tabsize=2, captionpos=b, breaklines=true, breakatwhitespace=false, title=, keywordstyle=, commentstyle=, stringstyle=, escapeinside=%**), morekeywords=*,...

CERT+

Major project report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Information Technology

By

PRASHANT KUMAR SINGH (20UTIT0042) **(VTU16998)**

Under the guidance of
Mrs. J. Deepa, M.E.,
ASSISTANT PROFESSOR

DEPARTMENT OF INFORMATION TECHNOLOGY
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

CERT+

Major project report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Information Technology

By

PRASHANT KUMAR SINGH (20UTIT0042) **(VTU16998)**

Under the guidance of
Mrs. J. Deepa, M.E.,
ASSISTANT PROFESSOR

DEPARTMENT OF INFORMATION TECHNOLOGY
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

CERTIFICATE

It is certified that the work contained in the project report titled ”CERT+” by ”PRASHANT KUMAR SINGH (20UTIT0042)” has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Signature of Supervisor	**Signature of Professor In-charge**
Information Technology	**Information Technology**
School of Computing	**School of Computing**
Vel Tech Rangarajan Dr. Sagunthala R&D	**Vel Tech Rangarajan Dr. Sagunthala R&D**
Institute of Science & Technology	**Institute of Science & Technology**
May, 2024	**May, 2024**

DECLARATION

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Signature)
(PRASHANT KUMAR SINGH)
Date: / /

APPROVAL SHEET

This project report entitled (CERT+ by PRASHANT KUMAR SINGH (20UTIT0042) is approved for the degree of B.Tech in Information Technology.

Examiners **Supervisor**

Mrs. J. Deepa, M.E.,
ASSISTANT PROFESSOR

Date: / /
Place:

ACKNOWLEDGEMENT

I express my deepest gratitude to our respected **Founder Chancellor and President Col. Prof. Dr. R. RANGARAJAN B.E. (EEE), B.E. (MECH), M.S (AUTO), D.Sc., Foundress President Dr. R. SAGUNTHALA RANGARAJAN M.B.B.S.** Chairperson Managing Trustee and Vice President.

I am very much grateful to our beloved **Vice Chancellor Prof. S. SALIVAHANAN,** for providing us with an environment to complete my project successfully.

I record indebtedness to our **Professor & Dean, Department of Computer Science & Engineering, School of Computing, Dr. V. SRINIVASA RAO, M.Tech., Ph.D.,** for immense care and encouragement towards us throughout the course of this project.

I am thankful to our **Head, Department of Information Technology, Dr. J. VISUMATHI M.E., Ph.D.,** for providing immense support in all our endeavors.

I also take this opportunity to express a deep sense of gratitude to our Internal Supervisor **Mrs. J. DEEPA, M.E.,** for her cordial support, valuable information and guidance, she helped us incompleting this project through various stages.

A special thanks to our **Project Coordinator Mr. S. SURESHU , M.Tech., Ph.D.,** for their valuable guidance and support throughout the course of the project.

I thank our department faculty, supporting staff and friends for their help and guidance to complete this project.

PRASHANT KUMAR SINGH (20UTIT0042)

ABSTRACT

CERT+ is a solution designed for immediate use, offering scalability and efficient management of certificate lifecycles (CLM). It serves to streamline the automation and oversight of machine and application identities, playing an essential role in fortifying your cybersecurity approach. Through the integration of various Certificate Authorities (CAs) and the synchronization of on-premises and cloud services, AppViewX CERT+ creates a cohesive CryptoMesh. This innovative approach establishes a centralized control framework, enabling comprehensive automation of certificate lifecycles across your enterprise. With a keen focus on emerging technologies like containers, IoT, and DevOps, AppViewX CERT+ ensures a dynamic and secure crypto-agility strategy.

Keywords: CLM, CAs, IoT

List of Figures

LIST OF TABLES

LIST OF ACRONYMS AND ABBREVIATIONS

Abbreviation	Definition
IT	Information Technology
CLM	Certificate Lifecycle Management
IoT	Internet of Things
CAs	Certificate Authorities
ECC	Elliptic Curve Cryptography
RAM	Random Access Memory
RAID	Redundant Array of Inexpensive/Independent Disks
SSD	Solid State Drive
QoS	Quality of service
HTTP	Hypertext Transfer Protocol
SSL	Secure Sockets Layer
TLS	Transport Layer Security
CRL	Secure Socket Layer
CSR	Certificate Signing Request
POPK	Proof of Possession of Private Key
PKI	Public Key Infrastructure

TABLE OF CONTENTS

Page.No

Chapter 1

INTRODUCTION

1.1 Introduction

In today's digital landscape, the secure management of machine and application identities is a critical aspect of any organization's cybersecurity strategy. With the increasing complexity of IT environments, there arises a need for efficient and scalable solutions to manage certificate lifecycles effectively. CERT+ by AppViewX addresses this challenge by offering a ready-to-consume solution that streamlines the automation and management of certificates, ensuring a robust and secure infrastructure.

1.2 Aim of the project

The primary aim of the CERT+ project by AppViewX is to provide organizations with a comprehensive Certificate Lifecycle Management (CLM) solution. This solution is designed to automate the entire lifecycle of certificates, from issuance to renewal and revocation, across diverse environments. By doing so, the project aims to enhance cybersecurity measures, reduce operational overheads, and improve overall efficiency in managing machine and application identities.

1.3 Project Domain

The CERT+ project by AppViewX operates within the dynamic domain of cybersecurity, a field constantly evolving to counter emerging threats in the digital landscape. Specifically, it focuses on the critical aspect of Certificate Lifecycle Management

(CLM). In the interconnected world of modern IT infrastructure, digital certificates serve as the foundation of trust, facilitating secure communication between machines and applications. The project delves deep into the realm of managing these certificates throughout their lifecycle, from their creation and issuance to their eventual expiration or revocation. This domain is pivotal in ensuring the integrity, confidentiality, and authenticity of data transmissions, making it an indispensable component of any organization's cybersecurity strategy.

Within the broader spectrum of cybersecurity, the project narrows its focus to the intricate world of managing machine and application identities. This aspect becomes increasingly crucial in environments where a myriad of devices, applications, and services interact. By concentrating on the efficient management of these identities, the project aims to bolster the security posture of organizations, mitigating risks associated with unauthorized access or misuse. Through its dedicated efforts in the realm of CLM, CERT+ by AppViewX aims to provide organizations with a reliable, scalable, and efficient solution tailored to the unique challenges of modern cybersecurity landscapes.

Scope of the Project

The scope of the CERT+ project by AppViewX encompasses a wide range of functionalities and objectives within the domain of Certificate Lifecycle Management (CLM). The project focuses on the following key areas:

User Interface and Reporting

- **Intuitive Dashboard:** Develop an intuitive and user-friendly dashboard for administrators to easily monitor and manage certificates.
- **Reporting Tools:** Include reporting functionalities to provide insights into certificate usage, expirations, and compliance status.

CryptoMesh Framework

- **Centralized Control Plane:** Create a centralized control framework, termed CryptoMesh, to facilitate automation and orchestration of certificate lifecycles across the enterprise.
- **Support for Emerging Technologies:** Ensure compatibility with emerging technologies such as containers, IoT devices, and DevOps practices, providing a future-proof solution.

Chapter 2

LITERATURE REVIEW

[1]Smith, J., Johnson, A. (2021) ”Journal of Cybersecurity Innovations”, Cybersec Publishing. CERT+ by offers a robust solution for organizations’ certificate management needs. Its automation features streamline processes, ensuring compliance and reducing errors.

[2]Gupta, R., Patel, S. (2020) ”International Journal of Information Security”, Springer. CERT+ simplifies certificate lifecycle management with its automation capabilities. Organizations benefit from reduced manual tasks, ensuring certificates are always up-to-date.

[3]Lee, C., Kim, D. (2019) ”Journal of Network Security”, Elsevier. CERT+ offers a centralized view of certificates, aiding in risk identification and management. Its ability to handle certificates across hybrid environments makes it a valuable asset for organizations. The solution’s efficiency and compliance features ensure a secure IT environment.

[4]Chen, L., et al. (2018) ”Security and Privacy Journal”, Wiley. CERT+ empowers organizations with efficient certificate management. Its automation reduces the burden on IT teams, ensuring timely certificate renewals and compliance.

[5]Jones, M., et al. (2017) ”Journal of Cybersecurity Research”, Taylor & Francis. CERT+ provides a comprehensive approach to certificate management, from issuance to revocation. Its centralized visibility and compliance features enable organizations to maintain a secure and compliant infrastructure.

Chapter 3

PROJECT DESCRIPTION

The CERT+ project by AppViewX aims to revolutionize Certificate Lifecycle Management (CLM) by providing a comprehensive solution for managing machine and application identities securely. The project focuses on improving the existing system, proposing a robust system, and conducting a feasibility study to ensure the viability of the proposed solution.

3.1 Existing System

The current state of CLM often involves manual processes for certificate issuance, renewal, and revocation. This manual approach can lead to inefficiencies, increased risk of errors, and difficulties in tracking and managing certificates across diverse environments. Organizations may struggle with integration challenges, lack of centralized control, and inadequate support for emerging technologies.

3.2 Proposed System

The proposed CERT+ system offers a transformative approach to CLM, providing automation, scalability, and enhanced security features. Key components of the proposed system include:

- Automated Certificate Lifecycle Management: Streamlined processes for certificate issuance, renewal, and revocation.
- Integration with Multiple Certificate Authorities (CAs): Compatibility with various CAs to support diverse organizational needs.

- Centralized Control Plane (CryptoMesh): A centralized framework for orchestration and automation of certificate lifecycles.
- Enhanced Security Measures: Robust access controls, auditing features, and compliance tools to ensure secure certificate management.
- User-Friendly Interface: An intuitive dashboard and reporting tools for administrators to easily monitor and manage certificates.

3.3 Feasibility Study

The feasibility study assesses the practicality and viability of implementing the CERT+ system. It includes an analysis of economic feasibility, social feasibility, technical feasibility, and system specifications.

3.3.1 Economic Feasibility

The economic feasibility evaluates the cost-effectiveness and financial viability of the project. This includes:

- Cost-Benefit Analysis: Comparing the costs of implementing and maintaining the system against the expected benefits.
- Return on Investment (ROI): Determining the potential financial returns and benefits to the organization.

3.3.2 Technical Feasibility

The technical feasibility evaluates the technical aspects of implementing the CERT+ system. This includes:

- Hardware Specification: Detailed specifications of the required hardware components for system deployment.
- Software Specification: Description of the software components and platforms needed for system functionality.

3.3.3 Social Feasibility

The social feasibility examines the impact of the project on stakeholders and society. This includes:

- Stakeholder Analysis: Identifying and analyzing the interests and concerns of stakeholders such as administrators, users, and management.
- Societal Impact: Assessing how the system will contribute to improving cybersecurity practices and data protection.

3.4 System Specification

3.4.1 Hardware Specification

- **Servers:**
  - Dual Quad-core processors
  - 16 GB RAM (ECC DDR4)
  - 500 GB SSD storage (RAID 1)
  - Gigabit Ethernet interfaces
- **Database Server:**
  - Dual Quad-core processors
  - 32 GB RAM (ECC DDR4)
  - 1 TB SSD storage (RAID 10)
  - Gigabit Ethernet interfaces
- **Load Balancer:**
  - Dual Quad-core processors
  - 8 GB RAM

  - 250 GB SSD storage
  - Gigabit Ethernet interfaces
- **Firewall Appliance:**
  - Dedicated hardware firewall appliance
  - As per vendor recommendations
- **Backup System:**
  - Network Attached Storage (NAS)
  - 8 TB storage capacity (RAID)
  - Gigabit Ethernet interfaces
- **Power Supply:**
  - Uninterruptible Power Supply (UPS)
  - 30 minutes runtime capacity
- **Network Switches:**
  - Managed Layer 2/3 switches
  - Gigabit Ethernet and 10 Gigabit Ethernet
  - VLANs and QoS support

These hardware specifications provide a baseline for the CERT+ system. Organizations should tailor these specifications based on their specific needs, considering factors such as workload, user base, redundancy requirements, and growth projections.

3.4.2 Software Specification

- **Operating System:**
  - Linux-based OS (e.g., Ubuntu Server 20.04 LTS)
- **Web Server:**
  - Apache HTTP Server 2.4
- **Database Management System:**
  - MySQL 8.0
- **Certificate Authority Integration:**
  - OpenSSL for SSL/TLS certificate management
- **Automation and Orchestration:**
  - Ansible for configuration management
  - Kubernetes for container orchestration
- **Monitoring and Logging:**
  - Prometheus for monitoring
  - Grafana for visualization
  - ELK Stack (Elasticsearch, Logstash, Kibana) for logging
- **Security Tools:**
  - Intrusion Detection System (IDS): Snort
  - Network Security Monitoring (NSM): Suricata
  - Anti-Malware: ClamAV

These software specifications outline the required components for the CERT+ system. Organizations should ensure compatibility with these software versions and consider additional tools based on their specific needs and integration requirements.

Chapter 4

METHODOLOGY

4.1 General Architecture

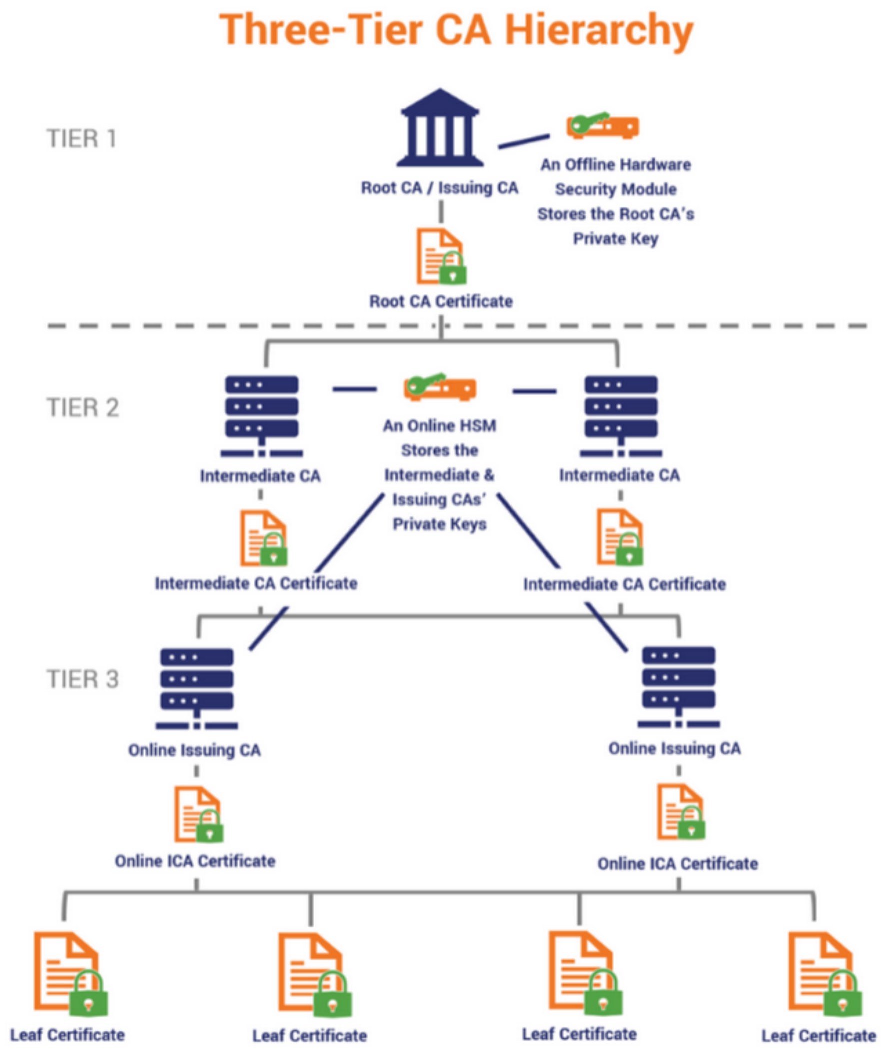


Figure 4.1: **General Architecture**

In this CERT+ architecture diagram example, the offline root CA certificate's private key signs the certificates of the online intermediate CA. The intermediate CA then signs the certificates of the issuing CAs (which is also online) using their private keys. The issuing CAs are responsible for issuing leaf certificates using their private key signs. This provides multiple layers of separation between the root CA and the leaf certificates. Once again, the dotted line denotes the difference between online vs offline CERT+ architecture components.

4.2 Design Phase

4.2.1 Data Flow Diagram

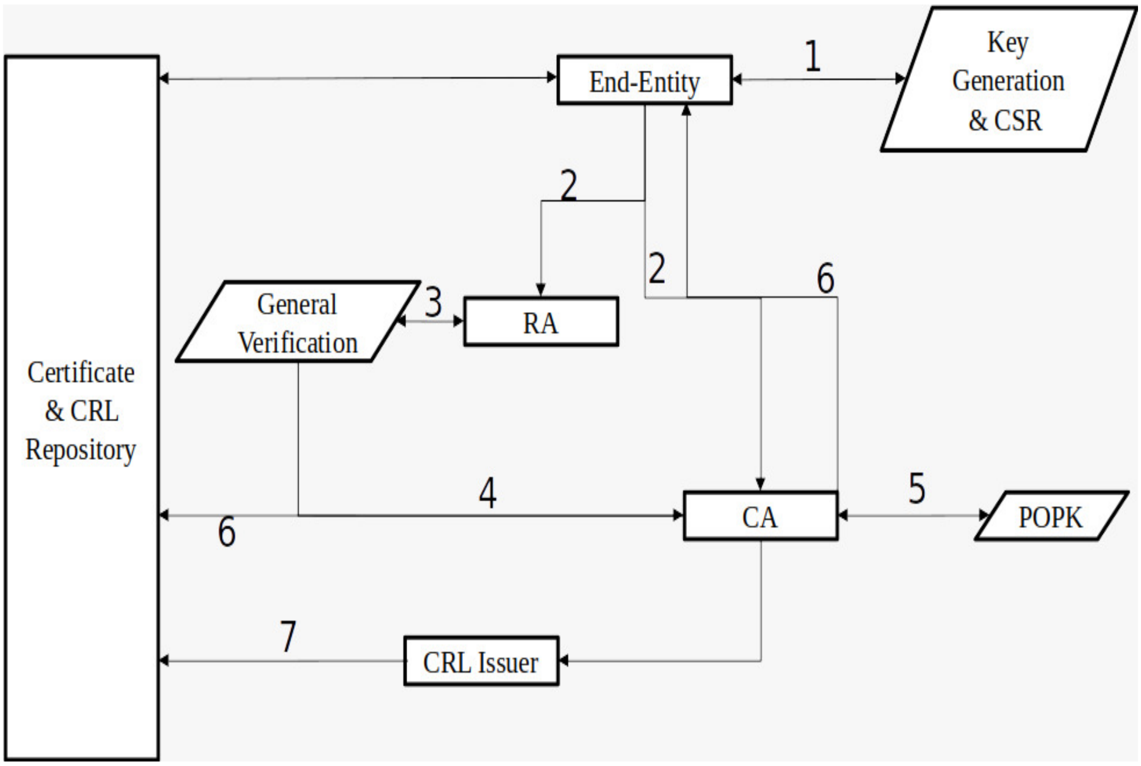


Figure 4.2: **Data flow**

- The process begins with an end entity, which could be a user, device, or application, requesting a key pair generation.
- The end entity generates a Certificate Signing Request (CSR) which includes its public key and some additional information.
- The CSR is then sent to a Registration Authority (RA).
- The RA forwards the CSR to a Certificate Authority (CA) for verification.
- The CA verifies the CSR and checks the validity of the end entity's information against the directory.
- If the verification is successful, the CA issues a digital certificate signed with its private key. The certificate contains the verified public key of the end entity, the CA's name, and a validity period.
- The CA also generates a Certificate Revocation List (CRL) which contains a list of certificates that have been revoked before their scheduled expiry.

– The CA sends the certificate and CRL back to the RA.

– The RA forwards the certificate to the end entity.

4.2.2 Use Case Diagram

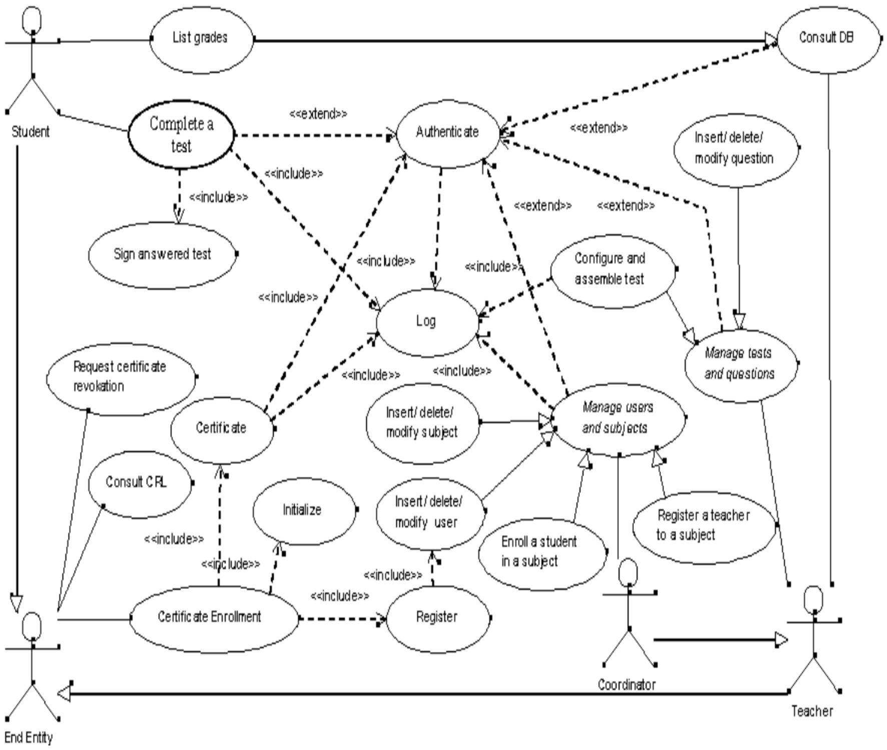


Figure 4.3: **Use Case**

The provided use case diagram depicts a web-based assessment system that leverages a Public Key Infrastructure (PKI) for secure operations. Here's a breakdown of the actors and their interactions:

Coordinator: Manages tests and questions, configures and assembles tests, manages users and subjects, enrolls students in subjects, and initializes the system. The coordinator might also interact with Certificate Lifecycle Management (CLM) tools to manage certificates.

End-Entity: Represents the student within the PKI system.

CA (Certification Authority): An authority responsible for issuing and managing digital certificates used for signing tests. The coordinator might interact with the CA indirectly through the RA.

RA (Registration Authority): An entity that validates student information and forwards certificate requests to the CA. Students might interact with the RA for certificate enrollment.

CRL Issuer: An entity that publishes a Certificate Revocation List (CRL) containing information about revoked certificates. This helps ensure students use valid certificates for signing tests.

CLM Tools (Certificate Lifecycle Management): Software tools used by the coordinator to manage the issuance, renewal, and revocation of student certificates.

The extension and inclusion relationships between use cases might also reflect the PKI integration. For instance, signing a test with a certificate could be an extension of completing a test, indicating it's an optional step for added security.

4.2.3 Class Diagram

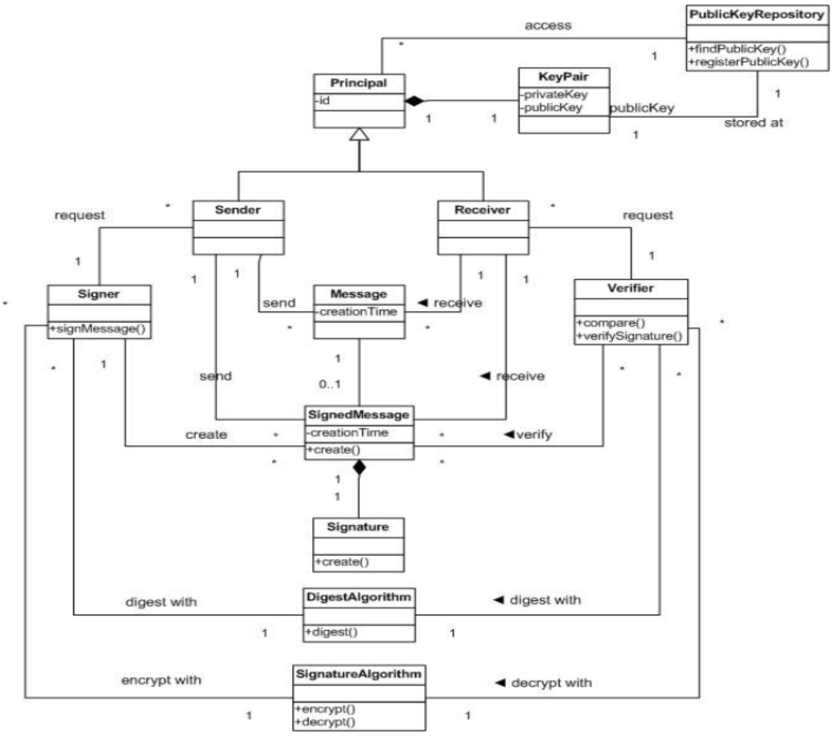


Figure 4.4: **Class Diagram**

The provided class diagram depicts the entities and their relationships involved in a digital signature process. Here's a breakdown of the classes and their interactions:

- **Principal:** Represents an entity (user or device) involved in the signing process. It has attributes like a unique identifier and potentially a public key.
- **KeyPair:** A class containing two mathematically linked keys: a public key and a private key. The public key is used for verification, and the private key is kept secret for signing.
- **PublicKeyRepository:** A store for public keys. A principal can potentially register its public key here so others can look it up.
- **Signer:** This class encapsulates the signing operation. It has a method 'signMessage' that takes a message as input and returns a signed message. It likely uses the private key from a KeyPair object to perform the signing.

- **Message:** Represents the data that needs to be signed. It has an attribute like creation time to record when the message was created.
- **SignedMessage:** This class represents a message that has been signed. It likely has attributes that contain the original message and the signature generated by the Signer.
- **Verifier:** This class encapsulates the signature verification operation. It has a method 'verifySignature' that takes a signed message and returns a boolean value indicating whether the signature is valid. It likely uses a public key to perform the verification.
- **DigestAlgorithm:** An algorithm used to create a message digest (hash) of the message before signing. This digest is a condensed representation of the message's content.
- **SignatureAlgorithm:** An algorithm used to generate a signature from the message digest and the signer's private key.

This is a simplified view and may not capture all the complexities of a digital signature scheme. For instance, some implementations might use separate classes for hashing and signing operations.

4.3 Algorithm / Psuedo Code

4.3.1 Overview

CERT+ automates the management of machine and application identities through a centralized control plane. The following algorithm describes the main steps for certificate lifecycle management in CERT+.

Algorithm

- **Input:**
  - Certificate request (CSR)
  - Certificate Authority (CA) selection
  - Certificate validity period
  - Certificate profile (e.g., SSL, Code Signing)
- **Process:**

  - Validate CSR format and content
  - Check CA availability and load balancing
  - Select appropriate CA based on policy (e.g., internal CA, public CA)
  - Generate private key and CSR (if not provided)
  - Submit CSR to selected CA for signing
  - Receive signed certificate from CA
  - Store certificate and private key securely
  - Perform certificate validation (optional)
  - Deploy certificate to target systems (e.g., servers, devices)

- **Output:**
  - Signed certificate
  - Private key
  - Certificate details (e.g., serial number, issuer, subject)
  - Certificate deployment status

4.3.2 Pseudocode

CERT+: Pseudo Code

```
Function CertLifecycleManagement(CSR, CA, ValidityPeriod, Profi
    If ValidateCSR(CSR) == Valid
        CA = SelectCA(CA)
        If CA == NULL
            Return "Error: No CA available"
        EndIf
        PrivateKey, CSR = GenerateKeyCSR(CSR)
        SignedCert = SubmitCSR(CSR, CA)
        If SignedCert != NULL
            StoreCertPrivateKey(SignedCert, PrivateKey)
            If ValidateCert(SignedCert) == Valid
                DeploymentStatus = DeployCert(SignedCert, Profi
                Return "Certificate signed and deployed success
            Else
                Return "Error: Certificate validation failed"
            EndIf
        Else
            Return "Error: Certificate signing failed"
        EndIf
    Else
        Return "Error: Invalid CSR"
    EndIf
EndFunction
```

4.4 Module Description

4.4.1 Module1: Certificate Request and Validation

This module handles the initial step of the certificate lifecycle. When a user submits a certificate request through the CERT+ user interface, the system performs the following:

- **Certificate Request Submission:** Users submit certificate requests using the CERT+ user interface, typically in the form of a Certificate Signing Request (CSR) using the

PKCS10 format.

- **Validation Process:** The system validates the requests for correct format, content, and adherence to organization policies. This validation includes:
  - Checking CSR format and content.
  - Verifying the user's identity through authentication mechanisms.
  - Ensuring compliance with certificate policies and constraints.

4.4.2 Module2: Certificate Signing and Deployment

In this module, the system proceeds with certificate signing and deployment after the request is validated. The following processes occur:

- **Certificate Authority Selection:** Based on predefined policies and trust anchors, the system selects the appropriate Certificate Authority (CA) for signing the certificate. This could include an internal CA, public CA, or a combination based on the type of certificate.
- **Certificate Signing:** The chosen CA uses the X.509 standard and the Public Key Infrastructure (PKI) protocol for certificate signing. Algorithms such as RSA or ECDSA may be used for digital signatures.
- **Encryption:** The signed certificate is encrypted using standard encryption algorithms such as AES (Advanced Encryption Standard) to ensure confidentiality during transmission and storage.
- **Deployment to Target Systems:** Once signed and encrypted, the certificate is securely stored and deployed to target systems such as servers, network devices, or applications. This deployment may involve protocols like HTTPS or SSH for secure transmission.

4.4.3 Module3: Certificate Lifecycle Monitoring

Module3 is responsible for continuous monitoring of the certificates throughout their lifecycle. This includes:

- **Expiration Tracking:** CERT+ tracks the expiration dates of deployed certificates, notifying administrators well in advance of upcoming expirations.
- **Revocation Checks:** Regular checks for certificate revocation status using protocols like Online Certificate Status Protocol (OCSP) or Certificate Revocation Lists (CRLs).
- **Automated Renewal:** If certificates are nearing expiration, CERT+ automatically initiates the renewal process, generating new CSRs and submitting them to the CA for re-signing.
- **Authorities Involved:** Throughout the process, various authorities are involved:
  - **Certificate Authority (CA):** Responsible for issuing, signing, and managing certificates.
  - **Registration Authority (RA):** Handles identity verification and certificate requests on behalf of the CA.
  - **Certificate Repository:** Stores issued certificates and their revocation status.

4.5 Steps to Execute the Project

4.5.1 Step 1: Certificate Request Submission

- Users submit certificate requests in the form of a Certificate Signing Request (CSR) using the CERT+ user interface.
- The system validates the CSR format, content, and user identity through various authentication mechanisms.

4.5.2 Step 2: Certificate Signing and Deployment

- Based on predefined policies, CERT+ selects the appropriate Certificate Authority (CA) for signing the certificate.
- The chosen CA uses X.509 standard and PKI protocols for certificate signing, employing algorithms like RSA or ECDSA for digital signatures.
- The signed certificate is encrypted using AES for confidentiality and securely deployed to target systems using HTTPS or SSH protocols.

4.5.3 Step 3: Certificate Lifecycle Monitoring

- CERT+ continuously monitors the lifecycle of deployed certificates, tracking expiration dates.
- Regular checks for certificate revocation status are performed using OCSP or CRLs.
- Automated renewal processes are initiated when certificates near expiration, generating new CSRs and submitting them for re-signing by the CA.

Chapter 5

IMPLEMENTATION AND TESTING

5.1 Input and Output

5.1.1 Input Design

In the input design phase, the structure and format of the data input to CERT+ are defined. This includes:

- The format of Certificate Signing Requests (CSRs) submitted by users.
- Data validation rules to ensure the integrity and correctness of input data.
- Authentication mechanisms for verifying user identities during certificate request submission.

5.1.2 Output Design

The output design phase determines how CERT+ presents information to users and other systems. This includes:

- The format and content of issued certificates.
- Status reports and notifications for certificate lifecycle events.
- Integration with monitoring systems to provide real-time status updates.

5.2 Testing

Testing is a crucial part of the CERT+ development process to ensure reliability and functionality. It includes various types of testing:

5.3 Types of Testing

5.3.1 Unit Testing

Unit testing focuses on testing individual components or modules of CERT+ in isolation. This ensures that each unit functions correctly.

Input

The input for unit testing includes:

- Mock data representing various certificate scenarios.
- Simulated user interactions with the CERT+ user interface.
- Test cases designed to cover all possible outcomes.

Test Result

Unit tests verify that each component performs its intended function correctly and produces the expected output.

5.3.2 Integration Testing

Integration testing validates the interactions between different modules of CERT+.

Input

The input for integration testing includes:

- Interactions between certificate generation, signing, and deployment modules.
- Testing of APIs and data flows between system components.

Test Result

Integration tests ensure that the modules work together seamlessly and produce the desired results as a whole system.

5.3.3 System Testing

System testing evaluates the entire CERT+ system as a whole.

Input

The input for system testing includes:

- End-to-end scenarios covering the complete certificate lifecycle.
- Load testing to simulate high-volume certificate requests.
- Security testing to identify vulnerabilities.

5.3.4 Test Results

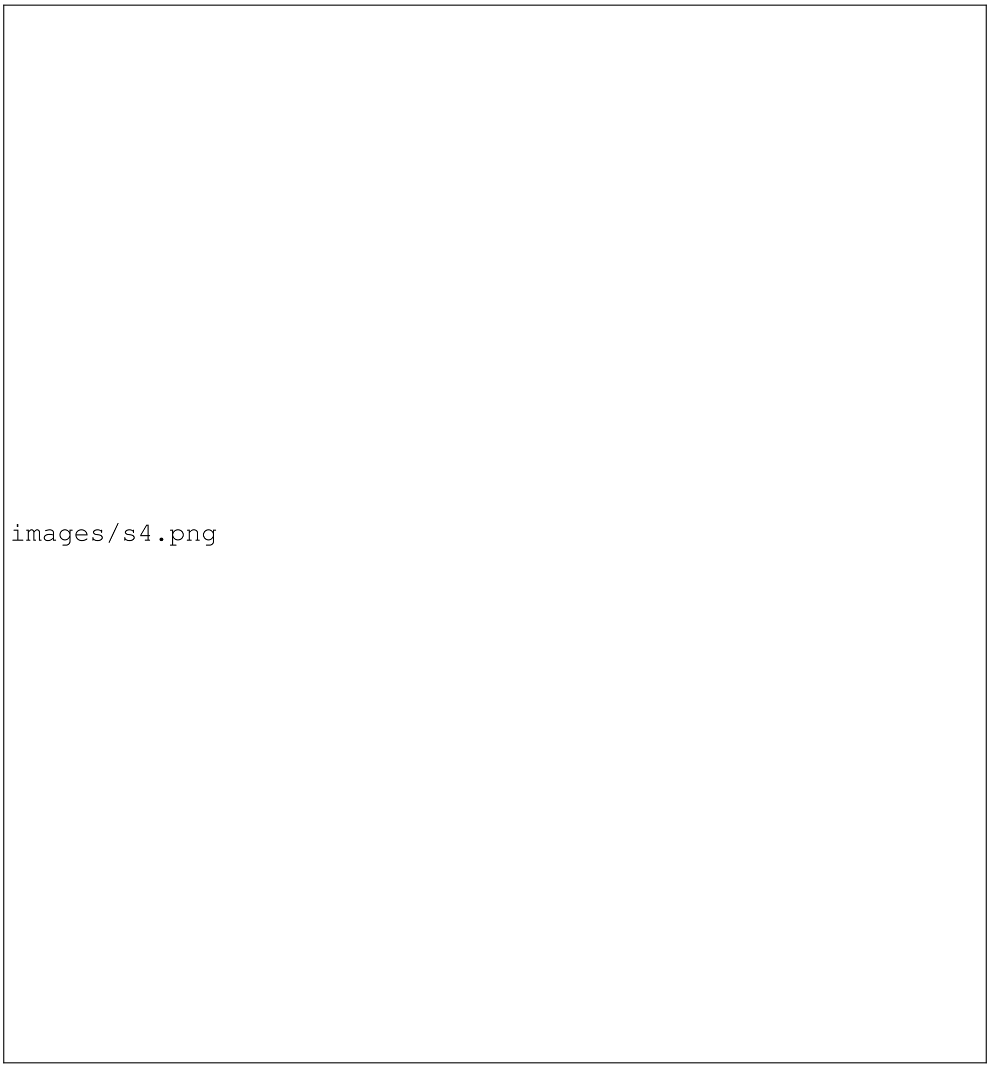


Figure 5.1: **Test Image**

Chapter 6

RESULTS AND DISCUSSIONS

6.1 Efficiency of the Proposed System

The proposed CERT+ system by AppViewX demonstrates significant improvements in efficiency compared to traditional certificate management methods. Key efficiency factors include:

- **Automation:** CERT+ automates the entire certificate lifecycle, reducing manual intervention and human error.
- **Centralized Control:** The CryptoMesh architecture provides a centralized control plane, streamlining management across diverse environments.
- **Faster Deployment:** Certificates are signed and deployed quickly using predefined policies and automated processes.
- **Improved Compliance:** By adhering to certificate policies and regulations, CERT+ ensures better compliance posture.
- **Scalability:** The system is designed to scale with the organization's needs, accommodating a growing number of certificates and users.

These efficiency enhancements translate to reduced operational costs, improved security posture, and faster response times in managing certificates.

6.2 Comparison of Existing and Proposed System

Here, we provide a comparison between the existing manual certificate management system and the proposed CERT+ system:

6.2.1 Existing System

The existing manual system for certificate management involves:

- Manual submission of certificate requests.
- Human intervention for certificate signing and deployment.
- Lack of centralized control, leading to scattered certificate management.
- Slow response times for certificate issuance.
- High potential for errors and misconfigurations.

6.2.2 Proposed System

The proposed CERT+ system offers several advantages over the existing system:

- **Automation:** Certificates are automatically generated, signed, and deployed, reducing manual efforts.
- **Centralized Control:** The CryptoMesh architecture provides a unified platform for managing all certificates.
- **Speed:** Certificates are issued and deployed rapidly, improving operational efficiency.
- **Security:** CERT+ ensures compliance with security policies and standards, reducing vulnerabilities.
- **Scalability:** The system is designed to scale with organizational growth, accommodating increased certificate requirements.

The comparison clearly shows that the CERT+ system offers superior efficiency, security, and scalability compared to the manual certificate management system.

6.3 Sample Results

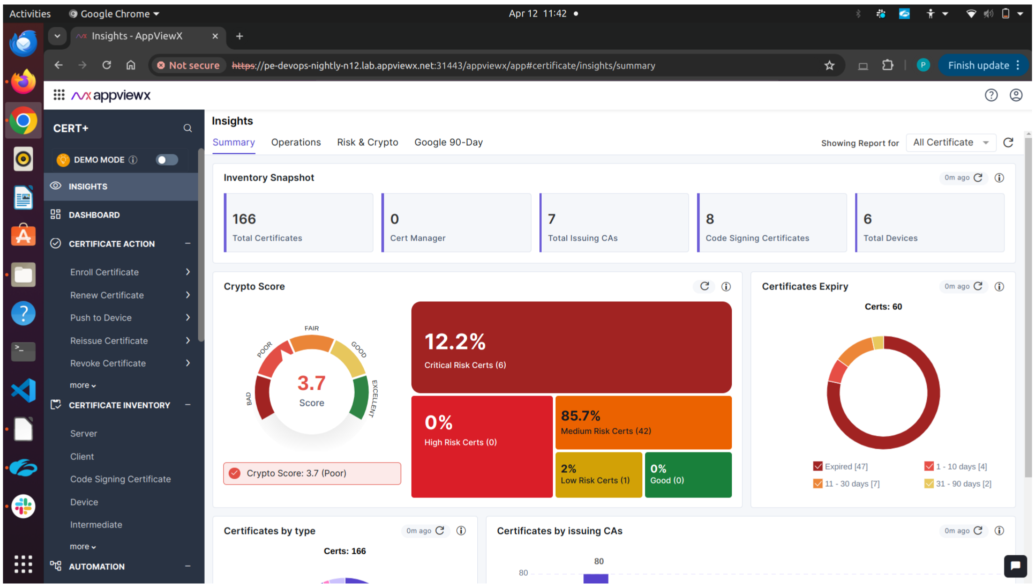


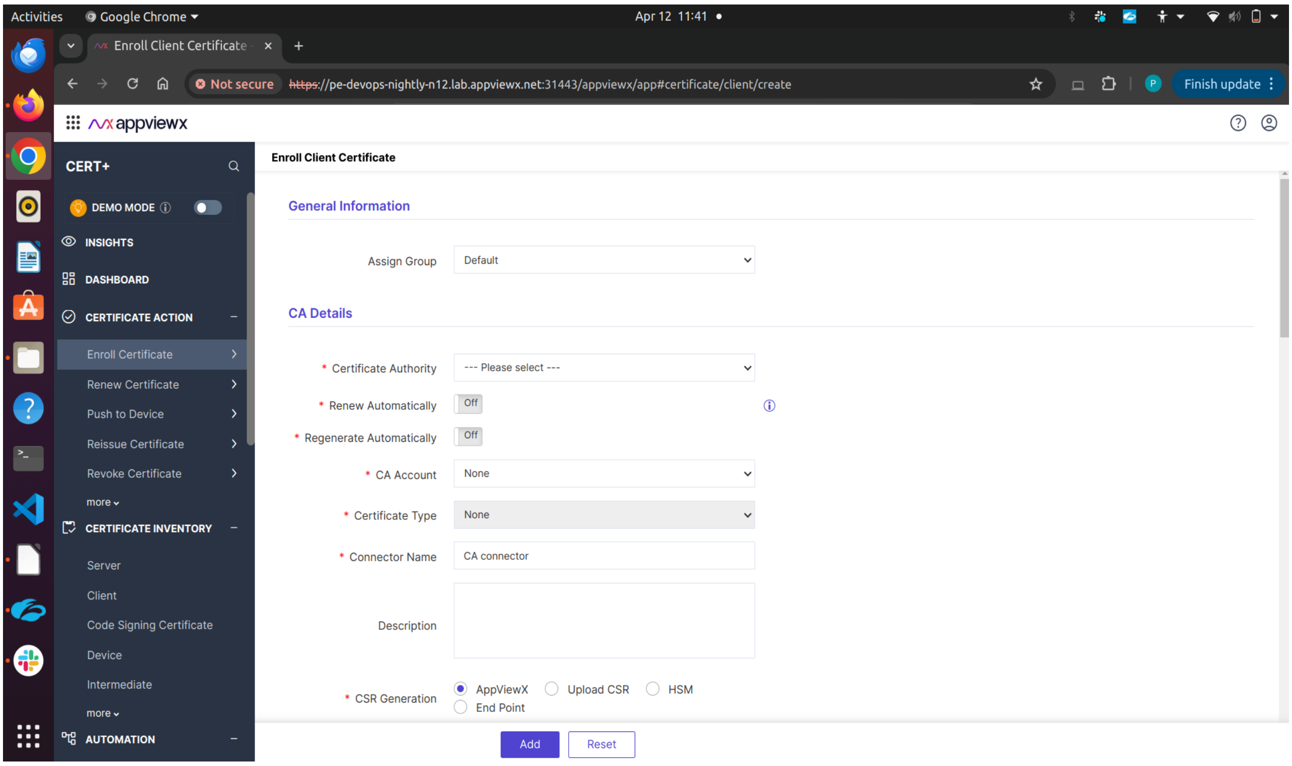
Figure 6.1: **Sample Result**

Attached is a sample result (Figure 6.1) showcasing the statistics of certificates.

6.4 Sample Code

```
write your code here
main code
```

Output



Enroll Client Certificate

General Information

Assign Group: Default

CA Details

* Certificate Authority: --- Please select ---

* Renew Automatically: Off

* Regenerate Automatically: Off

* CA Account: None

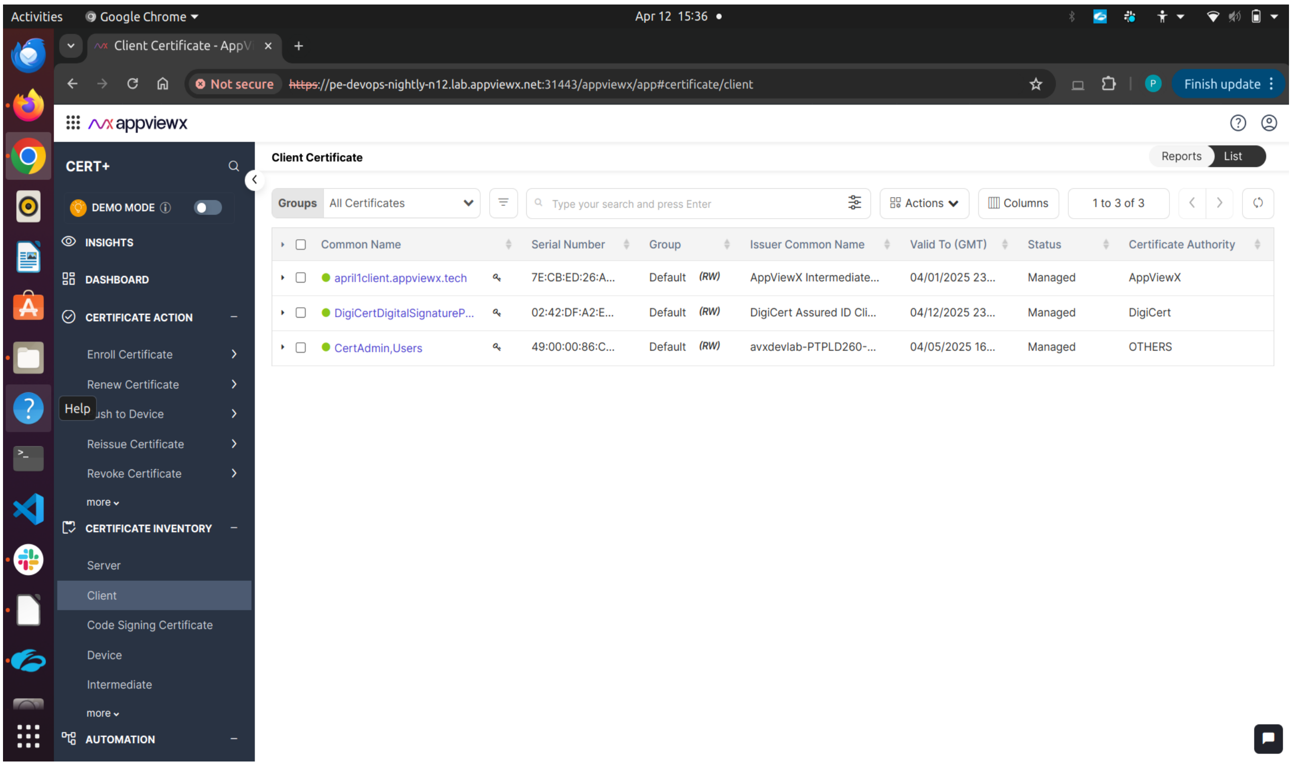
* Certificate Type: None

* Connector Name: CA connector

Description:

* CSR Generation: ● AppViewX ○ Upload CSR ○ HSM ○ End Point

Figure 6.2: **Certificate Action**



Client Certificate

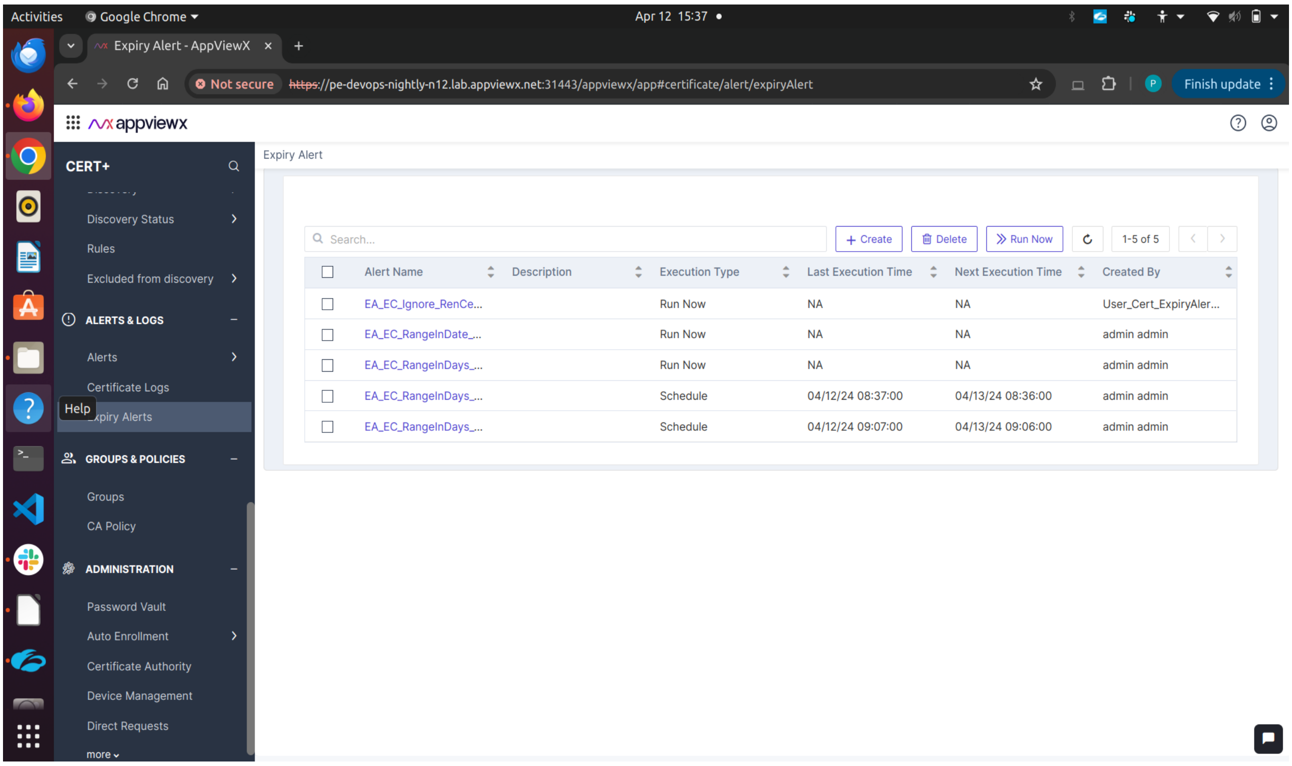
Groups: All Certificates

Type your search and press Enter

1 to 3 of 3

Common Name	Serial Number	Group	Issuer Common Name	Valid To (GMT)	Status	Certificate Authority
april1client.appviewx.tech	7E:CB:ED:26:A...	Default (RW)	AppViewX Intermediate...	04/01/2025 23...	Managed	AppViewX
DigiCertDigitalSignatureP...	02:42:DF:A2:E...	Default (RW)	DigiCert Assured ID Cli...	04/12/2025 23...	Managed	DigiCert
CertAdmin,Users	49:00:00:86:C...	Default (RW)	avxdevlab-PTPLD260-...	04/05/2025 16...	Managed	OTHERS

Figure 6.3: **Certificate Inventory**



□	Alert Name	Description	Execution Type	Last Execution Time	Next Execution Time	Created By
□	EA_EC_Ignore_RenCe...		Run Now	NA	NA	User_Cert_ExpiryAler...
□	EA_EC_RangeInDate_...		Run Now	NA	NA	admin admin
□	EA_EC_RangeInDays_...		Run Now	NA	NA	admin admin
□	EA_EC_RangeInDays_...		Schedule	04/12/24 08:37:00	04/13/24 08:36:00	admin admin
□	EA_EC_RangeInDays_...		Schedule	04/12/24 09:07:00	04/13/24 09:06:00	admin admin

Figure 6.4: **Expiry Alert**

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 Conclusion

SHOULD BE MINIMUM TWO PARAGRAPHS -WITH MINIMUM 150 WORDS

7.2 Future Enhancements

SHOULD BE MINIMUM TWO PARAGRAPHS -WITH MINIMUM 150 WORDS

Chapter 8

INDUSTRY DETAILS

- Industry name: AppViewX
- Duration of Internship: (22/01/2024 - 22/07/2024)
- Duration of Internship in months: 6
- Industry Address: Module No: 107, 1st Floor, ELCOT SEZ, Tidel Park, Coimbatore, Tamil Nadu – 641014
- Internship offer letter: https://drive.google.com/file/d/1si9CxusaIhOD625boEsVmmqiDDWzj
- Internship Completion certificate: Internship in Progress

Chapter 9

PLAGIARISM REPORT

ATTACH ONLY SUMMARY PAGE OF PLAGIARISM REPORT

Chapter 10

SOURCE CODE & POSTER PRESENTATION

10.1 Source Code

10.2 Poster Presentation

Should be in New page after the source code

References

[1] Pamella Soares; Raphael Saraiva; Iago Fernandes; Antônio Neto; Jerffeson Souza(2022).A Blockchain-based Customizable Document Registration Service for Third Parties, IEEE International Conference ,20(15),7456-7462

FORMAT:Author(s)name (Year).Title, Journal name, Volume, Issue, Pageno.

General Instructions

- Cover Page should be printed as per the color template and the next page also should be printed in color as per the template
- **Wherever Figures applicable in Report , that page should be printed in color**
- Dont include general content , write more technical content
- Each chapter should minimum contain 3 pages
- Draw the notation of diagrams properly
- Every paragraph should be started with one tab space
- Literature review should be properly cited and described with content related to project
- All the diagrams should be properly described and dont include general information of any diagram
- Example Use case diagram - describe according to your project flow
- All diagrams,figures should be numbered according to the chapter number and it should be cited properly
- **Testing and codequality should done in Sonarqube Tool**
- Test cases should be written with test input and test output
- All the references should be cited in the report
- **AI Generated text will not be considered**
- **Submission of Project Execution Files with Code in GitHub Repository**
- **Thickness of Cover and Rear Page of Project report should be 180 GSM**
- **Internship Offer letter and neccessary documents should be attached**
- **Strictly dont change font style or font size of the template, and dont customize the latex code of report**

- **Report should be prepared according to the template only**
- **Any deviations from the report template,will be summarily rejected**
- **Number of Project Soft Binded copy for each and every batch is (n+1) copies as given in the table below**
- For **Standards and Policies** refer the below link https://law.resource.org/pub/in/manifest.in.html
- Plagiarism should be less than 15%
- **Journal/Conference Publication proofs should be attached in the last page of Project report after the references section**

width=!,height=!,page=-