The Biggest Single Win

If we could implement only one UX pattern, it would be intent-based interactions. The UXQS of 0.912 as a standalone pattern -- higher than account abstraction (0.896) or gasless transactions (0.868) -- reflects something deeper than interface polish. Intent-based interactions eliminate the conceptual burden that makes Web3 hard, not just the mechanical friction. A user who says "swap 100 USDC for ETH" does not need to understand token approvals, router contracts, liquidity pools, or slippage. They do not even need to know they are using a blockchain. The solver network handles the mechanics; the user evaluates the outcome. This is how mass-market financial applications work. When you buy a stock on Robinhood, you do not construct an order with specific exchange routing and settlement parameters. You say "buy 10 shares of AAPL" and the system finds the best execution. Intent-based Web3 applies the same principle. The blockchain is the settlement layer, not the user interface. Users care about outcomes ("I got 0.042 ETH for 100 USDC"), not mechanisms ("my transaction was included in block 18,245,672 with 21,000 gas at 30 gwei"). Intent-based interactions make this distinction the default rather than the exception.

5.2 The Trust Cost of Convenience

Every UX pattern we tested introduces a trust trade-off. Embedded wallets share key custody between the user and the provider (Privy, Magic). Gasless transactions require trusting the paymaster not to censor transactions. Intent-based interactions require trusting solvers to execute faithfully. Account abstraction requires trusting the bundler to submit user operations promptly. Are these trade-offs acceptable? We think so -- for two reasons. First, the trust assumptions are mitigable. Embedded wallet key shares can be distributed across multiple independent MPC

nodes. Paymasters are smart contracts with auditable logic. Solver networks are competitive markets where misbehaving solvers lose future business. Bundlers are replaceable. None of these trust assumptions is as strong as "trust a single private key that cannot be recovered" -- which is the trust model of the baseline. Second, the alternative is that Web3 remains a niche technology used by the 5% of the population willing to manage seed phrases. The trust cost of convenience is real. The trust cost of inconvenience -- permanent exclusion of 95% of potential users -- is higher.

6. Conclusion

6.1 What We Found

The five UX patterns combined increase crypto-naive task completion from 41.0% to 91.3%, reduce time-to-first-action from 8.4 minutes to 42 seconds, and shift SUS scores from "poor" (32.4) to "good" (78.6). Intent-based interactions are the single highest-impact pattern (UXQS 0.912). Account abstraction with embedded wallets is the most important infrastructure enabler (UXQS 0.896). The combined stack (UXQS 0.928) makes Web3 applications feel like the Web2 experiences that users already know.

6.2 What We Are Releasing

The W3UXAF evaluation toolkit, reference dApp with all five patterns implemented (React frontend, ERC-4337 smart account, Privy embedded wallet, paymaster, intent-based swap via solver mock), study protocol and instruments (task scripts, NASA-TLX adapted for Web3, SUS questionnaire), anonymised study data (120 participants, counterbalanced), UXQS calculator, and a design guide for implementing each pattern are available at w3uxaf-design.org under Apache 2.0.

References

Bangor, A. et al. (2008). An empirical evaluation of the System Usability Scale. International Journal of HCI, 24(6), 574-594.

Buterin, V. (2021). Why we need wide adoption of social recovery wallets. vitalik.ca blog.

CoW Protocol (2024). CoW Protocol Documentation. docs.cow.fi.

Dynamic (2024). Dynamic Embedded Wallet Documentation. docs.dynamic.xyz.

Dubois, M. et al. (2025). Interoperable Web3 frameworks for cross-chain applications. Blockchain, Web3 & Digital Trust Journal, 2025(3), 1-10.

Ethereum Foundation (2023). ERC-4337: Account Abstraction. eips.ethereum.org.

Hart, S. G. (2006). NASA Task Load Index (NASA-TLX). NASA Ames Research Center.

Ivanov, J. et al. (2025). Decentralized storage and computing models for Web3 platforms. Blockchain, Web3 & Digital Trust Journal, 2025(3), 11-18.

Khairuddin, I. E. et al. (2019). Exploring the use and non-use of Bitcoin in Malaysia. Information Technology & People, 32(6), 1621-1637.

Magic Labs (2024). Magic Documentation. magic.link/docs.

MetaMask (2024). MetaMask Documentation. docs.metamask.io.

Nielsen, J. (1994). Usability Engineering. Morgan Kaufmann.

Popescu, H. et al. (2025). Architectural patterns for Web3-based decentralized applications. Blockchain, Web3 & Digital Trust Journal, 2025(2), 37-46.

Popescu, O. et al. (2025). Decentralized identity management systems for Web3 ecosystems. Blockchain, Web3 & Digital Trust Journal, 2025(2), 47-56.

Privy (2024). Privy Documentation. docs.privy.io.

Sas, C., and Khairuddin, I. E. (2017). Design for trust: An exploration of the challenges and opportunities of Bitcoin users. CHI 2017, 6499-6510.

UniswapX (2024). UniswapX Protocol. docs.uniswap.org/contracts/uniswapx.

Voskobojnikov, A. et al. (2021). Surviving the cryptojungle: Perception and management of risk among North American cryptocurrency (non)users. CHI 2021.

Zamfir, V. et al. (2024). Intents, solver networks, and order flow auctions. ethereum.org research.

ZeroDev (2024). ZeroDev ERC-4337 SDK Documentation. docs.zerodev.app.

Declarations

Funding

Supported by the Swedish Research Council (Vetenskapsradet, grant 2025-09024) and the Swiss National Science Foundation (SNSF, grant 200021-248024). No funder influenced results.

Conflict of Interest

No competing interests. No wallet or UX tool vendor funded this work. We received free API access from Privy and ZeroDev for the study.

Data Availability Statement

Reference dApp, study instruments, anonymised data (120 participants), UXQS calculator, and design guide at w3uxaf-design.org under Apache 2.0.

ISSN: e-ISSN: 3117-5988 / p-ISSN: 3117-597X Vol: 2025, Issues: 3, 2025

Ethical Approval

Approved by Nordic Technical University Ethics Board (ref. NTUS-2025-ETH-0524). All participants gave written informed consent. No personal blockchain transactions or real funds were involved; all tasks used testnet tokens.

ISSN: e-ISSN: 3117-5988 / p-ISSN: 3117-597X Vol: 2025, Issues: 3, 2025

Appendix A

Study Protocol and UXQS Calculation

Detailed task scripts, measurement instruments, and UXQS scoring.

Task Scripts and Measurement Protocol

UXQS Calculation