Epidemica: An Open, Agentic-Ready Research Platform for Participatory Epidemiology Studies

Andrés Colubri

Abstract

Epidemica is an open-source platform for building study apps that collect high-resolution, multi-modal epidemiological data — including proximity contacts, survey responses, and, in a companion transmission engine, a simulated infection outcome — and for running interventions over that data. It is organised as a set of versioned data and protocol contracts, implemented once each in Dart client packages, an Elixir/Phoenix server, and a Python transmission model, so that a study is authored as a configuration document rather than a fork of an app. This paper describes the platform's motivation, its architecture, a reference application (a seven-day transmission game whose infections are decided by a real contact network rather than scripted), and a purposeful extension of that same contracts-first discipline to the agents that help build it: machine-readable convention files that let a coding agent work in this codebase without re-deriving rules that have already been paid for in debugging time. We report the platform's current state and discuss next steps and further directions. Epidemica is available under the Apache-2.0 license at: https://github.com/colabobio/epidemica

ark:24975/genrxiv-2026-00002

1. Introduction

Digital tools for infectious disease surveillance improved considerably over the last decade, accelerated by the COVID-19 pandemic's demand for contact tracing at population scale [1–3]. What emerged, however, was mostly single-purpose: an app that traces contacts, a survey platform that collects symptoms, a simulation package that models spread — each built once, for one study, rarely reused by the next. A research group that wants to measure a contact network, ask participants how they felt about it, and feed the result into a transmission model still assembles that pipeline from parts that were never designed to fit together, and largely does so again for the next study.

Epidemica is an attempt at the alternative: infrastructure a study is *assembled from* rather than *built against*. Its central claim is that this is possible without sacrificing the two things a one-off app usually gets right by construction — a data model that means exactly what the study needed it to mean, and a epidemiological model a specific research question actually calls for — provided the boundary between "what the platform guarantees" and "what a study defines" is drawn as an explicit, versioned **contract** rather than left implicit in whichever code happens to run.

This paper describes that contract-first design, the reference implementation built against it, and a companion application — a short transmission game in which a phone's own measured contacts, not a scripted schedule, decide who a server-side epidemic model infects next. It also describes an extension of the same discipline to a newer kind of collaborator: the coding agents now routinely used to build and maintain a codebase like this one [4], for whom the same lesson applies — an undocumented convention re-learned by trial and error is exactly the failure mode contracts exist to prevent, whether the reader is a person or a language model.

We do not report epidemiological results. Epidemica has not yet run a study at the scale or duration its own milestones require, and this paper is explicit about that — see §8. What we report is a platform whose internal seams have been exercised end to end with real devices, whose behaviour is pinned down by an automated test suite in three languages, and whose architecture we believe generalises past its first reference application.

2. Related Work

Structured, empirical contact data has driven epidemiological modeling for decades. Diary-based studies such as POLYMOD established that contact patterns are age-structured and setting-dependent at a scale still cited in transmission models today [5], and RFID- or Bluetooth-based proximity studies — the Copenhagen Networks Study among the largest — showed that sensor-derived contact networks capture structure a diary cannot, at the cost of purpose-built, non-reusable infrastructure for each deployment [6]. The COVID-19 pandemic then produced a wave of literature on digital contact tracing specifically: modeling work argued that population-scale Bluetooth tracing could plausibly control an epidemic if adopted widely enough [2], and real-world deployment of contact tracing apps demonstrated measurable reductions in transmission [7], while comparative studies of tracing, testing and distancing measured what such interventions achieved in practice [3]. Classical network epidemiology, meanwhile, had already established that *which* contact network a model assumes changes its qualitative predictions, not just their magnitude [8] — an argument for measuring the real network rather than assuming a stylised one, which is Epidemica's proximity module's entire job.

On the modeling side, agent-based transmission simulators — Covasim and the wider Starsim family from the Institute for Disease Modeling being a widely used example — demonstrated that a general disease-and-network simulation framework, not a bespoke model per study, can serve a whole research program [9–11]. Epidemica adopts this position directly: rather than writing its own transmission mathematics, it treats an external simulator as the canonical authority and limits its own scope to getting a measured network and a study protocol into that simulator's native shape.

Research data platforms such as REDCap solved an adjacent problem — a

metadata-driven, reusable instrument for capturing structured research data across studies without bespoke software per project [12] — for questionnaire-style data. Epidemica's survey module is deliberately compatible in spirit (a versioned instrument definition, closed-response items, one contract every study reuses) while extending the same reusable-instrument idea to passively sensed data, which a form-based tool was never built to hold. FAIR data principles [13] motivate a design choice that runs through the whole platform: every observation carries a schema URI that resolves to a machine-readable description of what it means, so that a dataset remains interpretable independent of the software that produced it.

Epidemica's own lineage includes two prior single-purpose platforms built by the same lab — Operation Outbreak, a proximity-sensing outbreak simulation app originally developed by our group and deployed at over hundred of schools [14], and Travel Healthy, a participatory surveillance app for international travelers [15] — plus an early Epidemica-based prototype, Epigames, shown publicly as a proof of concept at the International Pandemic Sciences Conference in 2025 [16], and multiple empirical deployments of epigames in university campus settings that demonstrated measurable effects of risk perception and behavioural interventions on simulated disease transmission dynamics [17, 18]. The scientific rationale and broader vision for epigames as behavioural research instruments is further developed in a recent opinion piece in *Nature Health* [19]. Each platform solved its own problem well and duplicated infrastructure the next one needed again; Epidemica is the generalisation of what those projects had in common.

3. Design Principles

Four decisions shape everything described in the sections that follow.

Contracts before code. Epidemica's primary artifact is a versioned set of JSON Schema contracts — what an observation looks like, what a study protocol declares, what the server publishes back to a participant — with reference implementations of those contracts, not the contracts *as* an implementation detail of one. A module, in this platform, is not "a package that happens to collect proximity data"; it is a package that owns a payload contract, is activated and configured at runtime by a study's protocol document, and emits observations into a shared, append-only store without ever touching another module's data. This is what makes a second module (surveys, described in §5) additive rather than a rewrite, and it is what makes a study author's job writing a configuration document rather than forking an app.

One ingest path for every kind of data. Proximity episodes, survey answers, and a module's report of its own health all travel in the same *observation envelope*: a wrapper the platform controls (who, which study, which protocol version, when) around a payload the module controls and the platform never inspects. One sync engine, one offline queue, one retry policy, one export pipeline, regardless of how many kinds of module a study combines.

One canonical transmission model, not our own. Epidemica does not implement epidemiology. A study's transmission parameters are constrained to be directly loadable as parameters of an external, general-purpose simulator such as [11], and a measured contact network is exposed to that simulator as a first-class network object rather than reimplemented as a parallel, drift-prone mathematical model maintained in two languages. This was motivated by the earlier approach in the pilot Epigames app, which specified both a server-side and an on-device transmission implementation, and the risk that the two would silently disagree was judged worse than the cost of running the canonical model server-side and treating the phone as measurement, not computation.

A study's identity is its bytes. A protocol bundle is a signed, versioned configuration document, and the server stores and serves its exact bytes rather than a re-derived copy. A participant's device verifies the bundle's hash before activating any module. The consequence that matters most in practice: a study's identity *is* its bundle's hash, so a change to a study's rules — intentional or a typo — creates a new, distinct study that participants must (re-)join, rather than silently mutating one they already joined mid-run.

4. Architecture

Figure 1 traces one observation's round trip through the system. A study app embeds one or more **modules** at build time — proximity sensing over Bluetooth Low Energy via the Herald protocol library [20], and scheduled survey instruments are the two shipped today — each of which is handed a narrow `ModuleContext`: its own configuration block from the protocol bundle, the study and participant identifiers, a recorder, and nothing about any other module. Observations accumulate in a durable, on-device outbox (SQLite in write-ahead-logging mode, chosen after an earlier design's JSON-blob queue proved to cost an unbounded rewrite per append and offer no retry semantics) and are drained by a sync service whose only job is one upload pass; when to call it is left to whatever manages the platform's background execution, purposefully, because that policy differs by operating system and by how aggressively a study needs data in near-real-time.

One observation's path from a phone to a published conclusion and back.

Figure 1: One observation's path from a phone to a published conclusion and back.

The server is a single Phoenix/Elixir application over PostgreSQL. Ingest is a high-volume REST path built for constrained clients on unreliable networks: batched, idempotent, authenticated by short-lived, per-device tokens rather than a shared API key. Every accepted observation lands in one append-only table; everything else — a contact list, a coverage judgement, a day's simulated outcome — is a *projection* derived from it, and is required to be exactly reproducible from the observation store alone. This is what makes the observation

store the platform's actual system of record rather than one copy of the truth among several: if a derived table and a full projection rebuild ever disagree, the derived table is wrong by definition.

For studies that include a transmission model, a **twin** runtime settles one study-day at a time by handing a single JSON document — the day's reconciled contact network, the participants' declared choices, and each affected agent's prior state — to a Python subprocess that runs one simulated day against the canonical transmission model and returns the updated states. The settlement this produces is immutable: a study day, once decided, is never re-simulated against different inputs, because participants are told its result and a live study cannot retract a conclusion it has already published. Conclusions reach a participant's device through a **state channel** — a single document the server publishes and the client can only read, never negotiate — which keeps a hard line between what the platform has concluded (a settled score, a coverage judgement) and what is true right now (a radio is or is not scanning), a distinction that proved easy to blur in practice and is discussed further in §8.

Every payload contract, the protocol bundle format, and the participant-facing state documents are JSON Schema, closed to additional properties, with machine-checked valid and invalid examples for each. This is not incidental tooling: it is what lets analysis code, generated client types, and server-side validation all be produced from — and stay honest to — one specification rather than three independently maintained approximations of it.

5. Epigames: A Digital Twin Reference Application

The platform's first non-trivial application is a short transmission game: over seven simulated days, participants carry a phone that senses nearby participants, choose each day whether to spend points protecting themselves, and are scored on contacts made and avoided — while a real epidemic, seeded onto the *measured* contact network the participants themselves produced, decides who is actually infected. It exists to prove a specific claim about the platform: that the same client modules and the same server, unmodified, can run both a plain contact-logging study and a study with game mechanics layered on top, so that "a module" is a genuine reusable unit and not a description that only happens to hold for one app.

Three properties of the design carry the whole game. First, every infection is attributable to a specific, reconciled contact or to the study's *virtual population* — participants injected into the simulation to complete an epidemiologically plausible population size around a real cohort too small to sustain transmission on its own — never to an unexplained default. Second, the network aggregates both sides of every encounter server-side, which is strictly more information than either phone has alone and is why the earlier on-device transmission design (§3) was set aside for this application. Third, a settled day's score never changes retroactively except by an explicit, audited carry-over rule that recomputes

what was actually true on each earlier day it revisit.

Two extensions built on the same reference application illustrate how a study author varies a platform-provided mechanism without touching platform code. **Arms** let a study randomise its economics — different point values or protection costs — across participants drawn by a weighted, deterministic assignment at enrollment, with every rule resolved through one function that folds a study's defaults, its declared parameters, and a participant's arm into the parameters that actually govern their day; no scoring path is allowed a second, competing notion of what a participant's rules are. **Scheduled surveys** close a gap sensing alone cannot: the sensors measure what happened, not what a participant knew, believed, or felt about it, which for a study whose research question is about understanding transmission is often the actual outcome of interest. Survey items are restricted to closed responses by construction — no free-text field exists in the instrument contract — which turns "the store should be pseudonymous" from a policy statement into a structural guarantee the schema itself enforces.

6. Agentic Readiness

A platform built as an explicit contract between components generalises, we argue, to a contract between a codebase and the coding agents that increasingly help build and maintain it. The same failure mode that motivated §3 — a rule known only by whoever last debugged it, silently violated by the next change — recurs verbatim when the "next change" is proposed by an autonomous agent with no memory of the debugging session that established the rule in the first place.

Epidemica addresses this with the same discipline it applies to its data contracts: writing non-obvious knowledge down, once, in a place an agent is specified to read before acting. Two converging conventions serve this purpose. `AGENTS.md` is a plain-Markdown, vendor-neutral convention supported by multiple coding-agent tools, with nested files resolved nearest-first so that subdirectory-level conventions can override or extend the root. `CLAUDE.md` is a related, tool-specific convention that concatenates a hierarchy from managed policy to project-local file, and prefers brevity — content an agent can derive from the codebase should not be restated, leaving space for what cannot: pitfalls, rationale, and hard-won conventions. It also adds pointers to reusable Agent Skills — named, invokable procedures for recurring workflows.

This section is itself an instance of its own argument: this manuscript was substantially drafted by a coding agent operating under exactly these conventions, and the venue we submit it to — an archive built on the premise that AI involvement in research writing is the default rather than an exception requiring disclosure [21] — is evidence that the same contracts-first instinct extends naturally from data formats, to software interfaces, to documents that describe both to a reader that may not be human.

7. Safeguards for Agent-Assisted Development

Agentic readiness cuts in a second direction. A codebase built substantially by coding agents inherits two risks a documented convention file does not, by itself, address: an agent can reproduce a distinctive snippet from its training data under terms this project's Apache-2.0 license cannot absorb [22], and a body of code produced primarily through prompting invites the question of whether it carries the human creative control copyright protection generally requires. Epidemica treats both as engineering problems: name the risk, build a check, and state plainly what the check does not cover.

For the first, an open-source license-text detector is run against the repository's own source, flagging GPL-, AGPL-, or LGPL-family matches in project source for human review rather than auto-rejecting, on the reasoning that blocking every low-confidence match trains people to ignore the report. The check is explicitly scoped: it is a text-matching tool, not a semantic clone detector, and a clean run is not evidence that no code was derived from anything — a limitation the project states rather than elides. It runs on demand rather than in continuous integration, a deliberate choice recorded, like everything else reported here, rather than quietly walked back.

For the second, a companion tool renders a coding session's raw interaction log into a readable transcript — every message from both sides in full, tool invocations reduced to one line each — on the premise that the record of direction given, alternatives rejected, and output reviewed and revised is the evidence a human-authorship claim rests on, not the final diff. Neither tool is specific to this repository; a research group building their own study app can point either at their own source.

A third safeguard generalises furthest: guidance addressed to platform adopters states what license obligations follow from building on Apache-2.0 code, what a copyleft dependency does to a combined binary, and what agent-assisted development specifically asks of a study handling IRB-governed data — most concretely, that a general-purpose coding agent's context window is not bound by a study's own privacy protocol, and real participant data has no more business there than it does in a screenshot posted to a public forum. Writing this down follows the same instinct that produced the observation envelope in §4: the platform's job is to make the correct choice the legible one.

8. Current Status and Validation

Epidemica's status is best described as an *alpha*: the complete arc from sensing to a published, settled conclusion runs end to end, and has been exercised on real devices for two reference study types (a plain contact-logging study, and the transmission game of §5) — but it has not been operated unattended, at the scale or duration its own milestone documents specify as acceptance criteria, or self-hosted anywhere outside its own development environment. The platform

is under active development and publicly available at https://github.com/colabobio/epidemica.

More concretely, the platform's cross-language test suite — Elixir server, Dart client packages, and the Python transmission bridge — currently comprises over one thousand automated tests, with static analysis to report issues across the Elixir, Dart, and Python codebase. Ten JSON Schema contracts exist, each verified against machine-checked valid and invalid examples by a self-discovering test harness that requires every violation to be traceable to a single, named rule; two further contracts — the device-facing ingest API and the Bluetooth wire format — extend that same versioned-contract discipline to interfaces JSON Schema does not fit. The transmission game's field exercise used a compressed, purpose-built variant of its own study protocol — rounds of minutes rather than a day, so that a full arc could be exercised repeatedly against real Bluetooth hardware.

9. Applications and Future Directions

The reference application in §5 exercises one point on a wider design space the architecture was built for. The same contract — an observation envelope, a protocol bundle, a module boundary — should in principle support a plain observational cohort study using only the proximity module; a randomised controlled trial using the arms mechanism without any game mechanics at all; and a study whose research question is squarely about behaviour, using the survey module to pair passively sensed contact data with participants' stated beliefs and choices about it — a combination the platform's motivating research questions treat as a genuine current gap: how contact duration relates to measured infection risk, how digital contact-tracing strategies compare against each other, and how individual behavioural choices aggregate into population-level transmission outcomes are all questions that need both signals together, not either alone.

Further out, the architecture's roadmap anticipates channels beyond an installed app — SMS, voice, and plain mobile web — on the position that enrollment, consent, and instruments should not require a smartphone app even where sensing modules do, which matters directly for reaching participants in lower-resource settings where continuous app usage cannot be assumed. It also anticipates connectors to research-data infrastructure already in wide use — REDCap [12] and HL7 FHIR [23] among them — on the position that a platform for collecting novel data types should not require abandoning the infrastructure a research group already has for the data types it already knows how to handle. A further direction, particularly relevant for pandemic preparedness, is rapid adoption of standardised early-epidemic data schemas: the Global.health core schema [24], a minimum interoperable dataset for the first hundred days of an outbreak aligned with WHO reporting standards, could be implemented as a Tier 1 protocol bundle in Epidemica, enabling research groups to deploy studies that feed directly into established outbreak-response data pipelines without bespoke integration

work. None of this is built yet; it is named here because the architecture in §4 was deliberately shaped to make it additive rather than a redesign.

10. Limitations

Beyond the validation gaps named in §8, three limitations are structural rather than incidental. Epidemica's proximity module currently broadcasts a stable per-participant pseudonym in the clear over Bluetooth for its first reference deployment, which is an explicit, documented trade-off adequate for a consenting internal pilot and not for an externally facing study without rotating identifiers, a prerequisite the platform's own milestone record treats as scheduled rather than optional. The platform's canonical transmission model is an actively developed external dependency [9, 11]; the architecture's own risk register names pinning a model version per study protocol, and recording it in every export, as a required mitigation not yet implemented — an upstream release could otherwise change a study's results without the study's authors noticing. Finally, self-hosting — the platform's stated design goal of being deployable by a research group without dedicated infrastructure staff — has been exercised only in the authors' own development environment; whether the deployment tooling holds up when operated by someone who did not write it is, at the time of writing, untested and the single largest open question standing between the alpha status reported here and the platform's stated goal.

11. Conclusion

Epidemica generalises three single-purpose platforms built by the same lab into infrastructure a study is assembled from rather than built against, by drawing an explicit, versioned contract between what the platform guarantees and what a study defines, and by holding every component — a Bluetooth payload, a server endpoint, a transmission model's parameters, and, we argue, the conventions a coding agent needs to work safely in the codebase — to that same discipline. The platform's data and protocol contracts, reference implementations, and reference application are built, tested, and field-exercised end to end; what remains before its stated milestones are met is scale, duration, and independent operation, each named rather than assumed, and each the kind of empirical question this platform is specifically designed to help answer once it is put to that use.

AI-generated research. This article was generated or co-generated using AI and reviewed by the author(s) before submission to GenRxiv.

[1] J. A. Pandit, J. M. Radin, P. Niga, and E. J. Topol, "Smartphone apps in the COVID-19 pandemic," *Nature Biotechnology*, 2022, vol. 40, pp. 1013–1022.

[2] L. Ferretti, C. Wymant, M. Kendall, L. Zhao, A. Nurtay, L. Abeler-Dörner, M. Parker, D. Bonsall, and C. Fraser, "Quantifying SARS-CoV-2 transmission suggests epidemic control with digital contact tracing," *Science*, 2020, vol. 368, pp. eabb6936.

[3] A. J. Kucharski, P. Klepac, A. J. K. Conlan, S. M. Kissler, M. L. Tang, H. Fry, J. R. Gog, and W. J. Edmunds, "Effectiveness of isolation, testing, contact tracing, and physical distancing on reducing transmission of SARS-CoV-2 in different settings," *The Lancet Infectious Diseases*, 2020, vol. 20, pp. 1151–1160.

[4] MIT Sloan Management Review, "Agentic AI, explained," 2026, https://mitsloan.mit.edu/ideas-made-to-matter/agentic-ai-explained.

[5] J. Mossong, N. Hens, M. Jit, P. Beutels, K. Auranen, R. Mikolajczyk, M. Massari, S. Salmaso, G. S. Tomba, J. Wallinga, J. Heijne, M. Sadkowska-Todys, M. Rosinska, and W. J. Edmunds, "Social contacts and mixing patterns relevant to the spread of infectious diseases," *PLOS Medicine*, 2008, vol. 5, pp. e74.

[6] A. Stopczynski, V. Sekara, P. Sapiezynski, A. Cuttone, M. M. Madsen, J. E. Larsen, and S. Lehmann, "Measuring large-scale social networks with high resolution," *PLOS ONE*, 2014, vol. 9, pp. e95978.

[7] M. Kendall, D. Tsallis, C. Wymant, A. Di Francia, Y. Balogun, X. Didelot, L. Ferretti, and C. Fraser, "Epidemiological impacts of the NHS COVID-19 app in england and wales throughout its first year," *Nature Communications*, 2023, vol. 14, pp. 858.

[8] K. T. D. Eames, and M. J. Keeling, "Contact tracing and disease control," *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 2003, vol. 270, pp. 2565–2571.

[9] C. C. Kerr, R. M. Stuart, D. Mistry, R. G. Abeysuriya, K. Rosenfeld, G. R. Hart, R. C. Núñez, J. A. Cohen, P. Selvaraj, B. Hagedorn, L. George, M. Jastrzębski, A. S. Izzo, G. Fowler, A. Palmer, D. Delport, N. Scott, S. L. Kelly, C. S. Bennette, B. G. Wagner, S. T. Chang, A. P. Oron, E. A. Wenger, J. Panovska-Griffiths, M. Famulare, and D. J. Klein, "Covasim: An agent-based model of COVID-19 dynamics and interventions," *PLOS Computational Biology*, 2021, vol. 17, pp. e1009149.

[10] J. Panovska-Griffiths, B. Swallow, R. Hinch, J. Cohen, K. Rosenfeld, R. M. Stuart, L. Ferretti, F. Di Lauro, C. Wymant, A. Izzo, W. Waites, R. Viner, C. Bonell, C. Fraser, D. Klein, and C. C. Kerr, "Statistical and agent-based modeling of the transmissibility of different SARS-CoV-2 variants in england and impact of different interventions," *Philosophical Transactions of the Royal Society A*, 2022, vol. 380, pp. 20210315.

[11] Starsim Development Team, "Starsim: An agent-based modeling framework," 2026, https://starsim.org.

[12] P. A. Harris, R. Taylor, R. Thielke, J. Payne, N. Gonzalez, and J. G. Conde, "Research electronic data capture (REDCap)—a metadata-driven methodology and workflow process for providing translational research informatics support," *Journal of Biomedical Informatics*, 2009, vol. 42, pp. 377–381.

[13] M. D. Wilkinson, M. Dumontier, Ij. J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.-W. Boiten, L. B. da Silva Santos, P. E. Bourne, and others, "The FAIR guiding principles for scientific data management and stewardship," *Scientific Data*, 2016, vol. 3, pp. 160018.

[14] A. Colubri, M. Kemball, S. Kian, C. Boehm, K. Mutch-Jones, B. Fry, T. Brown, and P. C. Sabeti, "Preventing outbreaks through interactive, experiential real-life simulations," *Cell*, 2020, vol. 182, pp. 1366–1371.

[15] A. Colubri, N. Willing, A. Grozdani, Y. Dong, H. Hong, M. Khandpekar, E. Oliver, J. Thwing, E. T. Ryan, and R. C. LaRocque, "Travel healthy, a mobile app for participatory surveillance among u.s. International travelers," *Travel Medicine and Infectious Disease*, 2025, vol. 68, pp. 102922.

[16] A. Colubri, "Let the epigames begin: A MERS-x epidemic game at the international pandemic sciences conference," 2025, Medium, https://colabobio.medium.com/let-the-epigames-begin-a-mers-x-epidemic-game-at-the-international-pandemic-sciences-conference-f895b363c846.

[17] S. S. Musa, W. Mkandawire, T. Inekwe, Y. Dong, A. Grozdani, H. Hong, M. Khandpekar, S. A. Nowak, J.-G. Young, A. Wong, D. King, and A. Colubri, "App-based epidemic game in a university campus reveals how risk perception and behavioral interventions shape disease transmission dynamics," *Scientific Reports*, 2026, vol. 16.

[18] A. Colubri, A. Grozdani, M. Khandpekar, Y. Graytee, O. Al-Mohammedi, A. A. Al-Shabandar, W. Y. Shabeeb, Y. Ghassan, H. Swayedi, C. T. Bauch, J. Drury, J. Panovska-Griffiths, and D. King, "App-based epidemic game to model belief-behavior mapping and cost incentives in voluntary quarantine: A randomized controlled trial," 2026, medR*x*iv, https://doi.org/10.64898/2026.01.10.26343836.

[19] A. Colubri, D. Williams, T. Valente, C. T. Bauch, J. M. Drake, M. C. Mills, J. Drury, C. Fraser, L. Ferretti, and J. Panovska-Griffiths, "Understanding human behaviour for pandemic preparedness with epigames," *Nature Health*, 2026, vol. 1.

[20] A. Fowler, "Herald proximity: Open-source BLE proximity detection for mobile platforms," 2024, https://heraldprox.io/.

[21] R. Fenwick, "GenRxiv: An open archive for AI-generated research," 2026, GenRxiv, ark:99999/genrxiv-2026-00001, https://genrxiv.org/article/ark:99999/genrxiv-2026-00001.

[22] U.S. Copyright Office, "Copyright registration guidance: Works containing material generated by artificial intelligence," 2023, Federal Register, 88 FR 16190, https://www.federalregister.gov/documents/2023/03/16/2023-05321/copyright-registration-guidance-works-containing-material-generated-by-artificial-intelligence.

[23] HL7 International, "FHIR (fast healthcare interoperability resources)," 2026, https://www.hl7.org/fhir/.

[24] E. Kamau, S. Kelly, D. Darji, A. Y. Baidjoe, J. S. Brownstein, F. Campbell, A. Dasgupta, M.-A. Degail, A. Demidova, L. Ferretti, A. Han, S. L. Koyie, P. N. Ngamala, O. L. Polain, A. Rojek, J. Sauer, S. V. Scarpino, K. Sewalk, J. Sopko, S. Zakrzewski, L. Merson, and M. U. G. Kraemer, "Defining core early-epidemic data for interoperable outbreak response: The Global.health schema," *Wellcome Open Research*, 2026, vol. 10, pp. 524.