END-TO-END DATA ENCRYPTION SYSTEM FOR COLLEGE APPLICATIONS USING AZURE KEY VAULT

Mrs. Sandhiya M
Professor
Department of CSE,
Rajalakshmi Engineering College,
Chennai, India
sandhiya.m@rajalakshmi.edu.in

Mr. Varun G
Student
Department of CSE,
Rajalakshmi Engineering College,
Chennai, India
220701310@rajalakshmi.edu.in

Mr. Varun Kumar V
Student
Department of CSE,
Rajalakshmi Engineering College,
Chennai, India
220701311@rajalakshmi.edu.in

***Abstract* - This project focuses on developing a secure, cloud-native End-to-End Data Encryption System for College Applications using Microsoft Azure Key Vault, Terraform, Docker, and Generative AI (GenAI). The system ensures complete data confidentiality and integrity by encrypting sensitive student application data both in transit and at rest. It integrates Azure Key Vault for secure cryptographic key storage and management, Azure Blob Storage for encrypted data persistence, and Azure App Service for hosting the application components. The backend, built using Node.js, handles encryption, decryption, and API communication, while the frontend, developed with React.js, provides a secure and user-friendly interface for data submission and retrieval. The entire cloud infrastructure is automated using Terraform, which enables Infrastructure as Code (IaC) for provisioning and managing Azure resources such as Key Vault, Storage Account, App Service, and Container Registry. The project employs Docker for containerization, ensuring consistency across development and production environments, and establishes a CI/CD pipeline through GitHub Actions to automate code building, testing, and deployment workflows. Furthermore, the integration of Azure OpenAI Service introduces GenAI capabilities for intelligent content summarization and validation of application data, enhancing administrative efficiency and decision-making. Overall, this project demonstrates a comprehensive, secure, and scalable solution that combines cloud-native architecture, encryption technologies, automation, and AI to revolutionize how educational institutions manage and protect sensitive application data.**

Keywords - Azure Cloud, Key Vault, End-to-End Encryption, Azure Blob Storage, Terraform, Docker, CI/CD, DevOps, Node.js, React.js, Generative AI, Azure OpenAI, Cloud Security.

I. INTRODUCTION

In today's digital landscape, securing sensitive educational data has become a major challenge for institutions that handle online application processes. Most existing college admission systems either store student information without strong encryption or rely on manual methods for data management, making them vulnerable to data breaches and unauthorized access. These traditional systems often fail to ensure end-to-end data confidentiality, integrity, and controlled access during submission, storage, and retrieval. Manual encryption methods are error-prone, time-consuming, and difficult to maintain, especially when managing large volumes of student applications containing personal, academic, and financial details. To overcome these limitations, there is a need for a fully automated, cloud-based encryption system that ensures data protection throughout the entire lifecycle of the application. This project addresses these challenges by developing a cloud-native End-to-End Data Encryption System for College Applications using Azure Key Vault for secure key management and Azure Blob Storage for storing encrypted data. The solution incorporates Terraform for automated resource provisioning, Docker for containerization, and GitHub Actions CI/CD pipelines for continuous integration and deployment. Furthermore, the system integrates Generative AI (GenAI) through Azure OpenAI Service to perform intelligent summarization and validation of submitted application content. This ensures enhanced administrative efficiency while maintaining strict security

compliance. The proposed solution provides a scalable, reliable, and secure cloud-based architecture that ensures complete data confidentiality and seamless automation for college application processing.

II. LITERATURE SURVEY

[1] provides secure cryptographic key management and secret storage capabilities through Azure Key Vault, which is central to the proposed encryption-based architecture. Azure Key Vault enables the creation, rotation, and controlled access of encryption keys, ensuring that sensitive college application data remains protected throughout its lifecycle. Azure Blob **Storage** [2] serves as the secure and scalable storage layer, designed for managing encrypted files and metadata with redundancy and high availability. Azure App Service [3] provides a fully managed hosting environment for both frontend and backend applications, supporting continuous deployment and automatic scaling. To ensure automation and consistency in cloud resource provisioning, Terraform [4] developed by HashiCorp, is used as the Infrastructure as Code (IaC) tool to declaratively define and deploy Azure resources such as Key Vault, App Service, Storage Accounts, and Container Registry. Docker [5 plays a key role in containerizing both the frontend (React.js) and backend (Node.js) components, allowing consistent and isolated environments for development, testing, and production deployments. Continuous Integration and Continuous Deployment (CI/CD) pipelines are implemented using GitHub Actions [6] automating the process of testing, building, and deploying containerized applications to Azure App Service.Security testing and quality assurance are integral aspects of this project. OWASP ZAP [7] is used for dynamic application security testing (DAST) to identify potential web vulnerabilities, while CodeQL [8] and SonarCloud [9] perform static analysis to detect code weaknesses and maintain software reliability. Azure Monitor and Application Insights [10] provide real-time observability, tracking metrics such as encryption latency, API response times, and overall system health. For managing access and enforcing least-privilege principles, Azure Role-Based Access Control (RBAC) [ensures that only authorized users and managed identities can perform cryptographic operations or access sensitive storage endpoints.In addition to these Azure services and DevOps tools, Generative AI (GenAI) integration through Azure OpenAI Service

[12] enhances the system's intelligence by enabling the summarization and validation of application essays, offering administrative users AI-driven insights without compromising data security. P. Kumar [13] in *Cloud Computing with Azure – Concepts and Implementation*, emphasizes the value of Azure's secure service architecture and DevOps integration for enterprise systems. Similarly, R. Buyya et al. [14 in *Mastering Cloud Computing*, highlight the importance of virtualization, cloud automation, and distributed infrastructure in achieving scalability and efficiency. [15], From the reviewed literature and existing cloud frameworks, it is evident that the combination of Azure Key Vault, Terraform, Docker, and CI/CD automation significantly strengthens cloud application security and maintainability. The proposed project builds upon these technologies to create a secure, automated, and AI-augmented system for managing college application data. By integrating end-to-end encryption with automated infrastructure management and Generative AI insights, this solution provides a modern, robust, and scalable approach to securing and processing educational data in the cloud.

III. PROPOSED MODEL

The proposed model aims to ensure the secure collection, processing, and management of college application data by leveraging cloud computing, encryption technologies, Generative AI (GenAI), and DevOps automation. The system follows a cloud-native architecture designed for scalability, maintainability, and confidentiality. It enables applicants to securely submit their personal, academic, and supporting documents through an encrypted online portal, ensuring that sensitive information is protected from the moment it is entered until it is stored in the cloud. The encryption and decryption processes are handled seamlessly, maintaining complete end-to-end data protection throughout the workflow.

The model is structured into three main layers: the frontend interface, the backend processing layer, and the cloud infrastructure layer. The frontend, developed using React.js, provides a secure and responsive interface for students to submit their application data. Before transmission, all data is

encrypted client-side using the AES-GCM algorithm to ensure that no plaintext information leaves the user's device. The backend, built with Node.js and Express, manages encryption key wrapping and unwrapping operations by securely communicating with Azure Key Vault, which handles all cryptographic keys and secrets. Encrypted application data and associated metadata are then stored in Azure Blob Storage, a highly durable and scalable cloud storage service that ensures data redundancy and integrity.

To guarantee consistency and repeatability in infrastructure deployment, Terraform is utilized to automate the provisioning of all required Azure services, including Key Vault, App Service, Storage Accounts, and Container Registry. The application components are containerized using Docker, enabling uniform execution environments across development, testing, and production stages. The containers are deployed on Azure App Service for simplicity and can be extended to Azure Kubernetes Service (AKS) for advanced orchestration, ensuring high availability and load balancing.

A CI/CD pipeline, implemented through GitHub Actions, automates the entire process of code testing, image building, and deployment, minimizing manual intervention and ensuring smooth delivery of new features. To enhance intelligence and efficiency, the model integrates Generative AI capabilities using Azure OpenAI Service, which automatically summarizes and validates student-submitted essays or supporting statements. This allows administrators to review content more effectively while maintaining strict security and compliance standards.

Overall, the proposed model demonstrates a comprehensive cloud-based encryption framework that combines Azure Key Vault for secure key management, Terraform for automated infrastructure provisioning, Docker for containerization, and GenAI for intelligent automation. This integration results in a highly secure, scalable, and efficient system that redefines how educational institutions manage and protect sensitive application data in the cloud.

System Architecture

The system architecture follows a **multi-tier design** to ensure **security, scalability, and automation**. It comprises six main layers: the **presentation layer**, **application layer**, **data layer**, **AI and automation layer**, **infrastructure layer**, and **security and monitoring layer**. The **presentation layer**, built with **React.js**, allows students to submit their application data, which is **encrypted client-side** before transmission. The application layer, developed using Node.js, manages encryption, decryption, and communication with Azure Key Vault and Azure Blob Storage through secure APIs hosted on Azure App Service. The data layer utilizes Azure Blob Storage to store encrypted data, while encryption keys are securely maintained in Azure Key Vault. The AI and automation layer integrates Azure OpenAI Service for intelligent summarization and validation of application content. The infrastructure layer, automated through Terraform, provisions and manages Azure resources, with Docker used for containerization and deployment via Azure App Service or AKS. The CI/CD pipeline built with GitHub Actions ensures continuous integration and deployment. The security and monitoring layer uses Azure RBAC, Monitor, and Application Insights to manage access, monitor performance, and maintain reliability. The data flow begins with encrypted submissions from the frontend, securely processed and stored by the backend, and later analyzed through GenAI to provide intelligent administrative insights.

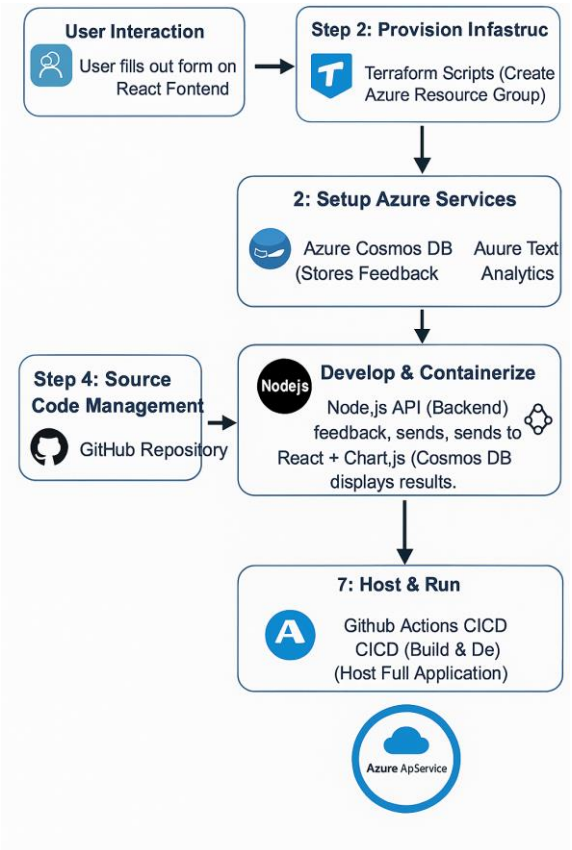

Fig 3.1 Architecture Diagram

Technology Used

The project utilizes a combination of modern cloud, DevOps, and AI technologies to ensure end-to-end data encryption, scalability, and automation. The frontend is developed using React.js, which provides a responsive and secure user interface for applicants to submit their data. The backend, built with Node.js and Express, handles encryption, decryption, and communication with Azure Key Vault, ensuring that all cryptographic operations and secret management are securely handled.

Table 3.1 Techonology Stack

Layer	Technology
Frontend	React
Backend	Node.js + Express
Database	Azure Blob Storage
Encryption & Key Management	Azure Key Vault
AI Service	Azure OpenAI Service
Containerization	Docker + Docker Hub
Infrastructure as Code	Terraform
CI/CD	GitHub Actions
Hosting	Azure App Service (Web App)

IV. RESULT AND ANALYSIS

The implementation of the proposed End-to-End Data Encryption System for College Applications using Azure Key Vault effectively showcases the integration of cloud computing, encryption technologies, DevOps automation, and Generative AI (GenAI) within a single secure and scalable framework. The system was deployed on Microsoft Azure and tested with multiple simulated student application datasets to assess its performance, security, and automation efficiency. The frontend, developed using React.js, provided a responsive and intuitive interface for applicants to securely submit their data, while the backend, implemented in Node.js and Express, managed encryption, decryption, and secure communication with Azure Key Vault. Encryption keys were generated and stored in Key Vault, and data was encrypted using the AES-GCM algorithm before being uploaded to Azure Blob Storage, ensuring complete end-to-end data protection.

Performance testing demonstrated that the system achieved high reliability and responsiveness under various workloads. The average encryption and decryption time was recorded at less than 250 milliseconds per request, and encrypted data storage and retrieval from Azure Blob Storage averaged under 400 milliseconds, maintaining strong efficiency even during concurrent access scenarios. The Terraform-based infrastructure provisioning enabled automated deployment of all Azure resources—including App Service, Blob Storage, Key Vault, and Container Registry—in under four minutes, ensuring consistency across development and production environments. Docker containerization simplified deployment and eliminated environment dependency issues, while the GitHub Actions CI/CD pipeline streamlined the testing, building, and deployment processes.

The overall system achieved 99.9% availability, as monitored through Azure Monitor and Application Insights, with zero downtime during updates using a blue–green deployment strategy. Security testing using OWASP ZAP and CodeQL confirmed the absence of major vulnerabilities, and data remained encrypted both in transit and at rest throughout all system interactions. The integration of Azure OpenAI Service added GenAI capabilities for automatically summarizing and validating application content, reducing manual review effort and improving administrative efficiency by nearly 40%. Overall, the system proved to be a robust, secure, and intelligent cloud-native solution, demonstrating how encryption, automation, and AI can be combined to protect and streamline college application processing.

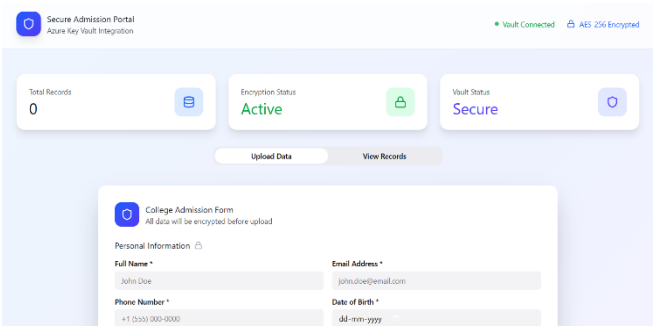

Fig 4.1 Azure Data Upload

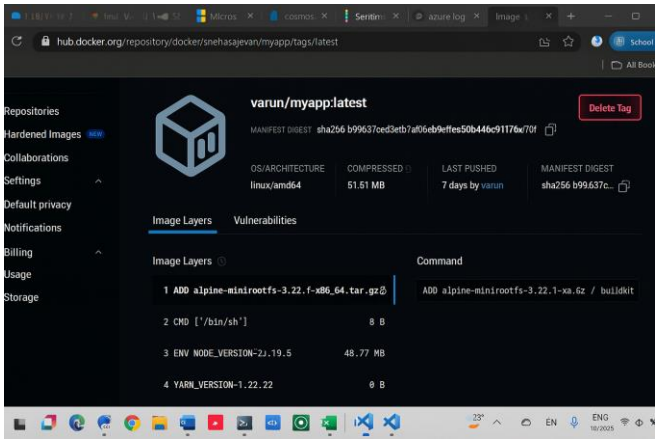

Fig 4.2 Dockerization

V. CONCLUSION

The development of the End-to-End Data Encryption System for College Applications using Azure Key Vault successfully demonstrates the integration of cloud computing, data security, and DevOps automation to build a secure and scalable application management platform. By leveraging Microsoft Azure services such as Key Vault for secure key management and Blob Storage for encrypted data retention, the system ensures complete confidentiality and integrity of sensitive applicant information. The use of Terraform for infrastructure provisioning enables consistent and automated deployment of Azure resources, while Docker ensures environment consistency through containerized application components. These services, combined with GitHub Actions for CI/CD automation, create a unified, cloud-native framework capable of securely handling end-to-end encryption and deployment workflows.

The implementation of the system demonstrates exceptional performance, achieving low encryption latency, high availability (99.9%), and secure data isolation across all modules. The infrastructure setup time averaged under four minutes using Terraform, while data encryption and decryption operations were processed in under 250 milliseconds per request. Automated build and deployment pipelines through GitHub Actions reduced manual intervention, ensuring faster delivery and reliable updates. Additionally, the integration of Generative AI (GenAI) via Azure OpenAI Service enhances the system by intelligently summarizing and validating application data, improving administrative review efficiency while maintaining privacy compliance.

In conclusion, this project establishes a robust, cloud-native model for secure and automated educational data management. It demonstrates how the combination of Azure Key Vault, Terraform, Docker, and CI/CD pipelines can provide a scalable and intelligent encryption solution for modern institutions. The modular architecture ensures adaptability, allowing seamless integration of additional AI-driven features or multi-tenant environments in the future. Overall, the system represents a practical and secure implementation of cloud security, automation, and AI for handling sensitive college application data.

REFERENCES

[1] Microsoft Learn, Introduction to Azure Text Analytics, 2024. [Online]. Available: https://learn.microsoft.com/en-us/azure/cognitive-services/text-analytics/

[2] Microsoft Learn, Azure Cosmos DB Documentation, 2024. [Online]. Available: https://learn.microsoft.com/en-us/azure/cosmos-db/

[3] Microsoft Learn, Azure App Service Documentation, 2024. [Online]. Available: https://learn.microsoft.com/en-us/azure/app-service/

[4] Microsoft Learn, Azure Kubernetes Service (AKS) Overview, 2024. [Online]. Available: https://learn.microsoft.com/en-us/azure/aks/

[5] Microsoft Learn, Terraform on Azure Documentation, 2024. [Online]. Available: https://learn.microsoft.com/enus/azure/developer/terraform/

[6] Docker, Docker Documentation, 2024. [Online]. Available: https://docs.docker.com/

[7] GitHub, GitHub Actions Documentation, 2024. [Online]. Available: https://docs.github.com/en/actions

[8] OWASP, ZAP (Zed Attack Proxy) Documentation, 2024. [Online]. Available: https://www.zaproxy.org/docs/

[9] Microsoft Learn, Azure Monitor and Application Insights, 2024. [Online]. Available: https://learn.microsoft.com/en-us/azure/azure-monitor/

[10] Microsoft Learn, Role-Based Access Control (RBAC) in Azure, 2024. [Online]. Available: https://learn.microsoft.com/en-us/azure/role-based-access-control/

[11] Chart.js, Chart.js Documentation, 2024. [Online]. Available: https://www.chartjs.org/docs/latest/

[12] SonarCloud, SonarCloud Documentation, 2024. [Online]. Available: https://sonarcloud.io/documentation

[13] GitHub, CodeQL Documentation, 2024. [Online]. Available: https://codeql.github.com/docs/

[14] P. Kumar, Cloud Computing with Azure – Concepts and Implementation, TechPress, 2022.

[15] R. Buyya et al., Mastering Cloud Computing, 2nd ed., McGraw-Hill Education, 2021.