Department of Electronic & Telecommunications Engineering
University of Moratuwa
EN3251 Internet of Things

Laboratory Exercise 3: The Constrained Application Protocol (CoAP)

2021 Batch **Semester 5**

Overview

In this exercise you will create your own CoAP server and add resources to it. You will use a CoAP browser to communicate with your server and examine the protocol operation with Wireshark. You will then learn an easy method to expose your local server to the Internet.

Objective

After this session you will be able to,

- Setup a local CoAP server
- Add and manage different types of resources on the server
- Integrate a Chromium based CoAP client
- Expose your local server to the Internet.

Pre-requisites

- A computer with an Internet connection
- The Edge browser (Chrome will not work)
- The Eclipse IDE preferred (You can use IntelliJ IDEA too)
- Wireshark

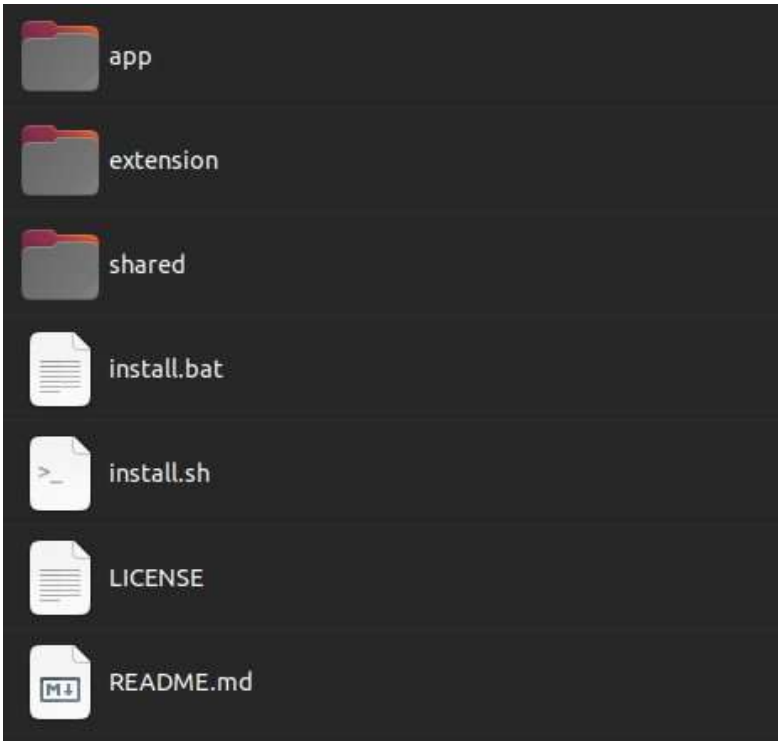
Download and Install required software and resources.

- Install the Eclipse IDE. Link
- Download the Copper for Chromium Client. Link
- Download the Californium framework from here. Link

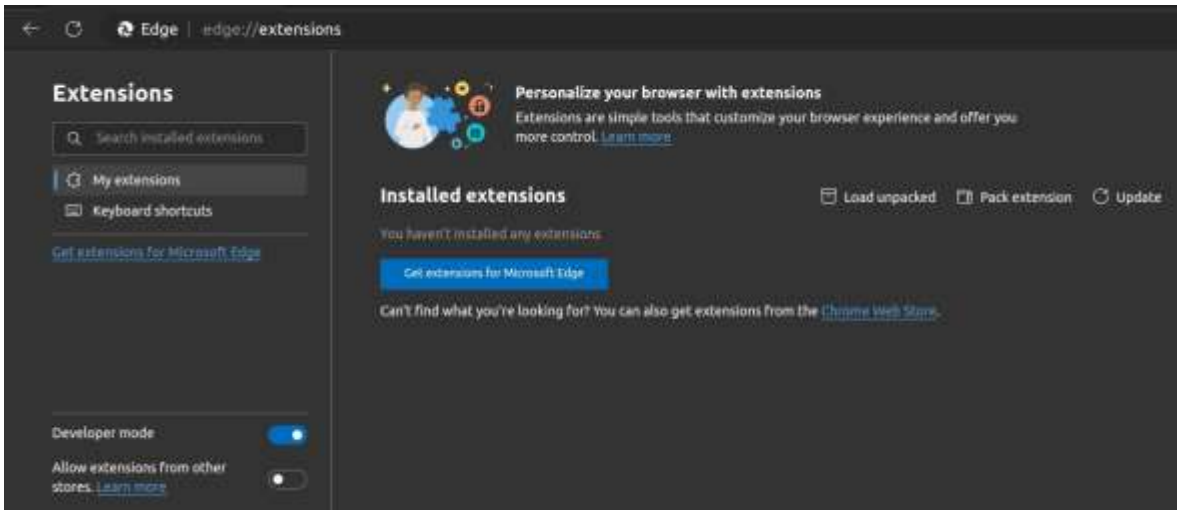
Task 1: Configuring the CoAP Client

At this point you need to have a copy of source code for the Chrome app and extension for Copper4Cr. If not, use this Link. We will be using this client to communicate with our Californium server we created above.

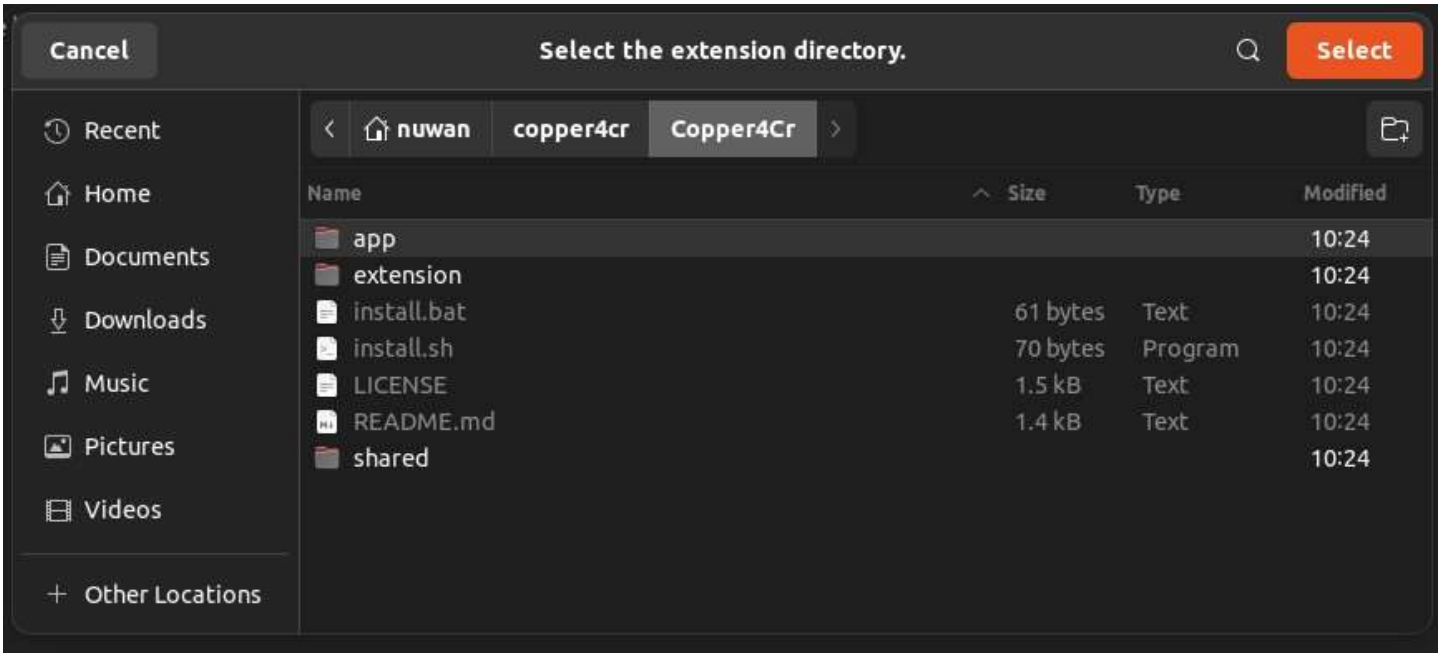
Also, Microsoft Edge will be used to integrate the Chromium app and the extension.



1. Run install.bat (install.sh for Linux) to prepare the codebase.
2. Now Open Edge and navigate to extensions, and turn on Developer Mode.

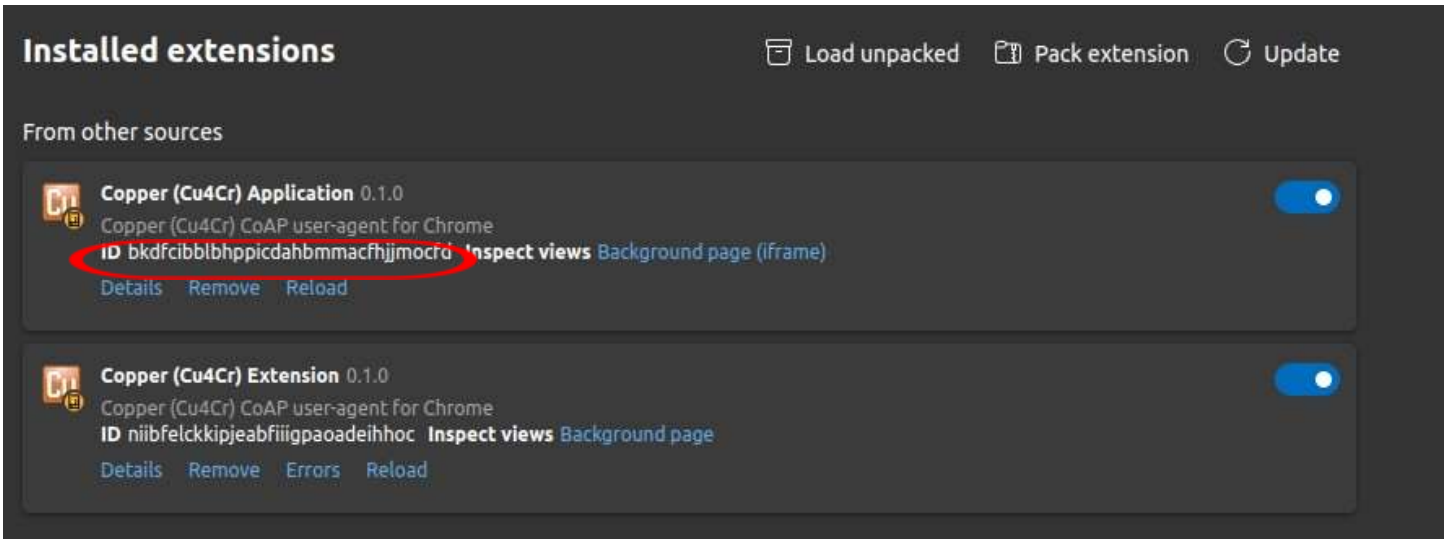


3. Go to the Load unpacked option and locate the app folder under the Copper4Cr folder. Load the entire app folder.

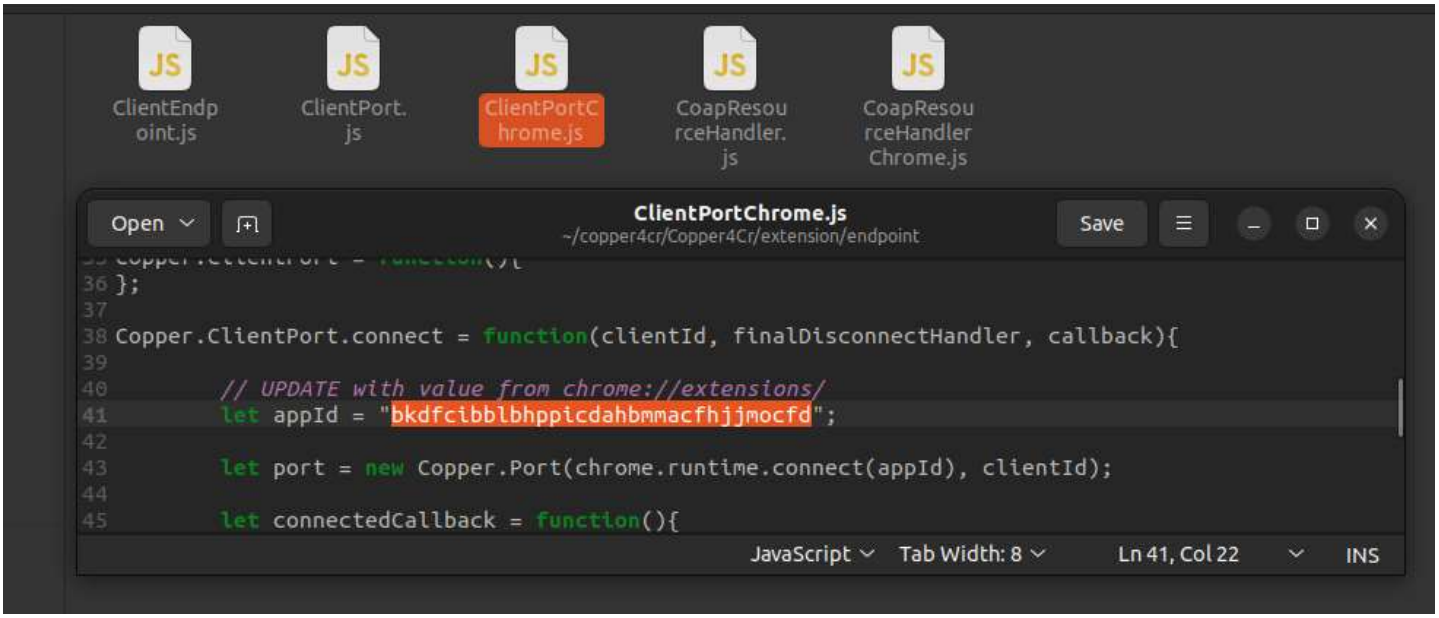


4. Do the same for the extension folder.

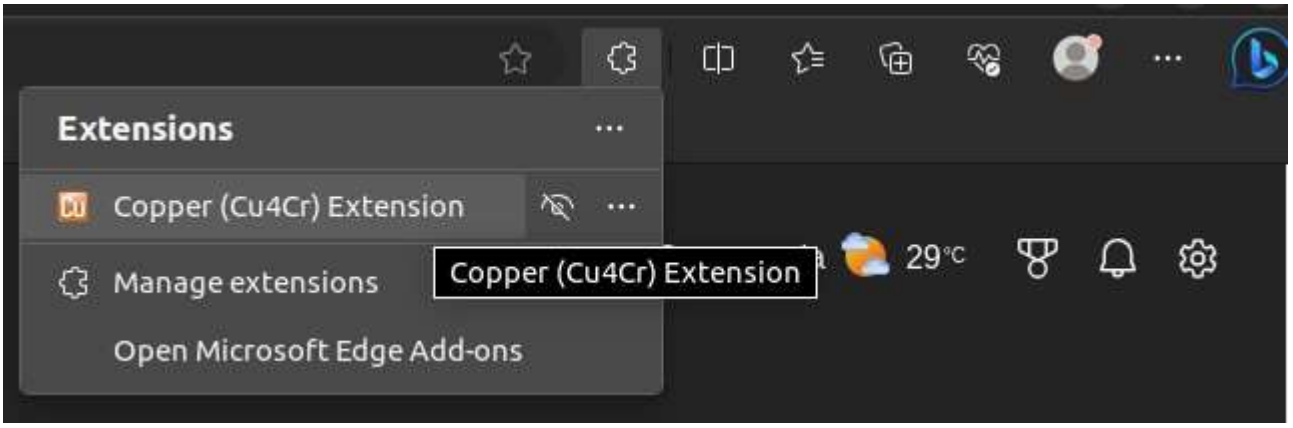
Now you should have both the app and extension integrated to your Edge browser.

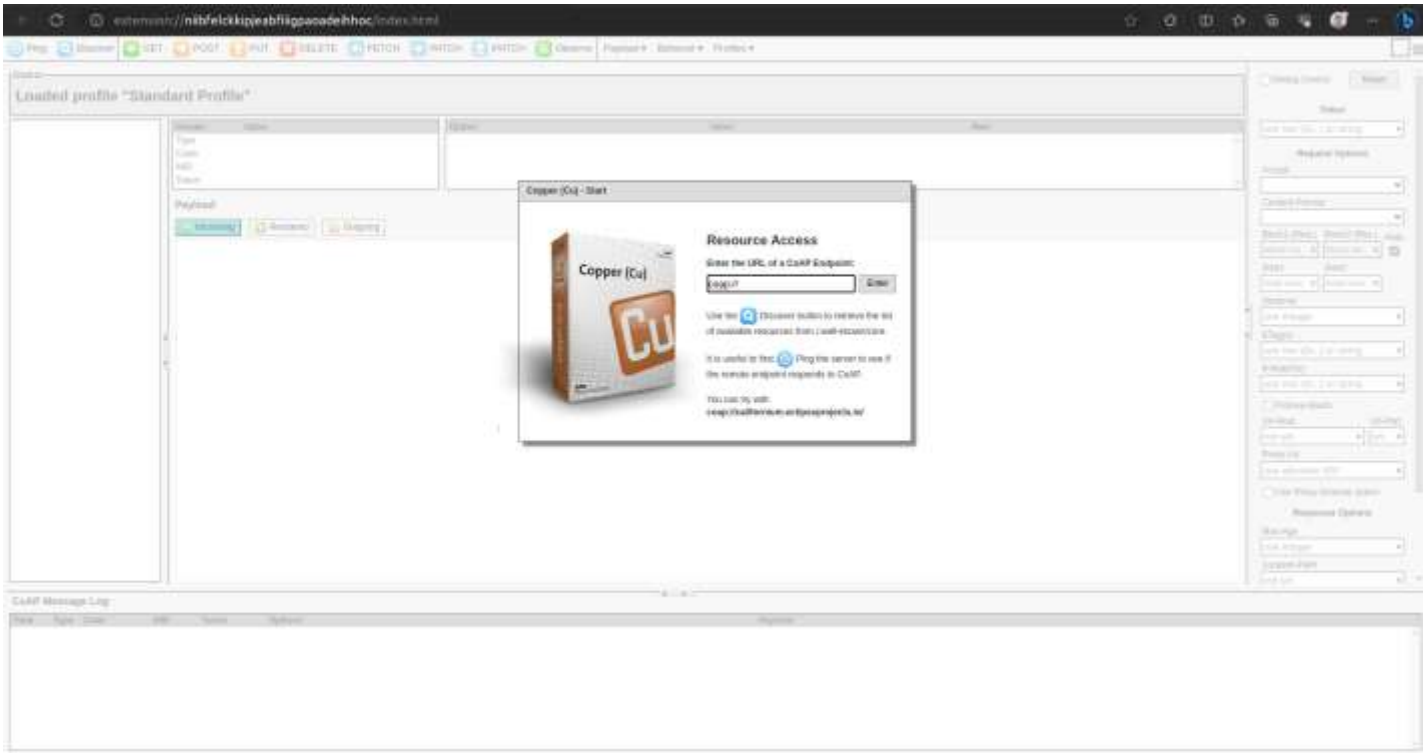


5. Copy your App ID and update the extension/endpoint/ClientPortChrome.js file and save it.



6. You can now test the client from Extensions->Copper(Cu4Cr) Extension

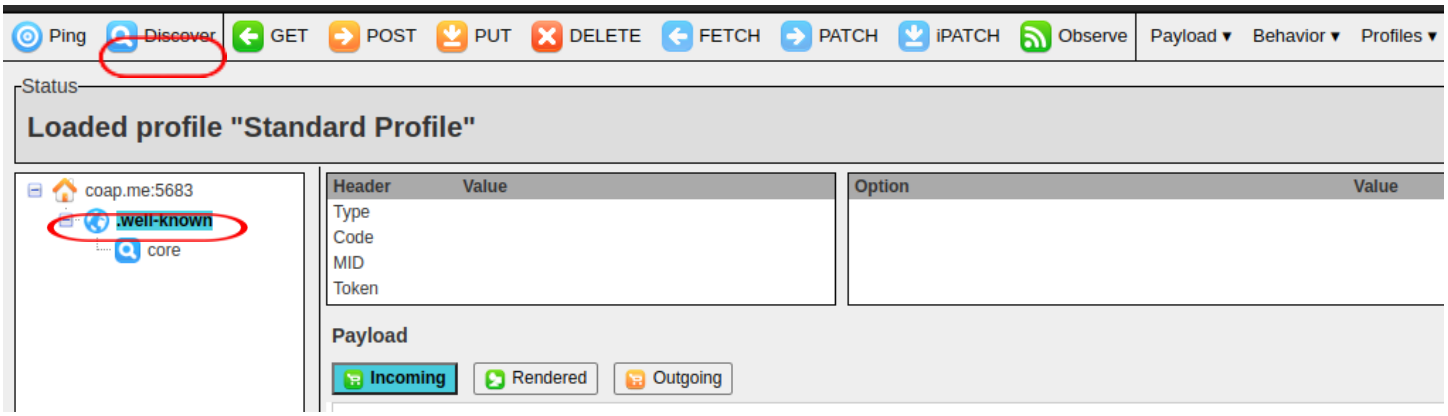



Loaded profile "Standard Profile"
Copper (Cu) - Start
Copper (Cu)
Resource Access
Enter the URL of a CoAP Endpoint:
coap://
Enter
CoAP Message Log

7. Now test it with your server. Enter

coap://coap.me:5683

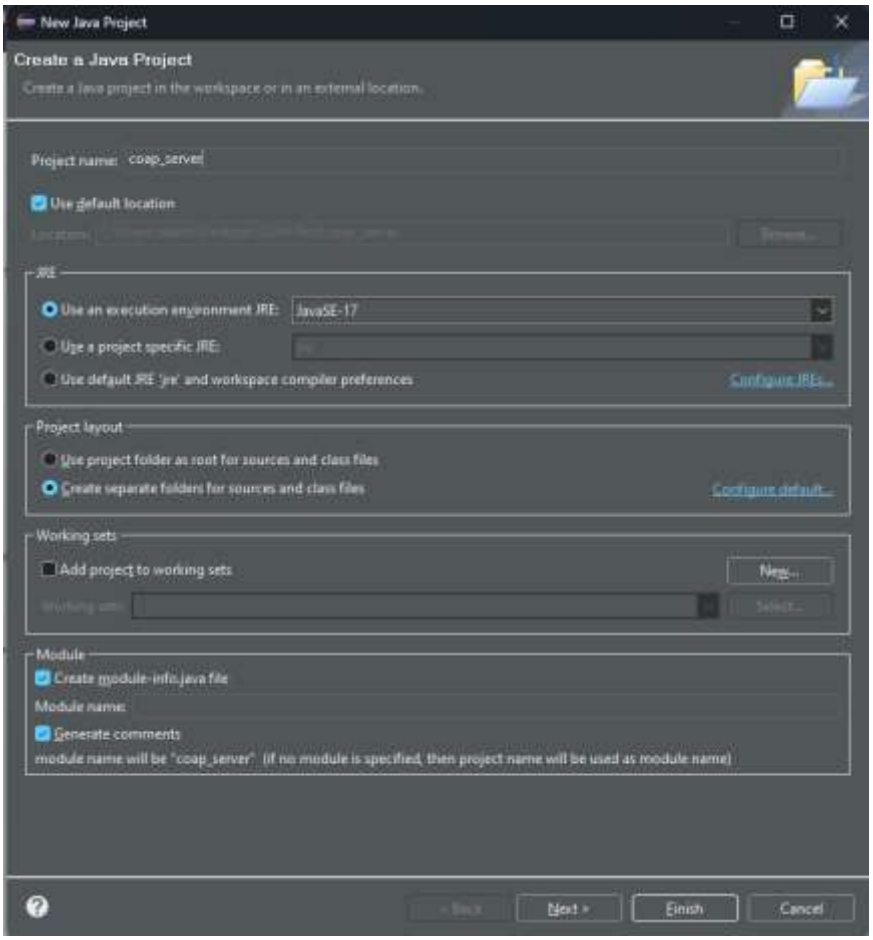
8. Now go to .well-known and press Discover to list out all the available resources in the server.



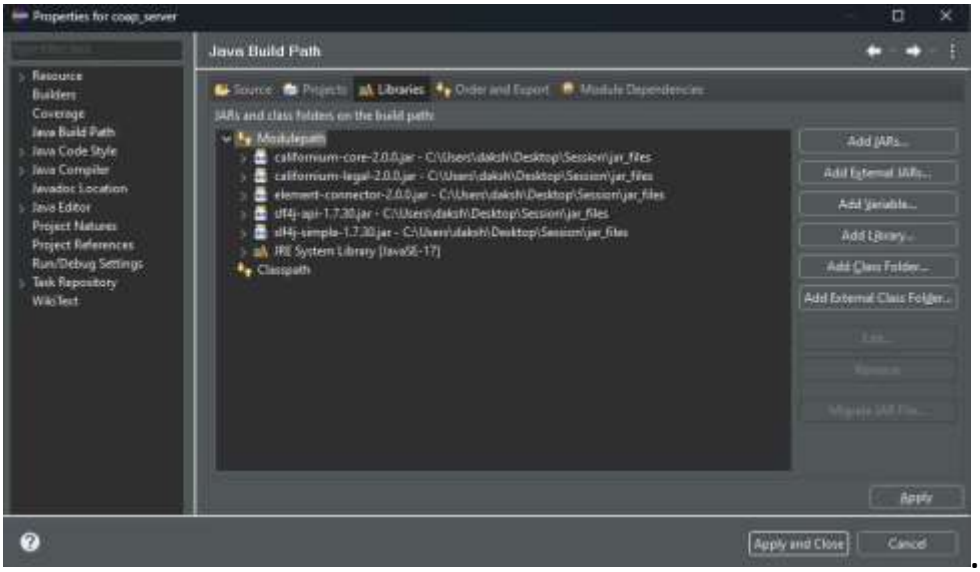
We will be creating our own server in the next step.

Task 2: Setting up and Configuring a CoAP Server

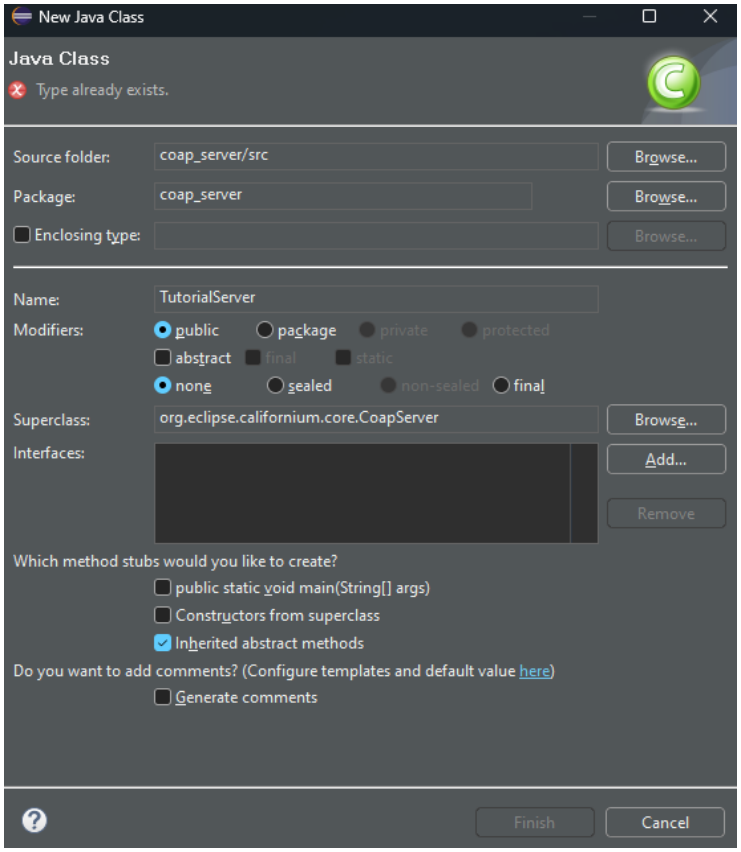
1. Create a Java project.
    - Open Eclipse IDE and Go to File>New>Java Project.
    - Set a Project name and then click Finish.



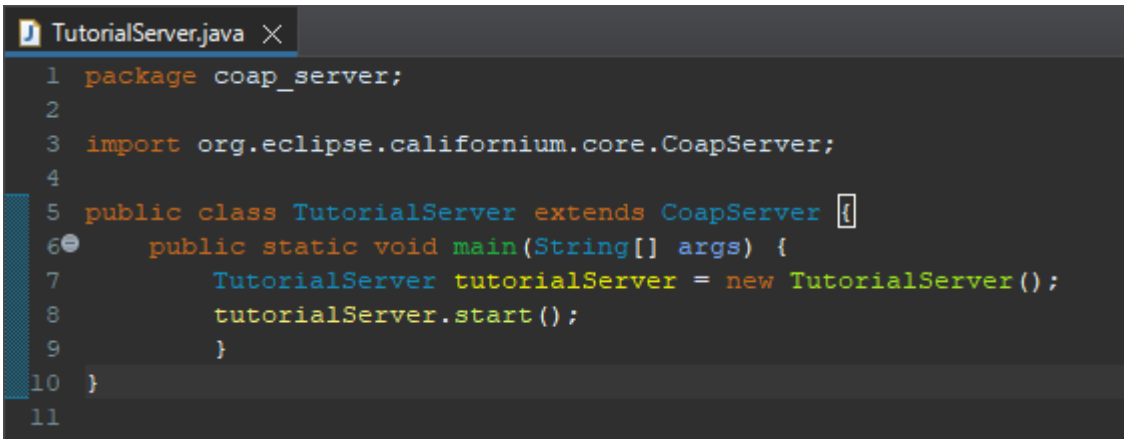
2. Add external JARs files.
    - Go to Project>Properties>Java Build Path>Libraries.
    - Click Modulepath and then click Add External JARs.
    - Select all the JARs files downloaded.
    - Apply and Close the window.



3. Create a Java class.
    - Right click on the src folder and go to New>Class.
    - Set Name as “TutorialServer”.
    - Set Superclass as “org.eclipse.californium.core.CoapServer”.

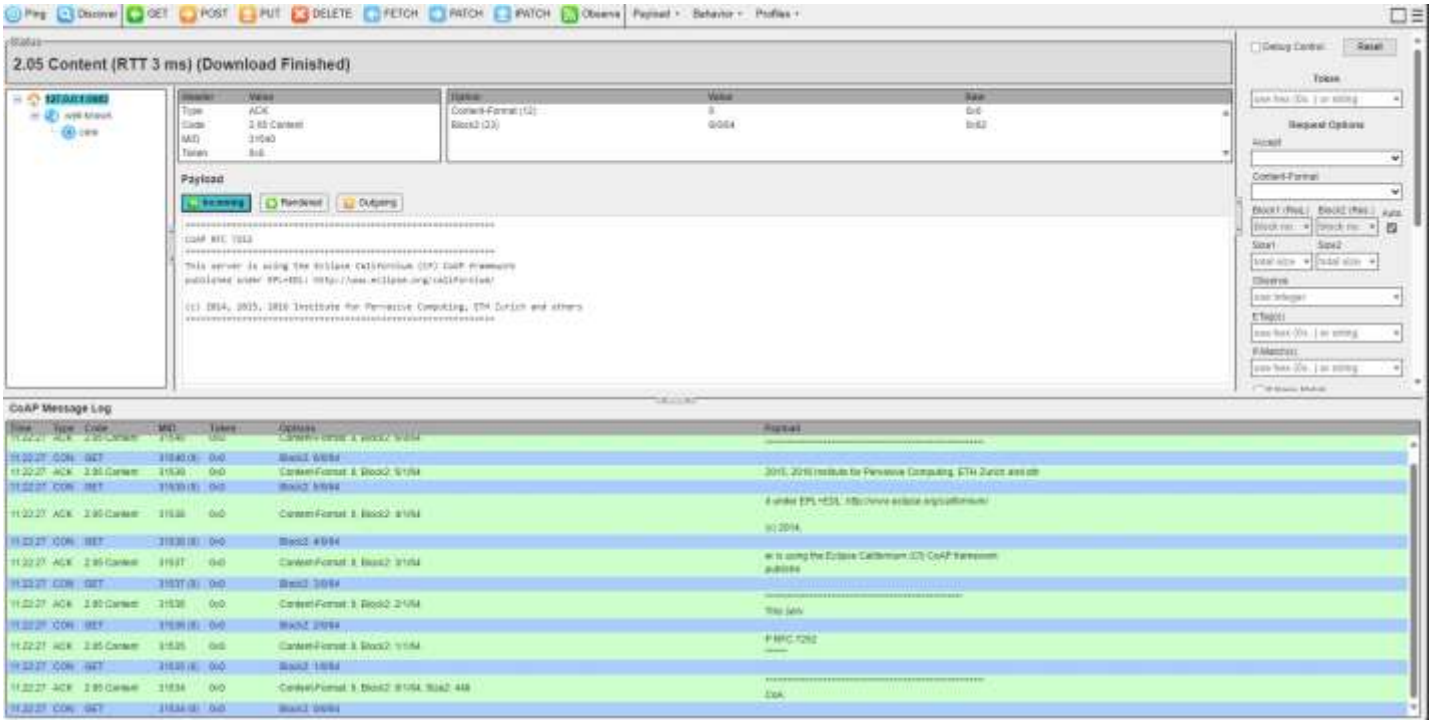


4. Start CoAp server.
    - Create an instance of the TutorialServer class as follows.



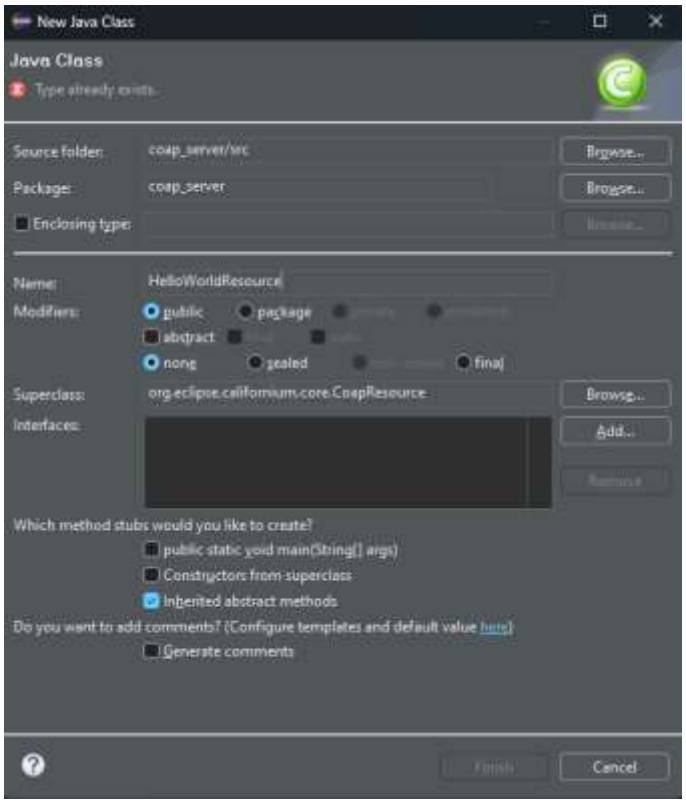
```
TutorialServer.java
1  package coap_server;
2
3  import org.eclipse.californium.core.CoapServer;
4
5  public class TutorialServer extends CoapServer {
6      public static void main(String[] args) {
7          TutorialServer tutorialServer = new TutorialServer();
8          tutorialServer.start();
9          }
10 }
11
```

    - Go to Run>Run and start the server.
    - Launch CopperCr client and query your COAP server by entering coap://127.0.0.1:5683/.
    - Click the GET button and you should see a response from the Coap server.
    - Observe the above communication with Wireshark.



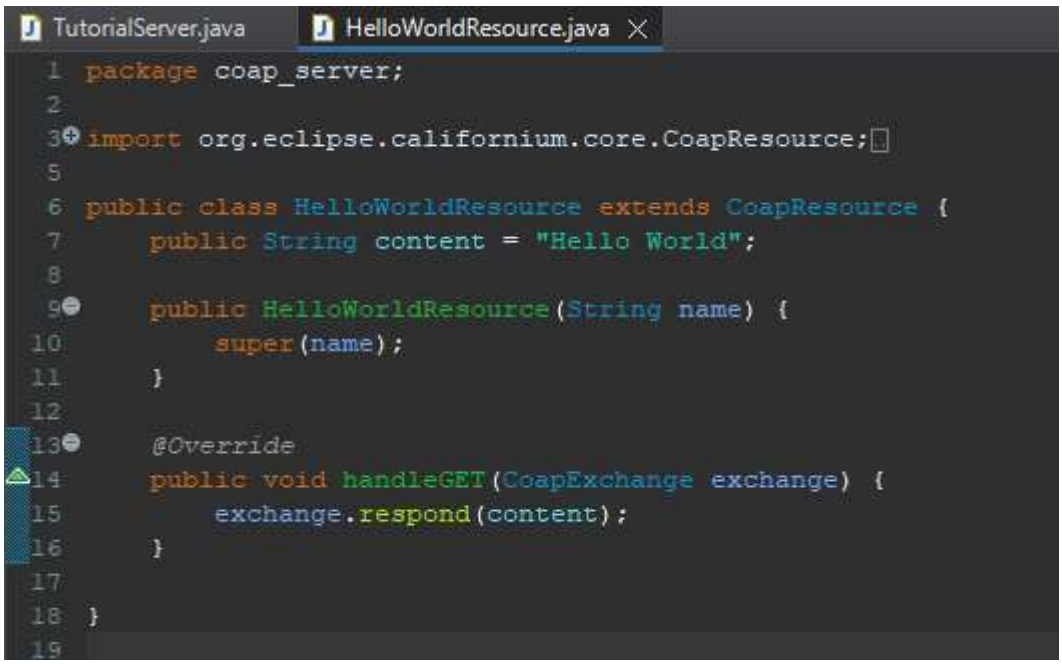
Task 3: Add Resources to Your CoAp Server

1. Create a new Java class.
    - Set name as HelloWorldResource.
    - Set Superclass as org.eclipse.californium.core.CoapResource



2. Add GET Resource Handler.
    - First, we need to configure the Coap handlers for GET request. Paste following code snip inside the HelloWorldResource class.

```
public String content = "Hello World";
public HelloWorldResource(String name) {
        super(name);
 }

 @Override
 public void handleGET(CoapExchange exchange) {
        exchange.respond(content);
 }
```



3. Add PUT Resource Handler.
   - Add PUT handlers to your resource as follows.

```
@Override
 public void handlePUT(CoapExchange exchange){
   byte[] payload = exchange.getRequestPayload();

   try {
     content = new String(payload, "UTF-8");
     exchange.respond(content);
   } catch (Exception e){
     e.printStackTrace();
     exchange.respond("Invalid String");
   }
 }
```

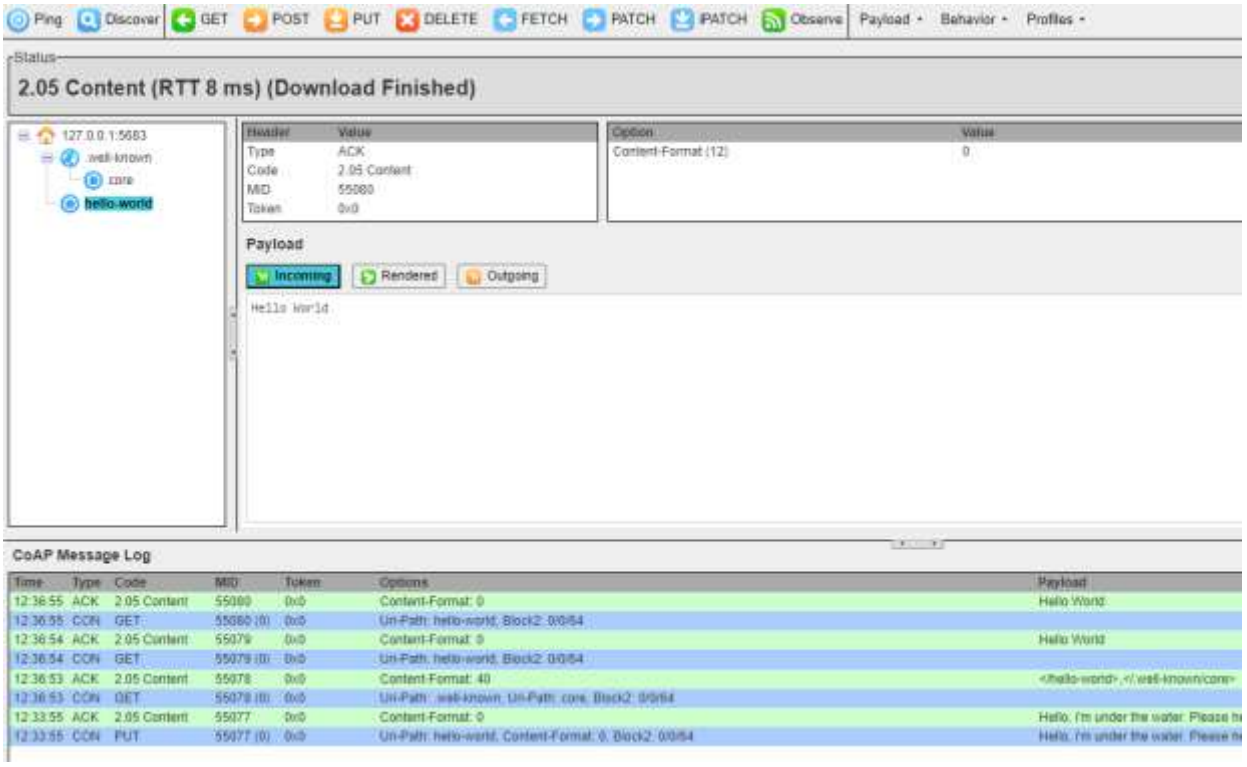
```java
TutorialServer.java    *HelloWorldResource.java

package coap_server;

import org.eclipse.californium.core.CoapResource;

public class HelloWorldResource extends CoapResource {
    public String content = "Hello World";

    public HelloWorldResource(String name) {
        super(name);
    }

    @Override
    public void handleGET(CoapExchange exchange) {
        exchange.respond(content);
    }

    @Override
    public void handlePUT(CoapExchange exchange){
        byte[] payload = exchange.getRequestPayload();

        try {
            content = new String(payload, "UTF-8");
            exchange.respond(content);
        } catch (Exception e){
            e.printStackTrace();
            exchange.respond("Invalid String");
        }
    }

}
```

- Then, we need to create an instance of HelloWorldResource inside our main server.

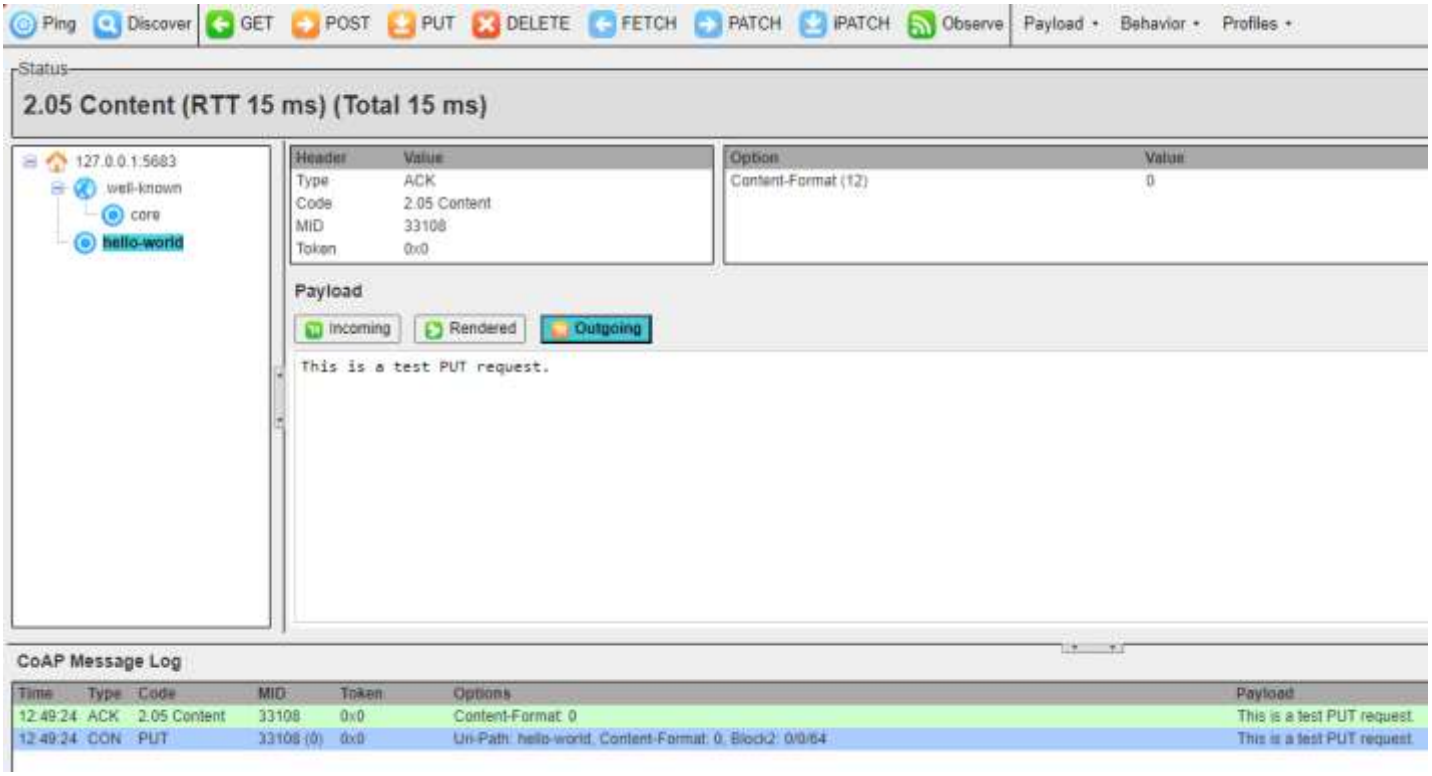
```java
package coap_server;

import org.eclipse.californium.core.CoapServer;

public class TutorialServer extends CoapServer {
    public static void main(String[] args) {
        TutorialServer tutorialServer = new TutorialServer();
        HelloWorldResource hello = new HelloWorldResource("hello-world");
        tutorialServer.add(hello);
        tutorialServer.start();
        }
}
```

4. Start CoAP server
   - Connect to the server as previously.
   - Click the Discover button. Now you should see a new resource called “hello-world”.

- Click on “hello-world” resource and click the GET button. Now, you should see the response from the resource.



- To test the PUT request, type a message inside the Outgoing tab as follows.

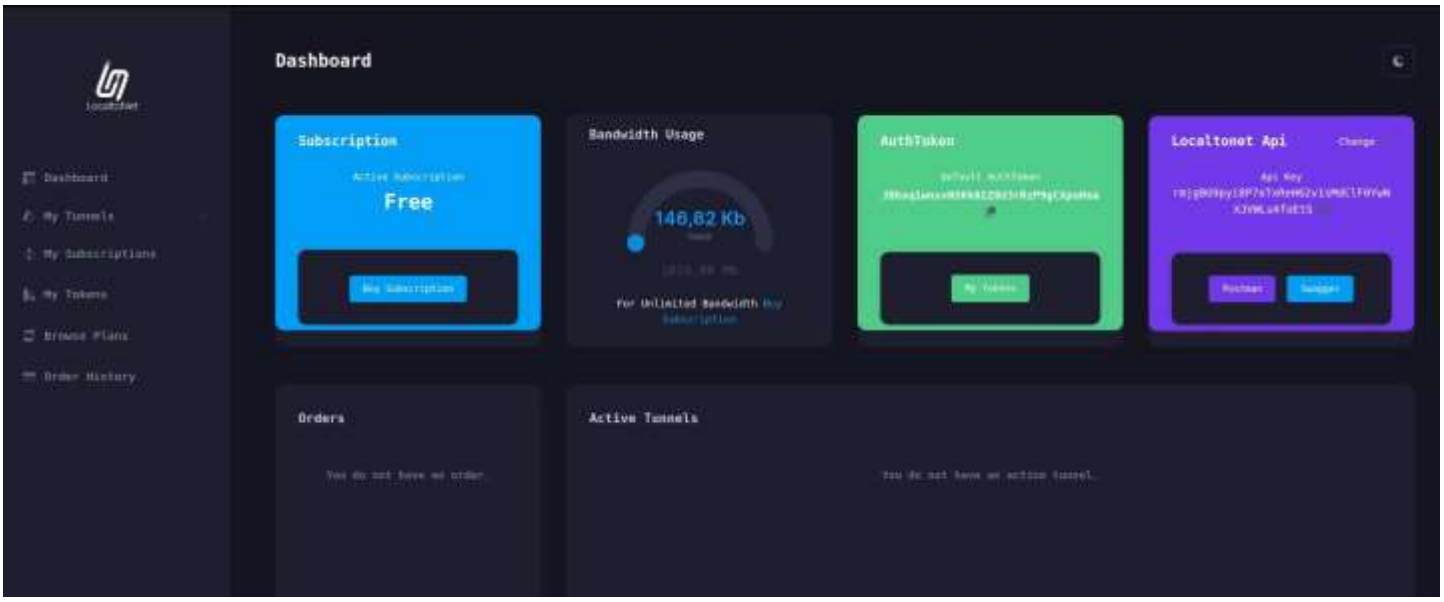


- Then click the PUT request button.
- Now, request a GET from the resource again. You should get the update value.
- Observe the above transactions with Wireshark.

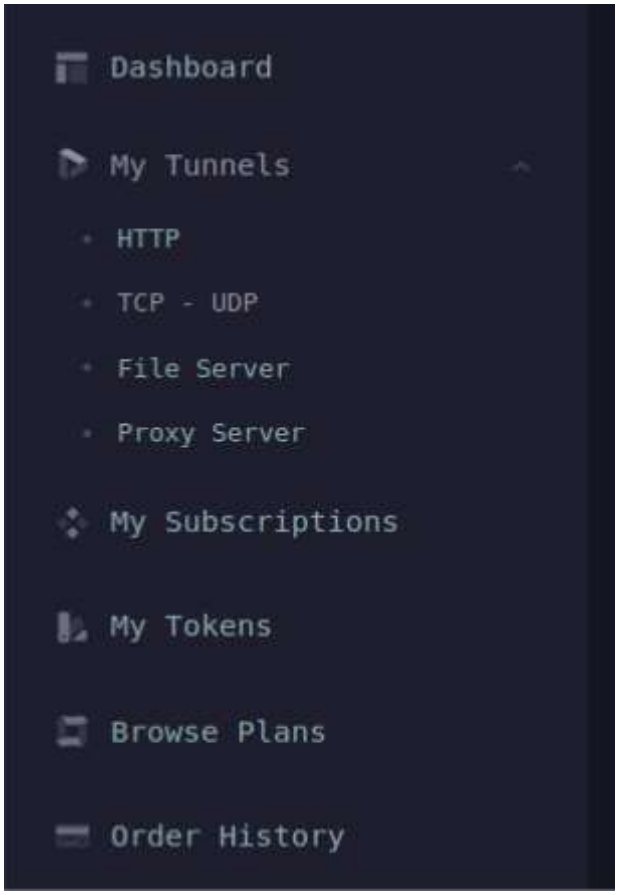
Task 4: Expose your CoAP server to Internet (optional)

There are very limited publicly accessible CoAP servers on the internet. For an individual who needs to test out their applications without having access to a VPS (virtual private server). You can either use coap.me but the functionality is very limited. So, you can expose your Californium server to the public internet.

1. Go to https://localtonet.com/ and create yourself an account.
2. Go to downloads and get yourself the relevant application for your OS.
3. After that Go the Dashboard

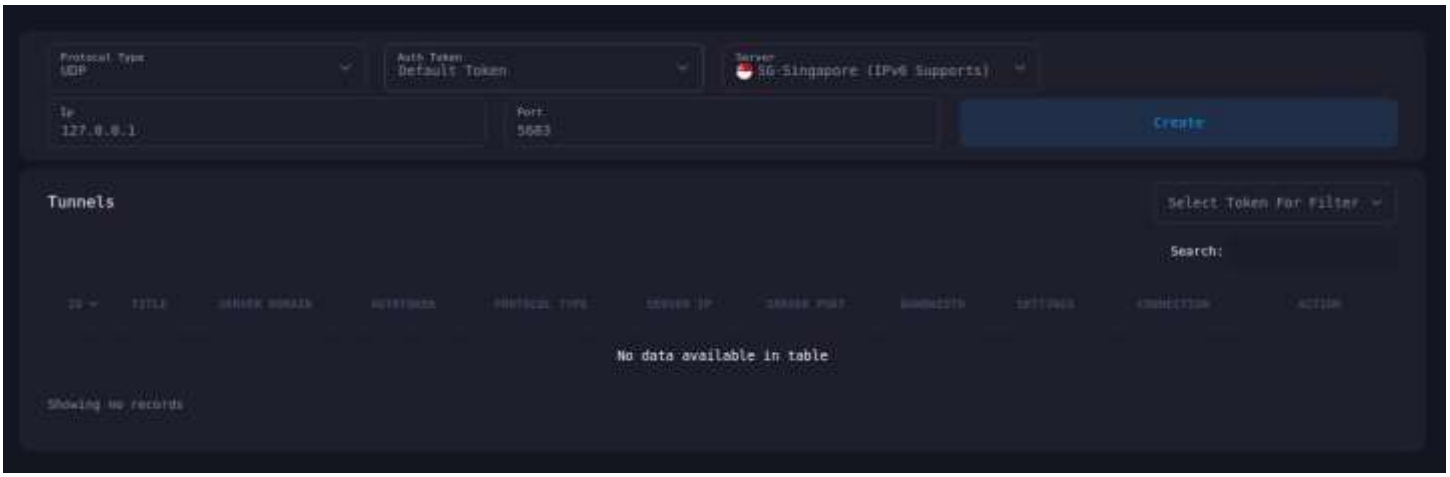


4. Go to the My Tunnels->TCP - UDP



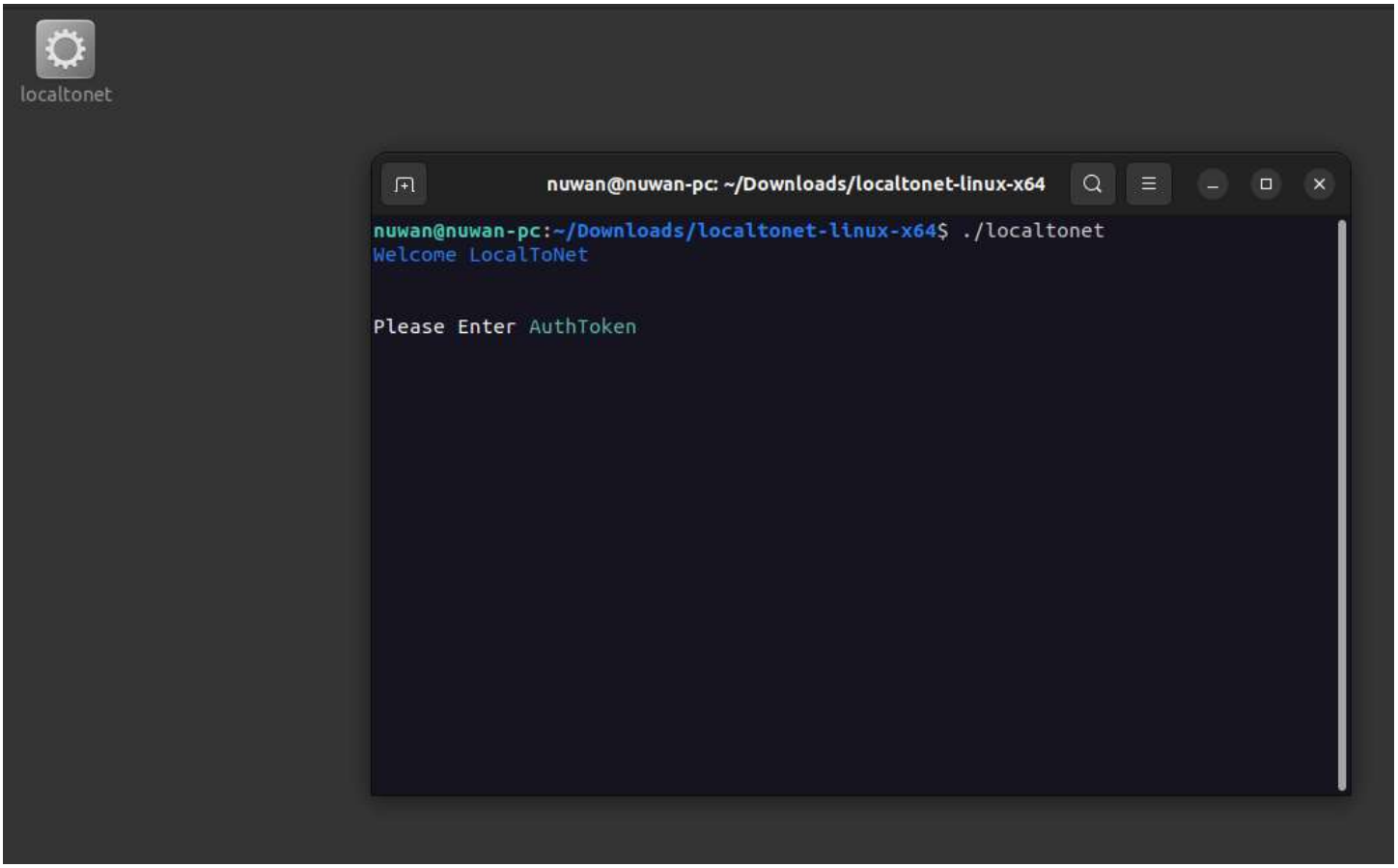
Since CoAP is powered by UDP, you need the UDP tunneling mode.

5. Enter the Port you need to expose and the localhost ip address, select the protocol to be UDP and server as the Singapore.



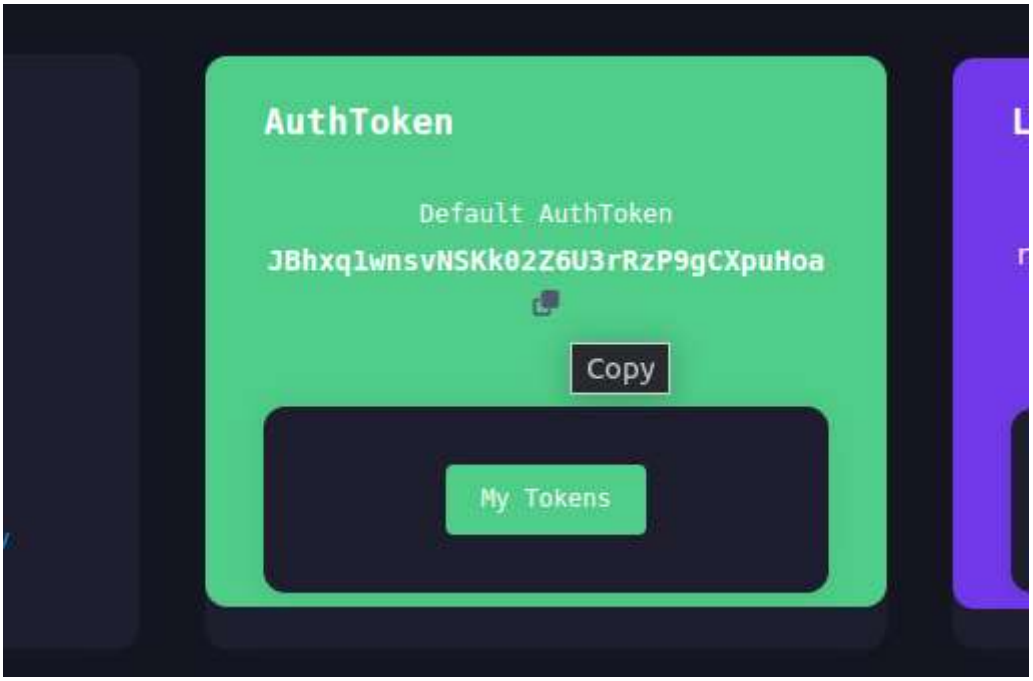
Press Create

6. Now go to the downloaded localtonet Client on your computer and run the file. Open it.

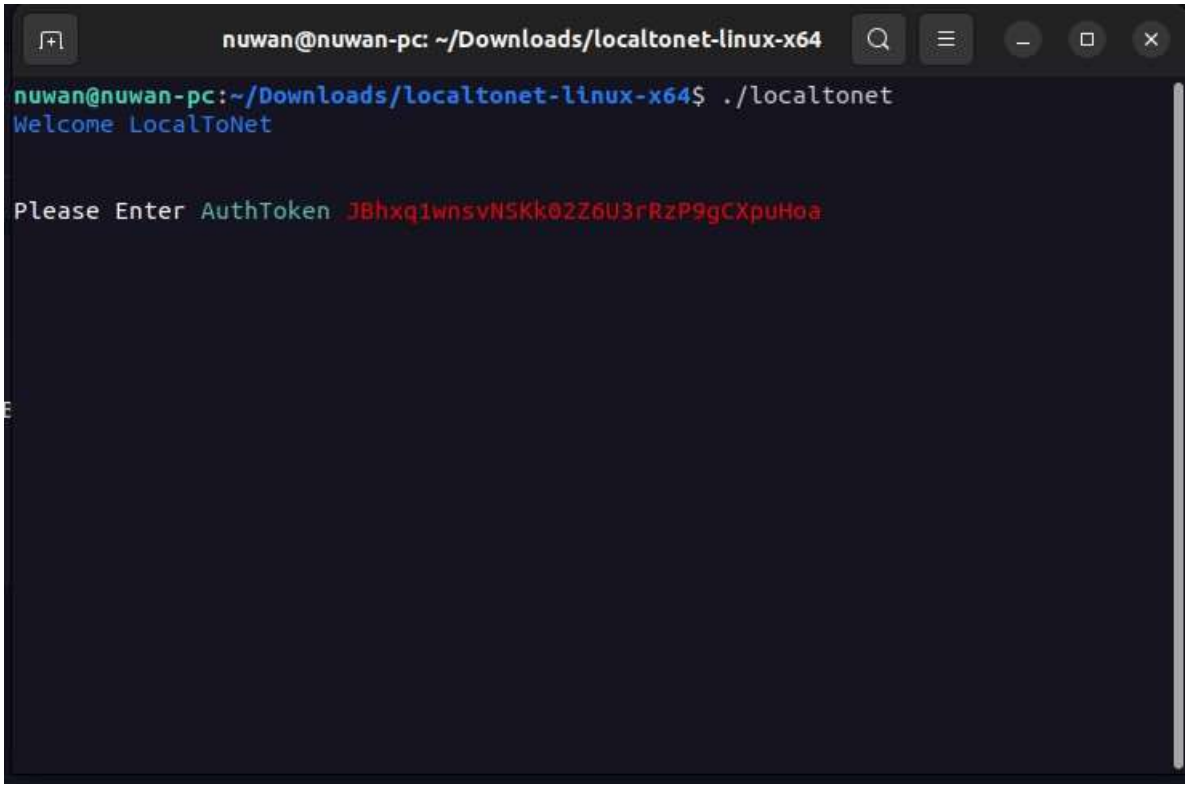


(For Ubuntu, you need to give execution access and run it using the terminal)

7. Copy the AuthToken from the website and paste it.



Provide it to the application.



Now you have established a link between the server and your pc.

8. Navigate to The Tunnels list and press Start.

Now your Localhost has a tunnel connectivity with the localtonet server and you can access it from the following url and the port

```
                    [ Session Status: Connected ]

    IP/Url        Protocol Type   Client IP   Client Port   Ping     Status
 sg4.localto.ne   UDP             127.0.0.1   5683          48        OK
    t:37236

 Token: Default                                              Plan: Free

                      Localtonet Version v3.7
```