**SSL HTTPS beszerzése a localhost számára**

2021. április 26

Ez a cikk végigvezeti az SSL-tanúsítványok helyi kiszolgálókhoz való kiépítésének folyamatán. Egy éles kiszolgálóhoz való webalkalmazás létrehozásához helyi tesztelés szükséges a telepítés előtt. Lehetőség van arra is, hogy a webalkalmazást helyi környezetben futtassuk, és az interneten keresztül is elérhető legyen.

Bevezetés

Ugyanilyen fontos, hogy ehhez kapjunk egy [SSL-tanúsítványt](https://www.cloudflare.com/en-gb/learning/ssl/what-is-an-ssl-certificate/) , hogy a felhasználók megbízhassanak az oldalon, és eltávolítsuk a *„Nem biztonságos”* üzenetet a címsorból.

Előfeltételek

A cikk követéséhez hasznos lesz, ha rendelkezel a következőkkel:

* [Node.js](https://nodejs.org/en/) telepítve.
* Postman telepítve.
* Az operációs rendszer lehet Windows [Cygwin](https://cygwin.com/install.html) vagy Linux alapú operációs rendszerrel telepítve.
* [Linux bash](https://www.linuxtrainingacademy.com/linux-commands-cheat-sheet/) parancsok, [Node.js](https://nodejs.dev/learn) és [Express](https://www.npmjs.com/package/express) munkaismeret .
* Működő [Firefox](https://www.mozilla.org/en-US/firefox/new/) böngésző telepítve.
* Szövegszerkesztő – [VS Code](https://code.visualstudio.com/download) .

Követendő lépések

1. Hozzon létre egy tanúsítványt.
2. Írjon alá egy SSL-tanúsítványt a localhost számára.
3. Fejlesszen ki egy kiszolgálót a Node.js használatával, amelyet egy localhost SSL-tanúsítvány használatával szolgálnak ki.
4. Konfigurálja a Firefox webböngészőt és a Postman API-klienst úgy, hogy engedélyezze az általunk hitelesítésszolgáltatóként aláírt tanúsítványokat.
5. Biztonságosan érje el a localhost HTTPS-sel a böngészőből vagy az API-kliensből.
6. lépés: generáljon CA-tanúsítványt   
   (aláíró cég tanúsítványa)

Név (CN)

Teleki Certificate Services

Szervezet (O)

Teleki Blanka Gimnazium

Szervezeti egység (OU)

Info\_13i

Az SSL-tanúsítványokat általában a Certificate Authority (CA) néven ismert harmadik fél cégek írják alá. Ők az internet megbízható kibocsátói, és minden tanúsítvány kiadása előtt alaposan megvizsgálják, hogy a webhely azt csinálja-e, amit tennie kell.

A CA azonban nem ad ki tanúsítványt a localhost számára, egyszerűen azért, mert senki sem birtokolja a localhostot. Esetünkben ugyanúgy aláírjuk a tanúsítványt, mint ahogy a CA teszi.

Az első lépés egy root CA-tanúsítvány létrehozása az alábbi parancsok segítségével:

$ mkdir cert

$ cd cert

$ mkdir CA

$ cd CA

$ openssl genrsa -out CA.key -des3 2048

A fenti parancsok egy **privát kulcsot** generálnak, és egyszerű jelszót kérnek a kulcshoz. A felhasználó beírja a jelszót, majd újra beírja a megerősítéshez.

Ezután a generált kulcs felhasználásával egy **root CA tanúsítványt** generálunk, amely esetünkben tíz évig lesz érvényes. A rendszer kéri a kulcshoz és a tanúsítványhoz tartozó jelszót. A felhasználó megadhatja a kívánt tanúsítványinformációt, vagy alapértelmezettként hagyhatja azt.

Futtassa az alábbi parancsot a fent tárgyalt eléréséhez:

$ openssl req -x509 -sha256 -new -nodes -days 3650 -key CA.key -out CA.pem

Jelenleg cert/CA folderben két fájlunk van, CA.keyés a CA.pem.

1. lépés: Tanúsítvány generálása

Név (CN) **localhost**

Szervezet (O) Bene Z

Szervezeti egység (OU) tanar

Most létrehoztuk a CA-kulcsot és a CA-tanúsítványt. Lehetőség van SSL-tanúsítványok aláírására, mivel már létrehoztuk a CA-t.

Ezután a cert/CAkönyvtárban hozzon létre egy új könyvtárat, localhost. Belül localhosthozzon létre egy új fájlt, localhost.ext.

$ mkdir localhost

$ cd localhost

$ touch localhost.ext

Az aláírt SSL-tanúsítványba írandó információkat ez a localhost.extfájl fogja tartalmazni.

A fájl az alábbiak szerint fog megjelenni:

authorityKeyIdentifier = keyid,issuer

basicConstraints = CA:FALSE

keyUsage = digitalSignature, nonRepudiation, keyEncipherment, dataEncipherment

subjectAltName = @alt\_names

[alt\_names]

DNS.1 = localhost

IP.1 = 127.0.0.1

Vegye figyelembe, hogy a tanúsítvány működni fog a **localhost** és a **127.0.0.1** számára is. Dönthet úgy, hogy további [tartományokat](https://www.wpbeginner.com/beginners-guide/beginners-guide-what-is-a-domain-name-and-how-do-domains-work/) vagy IP-címeket ad hozzá a fájlhoz, de ügyeljen az [/etc/hosts](https://www.howtogeek.com/howto/27350/beginner-geek-how-to-edit-your-hosts-file/) fájl szerkesztésére, hogy ezek a tartományok a helyi gépre mutassanak (127.0.0.1).

A következő lépésben létre kell hozni egy kulcsot, és a kulcs segítségével létrehozni egy [CSR](https://www.thesslstore.com/knowledgebase/ssl-generate/cetificate-signing-request-overview/) -t (Certificate Signing Request) az alábbi paranccsal.

$ openssl genrsa -out localhost.key -des3 2048

A parancs generálja a **localhost privát kulcsot**, és a rendszer kéri a kulcshoz tartozó jelszót, és a felhasználót megkéri annak megerősítésére.

A következő lépés a CSR létrehozása lesz a kulcs segítségével, majd a rendszer kéri a fent létrehozott jelszót. Bármilyen egyéb kért adat meghagyható alapértelmezettként vagy beírható.

Jegyezze fel a kért kihívás jelszavát; bármit be lehet írni.

Használja az alábbi parancsot a CSR létrehozásához:

$ openssl req -new -key localhost.key -out localhost.csr

Most ezzel a CSR-rel megkérjük a CA-t, hogy írjon alá egy tanúsítványt az alábbiak szerint. Vegye figyelembe, hogy a CA.keyés a CA.pemfájlok elérési útja attól függ, hogy a felhasználó honnan futtatja a parancsokat. Ebben az esetben az alábbi parancsok innen futnak /cert/CA/localhost.

$ openssl x509 -req -in localhost.csr -CA ../CA.pem -CAkey ../CA.key -CAcreateserial -days 3650 -sha256 -extfile localhost.ext -out localhost.crt

Ez a parancs beolvassa a CSR-t (**localhost.csr**), a CA tanúsítványt ( CA.pem és CA.key), valamint a tanúsítványkiterjesztés fájlt ( localhost.ext). Ezek a bemenetek egy **localhost.crt** tanúsítványfájlt generálnak, amely tíz évig érvényes.

A szervernek szüksége lesz a **localhost.crt** tanúsítványfájlra és a visszafejtett kulcsra, mivel a ***localhost.key*** fájlunk titkosított formában van.

Dekódolnunk kell a **localhost.key** fájlt, és tárolnunk kell az alábbiak szerint:

$ openssl rsa -in localhost.key -out localhost.decrypted.key

3. lépés: Node.js Express szerver létrehozása

Annak teszteléséhez, hogy a fenti kódrészletek a várt módon működnek-e, létre kell hoznunk egy Node.js Express szervert, amely sikerüzenetet ad ki.

Visszatérve a certkönyvtárunkba, inicializálunk egy Node.js projektet, és hozzáadjuk az expressz és HTTPS csomagokat az alábbi parancsokkal:

$ npm init -y

$ npm i express https

$ touch index.js

Ezután szerkessze a index.jsfájlt az alábbiak szerint:

const fs = require('fs');

const key = fs.readFileSync('./CA/localhost/localhost.decrypted.key');

const cert = fs.readFileSync('./CA/localhost/localhost.crt');

const express = require('express');

const app = express();

app.get('/', (req, res, next) => {

res.status(200).send('Hello world!');

});

const https = require('https');

const server = https.createServer({ key, cert }, app);

const port = 3000;

server.listen(port, () => {

console.log(`Server is listening on https://localhost:${port}`);

});

Futtassa a szervert az alábbi paranccsal:

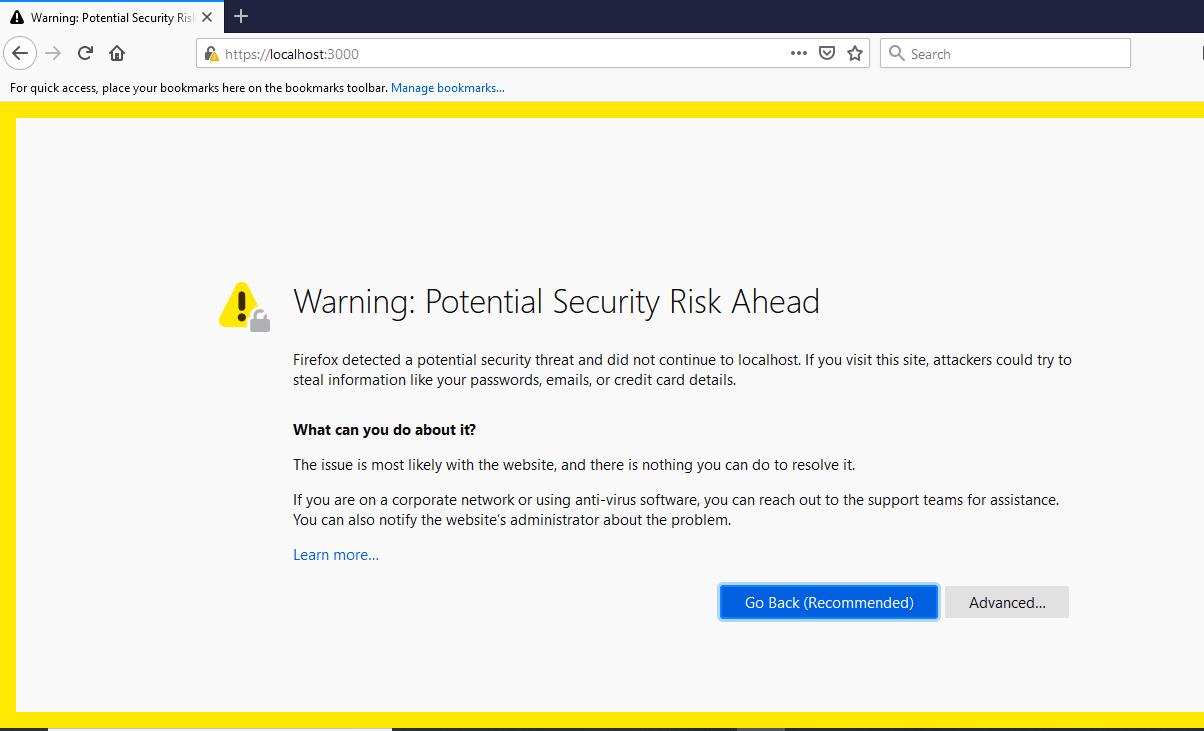
$ node index.js

A konzolon megjelenik egy sikerüzenet: A szerver figyel a https://localhost:3000 webhelyen.

**HTTPS működése:** https://love2dev.com/blog/how-https-works/

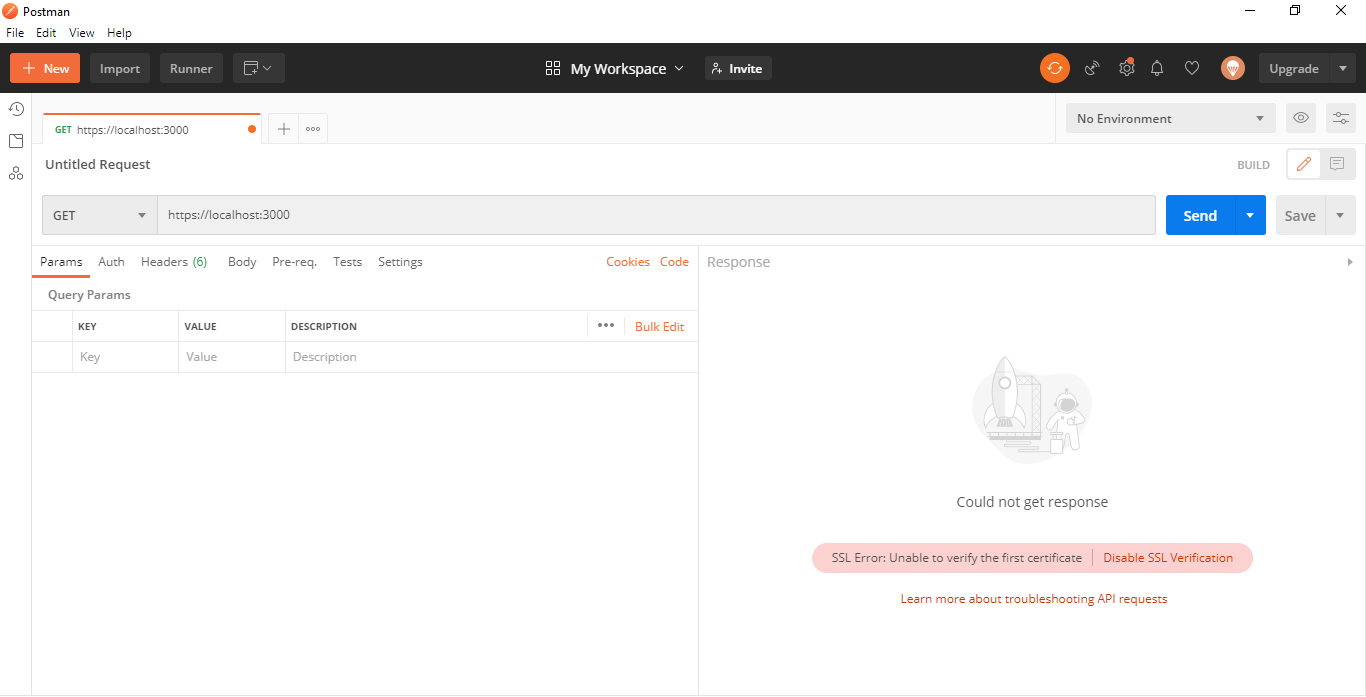
4. lépés: Tesztelje böngészőben és API-kliensben

Most, hogy szerverünk kiszolgálja az SSL-t, kipróbálhatjuk a [https://localhost:3000](https://localhost:3000/) hivatkozást a Firefox böngészőben az alábbiak szerint:



Közel vagyunk, de még nem fejeztük be. A Firefox még azután sem bízott meg a hitelesítésszolgáltatóban, amelyik aláírta ezt a tanúsítványt, ami így van. nem bízhatunk meg minket hitelesítésszolgáltatóként.

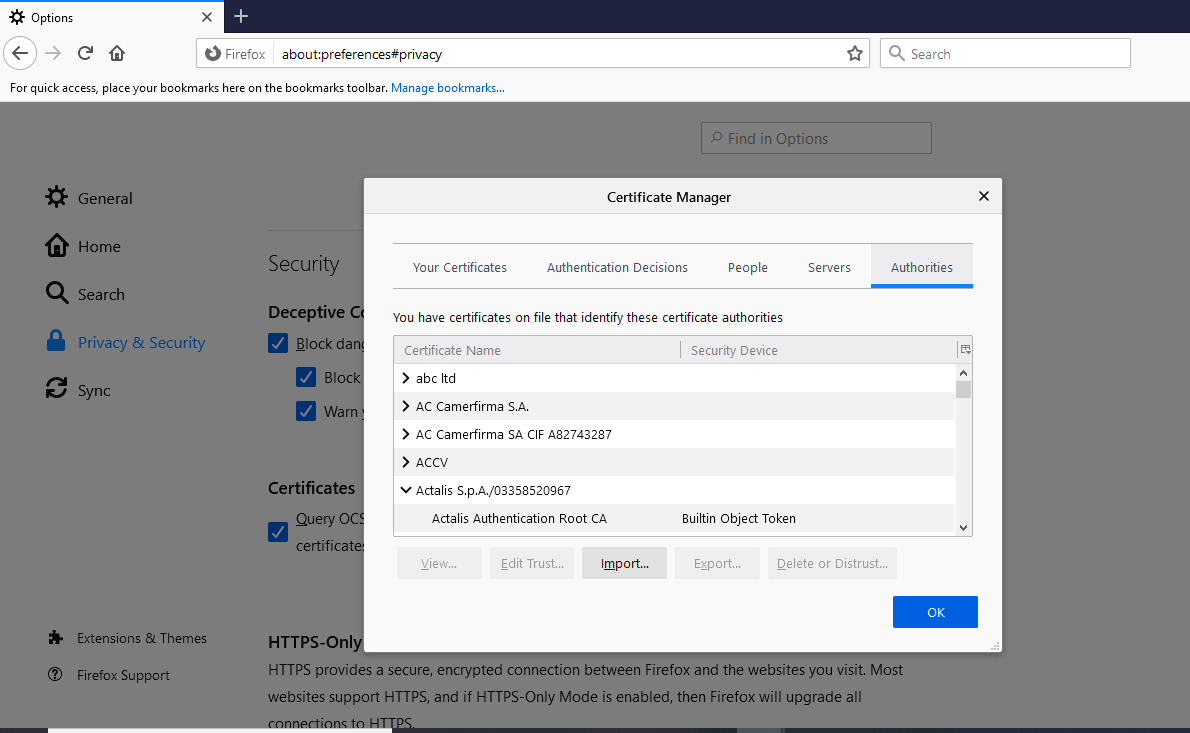
Ennek megoldásához először teszteljük a Postmannal való kapcsolatunkat az alábbiak szerint:



Mint fentebb látható, az SSL-tanúsítvány nem ellenőrizhető.

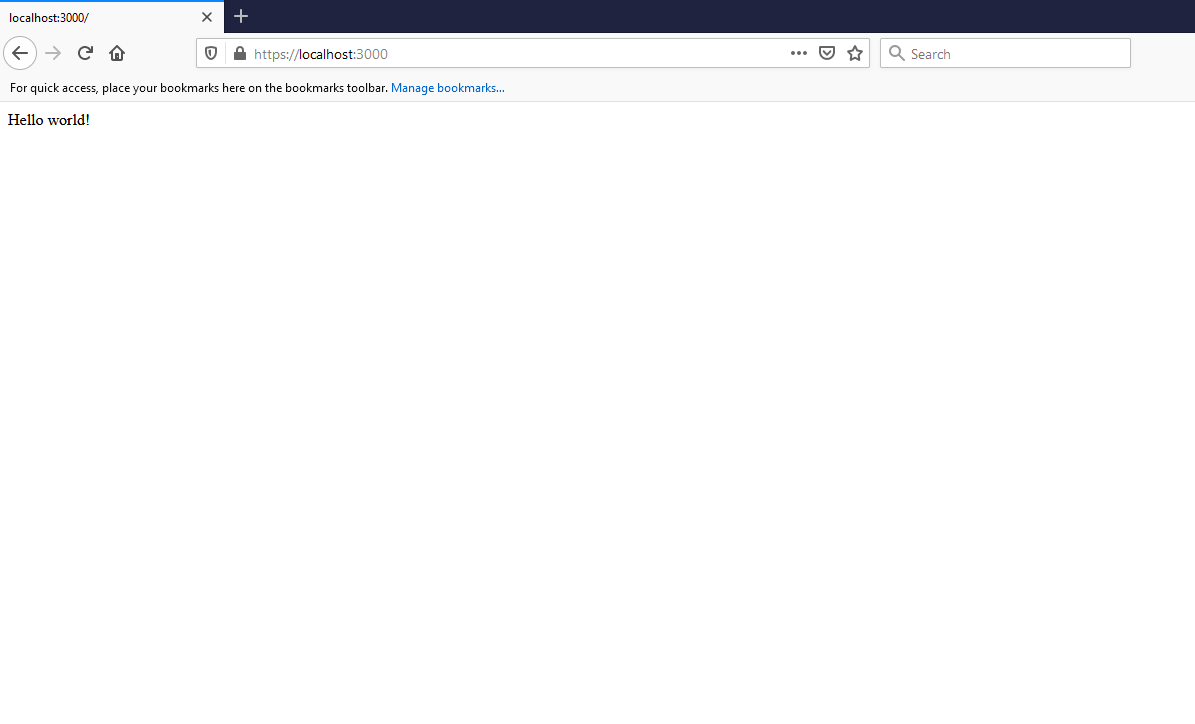
5. lépés: CA-tanúsítvány importálása a böngészőbe

Ahhoz, hogy a böngésző megbízhasson az általunk létrehozott CA-tanúsítványban, importálnia kell a tanúsítványt. A Firefoxban nyissa meg a *Beállítások* elemet, lépjen az *Adatvédelem és biztonság* elemre, és kattintson a *Tanúsítványok megtekintése* elemre az alábbiak szerint:



Ezután kattintson az importálásra,**CA.pem** és jelölje be a jelölőnégyzetet, hogy megbízzon az imént importált CA-ban, majd kattintson az OK gombra.

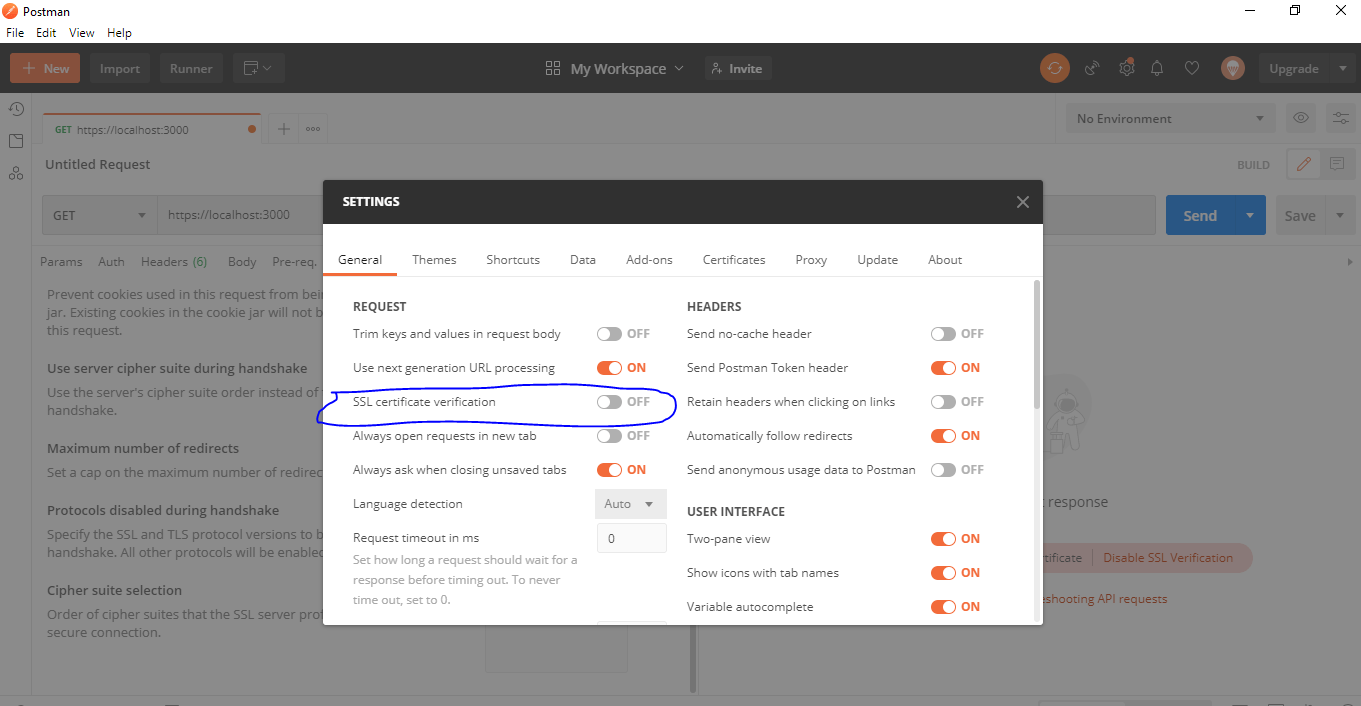
Ezután a Firefoxban teszteljük az alábbiak szerint:



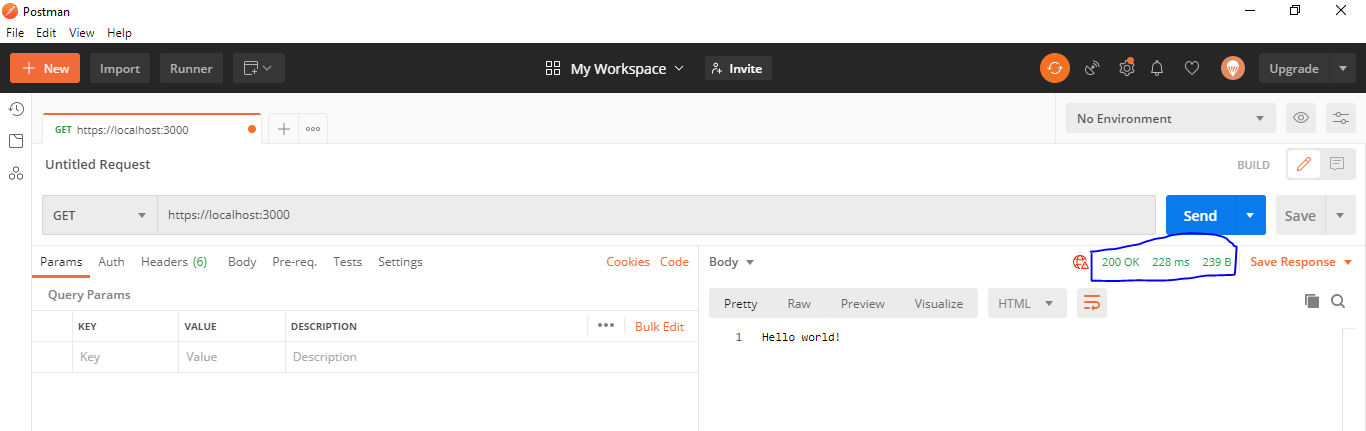
A böngésző most rendben van, és megbíznak a CA-ban és a tanúsítványban.

6. lépés: Az API-kliens konfigurálása és tesztelése

Az API-kliens konfigurálásának folyamata ugyanaz, mint a böngészőé, és a különbség az, hogy le kell tiltani a tanúsítvány érvényesítését. A Postman beállításainál győződjön meg arról, hogy az SSL-tanúsítvány-ellenőrzés ki van kapcsolva.



Ezután újra elküldjük a kérést a Postmanban, és az eredmény a következő lesz:

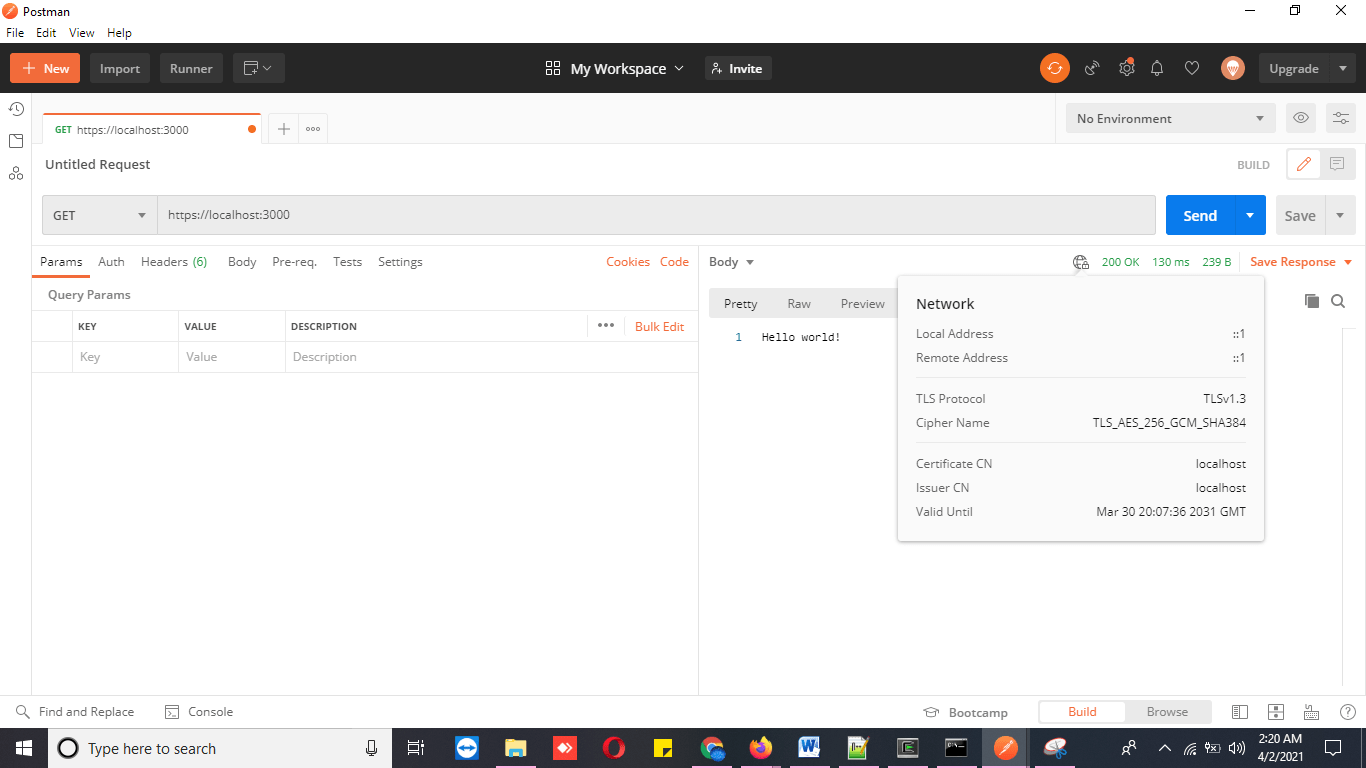


Kaptunk egy **200 OK** üzenetet, ami ebben a szakaszban jó. A probléma azonban az, hogy továbbra is piros biztonsági figyelmeztetésünk van, mert az SSL-tanúsítványunk ki van kapcsolva.

A figyelmeztetés részletesen így fog kinézni:

A probléma megoldásához hozzáadhatjuk CA-nkat a Postmanhoz. A Postman beállításainál győződjön meg arról, hogy a CA-tanúsítványok be vannak kapcsolva, és válassza ki a CA.pemfájlt.

Ezután futtassa újra a kérést az alábbiak szerint; ezúttal egy biztonságos **200 OK** üzenetet kapunk.



Következtetés

Ebben az útmutatóban láthattuk, hogy lehetséges-e a localhost SSL-lel kiszolgálni. Az útmutató segítségével be tudtuk állítani a CA-t, és ezzel aláírtuk az SSL-tanúsítványt. Módosítottuk a böngésző beállításait, valamint az API-kliens beállításait, hogy elfogadják a tanúsítványt.

Ne feledje, hogy soha nem tanácsos a tanúsítványokat meggondolatlanul hozzáadni a tanúsítványlistához. A rosszindulatú felhasználók mindenhol hitelesítésszolgáltatónak adják ki magukat, és ráveszik a felhasználókat, hogy hitelesnek fogadják el webhelyeiket, ami súlyos biztonsági megsértéshez vezet.

A folyamat segíthet a fejlesztőknek webhelyeiket helyi szervereken vagy továbbított portokon tárolni, hogy biztosítsák látogatóikat a webhely biztonságosságáról.

[A cikkben használt fájlok és kódrészletek a GitHub Repo webhelyen](https://github.com/lewe01/localhost-cert.git) érhetők el .

Boldog kódolást!