#SSL Certificate Manager project code:-

```
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)
```