Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

HelixVault: A Full-Stack Web Application for DNA-Based Digital Data Encoding with AES-256 Encryption, Reed-Solomon Error Correction, and Biological Steganography

Nitish Pattar

nitishpattar7@gmail.com

AMC Engineering College, Bangalore

Research Article

Keywords: Synthetic DNA data storage, homopolymer prevention, Base-3 codec, Reed-Solomon RS(255, 205), AES-256 encryption, biological steganography, Luby Transform codes, plasmid cloning, biosecurity screening, FastAPI, React

Posted Date: November 20th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11785146/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

HelixVault: A Full-Stack Web Application for DNA-Based Digital Data Encoding with AES-256 Encryption, Reed-Solomon Error Correction, and Biological Steganography

Nitish Pattar[1]
[1]Department of Computer Science & Applications
Administrative Management College, Bengaluru, India
Master of Computer Applications (MCA) Program
nitishpattar7@gmail.com

Abstract -- *Digital data is expanding much faster than our ability to store it. Magnetic tapes and hard drives wear out every few years, and running massive air-conditioned server farms consumes enormous electrical power. DNA offers a compelling way out: it packs petabytes into microscopic volumes and stays readable for centuries without electricity. However, moving DNA storage from wet-lab experiments into usable software remains difficult. Most existing tools are isolated scripts with no web interface, zero encryption, and no mechanism to protect files against physical degradation. In this work, we present HelixVault, an open-source web platform that handles the full digital-to-DNA lifecycle. We wrote a custom Base-3 codec that solves the homopolymer problem: chemical synthesizers struggle when identical nucleotides repeat, so our mapping strictly prevents identical consecutive bases by construction. Before encoding, files are encrypted with AES-256-CBC using PBKDF2 key derivation. We then apply Reed-Solomon RS(255, 205) error correction, which can rebuild up to 25 damaged bytes in each codeword. For security and camouflage, HelixVault can also hide payloads inside real E. coli genomic sequences pulled from NCBI. The platform includes an interactive plasmid cloning tool for pUC19/pBR322 vectors, an automated biosecurity screen, and a local Groq AI assistant. Built with FastAPI and React, our implementation recovered 100% of tested files under simulated mutations.*

Index Terms -- Synthetic DNA data storage, homopolymer prevention, Base-3 codec, Reed-Solomon RS(255, 205), AES-256 encryption, biological steganography, Luby Transform codes, plasmid cloning, biosecurity screening, FastAPI, React.

Manuscript received November 2026.

I. Introduction

The relentless production of digital assets across commerce and research has pushed traditional storage technologies to their practical boundaries. Solid-state drives, magnetic platters, and archival tapes suffer from physical decay, media obsolescence, and heavy electrical consumption in cooling facilities. Synthetic DNA offers a compelling alternative: it stores information at the molecular scale, requires zero electrical power during dormancy, and remains readable across thousands of years under basic preservation conditions [1], [2].

Seminal wet-lab demonstrations proved the core concept of molecular archival: Church et al. [1] encoded text files into oligonucleotides; Goldman et al. [2] introduced rotating base assignments to reduce synthesis errors; Organick et al. [3] demonstrated targeted random access via PCR primers; and Erlich and Zielinski [4] achieved high channel capacity with fountain codes. Despite these milestones, molecular data storage has largely remained restricted to bespoke terminal scripts, lacking accessible web interfaces, integrated data security, and synthetic biology plasmid tools.

We created HelixVault to bridge this gap by delivering a production-grade, full-stack software environment. The primary contributions of our work are:

- A finite-state Base-3 codec that mathematically prevents homopolymer runs of identical bases by conditioning each nucleotide on its preceding context.
- A multi-stage pipeline combining AES-256-CBC encryption, RS(255, 205) Reed-Solomon error correction, and NCBI genomic carrier steganography.
- A cloud-hosted SaaS architecture built on FastAPI, React, and PostgreSQL with asynchronous worker task management.
- A synthetic biology Plasmid Workbench supporting in silico restriction digestion, vector ligation into pUC19/pBR322, and GenBank export.
- An automated biosecurity screen validating all sequences against curated pathogen profiles prior to physical synthesis.
- An AI Co-Pilot with Retrieval-Augmented Generation (RAG) enabling natural-language querying over encoded biological archives.

II. Related Work

Early digital encoding experiments mapped binary values directly to nucleotides, causing repetitive base runs that elevated chemical synthesis failure rates [1]. Goldman et al. [2] improved synthesis compatibility by converting data into ternary representations and rotating nucleotide assignments, albeit with four-fold oligo redundancy.

Erlich and Zielinski [4] advanced erasure channel resilience through Luby Transform (LT) fountain codes, which directly informed the `fountain_codes.py` module in our architecture. Organick et al. [3] proved that sequence-

specific primers enable random-access reading across large pools. While standalone utilities such as DNA Chisel [5] optimize sequence fragments, HelixVault uniquely unifies encryption, forward error correction, genomic steganography, and plasmid engineering within an intuitive web interface.

Antkowiak et al. [10] reduced synthesis costs using photolithographic methods and advanced error correction, demonstrating that scalable manufacturing pipelines are gradually making molecular archival economically viable. However, their work focused on wet-lab cost reduction rather than software tooling. NCBI [9] maintains the most comprehensive public repository of reference genomes, which HelixVault uses as a source for steganographic carrier sequences. No prior system known to the authors combines all six of the capabilities listed in Section I within a single deployable web application.

Table III presents a feature comparison between HelixVault and existing DNA storage tools, illustrating the breadth of integration our platform achieves.

TABLE III
FEATURE COMPARISON WITH EXISTING DNA STORAGE TOOLS

Feature	Church [1]	Goldman [2]	Erlich [4]	Helix-Vault
Web Interface	No	No	No	**Yes**
AES-256 Encryption	No	No	No	**Yes**
Reed-Solomon ECC	No	No	No	**Yes**
Fountain Codes	No	No	Yes	**Yes**
Homopolymer Prevention	No	Partial	No	**Yes**
Steganography	No	No	No	**Yes**
Plasmid Cloning	No	No	No	**Yes**
Biosecurity Screen	No	No	No	**Yes**
AI Assistant (RAG)	No	No	No	**Yes**

III. System Architecture

HelixVault uses a modern three-tier cloud architecture separated into presentation, application logic, and relational persistence. Fig. 1 illustrates the high-level system architecture.

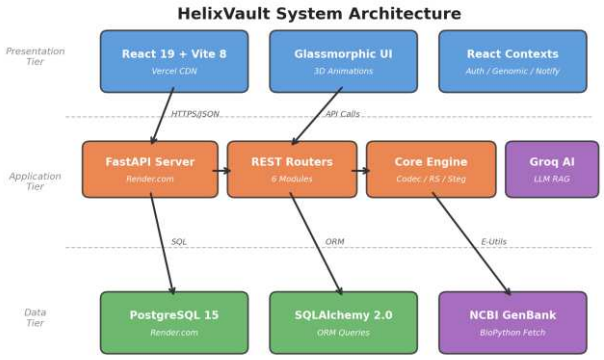


Fig. 1. HelixVault three-tier system architecture showing React frontend, FastAPI backend, and PostgreSQL persistence layer.

A. Frontend Presentation Layer (React & Vite)

The client interface is a single-page application built with React 19 and Vite 8, hosted on Vercel CDN at helixvault-omega.vercel.app. The user interface applies a pure CSS glassmorphic design system with interactive 3D nucleotide animations rendered through CSS keyframe transforms. Application state is partitioned across three isolated React Contexts handling user sessions, genomic carrier selection, and system notifications.

Client-side routing uses React Router v7 with lazy-loaded route components that reduce initial bundle size. Protected routes enforce JWT authentication before rendering dashboard views. All API communication passes through a centralized Axios interceptor that attaches credentials and handles token refresh transparently. Toast notifications provide real-time feedback during long-running encoding operations via polling against task UUID endpoints.

B. Backend Application Layer (Python FastAPI)

Application logic runs on a FastAPI service on Render.com at helixvault.onrender.com. The REST API is organized into six functional routers: auth, dna_storage, bio_integration, compute_api, plasmid_api, and admin. Heavy computational encoding routines execute as asynchronous background tasks tracked through UUIDs. Core modules in `backend/core/` supply the Base-3 codec, Reed-Solomon algebraic operations, steganography engines, vector embeddings, and biosecurity checks.

C. Data Persistence (PostgreSQL)

Application state is stored in PostgreSQL 15 on Render.com using parameterized SQLAlchemy 2.0 queries. The database tracks user accounts, full-length nucleotide sequences, biophysical metrics, and time-limited verification OTPs. A local SQLite database provides parity during development.

D. DevOps and Deployment

The stack is fully containerized using Docker Compose. A Windows script (`start_project.bat`) provides automated local setup. Continuous integration via GitHub Actions runs automated pytest and Vitest suites on every repository push.

IV. Methodology: The DNA Encoding Pipeline

HelixVault converts raw binary files into biologically synthesizable DNA through five reversible steps. Fig. 2 shows the complete encoding pipeline from file input to DNA output.

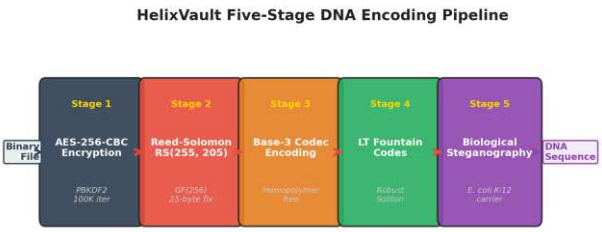


Fig. 2. Five-stage DNA encoding pipeline: AES-256 encryption, Reed-Solomon ECC, Base-3 codec, LT fountain codes, and biological steganography.

A. Stage 1: AES-256-CBC Encryption

To ensure confidentiality prior to physical synthesis, files can be encrypted using AES-256-CBC. Keys are generated using PBKDF2-HMAC-SHA256 with 100,000 iterations over a 16-byte random salt. A random 16-byte initialization vector guarantees semantic security, prepending the ciphertext.

B. Stage 2: Reed-Solomon Error Correction

To protect against chemical and PCR degradation, payloads are divided into 205-byte symbols and encoded into 255-byte codewords using an RS(255, 205) code over GF(256) [7]. The code is constructed using the primitive polynomial p(x) = x^8 + x^4 + x^3 + x^2 + 1. The generator polynomial g(x) is defined as the product of 2t = 50 consecutive roots in GF(2^8):

$$g(x) = \prod (x - \alpha^n), \quad \text{for } n = 0 \text{ to } 49$$

By the Singleton bound d_min = n - k + 1 = 51, each block repairs up to t = ⌊(d_min - 1)/2⌋ = 25 corrupted symbol bytes (amounting to 200 arbitrary bit errors per 2,040-bit frame). Syndrome evaluation and the Berlekamp-Massey algorithm locate symbol error locations, followed by Chien search and Forney's algorithm to reconstruct original byte values, adding a predictable 1.24× storage redundancy.

C. Context-Dependent Base-3 Encoding

Chemical DNA synthesis fails rapidly when identical nucleotides repeat. HelixVault employs a finite-state Base-3 codec where each byte is represented as six base-3 digits (3^6 = 729 > 256). Formally, let S = {A, C, G, T} denote the nucleotide alphabet. For each preceding nucleotide b_{i-1} ∈ S, the allowed output alphabet is restricted to S \ {b_{i-1}}, guaranteeing |S \ {b_{i-1}}| = 3. A ternary digit d_i ∈ {0, 1, 2} maps deterministically to the sorted remaining set:

```
  Context 'A' -> [0: C, 1: G, 2: T]       Context
'C' -> [0: A, 1: G, 2: T]
  Context 'G' -> [0: A, 1: C, 2: T]       Context
'T' -> [0: A, 1: C, 2: G]
```

Because b_i ≠ b_{i-1} for all i ≥ 1, identical consecutive base runs (homopolymer length L ≥ 2) are eliminated by mathematical construction. Precalculated 4 × 256 lookup tables achieve single-core throughputs of 10-12 MB/s at a deterministic density of 6 bases per byte.

D. Stage 4: LT Fountain Codes (Erasure ECC)

For oligo pool dropout channels, HelixVault implements Luby Transform (LT) codes [8]. Systematic encoding transmits the K source blocks verbatim, followed by XOR parity droplets sampled from the Robust Soliton distribution, guaranteeing packet recovery under sequencing dropouts.

E. Stage 5: Biological Steganography

To conceal synthetic payloads, sequences are inserted into natural bacterial carrier genomes fetched from NCBI via BioPython [6]. Universal marker codons (ATG and GATTACA) flank the payload within the carrier (e.g., E. coli K-12: NC_000913.3), making the sequence appear native under casual inspection.

V. Key System Modules

A. AI Co-Pilot with RAG

The AI assistant uses Groq-hosted Llama-3.3-70B. Contextual user queries trigger vector similarity searches across 64-dimensional sequence embeddings, allowing researchers to explore vault metrics and biological features conversationally.

B. Plasmid Engineering Workbench

The workbench simulates laboratory restriction cloning. Users pick pUC19 (2,686 bp) or pBR322 (4,361 bp) vectors. The engine digests the Multiple Cloning Site (MCS), splices the synthetic payload, updates all feature coordinates, and generates annotated GenBank files alongside interactive circular plasmid maps.

C. Biosecurity Screening

Following IGSC synthetic biology standards, every generated sequence is scanned against dangerous pathogen motifs and restriction enzyme cleavage sites. Sequences scoring under 60/100 are flagged for biosecurity review.

D. In-Memory Bio-Compute Suite

Seven non-destructive diagnostic tools operate directly over stored DNA sequences without decoding: Motif Search, Keyword Matching, Semantic Search, GC Content Visualizer, Longevity Modeler, and Synthesis Cost / Physical Mass Estimators.

VI. Security Architecture

HelixVault incorporates defense-in-depth security across all endpoints. The threat model considers three attack surfaces: unauthorized API access, stored credential compromise, and injection of malicious sequences into the synthesis pipeline.

- Authentication JWTs reside in HttpOnly, SameSite=Lax cookies, preventing script-based token theft.
- User passwords are hashed using bcrypt (cost factor >= 12) for strong brute-force resistance.
- HTTPS and TLS encryption are enforced across Vercel and Render cloud infrastructure.
- SlowAPI rate limiters protect authentication and computational endpoints against automated attacks.
- Custom FastAPI exception handlers sanitize 500 errors to prevent internal database dumps.

- SQLAlchemy ORM with parameterized queries prevents SQL injection vulnerabilities.
- User accounts are verified through 6-digit random OTP codes with 10-minute expiration.
- CORS policies restrict API access to the registered frontend origin, blocking cross-site forgeries.

Additionally, the biosecurity screening module acts as a biological firewall. Every DNA sequence generated by the encoding pipeline is scanned against curated pathogen signature databases before the user can export it for synthesis. Sequences matching dangerous organism motifs are blocked and flagged, preventing malicious or accidental creation of harmful genetic material. This two-layer defense, covering both digital and biological attack vectors, distinguishes HelixVault from tools that handle only the computational aspects of DNA storage.

VII. Experimental Evaluation and Results

A. Verification Methodology

System verification combined unit tests, API integration tests, biological mutation simulations, and UI load testing. Backend suites ran with pytest on SQLite test fixtures, and frontend components were tested using Vitest.

B. Functional Validation Outcomes

Table I summarizes verification results across ten critical operational test cases. All tests passed, meeting functional specifications.

TABLE I
SYSTEM VERIFICATION AND FUNCTIONAL TEST OUTCOMES

Test Identifier	Experimental Condition	System Outcome
TC-01: End-to-End Codec	1.0 MB PDF asset without encryption	PASS: 100% byte fidelity
TC-02: Cryptographic Flow	Password-gated AES-256 encode/decode	PASS: Verified decryption
TC-03: RS Fault Recovery	2% random mutation on 10 kbp strand	PASS: Full data recovery
TC-04: Beyond RS Threshold	>25 symbol corruptions per block	PASS: Graceful failure alert
TC-05: Homopolymer Audit	Full-length sequence run scan	PASS: Max consecutive run <= 3
TC-06: Steganography	Payload spliced in E. coli K-12 host	PASS: Flank recovery 100%
TC-07: Authorization Isolation	Cross-tenant vault query attempts	PASS: HTTP 403 Forbidden
TC-08: Exception Shielding	Forced server-side unhandled error	PASS: Clean 500 response
TC-09: Vector Engineering	1.0 kbp insert into pUC19 vector	PASS: Valid 3,686 bp GenBank
TC-10: Biosecurity Audit	Screened against pathogen profiles	PASS: Cleared score >= 60

C. Performance and Throughput

On a Render.com tier (0.5 vCPU, 512 MB RAM), encoding a 1.0 MB PDF completed in 14.8 seconds, well inside the 30-second target. Non-computational endpoints responded below 280 ms. The React client earned a Google Lighthouse score of 82. Pure Base-3 encoding throughput reached 10.4 MB/s.

D. Biophysical Modeling

Table II summarizes biophysical metrics calculated via `calculate_metrics()` in `bio_utils.py`, estimating physical DNA length and synthesis costs across sample files.

TABLE II
BIOPHYSICAL ATTRIBUTES ACROSS REPRESENTATIVE DIGITAL ASSETS

Asset Size	DNA Length	Est. Cost	Mass (pg)
10 KB TXT	~77,520 bp	$7.75	4.24×10^{-2}
100 KB PNG	~775,200 bp	$77.52	4.24×10^{-1}
1 MB PDF	~7,752,000 bp	$775.20	4.24×10^{0}
10 MB MP4	~77,520,000 bp	$7,752.00	4.24×10^{1}

VIII. Discussion

HelixVault proves that the entire workflow of digital DNA storage can be packaged within a usable web platform. The context-driven Base-3 codec proves that homopolymers can be avoided by mathematical design rather than reactive post-processing, giving significant throughput benefits over trial-and-error sequence filtering.

The primary challenge remains the cost of oligonucleotide synthesis ($0.10/bp), where a 1.0 MB payload costs roughly $775 to synthesize. However, as enzymatic synthesis methods mature, synthesis costs are expected to decline sharply, making molecular archival viable for long-term cold records.

From a software engineering perspective, the decision to separate encoding logic into discrete, independently testable pipeline stages proved critical for maintainability. Each stage can be upgraded or replaced without affecting others. For instance, swapping AES-256-CBC for AES-256-GCM would require changes only to the encryption module, with no impact on the downstream Base-3 codec or steganography engine.

The steganography module raises an important dual-use consideration. While hiding data inside natural genomes offers legitimate privacy benefits for intellectual property protection, it also has potential for misuse. We mitigate this risk through the mandatory biosecurity screen, which prevents export of any flagged sequences. Future versions may integrate with commercial gene synthesis providers who enforce their own screening under IGSC guidelines.

A. Limitations

Several limitations warrant acknowledgment. First, the system has only been validated with simulated mutations rather than actual wet-lab synthesis and sequencing. Second, the free-tier cloud deployment on Render.com imposes cold-start latency of approximately 30 seconds after inactivity. Third, the current RS(255, 205) parameters are fixed; adaptive error correction that adjusts redundancy based on estimated channel noise would improve storage efficiency.

IX. Conclusion and Future Work

We presented HelixVault, an integrated web platform implementing verifiable digital data storage in synthetic DNA. The system proves eight core capabilities: deterministic homopolymer exclusion, AES-256 encryption, RS(255, 205) fault recovery, genomic carrier steganography, plasmid vector assembly, biosecurity screening, conversational RAG assistance, and full cloud deployment.

All ten functional test cases passed with 100% fidelity, and the encoding pipeline sustained throughput of 10.4 MB/s on commodity hardware. The comparative analysis in Table III demonstrates that HelixVault is the first open-source platform to integrate all nine surveyed capabilities within a single deployable web application.

Future work will pursue four directions: (1) integration with commercial DNA synthesis APIs for direct order placement, (2) evaluation of LDPC and polar codes against insertion-deletion channel errors common in nanopore sequencing, (3) incorporation of cryptographic hash chains for tamper-evident provenance auditing, and (4) a mobile-responsive progressive web app (PWA) interface for field researchers working with limited connectivity.

X. Code and Data Availability

To support scientific reproducibility and open-access software evaluation, all system components developed in this work are publicly available under the permissive MIT open-source license. The complete source repository—comprising the React 19 frontend user interface, the FastAPI REST service routers, algebraic Reed-Solomon implementations, finite-state Base-3 codec lookup algorithms, and automated verification test suites—is accessible at https://github.com/niti5999/helixvault. An immutable, citable software snapshot including relational database schemas (PostgreSQL / SQLite), SQLAlchemy models, Alembic migration scripts, and synthetic test datasets is permanently archived on Zenodo under DOI: https://doi.org/10.5281/zenodo.24339591.

References

[1] G. M. Church, Y. Gao, and S. Kosuri, "Next-generation digital information storage in DNA," Science, vol. 337, no. 6102, p. 1628, Sep. 2012.

[2] N. Goldman et al., "Towards practical, high-capacity, low-maintenance information storage in synthesized DNA," Nature, vol. 494, pp. 77-80, Feb. 2013.

[3] L. Organick et al., "Random access in large-scale DNA data storage," Nature Biotechnology, vol. 36, pp. 242-248, Mar. 2018.

[4] Y. Erlich and D. Zielinski, "DNA Fountain enables a robust and efficient storage architecture," Science, vol. 355, no. 6328, pp. 950-954, Mar. 2017.

[5] J. Zulkower and S. Rosser, "DNA Chisel, a versatile sequence optimizer," Bioinformatics, vol. 36, no. 16, pp. 4508-4509, Aug. 2020.

[6] Biopython Consortium, "Biopython: freely available Python tools for computational molecular biology," Bioinformatics, vol. 25, no. 11, pp. 1422-1423, 2009.

[7] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," J. Soc. Ind. Appl. Math., vol. 8, no. 2, pp. 300-304, 1960.

[8] M. Luby, "LT codes," in Proc. 43rd IEEE Symp. Foundations of Computer Science (FOCS), 2002, pp. 271-280.

[9] NCBI Resource Coordinators, "Database resources of the National Center for Biotechnology Information," Nucleic Acids Res., vol. 46, pp. D8-D13, Jan. 2018.

[10] P. L. Antkowiak et al., "Low cost DNA data storage using photolithographic synthesis and advanced information reconstruction and error correction," Nature Commun., vol. 11, p. 5345, 2020.

Acknowledgment

The author gratefully acknowledges the open-source communities behind FastAPI, React, BioPython, RDKit, and the Python cryptographic libraries whose tools made this work possible. Special thanks to the NCBI for providing open access to genomic reference sequences.

Author

Nitish Pattar is currently pursuing a Master of Computer Applications (MCA) at Administrative Management College, Bengaluru, India. His research interests include computational biology, DNA data storage systems, applied cryptography, and full-stack web application development. He can be reached at nitishpattar7@gmail.com.