Hybrid Adaptive Software Engineering (HASE): A Woman-of-Color Founder's Smartphone-Directed Development of a Procurement-Readiness Platform as a Disability Accommodation

A Practitioner Case Study of Tapn.vip

Stephanie Nichole-Sinequante Comer, MBA

MBA, Western Governors University · Founder, Tapn.vip Inc., and Glossy Byrd Management LLC, Milwaukee, Wisconsin, United States

Glossybyrd@glossybyrd.com · ORCID 0009-0008-1528-206X

SSRN abstract ID 7692100 · DOI 10.2139/ssrn.7692100 (active once posted)

Working paper HASE-WP-2026-01 · Version 1.3 · 22 November 2026 · Not peer reviewed · Comments welcome

Abstract

Software development is usually described as the work of trained engineers at desktop workstations. This practitioner case study documents a different path. A Black woman founder with more than two decades of business-operations and consulting experience directed, as a disability accommodation, the development of a multi-module business identity and procurement-readiness platform (Tapn.vip) substantially from a Samsung Galaxy S24 Ultra smartphone, using cloud-hosted, AI-assisted development tools. Using an embedded single-case design with an explicit evidence hierarchy, the study traces how three services the founder had delivered by hand (back-office support, business audits and federal registration support) were translated into the platform's document vault, readiness-scoring and business-identity components. It explains why the device and a cloud architecture made a phone-based workflow feasible, and documents how platform constraints shaped the product, including the difference between NFC tag writing on Android and on iPhone. The study proposes Hybrid Adaptive Software Engineering (HASE), a descriptive model of founder-directed development that combines domain expertise, accessible tooling and human release judgment. A repository measurement on 13 November 2026 records about 118,600 lines of platform code and 351 automatic checkpoints, each a build step the founder directed and accepted, over 45 active build days; observed platform data show 69 member accounts in early access. The study does not

establish production reliability at scale, effects on contract awards or the combined effect of platform and curriculum, and it sets out an evaluation design for testing both. Its contribution lies in how these elements operate together: the literature searched documents mobile programming, low-code and conversational AI development, and disabled programmers' use of AI assistants separately, but no equivalent account of them combined in a founder-directed, multi-module commercial platform.

AI disclosure: The research, ideas, data, case history, HASE framework and findings in this paper are the author's own. Under the author's direction, generative AI assistants, including Claude (Anthropic), Gemini (Google) and AI Mode in Google Search, were used to restructure and format her historical records and materials, draft and edit text from them, search the literature and check citations. The author reviewed the paper and is responsible for all of its content. AI-assisted software development is also a subject of the study (Sections 4 and 6). Accessibility features used solely as assistive technology are not included in this disclosure.

Keywords: disability accommodation; ability-based design; end-user software engineering; generative AI; founder-directed development; mobile development; Black women entrepreneurs; procurement readiness; supplier diversity; NFC; digital identity

JEL classification: L26, J14, J15, J16, O33, M15, H57

1. Introduction

Small businesses often have the skills to deliver a service but lack the documentation, digital systems and procurement knowledge that institutional buyers expect. In more than twenty years of consulting with small businesses, the founder in this case repeatedly met the same barriers: business identifiers scattered across agencies, incomplete capability statements, fragmented customer records and uneven access to training. Building software to address those barriers raised a second access problem. Conventional software development assumes desktop equipment, programming education, financing and engineering staff, and each of those can exclude founders whose physical circumstances or resources require another way of working.

This study examines one founder's response to both problems. Between May and November 2026, Stephanie Nichole-Sinequante Comer directed the construction of Tapn.vip, a web platform that combines a shareable digital business card, NFC identity, a document vault, customer-relationship tools, event check-in, a learning system and a procurement-readiness assessment. She directed the build substantially from a smartphone, as a disability accommodation, using cloud-hosted development and AI tools.

The study asks four questions:

RQ1. How did prior business architecture and domain expertise shape the founder's AI-assisted development process?

RQ2. How did smartphone-based development function as an adaptive working method and disability accommodation?

RQ3. Which practices of the proposed HASE model are identifiable in the project history, and which may transfer beyond this case?

RQ4. What evidence distinguishes feature availability from usability, adoption and commercial value?

The study makes three contributions. First, it provides a documented account of founder-directed software development under a disability accommodation, with each claim classified by the strength of its evidence. Second, it proposes Hybrid Adaptive Software Engineering (HASE) as a descriptive, testable model of that practice. Third, it sets out an evaluation design for the questions the case cannot yet answer, in particular whether one integrated identity and documentation system reduces procurement-readiness friction, and whether education combined with software works better than either alone (Section 9). HASE is an author-proposed framework, not an established engineering standard or an independently validated method.

Section 2 reviews related work. Section 3 describes the research design and evidence levels. Section 4 presents the case. Section 5 reports findings. Section 6 sets out the HASE model and its propositions. Section 7 discusses contributions and rival explanations, Section 8 the limitations, and Section 9 the evaluation design. Section 10 concludes.

2. Related work

2.1 End-user software engineering and AI-assisted programming

Ko et al. (2011) survey how people who are not professional software engineers create, test, debug and maintain software. This case belongs to that field; it is not the first instance of non-programmers building software. What it adds for study is a combination: a disability-accommodated smartphone workflow, deep domain expertise, language-model-mediated implementation and procurement-focused requirements.

Research on programming with large language models is newer. Sarkar et al. (2022) argue that it should be treated as a distinct way of programming with its own properties and challenges, and they consider non-expert end-user programmers directly. In a controlled experiment, Peng et al. (2023) found that developers with an AI pair programmer completed a task 55.8% faster. That study is a working paper and its participants were professional developers, so it does not test this case's conditions. Bock and Frank (2021) describe low-code platforms, which assemble applications through configuration and generated components. The AI-assisted application builder used in this case shares features with low-code platforms, but whether it fits their definition has not been established.

The newest work studies conversational, agent-driven construction directly. Sarkar and Drosos (2025), in what they describe as the first empirical study of vibe coding, find that it does not remove the need for programming expertise but redistributes it toward managing context, evaluating generated code and deciding when to intervene. That finding matches the founder's role in this case, where domain expertise and release judgment, not code writing, carry the work. Two studies examine disabled programmers using AI assistants: Flores-Saviaga et al. (2025) found that an AI assistant helped developers with visual impairments but also introduced new accessibility barriers, and Chen et al. (2026) followed 16 screen reader users for two weeks and found that advanced assistants extended what they could build while making situational awareness harder. Both studies involve programmers working at computers. None of the studies found examines a non-programmer directing a production platform from a phone as a disability accommodation. Accessibility research adds two cautions. A systematic review of 38 primary studies found that low-code tools depend heavily on visual elements and that little work has improved the accessibility of low-code environments (Khalajzadeh & Grundy, 2025). An accessible process also does not guarantee an accessible product: Palmer and Oswal (2025)

examined both the workflows and the outputs of generative AI website builders for users with disabilities. Closest in method, Glazko et al. (2023) used autoethnography to examine how researchers with disabilities used generative AI for their own access needs, reporting benefits alongside concerns about verifiability. This case shares that first-person design but concerns the construction of a commercial platform.

2.2 Mobile development

Building software on a phone has precedent. Microsoft Research's TouchDevelop environment was designed on the principle that programs are both written and used on a mobile device (Tillmann et al., 2011). TouchDevelop used its own touch-oriented language; the workflow in this case instead combines browser-based builders and conversational AI tools. The precedent concerns the device, not the method. Its users were mainly students and hobbyists personalizing their phones, not founders building commercial platforms.

The name HASE should not be confused with two established terms. Adaptive Software Development is an iterative team methodology built on cycles of speculation, collaboration and learning (Highsmith, 2000), and hybrid adaptive software development describes how organizations combine agile and plan-driven methods (Gill, 2014). Both concern how professional teams organize development. HASE instead describes a single founder directing AI-assisted construction under accessibility constraints; it shares their emphasis on iteration but not their unit of analysis.

2.3 User innovation and design science

Von Hippel (2005) shows that users innovate for themselves as tools lower the cost of development. Tapn.vip is an instance of a user-founder addressing friction she met firsthand in client work. That literature does not establish that user-developed products are technically sound or commercially successful; those remain empirical questions here. Hevner et al. (2004) frame design science as knowledge produced by building and evaluating artifacts, which requires reporting design decisions, testable requirements and evaluation evidence, not only features.

2.4 Ability-based design, disability, race and entrepreneurship

Ability-based design (Wobbrock et al., 2011) moves accessible computing from a focus on disability to a focus on ability. Systems should fit the abilities a person has, instead of asking the person to fit the system. Most of this work looks at how people with disabilities use technology. This case applies the same idea to building software. The founder changed the device and tools she used to direct development so they fit her abilities, instead of adapting herself to a desktop workstation. Section 5.2 returns to this point.

Using national panel data, Renko et al. (2016) found that start-up efforts by nascent entrepreneurs with disabilities are less likely to result in a viable organization, which underlines why accommodations matter. Fairlie and Robb (2008) identify limited startup capital, linked to low household wealth, as a central reason Black-owned businesses trail other firms on sales, employment and survival. Both studies concern business formation generally and predate this case; they are cited for context, not as evidence about it.

2.5 Procurement readiness and supplier diversity

On the buyer side, Adobor and McMullen (2007) argue that supplier diversity can be a source of competitive advantage when it is integrated into corporate strategy, even as firms consolidate their supplier bases, and Shah and Ram (2006) document how large firms develop minority-owned suppliers in practice. Both concern corporate supplier diversity; public procurement, set-asides and certifications are governed separately. For this study, the relevant point is that buyers need suppliers whose qualifications are documented and easy to check.

2.6 Identity, verifiability and accessibility standards

NIST SP 800-63-4 distinguishes identity proofing, authentication and federation (Temoshok et al., 2025): a shareable profile, a verified identity claim and procurement eligibility are different things. The W3C Verifiable Credentials Data Model 2.0 states that the verifiability of a credential does not imply the truth of the claims it contains (World Wide Web Consortium, 2025). WCAG 2.2 provides testable accessibility criteria (World Wide Web Consortium, 2024), and the NIST generative AI profile and OWASP guidance provide risk categories for AI features, including prompt injection and excessive agency (Autio et al., 2024; OWASP, 2025).

3. Research design

3.1 Design and unit of analysis

The study is a longitudinal, embedded single-case study (Yin, 2018) with a design-science orientation (Hevner et al., 2004). Its design and reporting follow Runeson and Höst's (2009) guidelines for case study research in software engineering. The research questions are stated up front. Founder records are checked against a repository measurement, public registries and device documentation. Each claim is tied back to its source through the evidence levels in Section 3.3. The unit of analysis is the founder-directed development and early deployment of Tapn.vip between May and November 2026, set within the founder's consulting history from 2005. Embedded units are requirements translation, the mobile development workflow, cross-device NFC behavior, and early adoption. Disability is used as defined in the Americans with Disabilities Act (42 U.S.C. § 12102). Accommodation is used in its functional sense: a self-directed change to how the founder creates and directs her projects, made to manage her disability. Because she is a founder rather than an employee, it is not an employer accommodation under Title I of the Act, and the study makes no clinical claims.

3.2 Sources

Sources are the founder's dated case-study materials; dated platform specifications through release v1.19.3 (12 November 2026); the platform's public activity counter; public registries (USPTO trademark records and IngramSpark title records); device documentation from Samsung, Google and Apple; and published scholarship. The founder's research also drew on library research, her client case studies, and her conversations with, and feedback from, clients and platform members; no individual client, case or feedback is identified, quoted or reported. Construction drew on a founder-supplied corpus: her prior papers and case studies, business documents and policies, the workbook and curriculum, scoring rubrics and pricing spreadsheets, and pre-existing code. Its contents are not disclosed. The table below maps each type of material to the platform components it shaped (F1).

Founder-supplied material	**Platform components it shaped**
Prior papers, case studies and capability statements	Requirements (Section 5.1); AI-assisted document drafting
Business documents and policies	Document vault categories; public trust and policy pages
Workbook and curriculum	Learning academy
Scoring rubrics	Readiness scoring
Pricing spreadsheets	Plans and checkout
Pre-existing code	The first platform build, May 2026

Source code, prompts, scoring weights and other proprietary processes are not disclosed. Features are described by what they do, not how they are built.

3.3 Evidence levels

Each factual claim is assigned one of four levels.

Level	**Definition**	**Example in this study**
F1 Founder-reported	Recorded in the founder's materials or specifications	Capital history, client count, sales history
O1 Observed	Shown on the public platform counter or a public page on a stated date	Member accounts, site visits, card taps
T1 Technically reproduced	Produced by a documented, repeatable test or measurement that anyone with access can rerun	Repository build measurement (13 November 2026)
I1 Independently verified	Confirmed by an external record or third-party review	USPTO registration, ISBN records

As of this version, build statistics are T1 and most other claims are F1 or O1. The repository measurement was run by the build platform's own agent, read-only, against a recorded repository state, using commands that anyone with repository access can rerun; it is reproducible but has not yet been rerun by an independent party. Section 9 sets out how claims can be raised to T1 and I1.

3.4 Positionality and reflexivity

The author is also the founder whose work is studied. This gives direct access to the design rationale and requirements history, and it is also a source of confirmation bias. Three safeguards are built in: every claim carries an evidence level; rival explanations are addressed explicitly (Section 7.2); and claims that rest on the founder's records can be checked by an independent technology specialist (Section 9). The author identifies as a Black woman founder and advocate; her lived experience explains why the platform prioritizes business ownership, accessible identity and procurement participation. The disability accommodation is described functionally; no medical information is disclosed or needed for the analysis.

4. Case description

4.1 Founder and organizations

Glossy Byrd Management LLC is the founder's holding company. It has provided consulting and contract-readiness services since 2005 and was registered as a limited liability company in 2017. It wholly owns Tapn.vip Inc., formed in 2022; the platform's terms, footers and certificates name Glossy Byrd Management, LLC as the company behind Tapn.vip. Tapn.vip is a registered United States trademark (Registration No. 7,684,476). The founder brings more than twenty years of corporate experience and more than twenty years of entrepreneurship to the case. She holds an MBA in Healthcare Management from Western Governors University (2014) and a BA in Communication from the University of Minnesota (2004), where her capstone examined enterprise social media ethics for healthcare organizations. Technical work began early. Her mother worked in reinsurance and did data programming, and from the age of nine the founder went to work with her, programming, filing and working on green-screen terminals. In college she built websites in HTML for her coursework. Her work history maps directly onto the platform's components:

Experience	What it contributed	Where it appears in the platform
Assurant Health, eleven years: customer service, claims and benefits	Regulated record-keeping, privacy discipline and resolving customer problems against written rules	Document vault, privacy controls, member onboarding
Assurant Health: agent support and client services for managing general agencies (MGAs) handling millions of dollars in business each month; licensed agent in 48 states	Institutional client relationships, high-volume accuracy and compliance across jurisdictions	Procurement-readiness assessment, business identity, multi-country business settings
Assurant Health: trainer who created processes, procedures and manuals across departments and delivered corporate training	Turning operational know-how into written procedures and training	Guided workflows and the Academy
SKYGEN USA: federal and state health-plan account management, including responses to requests for proposals (RFPs), resolving complaints escalated to company presidents, and advising directors on how to address her accounts; worked directly with company presidents and government and state agencies	How government and institutional buyers evaluate vendors, from the RFP to account escalation	Procurement-readiness assessment, capability statements, business identity
SKYGEN USA: system implementation and de-implementation, audits, EDI (electronic data interchange) reporting, and analysis to identify and correct broken code	Hands-on experience with system rollouts, data exchange and diagnosing and correcting software	Analytics and reporting, and the founder's review and testing of AI-generated code
Payment processing at U.S. Bank (check processing and batch balancing)	Transaction accuracy and reconciliation	Payments and entitlement records
Confidential document processing at Northwestern Mutual	Secure handling and classification of sensitive documents	Document vault classification and access
Glossy Byrd Management consulting since 2005 (54+ clients): back-office support, business audits, SAM.gov registration, training and events	Direct knowledge of the barriers small businesses face	The requirements mapping in Section 5.1
Event production and NFC hardware sales since 2019	In-person identity exchange and event operations	Card, NFC identity, events and check-in

Writing procedures and manuals for other departments is the same skill HASE depends on: stating a business process precisely enough that someone else, or a tool, can carry it out. This experience explains the platform's requirements, workflow design and document handling, and it

is the basis of the requirements-translation finding (Section 5.1). Her early programming on green-screen systems, college web development and SKYGEN work gave her direct experience writing, analyzing and correcting code (F1). It does not by itself establish full-stack software engineering, which is why the case rests on her direction, review and correction of AI-assisted construction rather than on hand-written code (Section 6).

4.2 Chronology

Period	Event	Evidence
2005 onward	Consulting and contract-readiness services to small businesses (54+ clients)	F1
2019–2025	NFC wristbands and cards sold at events, online and in person (500+ units)	F1
2023–2026	Six event activations using NFC identity	F1
May 2026	First platform build: dashboard, document studio, administration, NFC activation and AI content tools	F1
June 2026	Founder’s curriculum workbook published (print and eBook ISBNs)	I1
27 July 2026	Version-control repository begins (not the first build date)	F1
30 August – 29 October 2026	Closed beta with invited members	F1
29 October 2026	Early access opened to the public	F1
10 November 2026	Release v1.16	F1
12 November 2026	Release v1.19: paid subscriptions opened; numbered founding-membership program; account email delivered to all members	F1

4.3 The platform

Tapn.vip is a browser-based web application, installable to a phone’s home screen. Its sixteen modules cover a digital business card; NFC identity; analytics; customer-relationship management; events and door check-in; payments; a learning academy; AI-assisted document drafting; a document vault; a community marketplace; an optional emergency-information profile; white-label tenancy; teams; platform metrics; administration; and public trust pages. Business-market settings cover 223 countries, with detailed templates for the United States,

Nigeria, Ghana, Kenya, South Africa and the United Kingdom. Members' business identifiers are labeled as reported by the business until an official registry confirms them. Checks against U.S. SAM.gov and U.K. Companies House are built and go live once registry access keys are in place; until then, and for other countries, identifiers keep the reported-by-the-business label, consistent with the verifiability principle in Section 2.6.

The document-drafting features are an application layer built on third-party language models, not a proprietary foundation model. They draft only from facts the member has on record and do not invent credentials, registrations or employment history.

5. Findings

5.1 Requirements came from services delivered by hand (RQ1)

The clearest mechanism in the case is the translation of existing client services into software requirements. Three of the founder's consulting services map directly onto platform components.

Service the founder delivered by hand	Recurring friction observed	Platform component	Testable requirements
Back-office administrative support	Business records scattered across email, paper and agency portals	Document vault	Secure upload, retrieval, classification, access permissions, version history
Business audits and operational assessments	Owners unable to see which gaps buyers would notice	Procurement-readiness assessment	Defined criteria, gap scores, evidence checklist, remediation steps
SAM.gov registration and supplier setup	Identifiers (UEI, CAGE, NAICS) recorded inconsistently or not at all	Business identity	Identifier fields, registry references, completeness checks
Training and coaching	Uneven access to structured learning	Academy, built on the founder's published curriculum	Lessons, assessments, completion records
Networking and event work	Contact details lost after events	Card, NFC identity, events and CRM	Tap and QR identity, check-in, contact capture, follow-up

The proposed traceability chain is: service delivered by hand, recurring friction, software requirement, implemented feature, verification test, measured user outcome. In this case the first four links are documented at level F1; the last two are not yet tested.

The readiness assessment provides one illustration. Applied to successive versions of the founder's own company capability statement, the platform's score rose from 42 (an initial two-paragraph version) to 67 (a third, fuller version), with consistent results on repeat scoring (F1). These are scores for one document, not a validation of the instrument; calibration against expert reviewers is part of the evaluation design (Section 9).

5.2 The smartphone worked because the heavy work ran in the cloud (RQ2)

The founder describes the smartphone as an accommodation, not a convenience: long periods of sitting and typing at a desktop are not workable for her. The workflow was feasible because compilation, storage, hosting and model inference all ran in cloud services: an AI-assisted development and hosting platform, a version-control repository and third-party model services. The phone did not need to build or host the application. It needed to let one person read, write, compare, test and approve work over long sessions. In ability-based design terms (Wobbrock et al., 2011), the work moved to a device and services that fit the founder's abilities. She did not have to meet the demands of a desktop workstation.

Documented device capability	Role in the build	Relevance as an accommodation
Snapdragon 8 Gen 3 for Galaxy processor; 12 GB of RAM; 256 GB to 1 TB of storage (GSMArena, 2024)	Several browser sessions, AI conversations, a live preview and documents open at once	Fewer reloads and restarts per session
6.8-inch QHD+ display, 120 Hz, up to 2,600 nits (GSMArena, 2024)	Code, previews and long specifications readable on the device	Create in whatever setting the founder needs
Built-in stylus with handwriting into text fields (GSMArena, 2024)	Precise selection, editing and annotation	Fine edits without keyboard and mouse
Samsung DeX desktop mode, wired or wireless (Samsung Electronics, n.d.)	The same phone drives a monitor as a desktop-style workstation	A larger screen when available, without a separate computer or fixed desk
Split-screen and pop-up windows	AI conversation beside the live preview	Compare and approve without switching devices
Dictation, Live Transcribe and Transcript Assist (Samsung Electronics UK, 2024)	Requirements and prompts spoken instead of typed	Less sustained typing
NFC hardware with Chrome's Web NFC (Beaufort, n.d.)	The platform's own tag writer and tester run on the build phone	The development device is also the test device
5,000 mAh battery; 5G and Wi-Fi 7 (GSMArena, 2024)	Long cloud-connected sessions	Work not tied to one room

The table lists documented capabilities of the device. Which capabilities the founder used in a given session, and for how long, has not yet been logged; a summary workflow log is part of the evaluation design.

5.3 Device platforms shaped the product (RQ2, RQ4)

Building on Android had a direct product consequence. Web NFC, the browser interface that lets a web page write to an NFC tag, is supported in Chrome for Android and in no browser on iPhone (Can I use, 2026; Beaufort, n.d.). The founder could therefore write, test and lock tags from inside the product she was building, and the platform's specification records a writer developed on the build phone.

The same constraint shapes how members use the platform. Reading a tag works on both platforms: Android opens the stored web link in the browser, and iPhone XS and later models read tags in the background and open web links with no app (Apple Inc., n.d.). Writing differs. On Android, members write and lock tags inside Tapn.vip. On iPhone, the platform guides members through NFC Tools, a free third-party app that writes tags on iPhone 7 and later (WAKDEV, 2026). Phones without NFC, and computers, open the same card through its QR code. All other functions run in the browser on every device type.

Task	Android with NFC and Chrome	iPhone	Device without NFC
Open a card by tapping a tag	In the browser	iPhone XS and later, no app needed	QR code
Write and lock a tag	Inside Tapn.vip	Through NFC Tools	Not applicable
All other functions	In the browser	In the browser	In the browser

This is a finding about design under platform constraints: the founder's device choice determined which core function could be built and tested end to end in the browser, and the product had to route iPhone users through a third-party step it does not control. The product's response is dated: in November 2026 the platform repositioned sharing from NFC-first to Beam, AirDrop or Quick Share, QR code and link, with NFC hardware optional, across its homepage, tutorial and shop (F1). A constraint first met during construction thus changed the product's positioning, not only its implementation.

5.4 Scale and pace of the build (RQ3)

A read-only measurement of the source repository on 13 November 2026 (T1) records 351 checkpoints over 45 active build days, from 27 July to 13 November 2026; 119 back-end route modules; 308 front-end component files; and 121,490 lines of code. Excluding a 2,865-line third-party interface kit, about 118,600 lines were written for the platform, including 17,833 lines of automated tests. Earlier figures in version 1.0 (about 112,000 lines and 298 commits over 41 days) were taken from the 9 November specification; the difference reflects four further days of building, not a correction.

Checkpoints are not hand-written commits. The development platform saves one automatically after each build step the founder directs and accepts, and its message is the build agent's reply. The checkpoint count therefore approximates the number of directed build steps and is reported as such. These figures measure construction activity; they do not by themselves show software quality, security or commercial scalability. By month, checkpoints numbered 53 in July, 53 in August, 96 in September, 36 in October and 113 in the first thirteen days of November. Activity spans all seven days of the week and most hours of the day, consistent with the adaptive pattern described in Section 5.2, though agent run-time may account for part of the spread.

Quality signals are reported as they stand. The repository holds 91 automated test reports. A November 2026 code review found import and naming defects, several since fixed; 23 lint warnings were open and two payment tests were failing at review time. A read-only security audit on 10 November 2026 found no critical or high-severity issues; lower-severity findings were fixed and hardening items remain open. A paid-plan production acceptance run covering later releases is the next step in the evidence record.

The founder's published curriculum workbook is a second product of the same method: she supplied the curriculum and content and directed AI-assisted tools to produce print-ready and eBook files, published with registered ISBNs in June 2026 (I1). It supports the repeatability of the method, not the platform's performance.

The founder's role was not only to request features but to reject or correct what the build agent produced. Her records list corrections to figures shown across the platform, biometric sign-in, NFC tag programming and locking, the customer-relationship tools, super-administrator controls, white labeling, Trust Center results, footer and site-map additions, search-engine indexing on Google, Bing and DuckDuckGo, and the domain's DNS connection (F1). She directed and checked every one of these corrections from the same Samsung Galaxy S24 Ultra smartphone. Three examples show the pattern.

First, the platform showed different numbers for the same measures in different places. The founder reconciled them and found that most logged requests were automatic server health checks, not people. The platform now reports requests separately from visits (Section 5.5).

Second, the agent reported that biometric sign-in was in place. Before accepting it, the founder confirmed that it was a true passkey under the Web Authentication standard, not a shortcut layered over a stored password.

Third, the Trust Center displayed a pass count and an evidence window that did not match the platform's own records. The founder had both corrected before the page went to an enterprise prospect.

In each case the tool did not flag its own error. The error was caught by someone who knew what a number, a credential or a claim had to mean to a buyer. This is the human release judgment the HASE model depends on (Section 6). Prompts and code for these corrections are not disclosed.

5.5 Early adoption and economic context (RQ4)

Measure	Figure	Evidence
Member accounts (no test accounts)	69	O1, 12 November 2026
Site visits since 11 August 2026	11,017	O1
Card taps / card opens (taps, scans and links)	868 / 1,062	O1
Countries of site visitors	51	O1
Paying subscribers, paid in cash and activated manually	6 (early November 2026)	F1, unaudited
Card payments through in-platform checkout	0 before checkout opened on 12 November 2026	F1
Closed beta (invited members)	300+ card taps; 2,300+ unique visitors from 5 countries	F1
Historical NFC hardware sales, 2019–2025	About $25,000; 500+ units	F1, unaudited
Business capital over 15 years	$180,000: $156,000 founder funds, $24,000 grants	F1, unaudited

The platform counter reports server requests separately from visits; requests are not reported here as people. The figures describe early adoption, not scale. The capital figure is cumulative business funding across hardware, events and the platform, not the cost of the software alone, and the largely self-funded path is consistent with the capital constraints Fairlie and Robb (2008) describe. Revenue and profitability are outside the scope of this study.

6. The HASE model

HASE is defined as an iterative, founder-directed practice in which a domain expert specifies business outcomes, uses language-model tools to draft and revise implementation artifacts, integrates them through a hosted development environment and version control, and applies human judgment, accessibility constraints, testing and release decisions. The claim is not that AI replaces engineering. It is that structured domain expertise and accessible tooling change who can initiate and direct software construction.

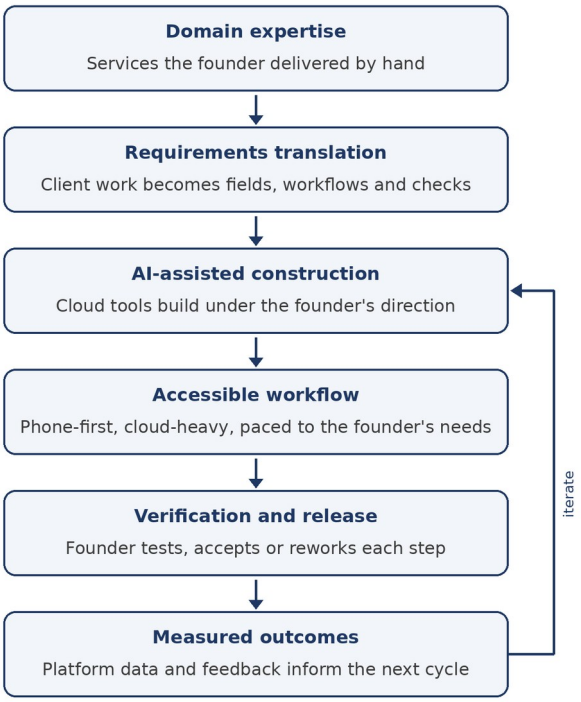

Figure 1. The HASE cycle as observed in this case. A descriptive synthesis of the method, not a validated engineering standard.

HASE overlaps with three established approaches but differs from each in who directs the work, what drives the requirements and what constrains the workflow.

Approach	Who builds	How software is produced	What HASE adds
End-user programming (Ko et al., 2011)	People who are not professional developers, usually for their own use	Writing or adapting programs, spreadsheets and scripts	A commercial, multi-module platform for other users, not a personal tool
Low-code development (Bock & Frank, 2021)	Business users within an organization	Configuring platform components and generated code	Requirements drawn from services the founder delivered by hand, outside any IT department
Conversational AI coding (Sarkar & Drosos, 2025)	Mostly programmers at computers	Prompting AI, then evaluating and editing its code	A non-programmer on a phone, with accessibility shaping the workflow and the founder holding release judgment

Dimension	Description	Evidence in this case	Evidence still needed
Problem ownership	The founder defines the user, economic and institutional problems	Consulting history; Section 5.1 (F1)	Interviews with former clients
Requirements translation	Services delivered by hand become data fields, workflows and acceptance criteria	Section 5.1 mapping (F1)	Dated requirement records
AI-assisted construction	AI tools draft and revise code and documents under the founder's direction	351 directed checkpoints; repository measurement (T1)	Independent rerun of the measurement
Provenance discipline	Version history and approvals are retained	Repository from 27 July 2026, measured at a recorded state (T1)	Independent review
Adaptive workflow	Development fits the founder's accessibility requirements	Section 5.2 (F1)	Summary workflow log
Verification and governance (emerging)	Privacy, permissions, payments, security and accessibility are tested	91 automated test reports; internal audit; open findings reported in Section 5.4 (F1, T1)	External security and WCAG audit
Release and learning	Outcomes are compared with platform data and user feedback	Public counter (O1)	Production acceptance run; cohort data

The model yields four testable propositions:

P1. Founders with deep domain expertise can specify software requirements that map directly to services they have delivered by hand.

P2. Where compilation, hosting and model inference run in cloud services, a current flagship smartphone can serve as a primary development workstation for directing AI-assisted construction.

P3. Device platform constraints shape which product functions a phone-directed founder can build and test end to end.

P4. Combining a readiness assessment with structured curriculum improves documentation completeness more than either alone.

This case offers support for P1 to P3 at the F1, O1 and T1 levels. P4 is untested, as are the cohort effects set out in Section 9.

7. Discussion

7.1 Contributions

For end-user software engineering, the case extends the population usually studied (Ko et al., 2011) to a founder directing production-scale construction through conversational AI tools on a phone. For accessibility research, it shows a mobile device serving not as an assistive add-on to a desktop workflow but as the workflow itself, made possible by moving computation to the cloud. For entrepreneurship research, it documents a founder converting consulting knowledge into a product under the capital constraints described by Fairlie and Robb (2008) and the formation barriers described by Renko et al. (2016).

Prior research documents each element of this case on its own: programming on mobile devices (Tillmann et al., 2011), low-code construction (Bock & Frank, 2021) and its accessibility barriers (Khalajzadeh & Grundy, 2025), conversational AI development (Sarkar & Drosos, 2025), and disabled programmers' use of AI assistants (Flores-Saviaga et al., 2025; Chen et al., 2026). The searches recorded in Appendix A found no equivalent account of these practices operating together in a founder-directed, multi-module commercial platform. HASE is therefore proposed as a descriptive and testable model of domain-expert-directed construction under accessibility constraints, not as a wholly new programming technique or a validated engineering method; comparative research is needed to test its transferability and effectiveness.

To the author's knowledge, this is also the first documented case of its kind led by a woman of color. The claim concerns the documented record, not every founder who has built software; undocumented cases may exist, and Appendix A lists related cases that each differ on at least one element.

7.2 Rival explanations

Four rival explanations are considered. First, the capabilities of the AI-assisted development platform, rather than the founder's direction, may account for much of the implementation; HASE accepts this and locates the founder's contribution in problem ownership, requirements translation and release judgment, which the tools do not supply. Second, the founder's prior experience may explain the product's coherence more than the method does; the case cannot separate the two, which is why P1 and P4 call for comparative study. Third, timing matters: the case occurs when AI-assisted builders had become capable of full-stack applications, so it may

not generalize to earlier tools. Fourth, differences between preview and production environments may overstate what is live; the acceptance run in Section 9 addresses this.

7.3 Implications

For founders with disabilities, the case suggests that the barrier to directing software construction may now lie less in the device than in domain knowledge, requirement clarity and access to cloud tools. For programs that support small and diverse suppliers, it suggests a design pattern: tools that turn readiness services delivered by hand into structured, reusable records, while labeling unverified claims as reported rather than verified.

8. Limitations

The study is a single case, so its findings are analytic, not statistical. The author is the subject, and most evidence is founder-reported (F1). Proprietary processes are withheld, which limits technical scrutiny to outcomes, scale and behavior. Production confirmation for later releases is pending. Adoption figures are early and small and lack activation and retention measures; activation, first-tap and 30-day return measures are scheduled for a platform deployment within 30 days and will be reported in a later version; the six paying subscribers paid in cash, and in-platform checkout opened only on 12 November 2026. Sales and capital figures are unaudited. The literature search in Appendix A is preliminary; a structured database search has not yet been run, and a protocol for it is set out there. The platform’s own interface has not been audited against the Web Content Accessibility Guidelines; an accessible development process does not by itself guarantee an accessible product. Some device evidence relies on manufacturer documentation and reviews, not measured use.

9. Evaluation design

Four steps would raise the evidence level of the case.

First, an independent technology specialist, who may use AI-assisted analysis tools but remains responsible for the conclusions, can confirm the build's dates, scale and tools by rerunning the repository measurement, test platform behavior on Android, iPhone and devices without NFC, and check public registries, without access to proprietary processes. Non-public business records can be reviewed under a confidentiality agreement. Automated code-quality and security scans, run under the same terms, would add further measures. Only summary results would be published, never code.

Second, a paid-plan production acceptance run would test checkout, entitlements, document generation and registry checks in production.

Third, external WCAG 2.2 and security reviews would test accessibility and the AI features against the criteria in Section 2.6.

Fourth, a prospective cohort study would test P4 and two questions this case cannot answer: to what extent one integrated identity and documentation system reduces procurement-readiness friction, and whether combining education with software improves procurement-readiness outcomes compared with either alone. Small businesses would be recruited with baseline readiness measures and assigned to platform plus curriculum, curriculum only, or ordinary support. Outcomes would be documentation completeness, accuracy, time to assemble a bid packet, qualified bids submitted, awards and participant confidence, with pre-specified definitions, informed consent, data minimization and an external evaluator. The readiness score would be calibrated against expert reviewers before use as an outcome.

10. Conclusion

This case documents a founder who converted two decades of hands-on business services into a multi-module platform, directing its construction from a smartphone as a disability accommodation. The evidence supports three findings: requirements were translated directly from services delivered by hand; the smartphone workflow was feasible because computation ran in the cloud; and the device platform shaped what could be built and tested. HASE names that practice and frames it as testable propositions. Whether this way of building widens participation in software production, and whether the resulting tools improve procurement outcomes for small suppliers, are questions for the evaluation this study sets out.

Declarations

Competing interests: The author is the founder and chief executive of Tapn.vip Inc. and Glossy Byrd Management LLC, which own the platform studied, and has a financial interest in it.

Funding: No specific funding was received for this research. Business grants previously received by Glossy Byrd Management LLC ($20,000 from Develop Wisconsin and $4,000 from The Business Council) supported business operations, not this study.

Data availability: Aggregate platform figures are from the public counter at tapnvip.com on 12 November 2026. Source code, prompts and proprietary processes are not available. Non-public business records, and a private appendix recording the repository measurement, its repository state and its commands (summarized in Appendix B), can be reviewed by an independent specialist under a confidentiality agreement.

Ethics: The study reports the author's own experience and aggregate, non-identifying platform counts. No individual user or client data are reported, client case studies are drawn on only in aggregate, and no human-subjects research was conducted.

AI disclosure: The research, ideas, data, case history, HASE framework and findings in this paper are the author's own. Under the author's direction, generative AI assistants, including Claude (Anthropic), Gemini (Google) and AI Mode in Google Search, were used to restructure and format her historical records and materials, draft and edit text from them, search the literature and check citations. The author reviewed the paper and is responsible for all of its content. Accessibility features used solely as assistive technology are not included in this disclosure.

Revision history: Version 1.0, 11 November 2026, initial posting. Version 1.1, 13 November 2026: replaces founder-reported build statistics with a repository measurement (T1) and discloses what a checkpoint is; reports quality signals, the hardware-optional pivot and the founder-supplied corpus; defines disability and accommodation as used here, with reference to the Americans with Disabilities Act; clarifies how the six paying subscribers paid and labels financial figures unaudited; states the corporate relationship; distinguishes HASE from Adaptive Software Development and hybrid adaptive development; adds a diagram of the HASE cycle and a comparison with end-user programming, low-code development and conversational AI coding;

adds a second literature search round and related work on vibe coding and disabled programmers; maps founder-supplied materials to platform components; adds the platform's evidence records (Appendix B); notes the scheduled activation and retention measures; adds the release v1.19 milestone to the chronology, corrects the description of registry checks in Section 4.3 (built, pending registry access keys), extends Section 3.2 sources to release v1.19.3, updates the platform figures in the abstract and Section 5.5 to the public counter on 12 November 2026, and adds ORCID and SSRN identifiers. Version 1.2, 15 November 2026: revises the novelty statement in the abstract and Section 7.1 so that it rests on the combination of practices rather than on any single element, and keeps the identity claim as a separate, qualified statement; adds a third literature search round and a planned scoping-review protocol (Appendix A); adds related work on the accessibility of low-code tools and AI-generated websites and an autoethnographic study of generative AI for accessibility (Section 2.1); updates Chen et al. to its published CHI 2026 version; and notes in Section 8 that the platform's interface has not been audited against the Web Content Accessibility Guidelines. Version 1.3, 22 November 2026: narrows the research questions to the four this case can address and moves the integrated-system and education-plus-software questions to the evaluation design (Section 9); states that design and reporting follow Runeson and Höst's (2009) case study guidelines (Section 3.1); adds ability-based design (Wobbrock et al., 2011) to the related work (Section 2.4); and adds automated code-quality and security scans, reported as summary results only, to the evaluation design; and adds examples of the founder rejecting or correcting the build agent's output (Section 5.4).

References

Adobor, H., & McMullen, R. S. (2007). Supplier diversity and supply chain management: A strategic approach. Business Horizons, 50(3), 219–229.

Apple Inc. (n.d.). Adding support for background tag reading. Core NFC documentation. Retrieved November 10, 2026, from https://developer.apple.com/documentation/corenfc/adding-support-for-background-tag-reading

Autio, C., et al. (2024). Artificial intelligence risk management framework: Generative artificial intelligence profile (NIST AI 600-1). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.AI.600-1

Beaufort, F. (n.d.). Interact with NFC devices on Chrome for Android. Chrome for Developers. Retrieved November 10, 2026, from https://developer.chrome.com/docs/capabilities/nfc

Bock, A. C., & Frank, U. (2021). Low-code platform. Business & Information Systems Engineering, 63(6), 733–740. https://doi.org/10.1007/s12599-021-00726-8

Can I use. (2026). Web NFC: Browser support tables. Retrieved November 10, 2026, from https://caniuse.com/webnfc

Chen, N., Qiu, L. K., Wang, A. Z., Wang, Z., & Yang, Y. (2026). Programmers who use screen readers in the vibe coding era: Adaptation, empowerment, and new accessibility landscape. In Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems. ACM. https://doi.org/10.1145/3772318.3790726

Fairlie, R. W., & Robb, A. M. (2008). Race and entrepreneurial success: Black-, Asian-, and White-owned businesses in the United States. MIT Press. https://doi.org/10.7551/mitpress/7961.001.0001

Flores-Saviaga, C., Hanrahan, B. V., Imteyaz, K., Clarke, S., & Savage, S. (2025). The impact of generative AI coding assistants on developers who are visually impaired. In Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems. ACM. https://doi.org/10.48550/arXiv.2503.16491

Glazko, K., Yamagami, M., Desai, A., Mack, K. A., Potluri, V., Xu, X., & Mankoff, J. (2023). An autoethnographic case study of generative artificial intelligence's utility for accessibility. In Proceedings of the 25th International ACM SIGACCESS Conference on Computers and Accessibility. ACM. https://doi.org/10.48550/arXiv.2308.09924

Gill, A. Q. (2014). Hybrid adaptive software development capability: An empirical study. Journal of Software, 9, 2614–2621.

GSMArena. (2024). Samsung Galaxy S24 Ultra: Full phone specifications. Retrieved November 10, 2026, from https://www.gsmarena.com/samsung_galaxy_s24_ultra-12771.php

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. MIS Quarterly, 28(1), 75–105. https://doi.org/10.2307/25148625

Highsmith, J. A. (2000). Adaptive software development: A collaborative approach to managing complex systems. Dorset House.

Khalajzadeh, H., & Grundy, J. (2025). Accessibility of low-code approaches: A systematic literature review. Information and Software Technology, 177, 107570. https://doi.org/10.1016/j.infsof.2024.107570

Ko, A. J., Abraham, R., Beckwith, L., Blackwell, A., Burnett, M., Erwig, M., Scaffidi, C., Lawrance, J., Lieberman, H., Myers, B., Rosson, M. B., Rothermel, G., Shaw, M., & Wiedenbeck, S. (2011). The state of the art in end-user software engineering. ACM Computing Surveys, 43(3), Article 21. https://doi.org/10.1145/1922649.1922658

OWASP. (2025). OWASP top 10 for large language model applications. OWASP Foundation.

Palmer, Z. B., & Oswal, S. K. (2025). Constructing websites with generative AI tools: The accessibility of their workflows and products for users with disabilities. Journal of Business and Technical Communication, 39(1). https://doi.org/10.1177/10506519241280644

Peng, S., Kalliamvakou, E., Cihon, P., & Demirer, M. (2023). The impact of AI on developer productivity: Evidence from GitHub Copilot (Working paper). arXiv. https://doi.org/10.48550/arXiv.2302.06590

Renko, M., Parker Harris, S., & Caldwell, K. (2016). Entrepreneurial entry by people with disabilities. International Small Business Journal, 34(5), 555–578.

Rethlefsen, M. L., Kirtley, S., Waffenschmidt, S., Ayala, A. P., Moher, D., Page, M. J., Koffel, J. B., & PRISMA-S Group. (2021). PRISMA-S: An extension to the PRISMA statement for reporting literature searches in systematic reviews. Systematic Reviews, 10, Article 39. https://doi.org/10.1186/s13643-020-01542-z

Runeson, P., & Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering. Empirical Software Engineering, 14(2), 131–164. https://doi.org/10.1007/s10664-008-9102-8

Samsung Electronics. (n.d.). Samsung DeX. Retrieved November 10, 2026, from https://www.samsung.com/us/dex/

Samsung Electronics UK. (2024). How to use Transcript Assist on Galaxy S24. Retrieved November 10, 2026, from https://www.samsung.com/uk/support/mobile-devices/how-to-use-transcribe-assist-on-the-galaxy-s24

Sarkar, A., Gordon, A. D., Negreanu, C., Poelitz, C., Ragavan, S. S., & Zorn, B. (2022). What is it like to program with artificial intelligence? In Proceedings of the 33rd Annual Conference of the Psychology of Programming Interest Group (PPIG 2022). https://doi.org/10.48550/arXiv.2208.06213

Sarkar, A., & Drosos, I. (2025). Vibe coding: Programming through conversation with artificial intelligence. In Proceedings of the 36th Annual Conference of the Psychology of Programming Interest Group (PPIG 2025). https://doi.org/10.48550/arXiv.2506.23253

Shah, M., & Ram, M. (2006). Supplier diversity and minority business enterprise development: Case study experience of three US multinationals. Supply Chain Management, 11(1), 75–81. https://doi.org/10.1108/13598540610642493

Temoshok, D., et al. (2025). Digital identity guidelines (NIST SP 800-63-4). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-63-4

Tillmann, N., Moskal, M., de Halleux, J., & Fähndrich, M. (2011). TouchDevelop: Programming cloud-connected mobile devices via touchscreen. In Proceedings of the 10th SIGPLAN Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software (Onward! ’11) (pp. 49–60). ACM. https://doi.org/10.1145/2048237.2048245

Tricco, A. C., Lillie, E., Zarin, W., O’Brien, K. K., Colquhoun, H., Levac, D., Moher, D., Peters, M. D. J., Horsley, T., Weeks, L., Hempel, S., Akl, E. A., Chang, C., McGowan, J., Stewart, L., Hartling, L., Aldcroft, A., Wilson, M. G., Garritty, C., … Straus, S. E. (2018). PRISMA extension for scoping reviews (PRISMA-ScR): Checklist and explanation. Annals of Internal Medicine, 169(7), 467–473. https://doi.org/10.7326/M18-0850

von Hippel, E. (2005). Democratizing innovation. MIT Press.

WAKDEV. (2026). NFC Tools [Mobile application]. App Store. Retrieved November 10, 2026, from https://apps.apple.com/us/app/nfc-tools/id1252962749

Wobbrock, J. O., Kane, S. K., Gajos, K. Z., Harada, S., & Froehlich, J. (2011). Ability-based design: Concept, principles and examples. ACM Transactions on Accessible Computing, 3(3), Article 9, 1–27. https://doi.org/10.1145/1952383.1952384

World Wide Web Consortium. (2024). Web Content Accessibility Guidelines (WCAG) 2.2. https://www.w3.org/TR/WCAG22/

World Wide Web Consortium. (2025). Verifiable Credentials Data Model v2.0 (W3C Recommendation). https://www.w3.org/TR/vc-data-model-2.0/

Yin, R. K. (2018). Case study research and applications: Design and methods (6th ed.). SAGE.

Appendix A. Literature and case search log

Round 1, 10 November 2026: preliminary web searches during manuscript preparation, plus verification of the references.

Round 2, 13 November 2026: web searches reaching publisher, preprint (arXiv) and research-lab pages, run to test the novelty claim before peer review. Terms: “TouchDevelop programming on smartphone”; “vibe coding empirical study”; “non-programmer founder built web application AI coding agent case study”; “disabled entrepreneurs generative AI software development”; “screen reader users vibe coding”; “programming on mobile phones only, developing countries”; “Black women founders AI no-code”; “low-code citizen developers systematic review generative AI”.

Round 3, 15 November 2026: targeted searches of publisher and repository pages on the accessibility of low-code and AI-generated software, and confirmation of the published version of Chen et al. Terms: “accessibility of low-code approaches systematic literature review”; “screen reader programmers vibe coding CHI 2026”; “generative AI website builders accessibility users with disabilities”; “autoethnography generative AI accessibility”. Sources added: Khalajzadeh and Grundy (2025); Palmer and Oswal (2025); Glazko et al. (2023); the CHI 2026 version of Chen et al.

The ACM Digital Library, IEEE Xplore and Scopus have not yet been queried directly. A scoping review following PRISMA-ScR principles (Tricco et al., 2018) and PRISMA-S search reporting (Rethlefsen et al., 2021) is planned under protocol HASE-LR-2026-01. It will search those databases from 2000 to the search date with concept-specific strings built from (“end-user programming” OR “citizen development” OR “low-code” OR “AI-assisted programming” OR “vibe coding”) AND (accessibility OR disability OR smartphone OR mobile OR entrepreneur OR founder), adapted to each database’s syntax. Eligible sources are peer-reviewed studies, conference papers, foundational books and clearly identified preprints that address one or more HASE components. The screening record will list each database, date, search string, result count, duplicates, exclusions, full-text decisions and included studies, and each included study will be compared against the dimensions of the HASE model in Section 6.

Search terms: “disabled founder built software platform smartphone AI no-code”; “Black woman founder disability built app on phone using AI”; “end-user programming smartphone disability accommodation entrepreneur software development”.

Result of the three rounds: no study or published case was found of a woman-of-color founder who directed the development of a working business platform from a smartphone as a disability accommodation. Related cases and how each differs:

Related case	What it shows	How it differs from this case
A founder with ADHD who built a productivity app with an AI app builder (news report, 2026)	Non-programmer founder building with AI	Not reported as built on a phone or as an accommodation; single app
A designer who built a mobile app with a no-code builder (vendor customer story)	Non-programmer building a mobile app	No disability or accommodation reported; vendor material
Blind and low-vision software professionals using generative AI (preprint, 2025)	AI as an accessibility aid in development	Professional programmers, not non-programmer founders
End-user programming tools for blind users (preprint, arXiv 2408.10499)	Users building their own assistive programs	Personal tools, not a commercial platform
A founder with chronic illness who launched an accessible dating platform (agency case study)	Disabled founder launching a platform	Built by an outside agency, not directed by the founder from a phone
TouchDevelop (Tillmann et al., 2011)	Programs written and run on a phone	Pre-language-model; students and hobbyists; own touch language
Phone-based programming for students in Kenya and South Africa (university studies, 2015–2016)	Phones as the only available programming device	Learners writing Java with scaffolding, not AI-directed platforms
Vibe coding study (Sarkar & Drosos, 2025)	Conversational, AI-driven construction	Programmers at computers; no disability or founder focus
Screen reader users with AI assistants (Chen et al., 2026; Flores-Saviaga et al., 2025)	AI assistants with disabled programmers	Programmers at computers, not a non-programmer founder on a phone
Researchers with disabilities using generative AI for their own access needs (Glazko et al., 2023)	First-person, autoethnographic study of AI and accessibility	Researchers' own access needs, not the construction of a commercial platform
Non-technical founders building with AI app builders (trade press and vendor stories, 2026)	Founders shipping products without engineers	Built at computers; no disability or accommodation reported
Black women founders building with no-code tools (vendor and trade press)	Women of color founders building their own apps	Visual no-code builders at computers; no disability or phone-directed workflow reported

Several sources are news items, vendor material and preprints rather than peer-reviewed studies. They are listed to show what the searches found and why the novelty claim is qualified: each related case shares one or more elements with this case, and none combines a non-programmer founder, a phone as the primary workstation, AI-directed production construction, disability accommodation and a woman-of-color founder.

Appendix B. Evidence record from the development platform

The development platform (Emergent) supplied the following records. They come from the vendor whose platform is part of the case, so they are reproducible records rather than independent verification.

Record	Detail	Level
Repository measurement	Run read-only by the platform's build agent on 13 November 2026 at 14:47:26 UTC	T1
Repository state measured	Commit 5d94c7429fd60759e5edd97dac3d828f857a25ad; latest checkpoint 13 November 2026, 14:45:18 UTC	T1
Measurement record fingerprint	SHA-256 7a6464ede0b99117c5df562902248548bfb4578252ac162002b4dc86d3887d32	T1
Scope of the attestation	The agent attests that it ran the recorded commands against that repository state and reported their output without manual change. It does not attest to authorship of repository contents, and it certifies measurement, not peer review.	T1
Automated test reports	91 test iterations by the platform's testing agent, stored in the repository	T1
Security audit	Read-only audit, 10 November 2026: no critical or high-severity findings	F1
Production deployment	Release v1.19 live on tapnvip.com, confirmed in writing by the platform's support team, 12 November 2026	F1

The measurement commands, internal platform identifiers and the full audit are held privately and are available to a reviewer under a confidentiality agreement.