**FYP – Client/Server AIOQUIC setup:**

**Step-by-Step Process for Setting Up the Certificate with SAN**

1. **Prepare the Configuration File for OpenSSL**

Create a configuration file named san.cnf in your project directory. This file will specify the SAN extension to include localhost as a valid domain.

**# san.cnf**

**[ req ]**

**default\_bits = 2048**

**distinguished\_name = req\_distinguished\_name**

**req\_extensions = req\_ext**

**prompt = no**

**[ req\_distinguished\_name ]**

**CN = localhost**

**[ req\_ext ]**

**subjectAltName = @alt\_names**

**[ alt\_names ]**

**DNS.1 = localhost**

* + **Explanation**:
    - [ req ]: Configures OpenSSL for certificate generation.
    - default\_bits: Sets the key size to 2048 bits.
    - distinguished\_name: Specifies the section that defines the certificate’s distinguished name.
    - req\_extensions: Points to the req\_ext section to define additional certificate extensions.
    - subjectAltName: Specifies the alternative names, such as DNS.1 = localhost.

1. **Generate the Certificate and Private Key with OpenSSL**

Use OpenSSL with the san.cnf configuration file to generate a certificate and private key. This command will create a self-signed certificate (cert.pem) and private key (privkey.pem).

bash

**openssl req -x509 -nodes -newkey rsa:2048 -keyout privkey.pem -out cert.pem -days 365 -config san.cnf**

* + **Explanation**:
    - -x509: Generates a self-signed certificate.
    - -nodes: Skips encryption of the private key.
    - -newkey rsa:2048: Creates a new RSA key with a 2048-bit size.
    - -keyout privkey.pem: Specifies the output file for the private key.
    - -out cert.pem: Specifies the output file for the certificate.
    - -days 365: Sets the certificate validity period to 365 days.
    - -config san.cnf: Tells OpenSSL to use the san.cnf file to include the subjectAltName.

1. **Verify the Certificate’s SAN**

Check that the subjectAltName is included in the certificate. This is crucial to confirm that it was generated correctly.

**openssl x509 -in cert.pem -text -noout**

* + Look for a section labeled X509v3 Subject Alternative Name. It should contain DNS:localhost, like this:

**X509v3 extensions:**

**X509v3 Subject Alternative Name:**

**DNS:localhost**

1. **Update the Server to Use the New Certificate and Key**

Ensure that your server (e.g., server.py) loads these files when configuring the SSL/TLS connection.

**from aioquic.quic.configuration import QuicConfiguration**

**configuration = QuicConfiguration(is\_client=False)**

**configuration.load\_cert\_chain(certfile="cert.pem", keyfile="privkey.pem")**

1. **Start the Server and Client**

Now you can start your server with python server.py, and then run the client with python client.py. The client should successfully connect without any subjectAltName errors.