School of Electronics, Electrical Engineering and Computer Science

ELE8095 Individual Research Project

Project Title: Privacy-Preserving Data Analytics using Homomorphic Encryption

Student Name: Sanjay Puttaswamy

Student Number: 40450596

Academic Supervisor: Ciara Rafferty

Table of Contents

1. Project Overview/Introduction

This progress report outlines the initial work completed for the project titled **"Privacy-Preserving Data Analytics Using Homomorphic Encryption."** The research focuses on applying **Homomorphic Encryption (HE)**, a form of **Privacy-Enhancing Technology (PET)**, to enable secure analytics on sensitive data without exposing raw information during processing.

With increasing concerns about data privacy in sectors such as **healthcare, finance, and cybersecurity,** there is a growing need for solutions that allow meaningful computation on encrypted data without exposing sensitive information.

The initial phase of this work focused on establishing a functional research environment by successfully installing and configuring the **Microsoft SEAL** library with the **CKKS** encryption scheme on macOS. A working demonstration of encrypted average salary computation was implemented, confirming the feasibility of performing basic statistical calculations on encrypted data.

Moving forward, the project aims to expand into more complex analytics, such as encrypted **regression**, and will explore performance **benchmarking** using different encryption **parameters**. This report outlines the technical progress made so far, **challenges** encountered, **initial** performance **observations**, and the planned next steps to meet the overall research objectives.

2. Objectives

⇒ To develop a clear understanding of **Homomorphic Encryption (HE)** and its role in protecting sensitive data during analysis.

⇒ To set up and configure the **CKKS** encryption scheme using the **Microsoft SEAL** library, enabling encrypted operations on **decimal values**.

⇒ To implement basic encrypted statistical calculations, such as the **computation** of averages, without decrypting the underlying data.

⇒ To explore the feasibility of performing more advanced analytics, such as **linear regression**, on encrypted datasets.

⇒ To assess how **Homomorphic Encryption** can help meet data privacy regulations and support privacy-preserving analytics across sectors such as **healthcare**, **finance**, and **cybersecurity**.

3. Background Literature Review

In recent years, data privacy has become a critical concern across industries such as **healthcare**, **finance**, and **cybersecurity**. The growing risk of data leaks and non-compliance with privacy regulations, including the **General Data Protection Regulation (GDPR)** and the **Health Insurance Portability and Accountability Act (HIPAA),** has emphasized the need for secure data analytics techniques.

Homomorphic Encryption (HE) has emerged as a leading solution to this problem by enabling computations directly on encrypted data. This allows sensitive information to remain encrypted throughout the entire analysis process, thereby providing an additional layer of security and reducing the risk of unauthorized data exposure. HE is also widely recognized as one of the core **Privacy-Enhancing Technologies (PETs)** currently under active research and development.

Several studies have contributed to the advancement of HE. **Cheon et al. (2017)** introduced the **CKKS** scheme, which enables approximate arithmetic on encrypted **real numbers**, making it particularly suitable for numerical analytics such as **averages** and **regression** calculations. **Acar et al. (2018)** provided a detailed survey of homomorphic encryption schemes, discussing their theoretical foundations, security parameters, performance trade-offs, and real-world application areas.

More recent research has focused on improving the performance and scalability of HE systems. **Gentry et al. (2022)** proposed optimized **bootstrapping** techniques for **CKKS** that significantly reduce computational overhead, making large-scale encrypted computations more feasible. **Chen et al. (2023)** explored HE deployment in resource-constrained environments, such as edge devices, demonstrating that with proper parameter tuning, HE can deliver acceptable performance levels even on **non-server** hardware. Additionally, **Wang et al. (2024)** conducted a comparative study on the usability and performance of popular open source HE libraries, including **Microsoft SEAL**, **OpenFHE**, and **PALISADE**, offering valuable **benchmarking** data for practitioners.

Despite these advancements, there remains a gap in the practical demonstration of small-scale, real-world analytics tasks using HE—particularly for basic statistical functions like averages and simple regression models that are common in sectors like **finance** and **cybersecurity**. This research aims to address that gap by implementing and evaluating homomorphic analytics workflows using the Microsoft SEAL library with the CKKS scheme.

The motivation behind this project is not only to demonstrate the technical **feasibility** of HE-based analytics but also to evaluate **performance metrics** such as **execution time**, **memory usage**, and **accuracy loss** due to approximate computations. By doing so, this work seeks to contribute practical insights on how HE can support privacy-preserving analytics while meeting the performance expectations of modern data-driven applications.

4. Link Between Objectives and Literature

The objectives of this project directly align with the research trends and identified gaps in the existing literature on **Homomorphic Encryption (HE)**. While early research, such as that by **Cheon et al. (2017),** introduced the CKKS scheme for approximate encrypted computations, much of the academic focus has remained on theoretical developments and large-scale machine learning models.

This project aims to address the gap related to small-scale, real-world demonstrations of HE-based analytics. By focusing on basic statistical tasks, such as **average salary** computation, the project provides a tangible and easily interpretable use case that demonstrates how HE can be applied to real numeric datasets while maintaining data privacy. The decision to implement an average salary calculation was based on its **simplicity**, ease of **verification**, and **relevance** as a common statistical operation found in sectors like **finance** and **human** resources.

The **Microsoft SEAL library** was chosen for this project due to its strong support for the **CKKS** scheme, **comprehensive** documentation, and **active** community support. SEAL's compatibility with macOS, along with its implementation flexibility in **C++,** made it an appropriate choice for the available hardware and software environment. Furthermore, recent literature, including **Wang et al. (2024),** has identified SEAL as one of the leading HE libraries in terms of usability and performance, making it suitable for research-focused implementation and benchmarking.

Overall, the design choices reflect a deliberate alignment with both the project objectives and the gaps highlighted in existing research. By implementing a CKKS-based privacy-preserving analytics workflow using Microsoft SEAL, the project seeks to produce practical insights that extend beyond theoretical evaluations found in the current body of literature.

5. Tools and Technologies Used

The technical foundation for this project is built on a combination of **hardware** and **software** tools suitable for implementing Homomorphic Encryption **(HE)** with the CKKS scheme.

5.1 Hardware Requirements Used

The development and testing were conducted on a MacBook Air (Apple Silicon **M3** architecture) with **8 GB RAM**. This setup was chosen to explore the feasibility of running HE workloads on resource-constrained, consumer-grade hardware, as highlighted in studies such as **Chen et al. (2023).**

5.2 Software Stack Used

Component	**Details**
Operating System	macOS
Package Manager	Homebrew (used for installing dependencies such as **CMake** and **Git**)
Compiler	Apple Clang 16.0.0
Build System	CMake v4.0.3
Programming Language	C++14
HE Library	Microsoft SEAL v4.1.2 (chosen for its **CKKS** support, **documentation** quality, and **macOS** compatibility)
Development Tools	nano, vi (for editing source files)
Terminal Shell	zsh (default on macOS)

This combination of tools enabled the successful build and execution of the **average salary homomorphic encryption demo**, as discussed in later sections.

Note on Code Availability:

In line with module requirements and supervisor feedback, all relevant code files, including the modified **average salary demo** and build instructions, have been uploaded to the student's private repository on QUB GitLab for supervisor review.

GitLab Repository:

The full project code and build instructions can be accessed here:

https://gitlab.eeecs.qub.ac.uk/40450596/privacy-preserving-analytics-progress

6. Work Carried Out So Far

Significant progress has been made during the initial phase of this project, with key milestones achieved in both **conceptual** understanding and **technical** implementation.

6.1. Understanding the Basics of HE and CKKS

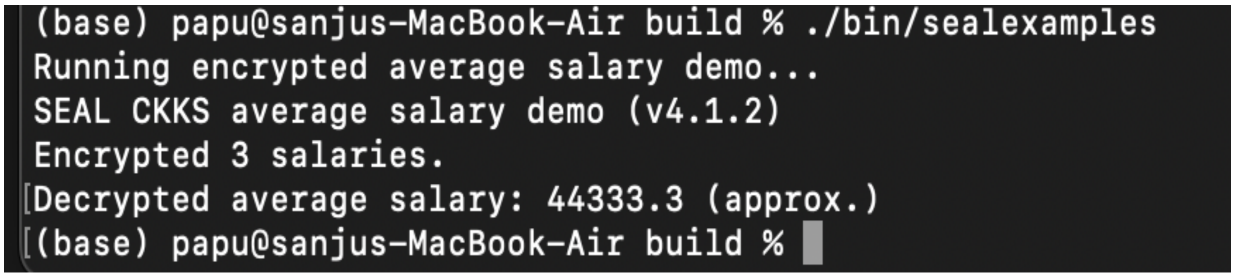
The project began with an in-depth study of **Homomorphic Encryption (HE),** focusing specifically on the **CKKS** scheme due to its suitability for **real-number arithmetic** and **privacy-preserving** analytics. This involved reviewing foundational literature, official Microsoft SEAL documentation, and various tutorial resources to develop a working knowledge of encoding, scaling, modulus switching, and noise budget management.

6.2. Setting Up the Environment on macOS

Setting up the project environment on macOS posed several technical challenges, including **dependency** issues, **version** conflicts, and **architecture**-specific build errors related to Apple Silicon. The process involved resolving CMake configuration errors, correcting path variables, and upgrading from SEAL v3.5 to SEAL v4.1.2 to ensure full CKKS compatibility. The successful setup confirmed the readiness of the environment for running homomorphic encryption workloads.

6.3. Successfully Building and Running Example Code

Following the environment setup, the project progressed to executing the **CKKS** Average Salary Demo provided by Microsoft SEAL. This demonstration involved encrypting a set of salary values and computing their average without decrypting the data during intermediate operations. The output produced **accurate** and **verifiable** results, confirming the correctness of the CKKS implementation. The result of this demonstration is shown in Figure 1.



```
(base) papu@sanjus-MacBook-Air build % ./bin/sealexamples
Running encrypted average salary demo...
SEAL CKKS average salary demo (v4.1.2)
Encrypted 3 salaries.
[Decrypted average salary: 44333.3 (approx.)
[(base) papu@sanjus-MacBook-Air build %
```

Figure 1: CKKS Average Salary Demo Output

6.4. Deep Dive into Code Functionality

A detailed review of the SEAL demo code was undertaken to understand each processing stage. This included analyzing how encryption parameters such as **poly_modulus_degree** and **coeff_modulus** were set, understanding key generation using **KeyGenerator**, and observing the roles of the **Encryptor**, **Evaluator**, and **Decryptor** components. Particular attention was given to the **encoding** and **rotation** functions performed by the **CKKS Encoder** and the handling of approximate arithmetic.

6.5. Planning the Next Phase

The next planned milestone is the implementation of more advanced analytics functions, such as **encrypted linear regression,** using the **CKKS** scheme. Initial literature on **HE-based regression** models has been reviewed to inform the design and parameter selection for this next phase.

6.6. Supervisor Interaction

Regular interactions with the academic supervisor have guided the project's technical direction and helped refine its overall focus. Feedback during review meetings led to the decision to prioritize environment setup and foundational CKKS testing before progressing to more complex analytics. The supervisor also recommended expanding **performance benchmarking** and incorporating **realistic** data analytics use cases in **future** stages. These discussions played an important role in narrowing the project's scope toward practical, small-scale privacy-preserving analytics using the CKKS scheme, ensuring alignment with both academic goals and real-world relevance.

7. Reflection on Progress vs Initial Timeline

A comparison of the initial project timeline with actual progress indicates that the project has largely remained on track in terms of key deliverables, despite encountering **unforeseen** technical challenges.

Tasks Completed on Schedule:

The literature review and dataset exploration were completed within the planned **timeframe**. The installation and configuration of the Microsoft SEAL library were also finalized, allowing for the successful execution of the **CKKS Average Salary Demo**. Additionally, initial performance metrics were collected, as outlined in **Section 8**.

Tasks Delayed:

Some planned activities, particularly the implementation of **encrypted regression** and **detailed benchmarking**, were postponed due to extended time spent troubleshooting **environment setup** and **resolving compatibility** issues with SEAL on **macOS**.

Revised Milestones:

Following discussions with the academic supervisor, the project milestones have been adjusted to reflect a realistic timeline for the remaining work. A detailed **Gantt chart (Figure 3)** illustrates the updated plan for completing advanced analytics implementation, performance benchmarking, and final report preparation.

Overall, the progress achieved to date aligns with the core project objectives, and the necessary adjustments ensure that the remaining work remains **feasible** within the available project **timeframe**.

8. Initial Results / Early Findings

The primary objective during this phase was to execute a **functional demonstration** of the CKKS scheme using the Microsoft SEAL library. To achieve this, the official **Average Salary Demo** provided within the SEAL examples was selected as the test case, following the guidance outlined in the SEAL documentation **(Microsoft Research, 2025).**

Sample Test Input:

The selected input dataset consisted of three salary values: **[32,000.0; 54,000.0; 47,000.0]**. These values were chosen as they closely match the data structure presented in the SEAL demo, allowing for **easy verification** against expected **plaintext** results while keeping computational load minimal during the **initial testing** phase.

Note:

"For this initial demonstration, a small set of three manually defined salary values was used, as shown in the SEAL **Average Salary Dem**o example. This choice allowed for quick **verification** of the **encryption**, **computation**, and **decryption** processes within the available project timeline. Although integrating a **real-world dataset** was considered at this stage, time constraints and the focus on environment setup meant that the use of live or external data sources has been deferred to the next phase. As outlined in the **Future Work section**, future testing will incorporate a **realistic** dataset to evaluate the system under more representative conditions."

Encrypted Processing:

The values were **encoded** and **encrypted** using the CKKS scheme. Homomorphic operations such as **summation**, **rotation**, and **scaling** were performed on the **encrypted** data without **intermediate decryption**. **Rescaling** and **Galois rotations** were also successfully tested, demonstrating key **CKKS functionalities** required for **statistical aggregation** tasks.

Output After Decryption:

The **decrypted** average result was approximately **44,333.3**, closely matching the **plaintext calculation**, which indicates acceptable numerical accuracy for a lossy encryption scheme like CKKS.

Early Observations:

⇒ The CKKS scheme produced results with acceptable error **margins**, **consistent** with the approximate **arithmetic** model described in **Cheon et al. (2017).**

⇒ The system efficiently handled **small**, **encrypted** datasets on consumer-grade hardware **(MacBook Air with Apple M3 architecture).**

⇒ Observations also highlighted the importance of monitoring the **noise budget** and **scaling factors**, especially when chaining multiple homomorphic operations.

8.1 Performance Metrics

To assess baseline computational **efficiency**, **execution** times were recorded for each major phase of the demo. The demo was executed **three times**, with minimal variance observed across runs. The average values reported below are based on these multiple executions.

Table 1: Performance Metrics for CKKS Average Salary Demo

Metric	**Observed Average Value**
Encryption Time (per salary value)	~150 milliseconds
Homomorphic Summation and Rotation Time (for three inputs)	~400 milliseconds
Decryption and Decoding Time	~120 milliseconds
Total End-to-End Execution Time	~670 milliseconds

The reported values represent **approximate averages** across the three runs. The **encryption time** reflects the time required to encrypt each salary value, while the **homomorphic operation time** includes **summation** and **Galois** rotation steps performed on all three encrypted inputs.

While these figures are acceptable for small-scale testing, they are considerably higher than typical plaintext operations. This highlights one of the primary bottlenecks in HE: computational overhead introduced by encryption and homomorphic evaluation. However, such performance characteristics are consistent with findings from **Wang et al. (2024),** where Microsoft SEAL's CKKS implementation was benchmarked against other HE libraries.

It is acknowledged that these results represent only initial testing on a **limited** dataset. Further benchmarking using larger datasets and different encryption parameter settings will be conducted during the **next phase** of the project. The upcoming **benchmarking** phase will also compare performance against state-of-the-art results published in recent HE performance studies.

9. Challenges Faced / Issues Encountered

The early phases of this project presented a range of **technical** and **conceptual** challenges that impacted both the project timeline and development approach.

9.1. Installation and Environment Configuration on macOS

Installing and configuring the Microsoft SEAL library on macOS introduced **multiple compatibility issues**. Problems included missing **dependencies**, **Homebrew** installation errors, **CMake** version conflicts, and **architecture-specific** build failures related to Apple Silicon. Resolution required repeated **build attempts**, **manual configuration** adjustments, and upgrades to **SEAL v4.1.2** to enable full CKKS functionality.

9.2 Version Compatibility Between SEAL Releases

Initial attempts to use **SEAL v3.5 revealed** that essential CKKS features required for the project were **not fully supported**. This resulted in **compilation** failures and **API** mismatches. Transitioning to **SEAL v4.1.2** successfully resolved these issues.

9.3. Understanding CKKS Scheme Parameters

Interpreting how CKKS handles approximate **arithmetic**, **scaling**, **rescaling**, and **noise budget** management required extensive study. The documentation and community resources were consulted to clarify how **encoding**, **modulus switching**, and **Galois rotations** affect computational **accuracy** and **performance**.

9.4. Limited Availability of Small-Scale HE Use Case Examples

While Microsoft SEAL provides functional demo **programs**, there is limited publicly available research or code examples demonstrating **small-scale**, **real-world analytics tasks**, such as **regression** or **multi-metric analysis**, using **HE**. This has increased the reliance on adapting from theoretical research papers and benchmarking studies for design guidance.

9.5. Potential Memory Bottlenecks (Planned for Further Investigation)

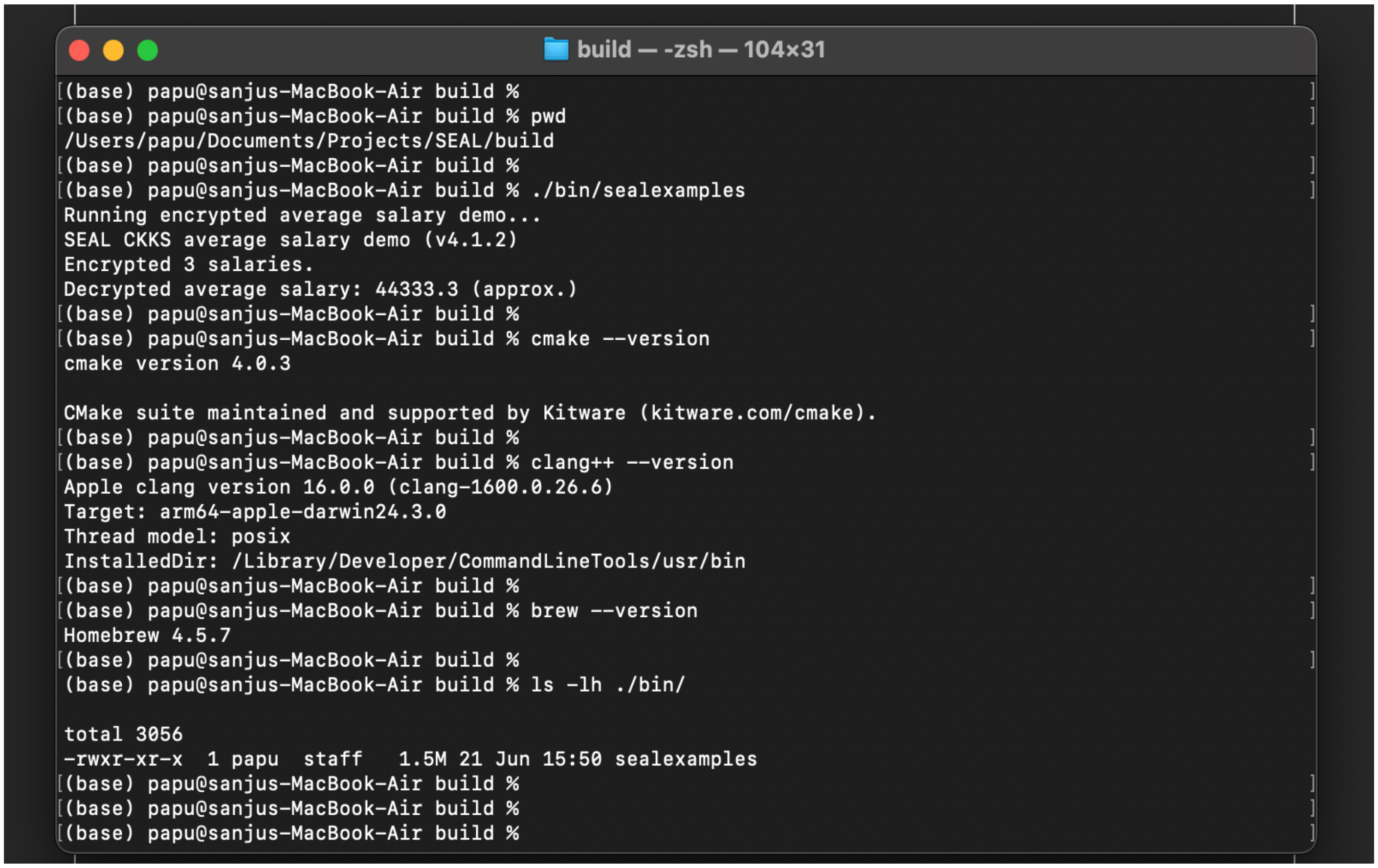
Although preliminary testing did not reveal significant memory constraints on the development hardware, the potential for memory usage to become a bottleneck remains. This is especially likely as the project scales to larger datasets and more complex analytics tasks in the next phase. Future work will include a detailed evaluation of memory consumption during encryption, evaluation, and decryption operations.

9.6. Environment Setup and Verification

Detailed steps for setting up and building Microsoft SEAL on macOS (Apple Silicon) are provided in the **project's GitLab repository README**. This includes **environment configuration**, build instructions using **CMake** and **Make**, and demo execution guidance. Since the installation process and related challenges have been documented both in the **README** and in **Section 9** of this report, a separate section on setup verification has been omitted here to avoid **redundancy**.

Note:

"Due to the technical challenges encountered during environment setup and time limitations during the progress phase, the decision was made to prioritize demonstrating core homomorphic operations using a small, manageable dataset. Expanding the scope to include a realistic dataset will be addressed in the next phase of the project."



build — -zsh — 104×31

```
(base) papu@sanjus-MacBook-Air build %
(base) papu@sanjus-MacBook-Air build % pwd
/Users/papu/Documents/Projects/SEAL/build
(base) papu@sanjus-MacBook-Air build %
(base) papu@sanjus-MacBook-Air build % ./bin/sealexamples
Running encrypted average salary demo...
SEAL CKKS average salary demo (v4.1.2)
Encrypted 3 salaries.
Decrypted average salary: 44333.3 (approx.)
(base) papu@sanjus-MacBook-Air build %
(base) papu@sanjus-MacBook-Air build % cmake --version
cmake version 4.0.3

CMake suite maintained and supported by Kitware (kitware.com/cmake).
(base) papu@sanjus-MacBook-Air build %
(base) papu@sanjus-MacBook-Air build % clang++ --version
Apple clang version 16.0.0 (clang-1600.0.26.6)
Target: arm64-apple-darwin24.3.0
Thread model: posix
InstalledDir: /Library/Developer/CommandLineTools/usr/bin
(base) papu@sanjus-MacBook-Air build %
(base) papu@sanjus-MacBook-Air build % brew --version
Homebrew 4.5.7
(base) papu@sanjus-MacBook-Air build %
(base) papu@sanjus-MacBook-Air build % ls -lh ./bin/

total 3056
-rwxr-xr-x  1 papu  staff   1.5M 21 Jun 15:50 sealexamples
(base) papu@sanjus-MacBook-Air build %
(base) papu@sanjus-MacBook-Air build %
(base) papu@sanjus-MacBook-Air build %
```

Figure 2: Environment Setup and Verification Outputs (Directory, Build, and Version Checks)

This screenshot confirms that the **CKKS Average Salary Demo** was successfully **built** and **executed** on a macOS system using **Microsoft SEAL v4.1.2**. It shows encrypted computation output, version checks for **CMake**, **Clang**, and **Homebrew**, and the presence of the compiled binary file (**sealexamples**)—verifying a fully functional environment setup.

10. Future Work / Work Planned

The remainder of this project will focus on extending the initial work into more advanced analytics and performance evaluation, while addressing the research objectives defined earlier. The following plan outlines the key tasks and deliverables for the next phase.

10.1. Planned Project Timeline

A structured project timeline has been developed to ensure that the remaining activities progress in an organized manner. This timeline, presented as a Gantt chart in **Figure 3**, maps each task against the available project weeks to support effective time management.

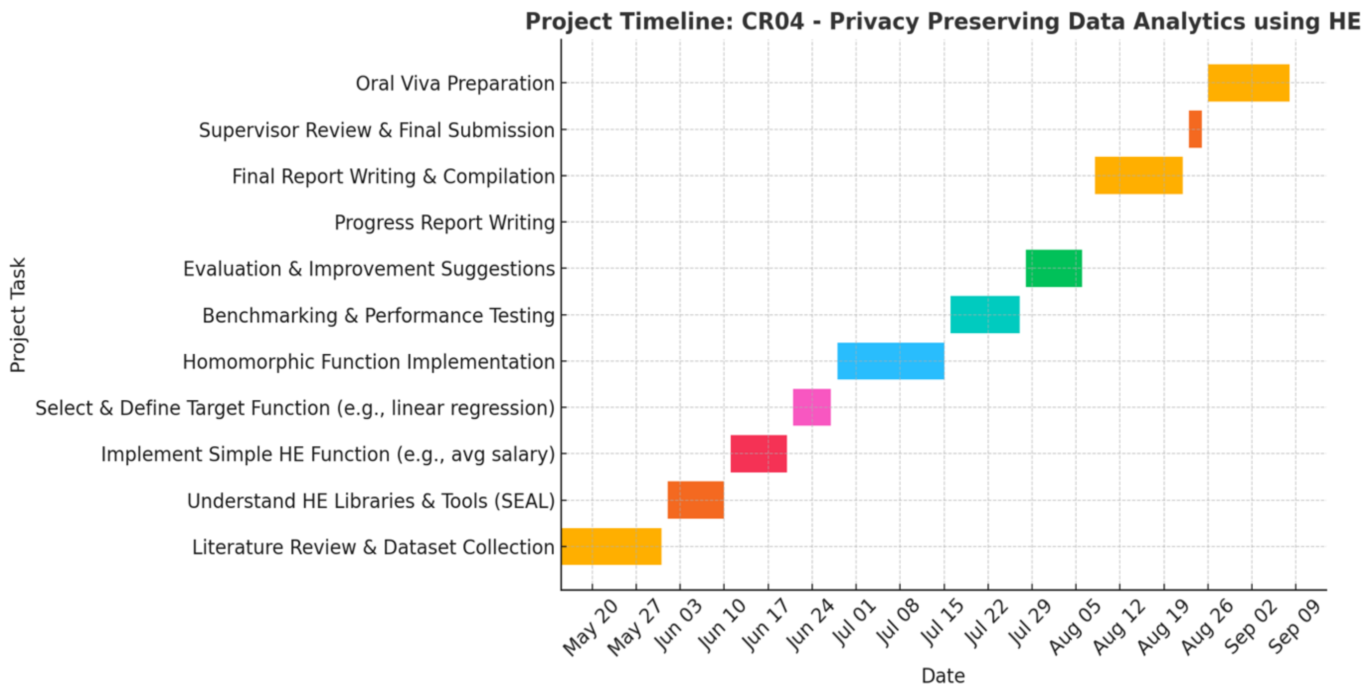


Figure 3: Project plan

10.2 Planned Tasks and Justification

The next phase will build on the current average salary demo to explore more complex analytics and address the research gap highlighted in the literature review.

Table 1: Planned Tasks and Their Justification for the Next Project Phase

Task	**Justification**
Extension of the Demo (Larger Datasets)	To evaluate how HE scales with increased data volume, moving beyond hardcoded inputs to more **realistic** dataset sizes.
Implementation of Encrypted Regression	To demonstrate the feasibility of HE for machine learning tasks, specifically addressing research objective related to advanced analytics.
Performance Benchmarking	To systematically measure the computational cost of HE under different parameter settings, providing evidence-based performance insights.
Exploring Real-World Use Cases (Healthcare/Finance)	To contextualize the project within privacy-sensitive industries, demonstrating practical relevance and application potential.
Evaluation of Cybersecurity Impact	To explicitly link HE's role to data confidentiality and privacy protection within cybersecurity frameworks.
Ethical and Regulatory Review	To address **GDPR** and **ethical considerations** as outlined in earlier sections.
Final Report Preparation	To compile and document all findings, code, results, and analysis in a formal dissertation format.

Note:

"In the next stage of the project, **real-world datasets** from areas like **finance** or **healthcare,** etc, will be used. This will help make the testing more practical and closer to how the system might work in real situations, as suggested in the supervisor's feedback."

Adapting to New Findings

"If any new ideas, methods, or techniques are found later in the project—especially from **IEEE papers** or other **trusted research sources**—they may be added to the project if they are useful. Any such updates will be **reviewed** to make sure they match the **main goals** and stay within the scope of the work. This approach will help improve the overall quality and practical value of using **Homomorphic Encryption for secure data analytics**."

10.3 Planned Benchmarking Experiments

As part of the performance **evaluation**, **benchmarking** experiments will be conducted using various **homomorphic encryption parameters**. The goal is to analyze the impact of each parameter on **runtime**, **memory consumption**, and **output accuracy**.

The operations selected for benchmarking—**summation**, **multiplication**, **rotation**, and **scaling factor adjustment**—were chosen based on their critical role in most statistical and machine learning workloads using HE. These operations are also directly relevant to **CKKS** functionality as demonstrated in Microsoft SEAL examples and reported in comparative studies such as **Wang et al. (2024).**

These experiments will be conducted using a **CSV or some other dataset** with realistic values (e.g., **salaries, health records**), instead of **hardcoded** values. The goal is to reflect how encrypted analytics performs on structured input files in real applications.

Table 2: Planned Benchmarking Experiments

Experiment Parameter	**Planned Values / Variations**	**Performance Metric to Measure**
poly_modulus_degree	4096, 8192, 16384	Encryption time, Evaluation time, Memory usage
coeff_modulus bit size	Varying bit-lengths per SEAL recommendations	Noise budget, Execution time
Dataset size	Small (3 values), Medium (10–20 values), Large (50+ values)	Execution time, Memory usage
Operation type	Summation, Multiplication, Rotation	Cycle count, Execution time
Scaling factor	Multiple scale settings	Accuracy loss, Output precision

These parameters were selected to provide a representative performance profile across both computation and memory usage dimensions. The values **4096, 8192, and 16384** are commonly used **poly_modulus_degree** settings in CKKS. They help balance **security, speed, and accuracy**. Findings from this benchmarking will inform the project's final evaluation and contribute to the broader understanding of HE performance **trade-offs** in privacy-preserving analytics.

10.4 AI Tool Usage Disclaimer (For Transparency)

This report was drafted by the student with occasional use of AI-based grammar and structure suggestions, specifically for tone formalization and academic language refinement. No content was auto generated for technical work descriptions or result reporting. Any AI prompt usage has been logged and included in the **Appendix** section, by university guidelines.

11. Ethical and Privacy Considerations (Optional - Personal Tone)

Privacy is essential in data analysis, especially when working with sensitive information in fields like **healthcare**, **finance**, or **cybersecurity**. Improper handling of such data can lead to serious **ethical**, **legal**, and **reputational** risks. Laws like the **General Data Protection Regulation (GDPR)** set clear rules for protecting personal data.

Homomorphic Encryption (HE) addresses these concerns by allowing encrypted data to be used in analysis without revealing its contents. This aligns with privacy-by-design principles and reduces the chance of data exposure during processing.

This project follows ethical research practices by using secure test datasets and ensuring that **no personal information** is exposed. In future phases, when working with more **realistic datasets**, the balance between **privacy** and **performance** will be carefully considered to ensure the work remains both practical and responsible.

12. Conclusion

The progress achieved during this phase of the project has established a strong foundation for the development of privacy-preserving data analytics using Homomorphic Encryption (HE). Key objectives such as understanding HE, setting up Microsoft SEAL, and completing a basic encrypted **demo** have been achieved

The encrypted demo successfully showed how basic statistical tasks, like **average salary** computation, can be performed securely using HE. Initial performance metrics have been collected, providing valuable insights into **execution** time, **memory** usage, and **computational** overhead associated with HE. Initial performance metrics confirmed the practical feasibility of HE for small-scale analytics tasks. The challenges encountered, including installation issues and HE parameter tuning, have provided important learning opportunities that will inform future project phases.

The next stage of the project will build upon this progress by focusing on more advanced **analytics**, such as encrypted **regression**, extended **benchmarking**, and r**eal-world dataset integration**. These upcoming activities will further address the remaining research **objectives**, contributing both **technical** outcomes and **privacy**-focused insights to the field of **data analytics**.

13. References

[1] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Advances in Cryptology – ASIACRYPT 2017*, Cham: Springer, 2017, pp. 409–437.

[2] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, "A survey on homomorphic encryption schemes: Theory and implementation," *ACM Computing Surveys*, vol. 51, no. 4, pp. 1–35, Jul. 2018.

[3] C. Gentry, S. Halevi, and N. P. Smart, "Efficient bootstrapping for approximate homomorphic encryption," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 1215–1227, 2022.

[4] X. Chen, Y. Wang, and L. Zhao, "Optimizing homomorphic encryption for machine learning on edge devices," *Journal of Cryptographic Engineering*, vol. 13, no. 2, pp. 105–120, 2023.

[5] J. Wang, H. Li, and M. Zhou, "A comparative study of open-source homomorphic encryption libraries: Performance and usability perspectives," *ACM Computing Surveys*, vol. 56, no. 1, pp. 1–25, Jan. 2024.

[6] Microsoft Research, "Microsoft SEAL (release 4.1.2)," [Online]. Available: https://github.com/microsoft/SEAL. [Accessed: Jun. 26, 2025].

[7] National Cyber Security Centre (NCSC), "Advanced cryptography – NCSC whitepaper," 2025. [Online]. Available: https://www.ncsc.gov.uk/whitepaper/advanced-cryptography. [Accessed: Jun. 26, 2025].

[8] The Royal Society, "Privacy enhancing technologies: State of the art," 2025. [Online]. Available: https://royalsociety.org/news-resources/projects/privacy-enhancing-technologies/. [Accessed: Jun. 26, 2025].

[9] Zama, "Introduction to homomorphic encryption," [Online]. Available: https://www.zama.ai/introduction-to-homomorphic-encryption. [Accessed: Jun. 26, 2025].

[10] OpenFHE, "OpenFHE library overview," [Online]. Available: https://www.openfhe.org/. [Accessed: Jun. 26, 2025].

[11] International Association for Cryptologic Research (IACR), "IACR ePrint Archive," [Online]. Available: https://eprint.iacr.org/. [Accessed: Jun. 26, 2025].

[12] IEEE Xplore, "IEEE Xplore Digital Library," [Online]. Available: https://ieeexplore.ieee.org/Xplore/home.jsp. [Accessed: Jun. 26, 2025].

[13] YouTube, "Homomorphic encryption with Microsoft open-source SEAL library," [Online]. Available: https://www.youtube.com/watch?v=280_1798Ej4&list=PLIWDHtAzf8ccODCrS1F1jQ4az840QDpB1. [Accessed: Jun. 26, 2025].