**BASAVARAJESWARI GROUP OF INSTITUTIONS**

**Ballari Institute of Technology & Management**

**AUTONOMOUS INSTITUTE UNDER VISVESVARAYA TECHNOLOGICAL UNIVERSITYJNANA SANGAMA, BELAGAVI 590018**

**INTERNSHIP**

**Report On**

**SSL Certificate Manager**

Submitted in partial fulfilment of the requirements for the award of degree of

**Bachelor of Engineering**

**In**

**COMPUTER SCIENCE AND ENGINEERING**

**Submitted by**

**Saraswath H D**

**3BR23CS139**

**Internship Carried Out By**

**EZ TRAININGS & TECHNOLOGIES PVT.LTD**

**HYDERABAD**

**Internal Guide Technical Trainer**

**S.STEFFI NIVEDITA RAGAVAN P**

**External Guide**

**VARADA ALEKHYA**

**BALLARI INSTITUTE OF TECHNOLOGY & MANAGEMENT**

NACC Accredited Institution\*

**(Recognized by Govt. of Karnataka, approved by AICTE, New Delhi & Affiliated to**

**Visvesvaraya Technological University, Belagavi "Jnana Gangotri" Campus, No.873/2, Ballari-Hospete Road,**

**Allipur, Ballar1-583 104 (Karnataka) (India) Ph: 08392 – 237100 / 237190, Fax: 08392 – 237197 2023-2024**

**BASAVARAJESWARI GROUP OF INSTITUTIONS**

**BALLARI INSTITUTE OF TECHNOLOGY & MANAGEMENT**

**Autonomous institute under VISVESVARAYA TECHNOLOGICAL UNIVERSITYJNANA SANGAMA,**

**BELAGAVI 590018**

NACC Accredited Institution\*

**(Recognized by Govt. of Karnataka, approved by AICTE, New Delhi & Affiliated to**

**Visvesvaraya Technological University, Belagavi)**

**"JnanaGangotri"Campus,No.873/2,Ballari-HospetRoad,Allipur,**

**Ballar1-583 104 (Karnataka)(India)**

**Ph: 08392 – 237100 / 237190, Fax: 08392 –237197**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**CERTIFICATE**

This is to certify that the Internship entitled **“ SSL Certificate Manager”** has been

successfully completed by **Saraswath H D** bearing USN **3BR23CS139** a bonafide student of

Ballari Institute of Technology and Management, Ballari. For the partial fulfilment of

the requirements for the **Bachelor’s Degree in Computer** **Science and Engineering**

of the VISVESVARAYA TECHNOLOGICAL UNIVERSITY, Belagavi during the

academic year 2023-2024.

**Signature of Internship Signature of HOD**

**Co-ordinator**

**S.STEFFI NIVEDITA R N KULKARNI**

**Asst.prof,CSE Prof. and HOD(CSE)**

**VARADA ALEKHYA**

**Asst. prof,CSE**

**DECLARATION**

I, **Saraswath H D,** second year student of, Computer Science and Engineering, Ballari Institute of Technology, Ballari, declare that Internship entitled **SSL Certificate Manager** is a part of Internship Training successfully carried out by **EZ TECHNOLOGIES & TRAININGS PVT.LTD ,Hyderabad** at “**BITM,BALLARI”.** This report is submitted in partial fulfilment of the requirements for the award of the degree, Bachelor of Engineering in Computer Science and Engineering of the Visvesvaraya Technological University, Belagavi.

**Date:28.09.2024 Signature of the Student**

**Place: Ballari**

**ACKNOWLEDGEMENT**

The satisfactions that a company the successful completion of my internship on “ SSL Certificate Recorder ” would be incomplete without the mention of people who made it possible, whose noble gesture, affection, guidance, encouragement and support crowned my efforts with success. It is my privilege to express my gratitude and respect to all those who inspired me in the completion of my internship.

I am grateful to our respective coordinator “S.Steffi Nivedita (Asst.prof,CSE) , Varada Alekya (Asst.prof,CSE)” for his noble gesture, support co-ordination and valuable suggestions given to me in the completion of Internship.

I also thank R N Kulkarni, H.O.D. Department of Computer science and engineering for extending all his valuable support and encouragement

* **Table of Contents**

|  |  |
| --- | --- |
| 1 | Company profile |
| 2 | Day to day Activities |
| 3 | Abstract |
| 4 | Introduction |
| 5 | System Requirements |
| 6 | Project description |
| 7 | Objective |
| 8 | System Architecture |
| 9 | Implementation |
| 10 | Key features and skills acquired |
| 11 | Challenges and solution |
| 12 | Result |
| 13 | Conclusion |
| 14 | Recommendations |
| 15 | References |

* **Abstract**

This project report presents my internship experience at EZ technologies where I worked on SSL Certificate Manager. The project aimed to ensure safe transfer of data over the network. Using python, its libraries such as Pandas , NumPy; and development environment, I have written a code for SSL certificate management. I successfully understood about python programming ,web development, data science and automation . Through this internship I gained valuable experience in advance DSA in python and deepened my understanding in implementation python in real life scenarios. The projects outcome have the potential to keep the data secure between the client and server.

­­­­

* **Introduction:**

**SSL (Secure Sockets Layer)** certificates play a vital role in securing online communication by encrypting data transmitted between web servers and clients. With an increasing number of websites and services adopting HTTPS, managing SSL certificates efficiently has become a critical task for organizations to avoid potential security risks. Failure to manage certificates properly can lead to certificate expiration, which may result in insecure connections, service downtime, or even exposure to attacks.

The aim of this project was to design and implement a Python-based SSL certificate manager to streamline and automate the process of generating, renewing, and managing SSL certificates. The SSL certificate manager reduces the manual efforts involved in the lifecycle of certificates and minimizes risks associated with expired or invalid certificates.

This report outlines the development process, tools, and technologies used to create the SSL certificate manager. It also discusses the project's objectives, challenges faced, and the solutions implemented to ensure secure management of SSL certificates.

The report further highlights the importance of SSL certificate management, particularly in environments where multiple servers and certificates need to be handled, ensuring that certificates remain valid and secure across various platforms.

* **System requirements**

Used programming languages, libraries, and tools used in the project are:

* **Languages:** Python 3.x
* **Libraries:**
* **ssl:** For creating secure socket layers and managing SSL contexts.
* cryptography: For generating and handling certificates.
* **requests:** For making HTTP requests to fetch certificate information.
* **OpenSSL:** Used for handling SSL/TLS protocols.
* **SQLite/MySQL:** For storing certificate records.
* **Development Environment:** PyCharm, VS Code, or Jupyter Notebook.
* **Version Control:** Git, GitHub
* **Objectives**
* **Automate SSL Certificate Lifecycle Management**:
* The SSL certificate manager should automate the entire lifecycle of SSL certificates, including generation, validation, renewal, and revocation. This reduces manual intervention and ensures that certificates are managed in a timely and error-free manner.
* **Prevent Certificate Expiration**:
* One of the primary objectives is to prevent SSL certificates from expiring by providing automated reminders and renewal mechanisms. This ensures uninterrupted secure communication between web servers and clients, preventing downtime or security vulnerabilities due to expired certificates.
* **Simplify Certificate Generation and Deployment**:
* Provide a user-friendly interface or script to generate new certificates (self-signed or from a Certificate Authority). The manager should also facilitate easy deployment of certificates to web servers, services, and other endpoints, ensuring seamless integration into existing systems.
* **Enhance Security through Centralized Management**:
* Centralize the management of multiple SSL certificates in a single tool, allowing administrators to monitor the status and validity of all certificates in one place. This reduces the risk of missing expired or misconfigured certificates and strengthens the overall security of the organization.
* **Ensure Compliance with Industry Standards**:
* Ensure that SSL certificates comply with industry standards and best practices to enhance security and maintain compatibility with modern browsers and systems.
* **Monitor and Alert for Certificate Expiration and Issues**:
* Continuously monitor the certificates’ expiration dates and alert administrators ahead of time to avoid expired certificates causing service outages or security breaches. This includes logging and sending notifications to responsible parties when certificates are near expiration or invalid.
* **Project Description**

The objective of the SSL Certificate Recorder is to develop a Python-based tool that monitors SSL certificates, records their details, and alerts users of upcoming expirations. The system improves web service security and ensures compliance with best practices for certificate management.

* **System Architecture Provide**

**Input:** Fetch SSL certificate details from various domains using Python scripts.

**Processing:** Parse certificate data, validate expiration dates, and store details in a database.

**Output:** Generate reports and send alerts for certificates nearing expiration.

* **Description**
* **What is SSL?**

SSL is a protocol for establishing secure links between networked computers.

* **Purpose of SSL**

SSL provides confidentiality, authentication and data integrity in internet communication. SSL is the predecessor to the modern TLS encryption used today.

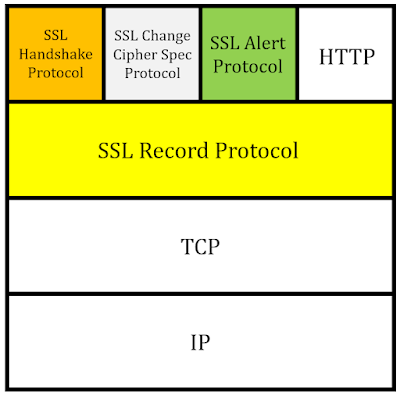


**Figure: Connection using HTTP**

In above figure user interact with server but server uses only normal http, it shows unsecured connection between server and client. Hacker or attacker can capture message from unsecured connection. Username and password are also sent in plain text form. It means hacker or attacker get user’s sensitive information.



**Figure: Connection using HTTPS (use of SSL Certificate)**



* ***SSL Handshake Protocol:*** Connection establishment.
* ***SSL Change Spec Protocol:*** Use of required cipher techniques for data encryption.
* ***SSL Alert Protocol:*** Alert (warning, error if any) generation.
* ***SSL Record Protocol:*** Encrypted data transmission and encapsulation of the data sent by the higher layer protocols. Two important SSL concepts are the SSL Connection and the SSL Session.
* ***SSL Connection:***It is a transport that provides a suitable type of service. Each connection is associated with one SSL session.
* ***SSL Session:***It is a set of cryptographic security parameters which can be shared among multiple SSL connections. An SSL session is an association between a client and a server.
* **Implementation: Flowchart**

**Start**

Initialize Certificate Manager object

Add certificate: cert \_id=1

Add certificate: cert \_id=2

Get certificate: cert \_id=1

Is certificate found?

**( no )**  **(yes)**

Display "Certificate not found" message Display certificate details

Update certificate: cert \_id=1

Renew certificate: cert \_id=2

Track certificate expiries

Are there certificates expiring soon?

**(no) (yes)**

Display "No expiring certs" Display “expiring certificates

Delete certificate: cert \_id=1

**End**

* **Implementation: Program**

from datetime import datetime, timedelta

# Define a Certificate class to store SSL certificate details

class Certificate:

def \_init\_(self, cert\_id, domain\_name, issue\_date, expiry\_date):

self.cert\_id = cert\_id

self.domain\_name = domain\_name

self.issue\_date = issue\_date

self.expiry\_date = expiry\_date

def \_repr\_(self):

return f"Certificate(cert\_id={self.cert\_id}, domain\_name='{self.domain\_name}', issue\_date={self.issue\_date}, expiry\_date={self.expiry\_date})"

# Define a CertificateManager class for managing SSL certificates

class CertificateManager:

def \_init\_(self):

# Use a dictionary to store certificates by their cert\_id for quick access

self.certificates = {}

# CRUD Operations

def add\_certificate(self, cert\_id, domain\_name, days\_valid):

issue\_date = datetime.now()

expiry\_date = issue\_date + timedelta(days=days\_valid)

cert = Certificate(cert\_id, domain\_name, issue\_date, expiry\_date)

self.certificates[cert\_id] = cert

print(f"Certificate added: {cert}")

def get\_certificate(self, cert\_id):

return self.certificates.get(cert\_id, "Certificate not found")

def update\_certificate(self, cert\_id, domain\_name=None, days\_valid=None):

if cert\_id in self.certificates:

cert = self.certificates[cert\_id]

if domain\_name:

cert.domain\_name = domain\_name

if days\_valid:

cert.expiry\_date = cert.issue\_date + timedelta(days=days\_valid)

print(f"Certificate updated: {cert}")

else:

print("Certificate not found")

def delete\_certificate(self, cert\_id):

if cert\_id in self.certificates:

del self.certificates[cert\_id]

print(f"Certificate {cert\_id} deleted")

else:

print("Certificate not found")

# Manage the issuance and renewal of SSL certificates

def renew\_certificate(self, cert\_id, extra\_days):

if cert\_id in self.certificates:

cert = self.certificates[cert\_id]

cert.expiry\_date += timedelta(days=extra\_days)

print(f"Certificate renewed: {cert}")

else:

print("Certificate not found")

# Track and alert on upcoming certificate expiries

def track\_certificate\_expiries(self, days\_ahead):

upcoming\_expiries = []

current\_date = datetime.now()

for cert in self.certificates.values():

if 0 <= (cert.expiry\_date - current\_date).days <= days\_ahead:

upcoming\_expiries.append(cert)

return upcoming\_expiries

# Example usage

if \_name\_ == "\_main\_":

manager = CertificateManager()

# Add certificates

manager.add\_certificate(cert\_id=1, domain\_name="example.com", days\_valid=90)

manager.add\_certificate(cert\_id=2, domain\_name="example.org", days\_valid=120)

# Retrieve a certificate

print(manager.get\_certificate(cert\_id=1))

# Update a certificate

manager.update\_certificate(cert\_id=1, domain\_name="new-example.com", days\_valid=180)

# Renew a certificate

manager.renew\_certificate(cert\_id=2, extra\_days=60)

# Track upcoming expiries within 30 days

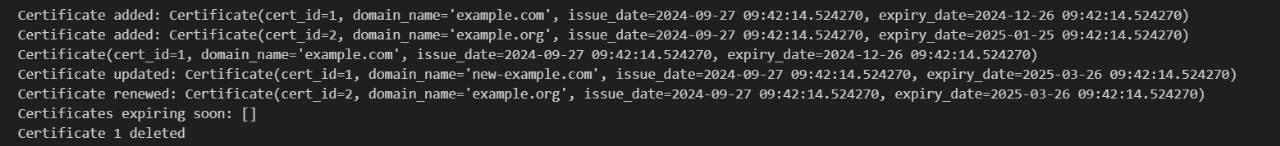
expiring\_soon = manager.track\_certificate\_expiries(days\_ahead=30)

print("Certificates expiring soon:", expiring\_soon)

# Delete a certificate

manager.delete\_certificate(cert\_id=1)

**OUTPUT:**

****

* **Key Features and Skills Acquired**
* **Automated Certificate Monitoring:** Automatically checks and records SSL certificate data from specified domains.
* **Database Integration:** Efficiently stores certificate details for future reference and analysis
* **Expiration Alerts:** Sends notifications when certificates are approaching their expiration date.
* **Python Programming:** Deepened understanding of Python, particularly in network programming and security.
* **SSL/TLS Management:** Gained knowledge of SSL certificates, their structure, and how to manage them programmatically.
* **Database Management:** Enhanced skills in managing data with SQLite/MySQL, including CRUD operations.
* **Automation:** Learned how to automate certificate fetching and alert systems
* **Challenges and Solutions**
* **Challenge:** Handling expired or invalid certificates without crashing the program.
* **Solution:** Implemented error handling using try-except blocks to catch SSL-related exceptions and log them appropriately.
* **Results and Impact**

The SSL Certificate Recorder successfully automated the monitoring of SSL certificates, reducing the risk of unexpected expirations and improving the organization’s security posture. The tool’s alert system has been particularly beneficial in maintaining compliance with security protocols.

* **Conclusion**

The SSL Certificate Recorder project was a rewarding experience that allowed me to apply my Python skills to solve real-world problems. The project not only enhanced my technical skills but also gave me a deeper understanding of cybersecurity practices related to SSL certificate management.

* **Recommendations**

Integrate a web interface to manage certificate monitoring and view reports more easily.

Expand the notification system to include SMS alerts or Slack integration for better accessibility.

* **References**
* Python Official Documentation: <https://docs.python.org/3/>
* OpenSSL Documentation: <https://www.openssl.org/docs/>
* Cryptography Library: <https://cryptography.io/en/latest/>
* **Our project presentation:**

<https://www.canva.com/design/DAGR144g78Q/gh1oS0_-cRo9XCOUogrUfw/view?utm_content=DAGR144g78Q&utm_campaign=designshare&utm_medium=link&utm_source=editor>