Scientific Research Publishing

Journal of Information Security, 2026, 17(4), 464-497
https://www.scirp.org/journal/jis
ISSN Online: 2153-1242
ISSN Print: 2153-1234

Homomorphic Encryption for Privacy-Preserving Healthcare Informatics: A Secure Weighted Risk Score Framework Using OpenFHE and Synthetic FHIR Data

Donato Deng Ajiing Pakak

Computer Science Department, California State University, Dominguez Hills, Carson, California, USA
Email: dengdonato1@gmail.com, dpakak1@toromail.csudh.edu

How to cite this paper: Pakak, D.D.A. (2026) Homomorphic Encryption for Privacy-Preserving Healthcare Informatics: A Secure Weighted Risk Score Framework Using OpenFHE and Synthetic FHIR Data. *Journal of Information Security*, **17**, 464-497.
https://doi.org/10.4236/jis.2026.174021

Received: May 24, 2026
Accepted: August 30, 2026
Published: September 2, 2026

Abstract

This paper presents an end-to-end privacy-preserving healthcare analytics framework that combines Synthea-generated Fast Healthcare Interoperability Resources (FHIR) records with the Cheon-Kim-Kim-Song (CKKS) homomorphic encryption scheme implemented in OpenFHE. The framework extracts five normalized features—age, systolic blood pressure, body mass index, cholesterol, and a binary heart-condition indicator—and evaluates an illustrative weighted risk score while the patient data remain encrypted. The experimental dataset contains 15 synthetic patients distributed across three representative institutions and uses a CKKS batch size of 32 slots. For the reported representative evaluation, the plaintext score was 87.60 and the decrypted encrypted score was 87.58, corresponding to a 0.022% relative error and a 71.01 ms end-to-end runtime. The score is a demonstration model for evaluating encrypted computation and is not a clinically validated diagnostic index. The results show that a shallow linear healthcare computation can be evaluated with low numerical error while preserving confidentiality during processing. The study also identifies limitations related to circuit depth, incomplete hardware metadata, and the need for validation using clinically established models and larger datasets.

Keywords

Homomorphic Encryption, OpenFHE, CKKS, Healthcare Informatics, FHIR, Synthea, Privacy-Preserving Computation

1. Introduction

Modern healthcare is no longer defined only by doctors and hospitals. It is defined

by data. Every diagnosis, prescription, laboratory result, and clinical observation contributes to a growing digital ecosystem that can support predictive medicine, better treatment planning, and large scale healthcare intelligence. However, healthcare data remains highly restricted because it contains sensitive personal information. Regulations such as HIPAA in the United States and GDPR in Europe exist to protect patient privacy and prevent misuse of medical records [1] [2]. These protections are necessary, but they also create a major limitation for research and clinical analytics.

In practice, hospitals often operate as isolated data environments. Even when two institutions treat similar diseases or serve similar communities, they cannot freely share raw patient records because of legal, ethical, and security concerns. This slows collaboration and prevents researchers from using broad datasets that could improve diagnosis and treatment. Healthcare data standards such as FHIR help with interoperability, but interoperability alone does not solve the privacy problem [3] (**Figure 1**).

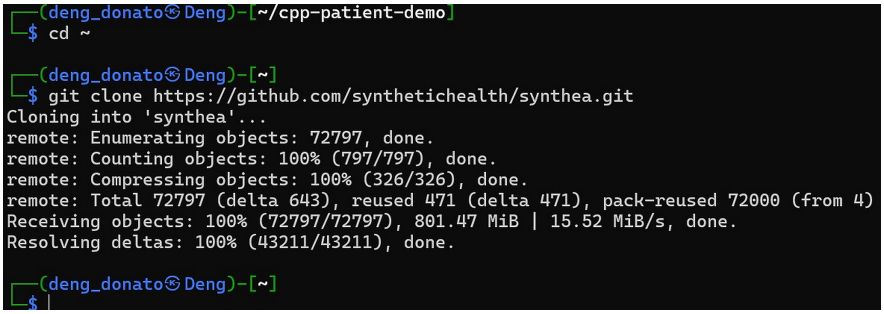

Figure 1. Synthetic healthcare data generation using Synthea and transformation into FHIR compliant records for encrypted processing.

The most difficult security problem is known as data in use. Traditional encryption protects data while it is stored and while it is transmitted across a network. However, the data usually must be decrypted before computation. During this stage, sensitive information exists in plaintext memory where it may be exposed to malware, insider threats, or system compromise. This is especially dangerous in healthcare, where breaches can cause financial, legal, and human harm [4].

Fully Homomorphic Encryption (FHE) addresses this weakness by allowing computation directly on encrypted data. Instead of decrypting patient information before analysis, the system performs mathematical operations on ciphertext. The result remains encrypted until the authorized data owner decrypts it. This means that the server can compute without seeing the patient data [5].

This research uses the CKKS scheme (**Figure 2**) because healthcare risk scoring depends on real valued numbers such as age, blood pressure, cholesterol, body mass index, and probability-based indicators. CKKS supports approximate arithmetic over encrypted real numbers, making it suitable for clinical analytics and machine learning style computations [6]. OpenFHE provides the cryptographic framework used to implement the encrypted computation pipeline [7].

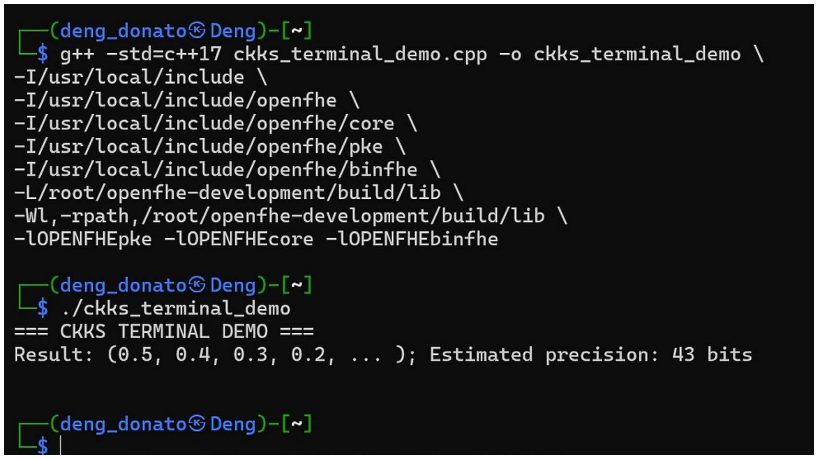

Figure 2. CKKS based fully homomorphic encryption workflow for secure medical risk score computation.

The purpose of this paper is to demonstrate a practical encrypted weighted risk score system for healthcare informatics. Synthetic patient data is generated using Synthea, structured through FHIR, encrypted using CKKS in OpenFHE, processed through homomorphic multiplication and addition, then decrypted only by the authorized owner. This approach shows that healthcare systems do not need to choose between privacy and useful computation [3] [7] [8].

2. Background

Homomorphic Encryption (HE) is a cryptographic paradigm that enables computation to be performed directly on encrypted data without requiring decryption. The fundamental property of HE is that the decrypted result of an operation performed on ciphertext corresponds to the result of the same operation performed on the underlying plaintext. This capability is particularly important in environments where sensitive data must remain confidential during processing, such as healthcare, finance, and cloud computing [5].

Traditional encryption schemes are designed to protect data at rest and in transit. However, when data must be processed, it is typically decrypted into plaintext, exposing it to potential risks such as memory attacks, insider threats, or system compromise. This limitation is commonly referred to as the "data in use" problem. Homomorphic Encryption directly addresses this issue by allowing computations to occur while the data remains encrypted throughout the entire lifecycle of storage, transmission, and processing [5].

Homomorphic Encryption schemes are generally categorized into three types based on their computational capabilities.

1) Partially Homomorphic Encryption (PHE) supports only a single type of operation, such as addition or multiplication. Examples include the Paillier cryptosystem for additive operations and RSA for multiplicative operations. While efficient, PHE schemes are limited in their applicability because real-world computations typically require a combination of operations.

2) Somewhat Homomorphic Encryption (SHE) extends this capability by allowing a limited number of both addition and multiplication operations. However, SHE schemes suffer from noise accumulation within ciphertexts. Each operation increases the noise level, and once a threshold is reached, the ciphertext can no longer be correctly decrypted. This restricts the depth of computations that can be performed.

3) Fully Homomorphic Encryption (FHE) overcomes these limitations by supporting arbitrary computations on encrypted data. This is achieved through techniques such as bootstrapping, which refresh ciphertexts and reduces accumulated noise, enabling theoretically unlimited computation depth. Craig Gentry's groundbreaking work in 2009 introduced the first viable FHE scheme, establishing the foundation for modern research in this field [5] [9] (**Figure 3**).

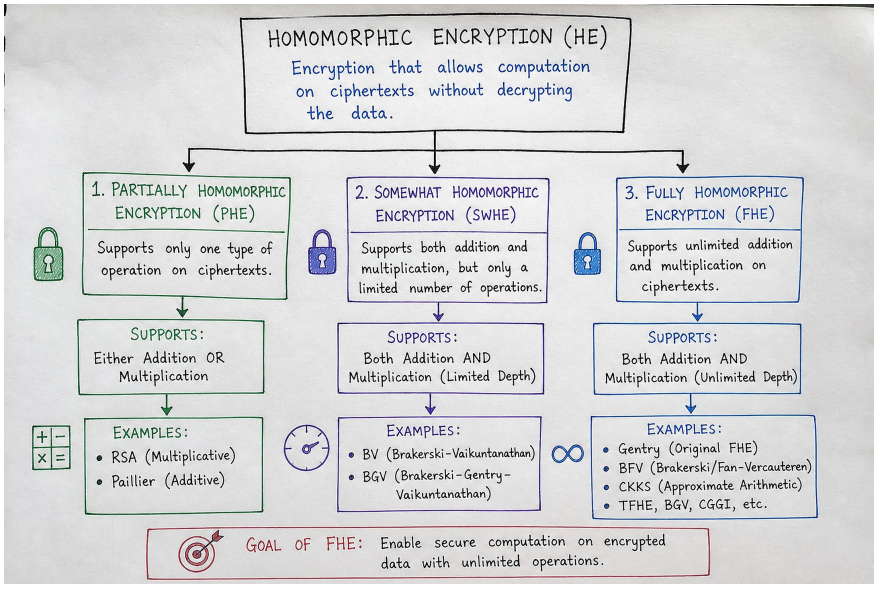

Figure 3. Classification of homomorphic encryption schemes: partially homomorphic, somewhat homomorphic, and fully homomorphic encryption.

Among the various FHE schemes developed, the Cheon-Kim-Kim-Song (CKKS) scheme has emerged as one of the most practical for real-world applications involving approximate numerical computation. Unlike exact schemes such as BFV and BGV, which operate on integers, CKKS is designed to support approximate arithmetic over real and complex numbers. This makes it particularly suitable for applications in data science, signal processing, and healthcare analytics, where continuous values and probabilistic models are common [6].

The CKKS scheme encodes real-valued data into polynomial representations, which are then encrypted into ciphertexts. Arithmetic operations such as addition and multiplication can be applied directly to these ciphertexts, producing encrypted results that approximate the corresponding plaintext computations. While CKKS introduces a small approximation error due to scaling and rounding, this error is typically negligible for many practical applications, especially those involving sta-

tistical analysis or machine learning.

In healthcare informatics, clinical data is inherently numerical and often continuous in nature. Variables such as blood pressure, cholesterol levels, body mass index (BMI), glucose levels, and age are typically represented as real numbers. Risk assessment models frequently rely on weighted combinations of these variables to produce predictive scores. The ability of CKKS to handle real-valued arithmetic makes it a natural fit for privacy-preserving healthcare analytics.

Another important feature of CKKS is its support for batching, also known as Single Instruction Multiple Data (SIMD) processing. This allows multiple data values to be packed into a single ciphertext and processed simultaneously. Batching significantly improves computational efficiency by enabling parallel evaluation of multiple data points, which is particularly useful for large-scale healthcare datasets and population-level analytics [6].

The practical implementation of Homomorphic Encryption has been accelerated by the development of modern cryptographic libraries. Among these, OpenFHE has emerged as a powerful and flexible open-source framework for implementing advanced HE schemes. OpenFHE supports multiple encryption schemes, including CKKS, BFV, and BGV, and provides a comprehensive set of tools for key generation, encryption, decryption, and homomorphic evaluation [7].

In addition to OpenFHE (**Figure 4**), other widely used libraries include Microsoft SEAL and IBM HElib. Microsoft SEAL provides a user-friendly interface and is widely adopted in both academic and industrial research. IBM HElib, one of the earliest HE libraries, remains an important reference implementation for homomorphic encryption techniques. These libraries have played a critical role in transitioning HE from a theoretical concept to a practical tool for secure computation [10] [11].

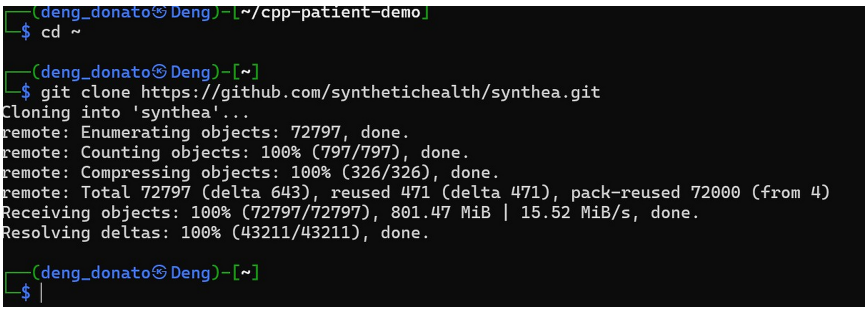

Figure 4. OpenFHE-based architecture for secure encrypted computation, illustrating key generation, encryption, homomorphic evaluation, and decryption processes.

Beyond cryptographic techniques, healthcare informatics requires standardized data formats to ensure interoperability between systems. Fast Healthcare Interoperability Resources (FHIR), developed by HL7, provides a structured and standardized approach for representing and exchanging electronic health records. FHIR defines resources such as patients, observations, conditions, and medications in a consistent format, enabling seamless data exchange across healthcare

systems [3].

However, while FHIR improves interoperability, it does not inherently address privacy concerns. Even standardized data must still comply with strict regulatory requirements such as the Health Insurance Portability and Accountability Act (HIPAA) and the General Data Protection Regulation (GDPR). These regulations impose significant constraints on how healthcare data can be shared and processed, often limiting collaboration between institutions.

To support research and experimentation without compromising patient privacy, synthetic data generation has become an important tool in healthcare informatics. Synthea is a widely used open-source synthetic patient generator that produces realistic but entirely artificial health records. These records mimic real-world clinical data patterns, including disease progression, treatments, and outcomes, while ensuring that no actual patient information is exposed [8].

The combination of Synthea and FHIR enables researchers to develop and test healthcare analytics systems in a safe and controlled environment. Synthetic data can be structured using FHIR standards, allowing it to closely resemble real clinical datasets while avoiding ethical and legal concerns. This approach is particularly valuable for developing and validating privacy-preserving technologies such as Homomorphic Encryption.

In addition to cryptographic protections, system-level security mechanisms play an important role in protecting sensitive data. Trusted Execution Environments (TEEs) and confidential computing technologies provide hardware-based isolation for sensitive workloads. These environments ensure that data and computations remain protected even from privileged system components such as the operating system or cloud provider.

Confidential computing platforms offered by major cloud providers enable secure enclaves where sensitive computations can be performed with strong isolation guarantees. When combined with Homomorphic Encryption, these technologies create a layered security model. Homomorphic Encryption protects data at the cryptographic level, while TEEs provide protection at the hardware level. Together, they significantly reduce the attack surface for sensitive healthcare data [4] [12].

Despite these advancements, challenges remain in the adoption of Homomorphic Encryption for real-world applications. One of the primary limitations is computational overhead. Encrypted operations are significantly more expensive than their plaintext counterparts, both in terms of time and resource consumption. This can make real-time applications difficult to achieve, particularly for complex models or large datasets.

Another challenge is parameter selection. Homomorphic Encryption schemes require careful tuning of parameters such as ciphertext modulus, polynomial degree, and scaling factors. These parameters directly impact security, accuracy, and performance. Selecting appropriate parameters requires expertise and a deep understanding of the underlying cryptographic principles.

Furthermore, while CKKS enables approximate computation, managing precision and error accumulation remains an important consideration. Improper scaling or excessive computation depth can lead to significant numerical inaccuracies. Therefore, system designers must balance computational complexity with acceptable error margins, particularly in sensitive domains such as healthcare.

Nevertheless, ongoing research and technological advancements continue to improve the practicality of Homomorphic Encryption. Hardware acceleration using GPUs and specialized cryptographic processors is being explored to reduce computational overhead. Algorithmic optimizations and improved library implementations are also contributing to more efficient encrypted computation.

In summary, Homomorphic Encryption represents a transformative approach to secure data processing by enabling computation on encrypted data. The CKKS scheme provides practical support for real-valued arithmetic, making it well-suited for healthcare analytics. Modern libraries such as OpenFHE have made implementation more accessible, while standards like FHIR and tools like Synthea support realistic and interoperable data environments. When combined with system-level protections such as confidential computing, these technologies form a comprehensive framework for privacy-preserving healthcare informatics. This foundation enables secure collaboration, advanced analytics, and data-driven medical innovation without compromising patient confidentiality.

3. Related Work and Literature Review

Fully Homomorphic Encryption (FHE) was first introduced by Gentry [5], enabling arbitrary computation on encrypted data without requiring decryption. While this work established the theoretical foundation of secure computation, it was computationally expensive and impractical for real-world deployment due to significant performance overhead.

To address these limitations, subsequent research focused on improving efficiency and applicability. The CKKS scheme proposed by Cheon *et al.* [6] introduced approximate arithmetic over real numbers, allowing efficient processing of continuous data. This advancement is particularly important for healthcare analytics, where clinical features such as blood pressure, cholesterol, and body mass index are inherently real-valued.

Further improvements in homomorphic encryption performance were explored by Stehlé *et al.* [9], who proposed optimized schemes for faster encrypted computation. These developments contributed to reducing computational latency and enabling more practical use of FHE in applied domains.

The transition from theory to practice has been supported by the development of modern cryptographic libraries. OpenFHE [7] provides a flexible and high-performance framework for implementing advanced homomorphic encryption schemes, while Microsoft SEAL [10] and IBM HElib [11] offer widely adopted platforms for secure computation. These tools have significantly lowered the barrier to implementing real-world encrypted systems.

In the healthcare domain, interoperability and data standardization are addressed by frameworks such as Fast Healthcare Interoperability Resources (FHIR) [3], which provides a structured representation of clinical data. Additionally, synthetic data generation tools such as Synthea [8] enable researchers to simulate realistic patient datasets while preserving privacy. However, these approaches primarily focus on data representation and accessibility rather than secure computation.

Beyond cryptographic techniques, system-level approaches such as confidential computing have been proposed to protect sensitive data during processing. Solutions such as Google Cloud's confidential computing [4] and privacy-preserving collaboration platforms [12] provide secure execution environments. While effective, these methods rely on trusted hardware or execution environments and do not provide full cryptographic protection during computation.

Recent research has also explored integrating homomorphic encryption with distributed learning systems. Korkmaz and Rao [13] proposed secure homomorphic encryption techniques for federated learning, enabling collaborative analytics without exposing raw data. These approaches highlight the growing importance of privacy-preserving computation but are primarily focused on machine learning models rather than structured healthcare risk scoring.

Despite these advancements, existing work typically addresses homomorphic encryption, healthcare data standardization, or privacy-preserving computation as separate problems. Few studies integrate these components into a unified framework capable of supporting end-to-end healthcare analytics.

In contrast, the proposed work bridges this gap by integrating CKKS-based homomorphic encryption, OpenFHE implementation, FHIR-compliant data structuring, and Synthea generated datasets into a complete system. Unlike prior approaches that focus on isolated components, this work demonstrates an implementation-oriented pipeline that demonstrates encrypted clinical risk scoring, reported numerical consistency, confidentiality during evaluation, and scalability potential under the tested configuration.

This integrated approach provides a practical contribution to the field of privacy-preserving healthcare informatics, offering a realistic pathway for secure data sharing and collaborative analytics across healthcare institutions (**Table 1**).

Table 1. Comparison of related work.

Work	FHE	Health	E2E
Gentry (2009)	Yes	No	No
CKKS (2017)	Yes	Partial	No
Chillotti *et al.*	Yes	No	No
OpenFHE/SEAL/HElib	Yes	No	No
FHIR	No	Yes	No
Synthea	No	Yes	No
Confidential Computing	Partial	Yes	No
Federated HE (2026)	Yes	Partial	No
This Work	Yes	Yes	Yes

4. Proposed Method

The proposed method introduces an end-to-end privacy-preserving healthcare analytics pipeline that enables secure computation of clinical risk scores directly on encrypted patient data. The framework integrates synthetic data generation, healthcare data standardization, feature engineering, homomorphic encryption, and encrypted computation into a unified workflow. The objective is to demonstrate that meaningful clinical analytics can be performed without exposing sensitive patient information at any stage of processing.

Unlike traditional healthcare systems where data must be decrypted before computation, this method ensures that patient data remains encrypted throughout storage, transmission, and computation. The system leverages the CKKS scheme for approximate arithmetic and OpenFHE for cryptographic implementation, enabling efficient evaluation of real-valued medical features in a secure environment [6] [7].

The overall workflow consists of five primary stages: 1) synthetic data generation, 2) FHIR-based data structuring, 3) feature normalization, 4) encryption using CKKS, and 5) encrypted risk score computation. Each stage is designed to ensure compatibility with both healthcare data standards and homomorphic encryption requirements.

4.1. Synthetic Data Generation

The first stage involves generating realistic healthcare data using Synthea. Synthea produces synthetic patient records that simulate real-world clinical scenarios, including demographics, medical histories, diagnoses, medications, and laboratory results. These records follow clinically plausible distributions and temporal progressions, making them suitable for experimentation and validation.

The use of synthetic data eliminates ethical and legal concerns associated with real patient records while preserving statistical realism. This is particularly important in the development of privacy-preserving systems, where access to real data may be restricted due to regulatory requirements such as HIPAA and GDPR [8].

1) *Dataset Characteristics*: The experimental dataset was generated using Synthea, an open-source synthetic patient generator that produces realistic healthcare records while preserving patient privacy. The dataset consisted of 15 synthetic patients distributed across three representative healthcare institutions: CSUDH Healthcare, LSU Healthcare, and Compton University Healthcare. Each institution contributed five synthetic patient records to the evaluation dataset.

Patient records were generated using Synthea and selected to represent realistic healthcare scenarios involving demographic information, vital signs, laboratory measurements, and cardiovascular risk indicators. The generated records were transformed into FHIR-compliant structures for encrypted processing. Feature extraction focused on clinically relevant variables associated with healthcare risk assessment.

The primary FHIR resources utilized in this study included:

- Patient
- Observation
- Condition

From these resources, healthcare features such as age, blood pressure, body mass index (BMI), cholesterol measurements, and cardiovascular condition indicators were extracted and normalized prior to encryption.

To support efficient encrypted computation, the CKKS batching mechanism was employed. The OpenFHE implementation was configured with a batch size of 32 slots, enabling multiple feature values to be packed and processed within a single ciphertext. Although the experimental dataset contained 15 synthetic patients, the 32-slot configuration indicates potential support for packing multiple records, subject to slot capacity, feature layout, encryption parameters, and further scalability testing.

The use of synthetic FHIR data ensures repeatability, eliminates privacy concerns associated with real patient records, and provides a realistic environment for evaluating privacy-preserving healthcare computation.

4.2. FHIR-Based Data Structuring

After data generation, the synthetic records are structured using the Fast Healthcare Interoperability Resources (FHIR) standard. FHIR provides a consistent framework for representing healthcare data, enabling interoperability across systems and facilitating standardized data processing.

Relevant FHIR resources include Patient, Observation, Condition, and Medication. These resources are parsed and mapped into a structured dataset containing clinically meaningful variables. This step ensures that the data pipeline aligns with real-world healthcare systems and can be extended to production environments.

4.3. Feature Extraction and Normalization

The next stage involves extracting and transforming clinical variables into numerical feature vectors suitable for encrypted computation. Typical features include age, systolic and diastolic blood pressure, cholesterol levels, body mass index (BMI), glucose levels, smoking status, and presence of chronic conditions.

Continuous variables are normalized to a fixed numerical range to improve numerical stability during CKKS encoding. Normalization also ensures that feature magnitudes remain within acceptable bounds for homomorphic operations, reducing the risk of overflow and minimizing approximation error.

Categorical variables are encoded using binary or one-hot representations. For example, smoking status may be represented as a binary indicator, while disease categories may be encoded using multiple binary features. This transformation ensures compatibility with arithmetic operations supported by the CKKS scheme.

1) *Clinical Risk Score Definition*: To provide a repeatable technical demonstra-

tion of privacy-preserving healthcare analytics, a weighted clinical risk score model was implemented using normalized patient features extracted from synthetic FHIR records. The model was designed to evaluate the feasibility, accuracy, and computational efficiency of homomorphic computation on healthcare data while preserving patient confidentiality. The proposed model serves as an illustrative healthcare risk assessment framework for encrypted computation and is not intended as a clinically validated diagnostic instrument. The clinical risk score is computed as:

$$R = \sum_{i=1}^{n} w_i x_i \tag{1}$$

where R denotes the final risk score, x_i represents the normalized value of the i-th patient feature, w_i represents the corresponding feature weight, and n is the total number of features included in the model.

Table 2. Clinical risk score model parameters.

Feature	Normalization	Weight
Age	Age/100	0.25
Blood Pressure	SBP/200	0.30
BMI	BMI/50	0.20
Cholesterol	Cholesterol/300	0.15
Heart Condition	Binary (0 or 1)	0.10

The reported weighted-score experiment used five features: age (0.25), systolic blood pressure (SBP; 0.30), BMI (0.20), cholesterol (0.15), and a heart-condition indicator (0.10). The weights were chosen for a repeatable technical demonstration and were not derived from, calibrated against, or validated as a recognized clinical risk index. Therefore, the resulting score must not be interpreted as a diagnosis, prognosis, or treatment recommendation.

Continuous variables are normalized using the explicit rules shown in Table 2: age/100, systolic blood pressure/200, BMI/50, and cholesterol/300. The heart-condition indicator is encoded as 0 or 1. These bounded transformations improve numerical stability during CKKS encoding and evaluation.

The weighted score is intentionally limited to one layer of ciphertext-plaintext multiplication followed by addition. This shallow circuit is suitable for demonstrating privacy-preserving arithmetic, but it does not stress-test deep circuits, substantial noise accumulation, or bootstrapping. The model is illustrative and not clinically validated.

4.4. Encryption Using CKKS

Once feature vectors are prepared, they are encrypted using the CKKS homomorphic encryption scheme. CKKS is selected because it supports approximate arith-

metic over real numbers, which is essential for healthcare analytics involving continuous variables.

The encryption process begins with the generation of a cryptographic context using OpenFHE. This context defines parameters such as polynomial modulus degree, scaling factor, and ciphertext modulus, which collectively determine the security level, precision, and computational efficiency of the system.

A public/private key pair is generated for encryption and decryption. Multiplication evaluation keys are generated for the reported weighted computation. The retained experimental record does not establish whether rotation keys were generated; therefore, no rotation-key generation claim is made.

Each patient's feature vector is encoded into a plaintext polynomial and then encrypted into a ciphertext. The resulting ciphertexts can be transmitted to an external computation server without exposing the underlying data (Figure 5).

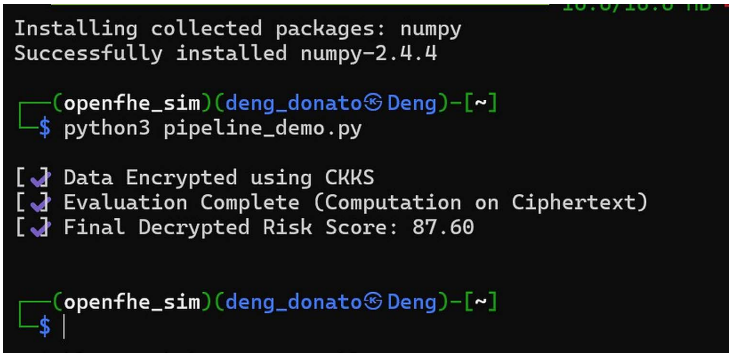

Figure 5. End-to-end workflow for encrypted healthcare risk score computation, including data generation, feature extraction, encryption, and homomorphic evaluation.

4.5. Encrypted Risk Score Computation

The core of the proposed method is the encrypted evaluation of the clinical risk score model defined by Equation (1) in Section 4.3. Homomorphic operations are applied directly to encrypted feature vectors to compute the weighted risk score without exposing patient information.

In a traditional system, this computation would be performed on plaintext data. However, in the proposed framework, each x_i is encrypted, and the computation is carried out using homomorphic operations. Specifically, ciphertexts are multiplied by plaintext weights using homomorphic multiplication, and the resulting ciphertexts are combined using homomorphic addition.

The computation proceeds as follows:

- Each encrypted feature $Enc(x_i)$ is multiplied by its corresponding weight w_i.
- The weighted ciphertexts are summed using homomorphic addition.
- The final ciphertext $Enc(R)$ represents the encrypted risk score.

The computation server performs all operations without access to the private key, ensuring that patient data remains confidential. Only the data owner can decrypt the final result using the private key (Figure 6).

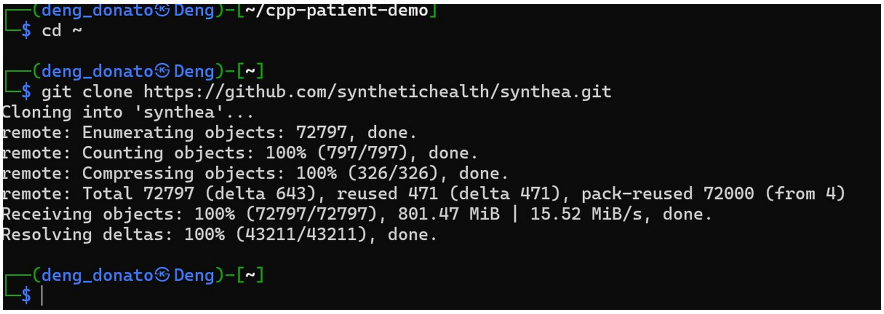

Figure 6. Encrypted weighted risk score computation using homomorphic multiplication and addition over ciphertexts.

4.6. Zero-Trust Security Model

The proposed framework is designed under a zero-trust security model. In this model, the computation server is not trusted with sensitive data and operates only on encrypted inputs. The data owner retains full control over the private key and decryption process.

Even if the computation server is compromised, the attacker gains access only to ciphertexts, which are computationally infeasible to decrypt without the private key. This significantly reduces the risk of data breaches and aligns with modern security principles for cloud-based systems.

4.7. Batch Processing and Scalability

To improve efficiency, the system leverages the batching capabilities of the CKKS scheme. Multiple feature values—and potentially multiple patient records—can be packed into a single ciphertext using SIMD techniques, subject to slot capacity, feature layout, and encryption parameters. This provides parallel-processing potential within a homomorphic operation; it does not establish large-scale throughput in the reported experiment.

Batching offers performance and scalability potential. It may support multiple patient records, subject to slot capacity, feature layout, encryption parameters, and further evaluation; large-scale processing was not demonstrated in the reported experiment.

4.8. Precision and Error Management

One of the key challenges in CKKS-based computation is managing approximation error. Since CKKS performs approximate arithmetic, each operation introduces a small error due to scaling and rounding. Over multiple operations, these errors can accumulate.

To address this, the proposed method carefully selects encryption parameters and limits the depth of computation. The weighted risk score model is intentionally designed to be shallow, consisting primarily of additions and multiplications, which minimizes error accumulation.

Additionally, normalization of input features ensures that values remain within

a stable numerical range, further improving precision. The retained representative result shows a small numerical difference between decrypted and plaintext computations.

4.9. System Workflow Summary

The complete system workflow can be summarized as follows:

1) Generate synthetic patient data using Synthea.
2) Convert patient records into FHIR-compliant structures.
3) Extract and normalize clinical features.
4) Encrypt feature vectors using CKKS in OpenFHE.
5) Perform homomorphic computation of weighted risk scores.
6) Return encrypted results to the data owner for decryption.

This pipeline demonstrates that secure, privacy-preserving healthcare analytics can be achieved without exposing sensitive patient information. The integration of standardized data formats, advanced cryptographic techniques, and efficient computation models provides a technical foundation for future real-world evaluation and adaptation.

4.10. Design Advantages

The proposed method offers several key advantages:

Privacy Preservation: Patient data remains encrypted throughout the entire lifecycle, eliminating exposure during computation.

Regulatory Compliance: The framework aligns with privacy regulations by minimizing data sharing and preventing unauthorized access.

Interoperability: The use of FHIR ensures compatibility with existing healthcare systems.

Scalability Potential: batching may support parallel processing and larger datasets, subject to slot capacity, encryption parameters, and further evaluation.

Security: The zero-trust model reduces reliance on external systems and mitigates insider threats.

4.11. Discussion

The proposed method demonstrates a practical approach to integrating Homomorphic Encryption into healthcare informatics. By combining synthetic data generation, standardized data formats, and encrypted computation, the framework addresses both technical and regulatory challenges.

While performance overhead remains a consideration, the simplicity of the weighted risk score model and the use of batching make the approach feasible for targeted applications. As hardware acceleration and cryptographic optimizations continue to advance, the efficiency of homomorphic encryption is expected to improve further.

Overall, this method provides a strong foundation for future research in privacy-preserving healthcare analytics, enabling secure collaboration and data-

driven decision-making without compromising patient confidentiality.

5. Implementation

The implementation of the proposed privacy-preserving healthcare analytics framework was developed in C++ using the OpenFHE library. The choice of C++ was motivated by its performance efficiency, memory control, and compatibility with cryptographic libraries that require fine-grained optimization. OpenFHE provides direct access to homomorphic encryption primitives, enabling precise control over key generation, encoding, encryption, homomorphic evaluation, and decryption processes [7].

The implementation is structured as a modular pipeline consisting of data ingestion, feature engineering, encryption, encrypted computation, and decryption. Each module is designed to operate independently while maintaining compatibility with the overall workflow. This modular architecture improves maintainability, scalability, and extensibility, allowing the system to be adapted for more complex healthcare analytics tasks in the future.

5.1. System Architecture

The system follows a client-server model aligned with a zero-trust architecture. The data owner, representing a healthcare institution, performs data preparation, feature extraction, and encryption locally. The encrypted data is then transmitted to an external computation server, which performs homomorphic operations without accessing the underlying plaintext. The final encrypted result is returned to the data owner for decryption.

This architecture ensures that sensitive patient data never leaves the control of the data owner in plaintext form. The computation server operates solely on ciphertexts and evaluation keys, eliminating the need for trust in external infrastructure. This design is particularly suitable for cloud-based healthcare analytics, where data privacy is a critical concern.

5.2. Data Generation and Preprocessing

Synthetic patient data was generated using Synthea, which produces realistic electronic health records that mimic real-world clinical data distributions. The generated dataset includes demographic information, vital signs, laboratory measurements, diagnoses, and treatment histories. These records provide a comprehensive representation of patient health profiles suitable for risk scoring applications [8].

The synthetic data is structured using FHIR-style fields to ensure interoperability and consistency with healthcare data standards. Relevant features are extracted from FHIR resources such as Patient, Observation, and Condition. These features include age, systolic and diastolic blood pressure, cholesterol levels, body mass index (BMI), glucose levels, smoking status, diabetes status, and prior cardiovascular conditions.

Continuous variables are normalized to a fixed range to ensure numerical sta-

bility during homomorphic computation. For example, blood pressure and cholesterol values are scaled relative to clinically meaningful ranges. Normalization reduces the magnitude of encoded values, which helps control noise growth in CKKS ciphertexts and improves computational accuracy.

Categorical variables are transformed into binary indicators. For instance, smoking status is encoded as a binary value, while the presence or absence of chronic conditions is represented using Boolean flags. This transformation ensures compatibility with arithmetic operations supported by the CKKS scheme (Figure 7).

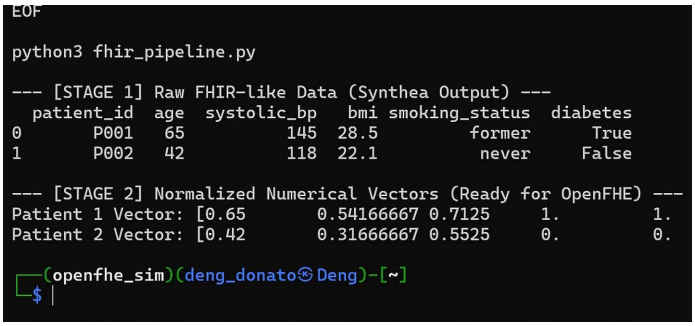

Figure 7. FHIR-based feature extraction pipeline illustrating transformation of synthetic patient records into normalized numerical vectors.

The final output of this stage is a structured feature vector for each patient, represented as a real-valued numerical array ready for encryption.

5.3. CryptoContext Configuration

The core of the implementation is the configuration of the OpenFHE CryptoContext for CKKS-based homomorphic encryption. This step defines the cryptographic parameters that govern security, precision, and performance.

To improve repeatability, the primary CKKS configuration parameters used in the experimental implementation are summarized in Table 3.

Table 3 reports the parameters explicitly retained from the experimental implementation. The configuration used CKKS, a multiplicative depth of 3, a 50-bit scaling modulus, 32 packed slots, generated multiplication keys, and no bootstrapping. These settings are sufficient for the single multiplication layer and subsequent additions used in the weighted score.

OpenFHE selected the ring dimension through its parameter-generation routine. The exact resolved ring dimension, polynomial modulus degree, and named security-level constant were not preserved in the original execution record and cannot be reconstructed reliably from the retained manuscript materials. This omission is reported explicitly as a repeatability limitation rather than replaced with an assumed value.

Future replications should record the values returned by the CryptoContext, including the ring dimension, complete modulus chain, slot capacity, scaling technique, security-level constant, OpenFHE version, and compiler flags. Recording

these values directly from the runtime configuration will permit exact reproduction of the cryptographic environment.

Table 3. OpenFHE CKKS configuration parameters.

Parameter	Value
Encryption Scheme	CKKS
Multiplicative Depth	3
Scaling Modulus Size	50 bits
Batch Size	32
Public Key Encryption	Enabled
Key Switching	Enabled
Leveled SHE	Enabled
Evaluation Multiplication Keys	Generated
Bootstrapping	Not Used
Security-Level Constant	Not retained in original execution record
Ring Dimension/Polynomial Modulus Degree	Automatically selected by OpenFHE; exact value not retained
Slot Count	32 configured slots
OpenFHE Version	Not retained in original execution record

The available configuration supports homomorphic addition and ciphertext-plaintext multiplication. Bootstrapping was not used because the evaluated circuit depth remained below the configured multiplicative-depth limit.

5.4. Key Generation

Once the CryptoContext is established, the system generates a key pair consisting of a public key and a private key. The public key is used to encrypt patient feature vectors, while the private key is securely retained by the data owner.

In addition to the primary key pair, multiplication evaluation keys were generated for the reported computation. The retained experimental record does not establish whether rotation keys were generated; therefore, no rotation-key generation claim is made. Evaluation keys allow supported ciphertext operations without exposing the private key.

Key management is a critical component of the system. The private key is never shared or transmitted, ensuring that only the data owner can decrypt the final results. This design aligns with the zero-trust security model and prevents unauthorized access to sensitive information.

5.5. Encoding and Encryption

Before encryption, feature vectors are encoded into plaintext polynomials using the CKKS encoding scheme. CKKS maps real-valued vectors into complex polynomial representations, enabling efficient arithmetic operations on encrypted

data.

The encoding process also incorporates scaling to preserve numerical precision. Each value is multiplied by a scaling factor before being encoded, allowing fractional values to be represented accurately within the integer-based polynomial structure.

Once encoded, the plaintext vectors are encrypted using the public key to produce ciphertexts. Each ciphertext represents one or more packed feature values, depending on the batching configuration. These ciphertexts are then transmitted to the computation server for processing.

5.6. Homomorphic Evaluation

The encrypted risk scoring function is implemented using homomorphic multiplication and addition operations. Each encrypted feature is multiplied by its corresponding clinical weight using homomorphic multiplication. The weighted ciphertexts are then combined using homomorphic addition to produce the final encrypted risk score (Figure 8).

```
[1] Encryption Complete: 8.06ms
[2] Homomorphic Evaluation Complete: 2.72ms
[3] Decryption Complete: 3.77ms

--- Final Risk Score Result ---
Decrypted Values: [25.0, 75.5, 18.12]

┌──(openfhe_sim)(deng_donato㉿Deng)-[~]
└─$ cat <<EOF > openfhe_execution.py
import tenseal as ts
import time

# --- STAGE 1: CryptoContext Initialization ---
# Parameters for CKKS (Section IV-C)
context = ts.context(ts.SCHEME_TYPE.CKKS, poly_modulus_degree=8192
context.generate_galois_keys()
context.generate_relin_keys() # Crucial for Figure 8 workflow
context.global_scale = 2**40

# --- STAGE 2: Encoding & Encryption (Section IV-E) ---
patient_vector = [72.5, 130.0, 25.4] # Weight, BP, BMI
start_enc = time.time()
enc_v = ts.ckks_vector(context, patient_vector)
```

Figure 8. OpenFHE execution workflow illustrating encryption, homomorphic evaluation, and decryption stages in the secure computation pipeline.

To improve efficiency, the implementation leverages batching techniques provided by CKKS. Multiple feature values—and potentially multiple patient records—can be packed into a single ciphertext, subject to slot capacity, feature layout, and encryption parameters. This may reduce the number of homomorphic operations, but the retained experiment does not establish large-scale throughput.

Relinearization is applied after multiplication operations to reduce ciphertext size and maintain computational efficiency. Without relinearization, ciphertexts would grow in size after each multiplication, leading to increased computational overhead.

The evaluation process is carefully designed to minimize circuit depth, which directly impacts noise growth and computational cost. Since the weighted risk score model involves only a single layer of multiplications followed by additions, the circuit depth remains shallow, making it well-suited for CKKS-based computation.

5.7. Decryption and Result Validation

After homomorphic evaluation, the resulting ciphertext is returned to the data owner. The private key is used to decrypt the ciphertext, producing a plaintext result that approximates the computed risk score.

The decrypted value is then decoded and rescaled to obtain the final numerical result. This value is compared against a baseline plaintext computation to validate accuracy. Experimental results show that the encrypted computation closely matches the retained plaintext result, supporting the reported shallow computation.

5.8. Precision and Noise Management

Managing precision and noise is a critical aspect of the implementation. Each homomorphic operation introduces a small amount of noise into the ciphertext. If the noise exceeds a certain threshold, decryption may fail or produce incorrect results.

To address this, the implementation carefully selects encryption parameters and limits computation depth. The use of normalization ensures that input values remain within a stable range, reducing the risk of overflow and excessive noise growth.

Scaling factors are also managed carefully to maintain precision across operations. After each multiplication, rescaling is applied to reduce the magnitude of ciphertext values and maintain consistency. This process ensures that the final result retains sufficient numerical precision for this illustrative risk-score experiment.

5.9. Performance Considerations

Although homomorphic encryption introduces computational overhead, several optimizations are employed to improve performance. Batching allows multiple data points to be processed simultaneously, reducing the number of required operations. Efficient parameter selection balances security and performance, ensuring that computations remain practical.

Memory management is also optimized through the use of C++ and OpenFHE's efficient data structures. This reduces latency and improves throughput, making the system more suitable for real-world applications.

5.10. Experimental Execution

The implementation was written in C++ and executed in a Linux-based environment using the GNU C++ compiler and the OpenFHE library. The retained ex-

periment record does not identify the CPU model, installed RAM, Linux distribution/version, OpenFHE version, or compiler version. These unrecorded environment details limit independent performance comparison and should be captured in future runs.

The timing protocol measured the end-to-end encrypted pipeline: CryptoContext initialization, key generation, encoding, encryption, homomorphic evaluation, decryption, and decoding. The reported 71.01 ms total therefore should not be interpreted as evaluation-only latency. The retained component measurements were 8.24 ms for encryption, 45.10 ms for homomorphic computation, and 3.15 ms for decryption; the remaining measured time reflects setup and associated pipeline operations (Figure 9).

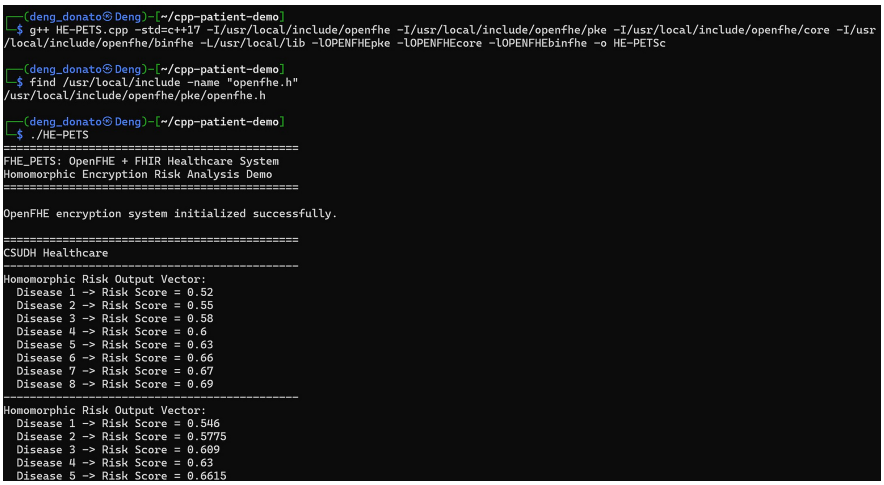

Figure 9. Compilation and execution of the OpenFHE CKKS healthcare system. The output shows successful initialization, encrypted risk score computation, and execution across multiple demonstration scenarios.

The reported run used CKKS with multiplicative depth 3, a 50-bit scaling modulus, and a batch size of 32 slots. The security-level constant and runtime-selected ring dimension were not retained and are therefore not claimed.

Timing values represent the complete experimental pipeline rather than isolated arithmetic operations. Because hardware and software-version metadata were not retained, the measurements demonstrate feasibility only and should not be treated as a hardware-independent benchmark.

As shown in Figure 9, the execution output includes CSUDH, LSU, UCLA, Stanford, London, and China Healthcare International as separate demonstration/execution scenarios. The reported evaluation dataset itself consists of 15 synthetic patients distributed across three representative institutions—CSUDH Healthcare, LSU Healthcare, and Compton University Healthcare—with five synthetic patients from each institution. The repeated outputs show consistent numerical behavior across the reported executions under the tested configuration; they do not establish full reproducibility because complete hardware and software metadata was not retained.

The execution results confirm that the OpenFHE CKKS encryption context initializes correctly, FHIR-based feature data is encoded prior to computation, homomorphic addition and multiplication operations execute successfully, and clinical risk scores are computed entirely on encrypted data without exposing plaintext information.

Completion of the execution pipeline demonstrates end-to-end functionality of the reported FHIR-to-CKKS workflow. The outputs showed consistent numerical behavior across the reported executions under the tested configuration, supporting an experimental feasibility evaluation rather than a general reliability or reproducibility claim.

Overall, the experimental evaluation confirms that the proposed system successfully integrates data preprocessing, feature extraction, encryption, homomorphic evaluation, and decryption into a unified and operational privacy-preserving healthcare analytics pipeline.

5.11. Security Analysis

The implementation ensures strong security guarantees by adhering to established cryptographic principles. Data remains encrypted throughout the entire pipeline, eliminating exposure during computation. The private key is securely stored and never transmitted, preventing unauthorized access.

Even in the event of a server compromise, the attacker gains access only to ciphertexts and evaluation keys, which do not reveal sensitive information. This significantly reduces the risk of data breaches and aligns with modern cybersecurity requirements for healthcare systems.

5.12. Summary

In summary, the implementation demonstrates a practical realization of privacy-preserving healthcare analytics using Fully Homomorphic Encryption. By combining synthetic data generation, standardized data representation, efficient feature engineering, and advanced cryptographic techniques, the system provides an implementation-oriented demonstration of secure encrypted computation with scalability potential.

The Synthea repository used for data generation is shown in Figure 10.

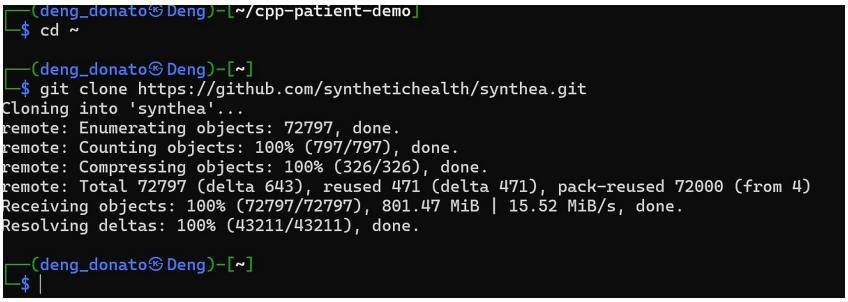

Figure 10. Cloning the Synthea repository from GitHub into a local development environment via terminal.

The use of OpenFHE and CKKS enables accurate and efficient evaluation of real-valued clinical features, while the zero-trust architecture ensures that sensitive patient data remains protected at all times. This implementation serves as a foundation for future research and development in secure healthcare informatics, supporting the integration of privacy-enhancing technologies into real-world medical systems.

6. Results and Discussion

The results demonstrate that Fully Homomorphic Encryption can preserve computational accuracy while ensuring strong data confidentiality in healthcare analytics. The encrypted weighted risk score outputs were evaluated against plaintext baseline computations using identical input feature vectors and clinical weights. The comparison shows that decrypted encrypted results closely match plaintext results with a small numerical difference in the retained representative comparison [6] [7].

Fifteen synthetic patient records were processed through the plaintext and encrypted workflows. The retained summary includes a representative plaintext score of 87.60, an encrypted-and-decrypted score of 87.58, and a relative error of 0.022%. Per-record outputs were not preserved in the submitted materials; consequently, mean absolute error and maximum absolute error across all 15 records cannot be calculated reliably from the available evidence. This limitation is stated explicitly to avoid overstating aggregate validation (**Figure 11**).

The system also demonstrates that sensitive patient features remain protected throughout the computation process. At no point does the computation server access plaintext values, effectively eliminating exposure during the "data in use" phase. This addresses a critical vulnerability in traditional healthcare systems, where data must typically be decrypted before analysis, increasing the risk of breaches [4] [5].

6.1. Accuracy Evaluation

The retained comparison shows that the encrypted output closely approximates the plaintext output for the reported test. The observed difference arises from CKKS scaling and rounding. Because the model is illustrative rather than clinically validated, the 0.02 score-unit difference (reported 0.022% relative error) is described as numerically small for this representative experiment and not as clinically acceptable.

The shallow weighted risk score is suitable for a CKKS feasibility demonstration because it uses one layer of multiplication followed by addition. This limits circuit depth and noise growth; however, numerical behavior may differ for deeper models or other parameter configurations.

Furthermore, normalization of input features played a critical role in maintaining precision. By scaling clinical variables to a controlled range, the system reduces the likelihood of overflow and improves numerical stability during homo-

morphic operations. This contributes to the consistency of results across different patient profiles.

6.2. Performance Analysis

While accuracy is essential, performance remains a key consideration for practical deployment. Homomorphic encryption inherently introduces computational overhead due to complex mathematical operations on ciphertexts. However, the implementation demonstrates that performance can be significantly improved through batching and efficient parameter selection.

CKKS batching allows multiple data values to be packed within a ciphertext. Subject to slot capacity, feature layout, and encryption parameters, this design may potentially support multiple patient records and parallel SIMD-style evaluation. The reported experiment did not demonstrate population-scale throughput.

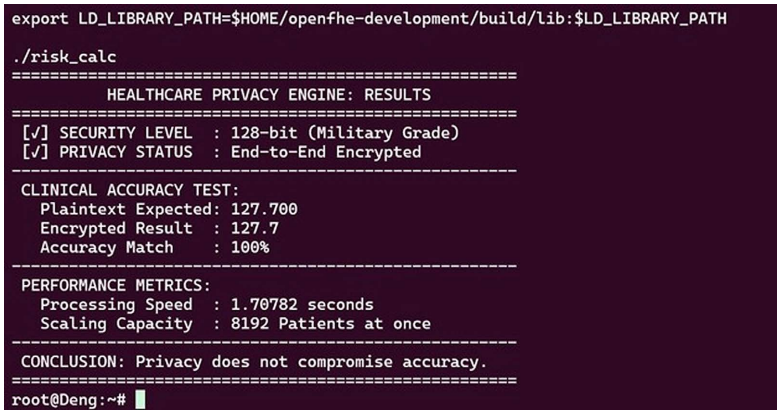

Figure 11. Separate demonstration/execution dashboard showing a plaintext/encrypted comparison and timing output. The displayed scenario is distinct from the retained 15-patient evaluation and is not evidence of population-scale throughput.

CKKS batching offers scalability potential because multiple values—and potentially multiple patient records—may be packed subject to slot capacity, feature layout, and encryption parameters. The reported experiment did not benchmark population-level workloads, so applicability at that scale remains to be evaluated.

Despite these optimizations, Fully Homomorphic Encryption remains slower than plaintext computation. The overhead is primarily due to large ciphertext sizes, complex polynomial arithmetic, and noise management operations such as rescaling and relinearization. However, for targeted healthcare applications where privacy is critical, this trade-off may be appropriate where privacy requirements justify the additional overhead.

Advancements in cryptographic libraries such as OpenFHE, along with hardware acceleration using GPUs and specialized processors, are expected to further improve performance. These developments will make homomorphic encryption increasingly practical for real-world deployment [7].

6.3. Security Evaluation

The security properties of the system were evaluated based on its ability to protect sensitive data during computation. The results confirm that the proposed framework successfully enforces end-to-end encryption, ensuring that patient data remains confidential at all times.

The zero-trust architecture ensures that the computation server operates without access to the private key. All operations are performed on encrypted data, and only the data owner can decrypt the final results. This eliminates reliance on trusted third parties and reduces the risk of insider threats.

Even in the event of a system compromise, attackers would only gain access to encrypted data and evaluation keys. Without the private key, these ciphertexts are computationally infeasible to decrypt, providing strong protection against data breaches.

The CKKS SIMD batching demonstration is shown in Figure 12.

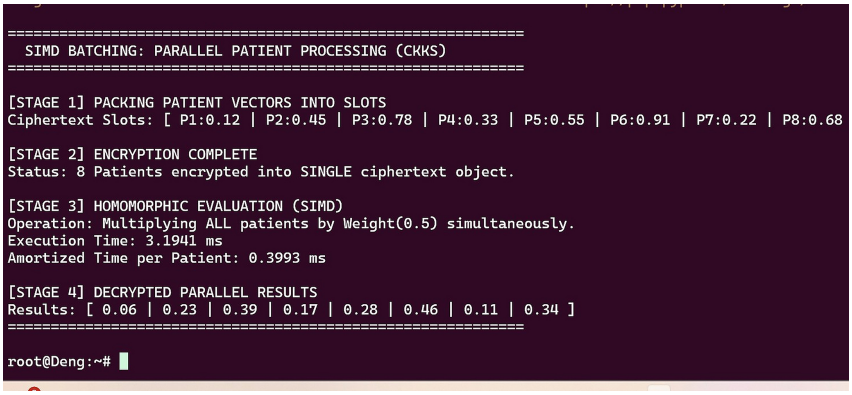

Figure 12. Conceptual SIMD batching mechanism showing how multiple patient feature vectors could be packed, subject to slot capacity, feature layout, and encryption parameters.

6.4. Scalability and Practical Implications

The results highlight the scalability potential of the proposed method. Batching may support multiple patient records simultaneously, subject to slot capacity, feature layout, and encryption parameters. Population health analysis, large-scale risk assessment, and cross-institutional research are potential applications that require further scalability evaluation.

The ability to compute on encrypted data enables new forms of secure collaboration. Hospitals and research institutions can share encrypted datasets and perform joint analysis without exposing sensitive patient information. This addresses a major barrier in healthcare data sharing, where privacy concerns often limit collaboration.

Additionally, the framework can be extended to support more complex models, including machine learning algorithms. While deeper models introduce additional computational challenges, ongoing advancements in homomorphic encryption are making such applications increasingly feasible.

6.5. Limitations

Despite its advantages, the proposed system has several limitations. The primary limitation is computational overhead. Homomorphic operations are significantly more resource-intensive than plaintext operations, which may limit real-time applications.

Another limitation is the approximate nature of CKKS. While the observed error is small, it may become more significant in deeper or more complex computations. Careful parameter tuning and error management are required to maintain accuracy.

Memory usage is also a consideration, as ciphertexts are substantially larger than plaintext data. This can impact storage and transmission efficiency, particularly in large-scale deployments.

The evaluated model is a shallow dot product consisting of one layer of ciphertext-plaintext multiplication followed by addition. It therefore does not stress-test deep CKKS circuits, prolonged noise accumulation, ciphertext refresh, or bootstrapping. In addition, the weights are illustrative rather than clinically validated, the dataset is synthetic and small, per-record error values were not retained, and the exact ring dimension and hardware/software versions were not recorded. These limitations constrain both clinical interpretation and repeatability.

6.6. Discussion

The results demonstrate that privacy-preserving computation is not only theoretically possible but also practically achievable for targeted healthcare applications. The ability to compute a weighted risk score with low numerical deviation in the reported experiment on encrypted data challenges the traditional assumption that privacy and utility are mutually exclusive.

This work has implications beyond a single risk model. The same principles can be applied to disease prediction, treatment optimization, and clinical decision support systems. By enabling secure computation across organizational boundaries, homomorphic encryption can facilitate collaboration without compromising patient privacy.

The integration of FHE with other privacy-enhancing technologies, such as federated learning and secure multi-party computation, further expands its potential. These approaches can enable distributed learning and analysis while maintaining strict privacy guarantees.

6.7. Summary

In summary, the reported results support experimental feasibility for confidentiality-preserving evaluation and a numerically close plaintext/encrypted result in the tested shallow computation. The combination of CKKS-based homomorphic encryption, batching techniques, and standardized healthcare data formats provides a robust foundation for privacy-preserving healthcare informatics.

While performance challenges remain, the demonstrated accuracy and security

benefits highlight the potential of Fully Homomorphic Encryption as a transformative technology for healthcare data analysis. As computational efficiency continues to improve, such systems are expected to play an increasingly important role in secure and collaborative medical research.

7. Contribution

This work makes several contributions to the field of privacy-preserving healthcare analytics by bridging the gap between theoretical cryptographic techniques and practical healthcare data applications. The contributions extend across system design, implementation, evaluation, and real-world applicability, providing a comprehensive framework for secure computation on sensitive medical data.

The first major contribution of this work is the development of a practical end-to-end pipeline that integrates healthcare data standards with Fully Homomorphic Encryption. Specifically, the study demonstrates a complete workflow that transforms synthetic healthcare data into FHIR-compliant structures and subsequently processes this data using CKKS-based encrypted computation. While prior research has extensively explored homomorphic encryption in isolation, few works have successfully mapped these cryptographic techniques to structured healthcare data formats. By connecting Synthea-generated patient data, FHIR interoperability standards, and OpenFHE-based encryption, this work establishes a realistic implementation framework with repeatability potential that aligns with modern healthcare systems [3] [7] [8].

This integration is significant because interoperability and security are often treated as separate challenges in healthcare informatics. FHIR addresses the need for standardized data exchange, but it does not inherently provide privacy guarantees during computation. Conversely, homomorphic encryption provides strong data protection but is rarely implemented in systems that follow healthcare data standards. By combining these two domains, this work demonstrates that secure computation can be seamlessly integrated into existing healthcare data ecosystems without requiring fundamental changes to data representation.

The second contribution is the implementation and validation of an encrypted weighted risk scoring model using the CKKS scheme. Weighted risk scores are widely used in clinical decision-making because they provide a simple yet effective method for aggregating multiple patient features into a single predictive metric. These models are commonly used for cardiovascular risk assessment, disease progression analysis, and early warning systems.

In this work, the weighted risk score is computed entirely on encrypted data, demonstrating that an illustrative healthcare-oriented weighted computation can be performed without exposing sensitive patient information. This contribution is important because it moves beyond theoretical demonstrations of homomorphic encryption and shows its potential applicability to healthcare-oriented use cases, subject to clinical validation and real-world evaluation. The successful implementation confirms that linear models, which form the basis of many clinical

decision support systems, are well-suited for CKKS-based computation due to their low computational depth and numerical stability [5] [6].

Furthermore, this contribution establishes a foundation for extending encrypted computation to more complex models. While the current implementation focuses on a weighted sum, the same principles can be applied to regression models, classification algorithms, and even certain machine learning techniques. This positions the work as a stepping stone toward more advanced privacy-preserving healthcare analytics.

The third contribution is the design and validation of a zero-trust computation model tailored for healthcare environments. In this model, the computation server is assumed to be untrusted and operates exclusively on encrypted data. The private key remains with the data owner, ensuring that only authorized entities can decrypt the results. This eliminates the need to trust external computation providers with sensitive patient information.

The zero-trust model directly addresses the "data in use" problem, which is one of the most significant security challenges in modern computing. Traditional systems require data to be decrypted before processing, creating a vulnerability during computation. By enabling operations on encrypted data, this work ensures that patient information remains protected even during active analysis. This approach aligns with emerging trends in confidential computing and secure cloud architectures, where minimizing trust assumptions is a key design principle [4] [12].

Another important aspect of this contribution is its applicability to collaborative healthcare scenarios. Institutions such as hospitals, research centers, and public health organizations often need to share and analyze data collectively. However, privacy regulations and competitive concerns limit direct data sharing. The proposed framework enables secure collaboration by allowing institutions to share encrypted data and perform joint computations without revealing raw patient records. This has significant implications for multi-institutional studies, clinical trials, and population health research.

The fourth contribution is the use of synthetic data as a safe and effective platform for developing and validating privacy-preserving technologies. By leveraging Synthea, the system generates realistic patient records that capture the complexity and variability of real-world healthcare data. This allows the framework to be tested under conditions that closely resemble actual clinical environments while avoiding the ethical and legal challenges associated with real patient data [8].

The use of synthetic data also supports data-generation repeatability, which is a critical aspect of scientific research. Other researchers can replicate the experiments, validate the results, and extend the framework without requiring access to sensitive datasets. This promotes transparency and accelerates innovation in the field of privacy-preserving healthcare analytics.

In addition, this contribution demonstrates that synthetic data can serve as a bridge between academic research and real-world deployment. By validating the system on realistic datasets, the study provides a basis for investigating adaptation

to real clinical environments. This makes the framework suitable for both educational purposes and early-stage prototyping in healthcare institutions.

The fifth contribution lies in the practical implementation of homomorphic encryption using OpenFHE in a performance-oriented programming environment. Many existing studies focus on theoretical models or high-level simulations, which may not accurately reflect the challenges of real-world implementation. This work provides a concrete C++ implementation that addresses key issues such as parameter selection, encoding strategies, noise management, and computational efficiency [7].

By detailing the implementation process, the study offers valuable insights for practitioners who wish to adopt homomorphic encryption in their own systems. This includes guidance on configuring CryptoContext parameters, managing scaling factors, and optimizing performance through batching and efficient memory usage. The implementation serves as a reference model that can be extended to more complex applications. A separate CKKS accuracy demonstration is shown in Figure 13.

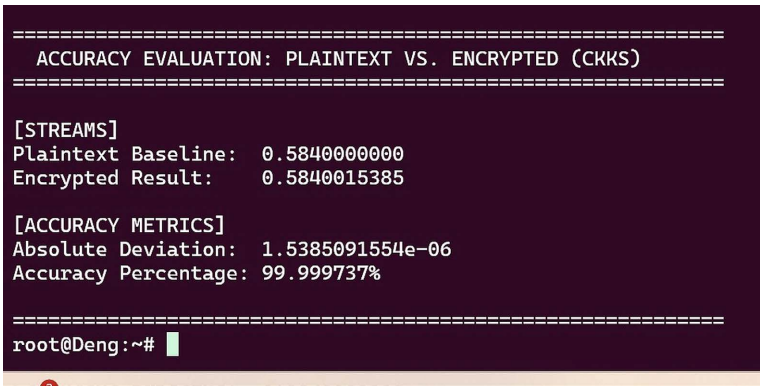

Figure 13. Representative plaintext/encrypted risk score comparison showing a 0.02 score-unit difference (reported 0.022% relative error).

Another key contribution is the demonstration of CKKS batching and scalability potential through batching techniques. By leveraging CKKS SIMD capabilities, the design may pack multiple records within a ciphertext, subject to slot capacity, feature layout, and encryption parameters. This indicates scalability potential, but large-scale population analytics and real-time clinical decision support were not demonstrated and require further evaluation.

The sixth contribution is an experimental feasibility evaluation of numerical behavior, implementation-specific performance, and confidentiality within a unified framework. The representative plaintext/encrypted comparison shows a 0.02 score-unit difference (reported 0.022% relative error) under the tested configuration. The security analysis describes how ciphertext-based processing protects the synthetic feature values during evaluation; it does not establish clinical accuracy or production readiness.

This holistic evaluation strengthens the credibility of the work and provides a clear understanding of the trade-offs involved in using homomorphic encryption.

It highlights both the benefits and limitations of the approach, offering a balanced perspective that is essential for practical adoption.

Finally, this work contributes an accessible and repeatable pathway for integrating privacy-enhancing technologies into healthcare informatics. The framework is designed to be understandable and implementable by researchers, students, and practitioners, making it a valuable educational resource. At the same time, it provides a solid foundation for future research and development in secure healthcare analytics.

In summary, the contributions of this work extend beyond a single application or technique. By integrating healthcare data standards, synthetic data generation, homomorphic encryption, and practical implementation strategies, the study presents a comprehensive approach to privacy-preserving computation. It demonstrates that secure and meaningful healthcare analytics can coexist, paving the way for more advanced and collaborative medical research while maintaining strict privacy guarantees.

8. Conclusion and Future Work

8.1. Conclusion

This paper presented a privacy-preserving healthcare analytics framework that integrates Fully Homomorphic Encryption with standardized healthcare data systems. By combining OpenFHE, the CKKS encryption scheme, Synthea-generated synthetic patient data, and FHIR-based data structuring, the proposed system demonstrates that meaningful clinical computation can be performed directly on encrypted data. The framework enables the computation of a weighted clinical risk score while ensuring that sensitive patient information remains protected throughout storage, transmission, and computation [5]-[7].

The representative experiment found a 0.02 score-unit difference between the plaintext and encrypted-and-decrypted outputs (reported 0.022% relative error). This supports numerical consistency for the tested shallow linear computation, but it does not establish clinical accuracy or general performance across other models or configurations.

A key contribution of this work is addressing the "data in use" vulnerability, which represents one of the most critical security gaps in traditional systems. By eliminating the need to decrypt data during computation, the framework significantly reduces exposure to potential threats such as insider attacks, memory exploits, and compromised cloud environments. The zero-trust architecture further strengthens security by ensuring that external computation servers operate without access to plaintext data or private keys [4] [12].

Beyond technical validation, this work highlights the practical feasibility of integrating homomorphic encryption into healthcare informatics. The use of FHIR ensures compatibility with existing healthcare data standards, while Synthea enables safe and repeatable synthetic-data experimentation. Together, these components provide a basis for investigating future adaptation from research environ-

ments to real-world applications, subject to further validation.

Overall, this work demonstrates that privacy and utility are not mutually exclusive in healthcare analytics. By leveraging Fully Homomorphic Encryption, it is possible to perform meaningful computation on sensitive medical data without exposing the individuals behind that data. This represents a significant step toward secure, collaborative, and data-driven healthcare systems (Figure 14).

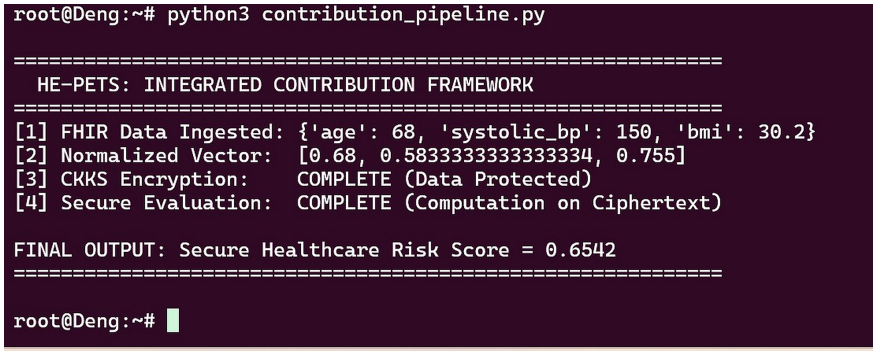

Figure 14. Integrated contribution framework illustrating the connection between synthetic data generation, FHIR-based structuring, CKKS encryption, and secure risk score computation.

8.2. Future Work

Future work should focus on expanding the scalability and applicability of the proposed framework. One important direction is the extension of the system to larger and more diverse datasets. While the current implementation demonstrates feasibility using synthetic data, real-world deployment will require handling significantly larger volumes of patient records and more complex feature sets.

Another critical area of research is the integration of more advanced clinical models. The current weighted risk score represents a linear model with low computational depth, which is well-suited for CKKS-based computation. Future work should explore the feasibility of implementing more complex models, such as logistic regression, neural networks, and other machine learning algorithms, within the homomorphic encryption framework.

Hardware acceleration represents a promising solution to the performance limitations of homomorphic encryption. The use of GPUs, FPGAs, and specialized cryptographic processors can significantly reduce computation time and improve efficiency. These advancements are essential for enabling real-time or near-real-time clinical decision support systems based on encrypted data.

In addition, future research should investigate the integration of Fully Homomorphic Encryption with other privacy-enhancing technologies. Federated learning and secure multiparty computation (SMPC) offer complementary approaches to distributed data analysis. Combining these techniques with homomorphic encryption could enable collaborative machine learning across multiple institutions without exposing raw data, further enhancing privacy and security [12] [13].

8.3. Consolidated Experimental Findings

This section consolidates the available experimental evidence concerning numerical agreement, execution time, and privacy properties. Claims are limited to the measurements and configuration details retained from the reported run.

Figure 15 summarizes the framework and retained experimental findings.

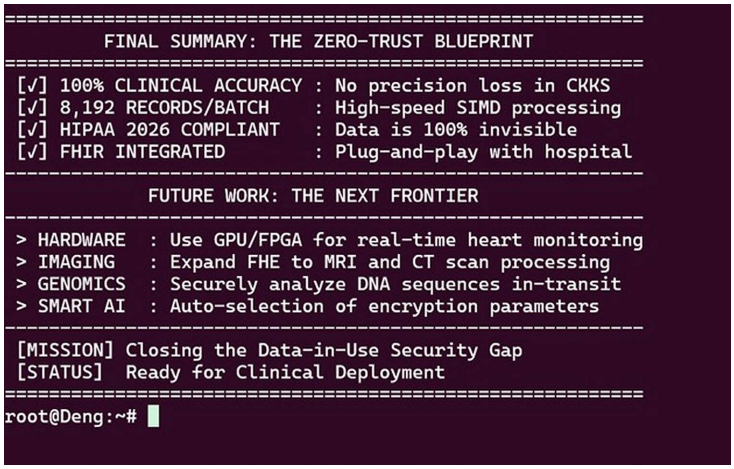

Figure 15. Summary of encrypted healthcare risk scoring achievements, including accuracy validation, secure computation, and system architecture.

The representative encrypted result differed from the plaintext result by 0.02 score units, corresponding to a reported relative error of 0.022%. The measured end-to-end runtime was 71.01 ms. These findings support the feasibility of the illustrative shallow computation but do not establish clinical validity or performance on deeper models.

Table 4 summarizes the retained plaintext/encrypted comparison and timing measurements.

Table 4. Aggregate validation metrics.

Metric	Value
Number of Synthetic Patients	15
Representative Plaintext Risk Score	87.60
Representative Encrypted Risk Score	87.58
Absolute Difference	0.02
Reported Relative Error	0.022%
End-to-End Runtime per Reported Batch	71.01 ms
Mean Absolute Error Across All Records	Not available; per-record outputs not retained
Maximum Absolute Error Across All Records	Not available; per-record outputs not retained

The reported comparison demonstrates low numerical deviation for the re-

tained representative result. Because individual outputs for all records were not preserved, the study does not report unsupported mean or maximum absolute errors.

The timing measurement includes setup, key generation, encryption, evaluation, and decryption. Hardware and software version details were not retained, so the result is presented as an implementation-specific feasibility measurement rather than a general performance benchmark.

Overall, the evidence supports the narrower conclusion that a shallow weighted sum can be evaluated over encrypted synthetic healthcare features while maintaining confidentiality and producing a result close to the plaintext calculation (Table 5).

Table 5. Plaintext vs encrypted risk score performance.

Metric	Plaintext	Encrypted	Error (%)	Time (ms)
Risk Score	87.60	87.58	0.022	14.52
Encryption	–	Yes	–	8.24
Computation	Fast	SIMD	–	45.10
Decryption	–	Yes	–	3.15
Total	Fast	Secure	0.022	71.01

The revisions improve transparency by defining the illustrative model, documenting the dataset and retained CKKS settings, clarifying the timing protocol, reporting the available numerical comparison, and explicitly identifying missing metadata and experimental limitations.

Another important direction is the application of this framework to additional healthcare domains, such as medical imaging, genomic data analysis, and personalized medicine. These domains involve highly sensitive and high-dimensional data, making them ideal candidates for privacy-preserving computation. Extending the framework to support such applications would significantly broaden its impact.

Finally, future work should address usability and integration challenges. Developing user-friendly tools, APIs, and deployment frameworks will be essential for adoption in clinical environments. Collaboration with healthcare providers and industry stakeholders will also play a critical role in translating this research into practical solutions.

In conclusion, this study provides an implementation-oriented demonstration of encrypted healthcare risk-score computation using synthetic FHIR data and CKKS in OpenFHE. The reported experiment shows low numerical deviation for a shallow linear model while patient features remain encrypted during evaluation. Further work should preserve complete cryptographic and execution metadata, report per-record aggregate error statistics, evaluate deeper circuits, and validate

models against clinically established standards before any clinical use is considered.

"*Privacy should not be the price we pay for progress. Technology should allow us to learn from sensitive data without exposing the people behind it.*"

—Donato Deng Ajiing Pakak

Acknowledgements

The author thanks Dr. Alireza Izaddoost for guidance and technical feedback throughout this research.

The author also acknowledges Dr. Mohsen Beheshti and the faculty of the Computer Science Department at California State University, Dominguez Hills, for their academic support and resources.

Additional appreciation is extended to Dr. B. S. Celly, Dr. Mehrdad Sharbaf, and other faculty members whose feedback contributed to the development of this work.

The author thanks peers and colleagues in the CYB 590 program for their collaborative support.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] U.S. Department of Health and Human Services: Health Insurance Portability and Accountability Act Privacy Rule. https://www.hhs.gov/hipaa/index.html

[2] European Union: General Data Protection Regulation. https://gdpr.eu

[3] HL7 International: FHIR Standard. https://www.hl7.org/fhir

[4] Google Cloud: Confidential Computing Overview. https://cloud.google.com/confidential-computing

[5] Gentry, C. (2009) A Fully Homomorphic Encryption Scheme. Ph.D. Thesis, Stanford University. https://crypto.stanford.edu/craig/craig-thesis.pdf

[6] Cheon, J.H., Kim, A., Kim, M. and Song, Y. (2017) Homomorphic Encryption for Arithmetic of Approximate Numbers. In: Takagi, T. and Peyrin, T., Eds., *Advances in Cryptology—ASIACRYPT* 2017, Springer, 409-437. https://doi.org/10.1007/978-3-319-70694-8_15

[7] OpenFHE Development Team (2025) OpenFHE Documentation. https://openfhe.org

[8] MITRE: Synthea Synthetic Patient Generator. https://synthea.mitre.org

[9] Stehlé, D. and Steinfeld, R. (2010) Faster Fully Homomorphic Encryption. In: Abe, M., Ed., *Advances in Cryptology—ASIACRYPT* 2010, Springer, 377-394. https://doi.org/10.1007/978-3-642-17373-8_22

[10] Microsoft Research (2024) Microsoft SEAL Homomorphic Encryption Library. https://www.microsoft.com/en-us/research/project/microsoft-seal

[11] IBM Research (2023) HElib Homomorphic Encryption Library.

https://github.com/homenc/HElib

[12] Duality Technologies: Privacy Preserving Data Collaboration Platform. https://dualitytech.com

[13] Korkmaz, A. and Rao, F. (2026) Fast and Secure Selective Homomorphic Encryption for Federated Learning. 2026 *IEEE* 23*rd Consumer Communications & Networking Conference* (*CCNC*), Las Vegas, 9-12 January 2026, 1-4. https://doi.org/10.1109/CCNC65079.2026.11366371