# Postman Notes

Contents

[Postman Notes 1](#_Toc12454632)

[To find and create request for the existing APIs 1](#_Toc12454633)

[RoutePrefix 2](#_Toc12454634)

[How to Create an Environment 7](#_Toc12454635)

[Base\_URL and Different Environments 9](#_Toc12454636)

[Saving Collections to Source Control (in this case TFS via Visual Studio) 10](#_Toc12454637)

[Checking in Postman Environments and Collections 10](#_Toc12454638)

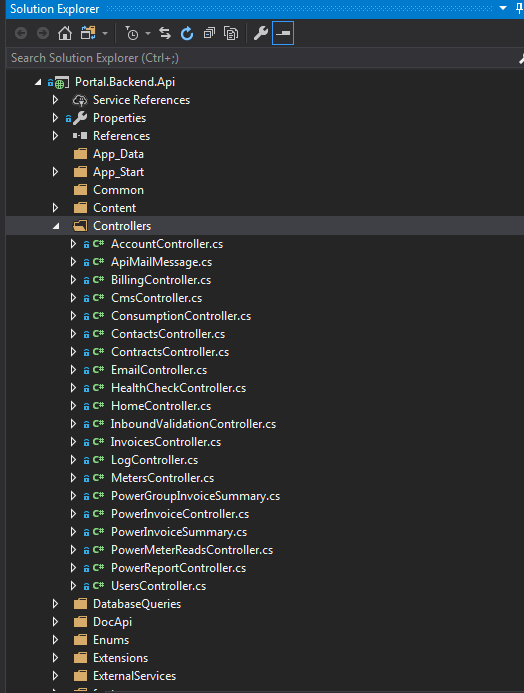
[To get Visual Studio to pick up the changes. 11](#_Toc12454639)

[Using Newman Command Line Interface (CLI) 11](#_Toc12454640)

## To find and create request for the existing APIs

In Visual Studio go to Portal.Backend.APi -> Controllers

The various APIs are within these folders



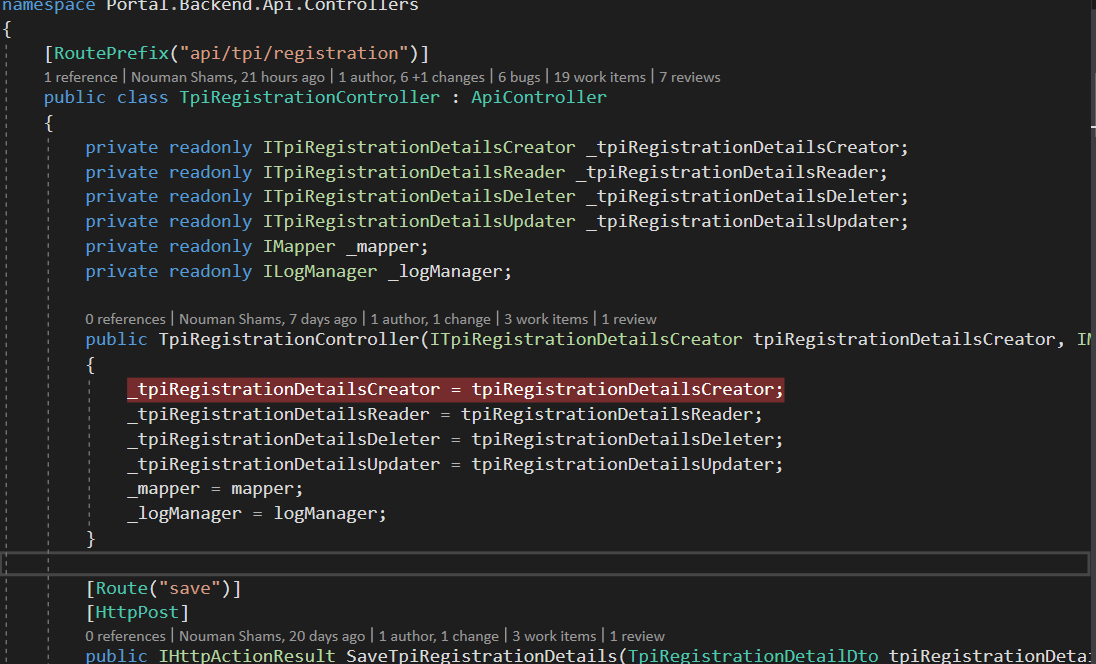
## RoutePrefix

The RoutePrefix will the part of the URL that comes after the server name (e.g. <http://porapp01qa:81>)

This will be followed by the Route for each service

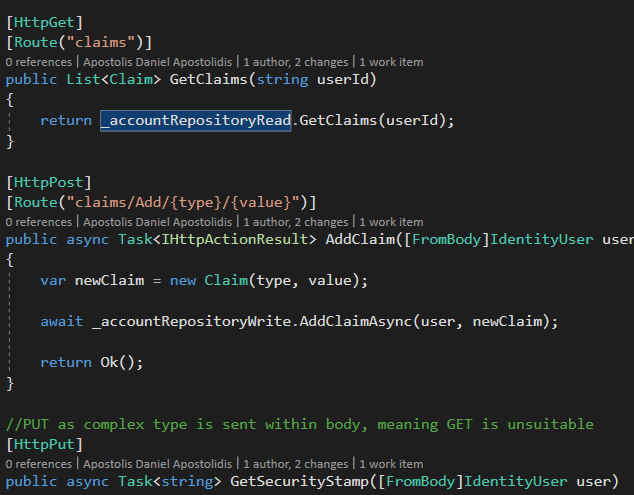
In this example, the URL for the Save service would be :

<http://porapp01qa:81/api/tpi/registration/save>



Double click on a folder to see the APIs.

API methods will have annotation of [HttpGet]. [HttpPost] etc.



The URL of the request will start with:

<http://porapp01qa:81/api/>

or

<http://porapp01dv:81/api/>

or

<http://localhost:60212/api/>

For QA, Dev and Local Host respectively

The next part will be the name of the method.

If the method has [Route] annotation, e.g. [Route("claims")], then that will replace method name in the URL as well

E.G.

[HttpGet]

public async Task<IdentityUser> Username(string username)

{

return await \_accountRepositoryRead.GetIdentityUserByUsernameAsync(username);

}

Would be:

<http://porapp01qa:81/api/Account/Username>

And

[HttpGet]

[Route("claims")]

public List<Claim> GetClaims(string userId)

{

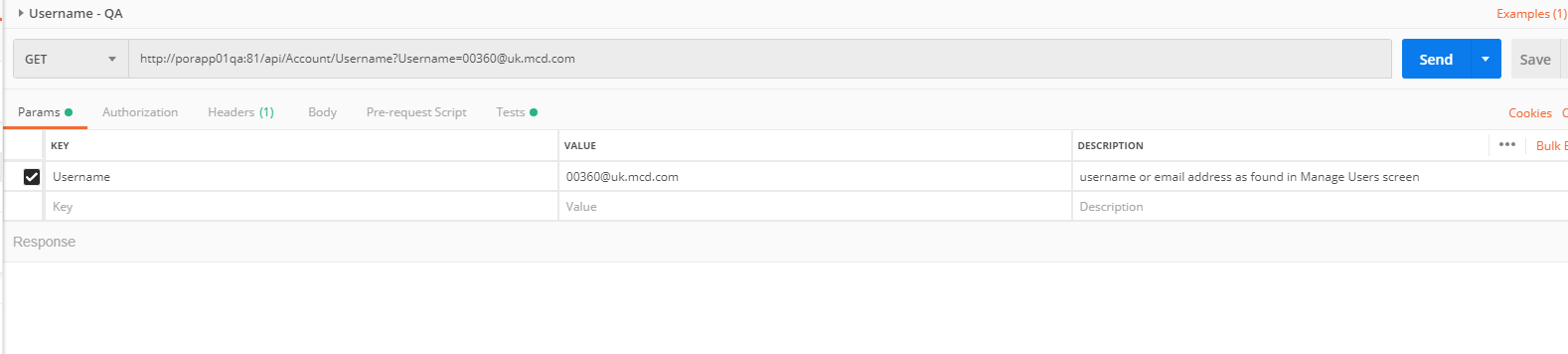
return \_accountRepositoryRead.GetClaims(userId);

}

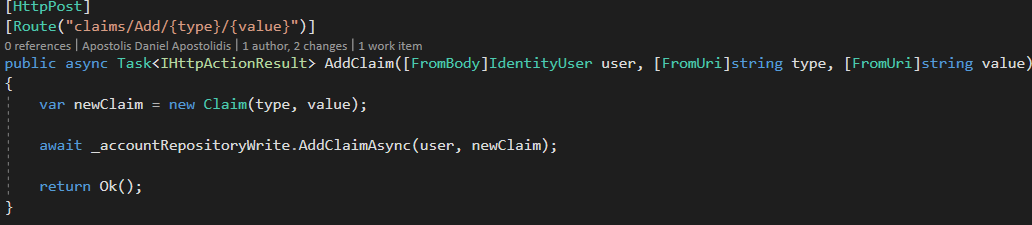
Would be

<http://porapp01qa:81/api/Account/claims>

You would add the parameters (e.g. username) via the params tab in Postman



If you have something like this:



The /{type}/{value} parameters are part of the URL.

The user, type and value parameters are added via params as above.

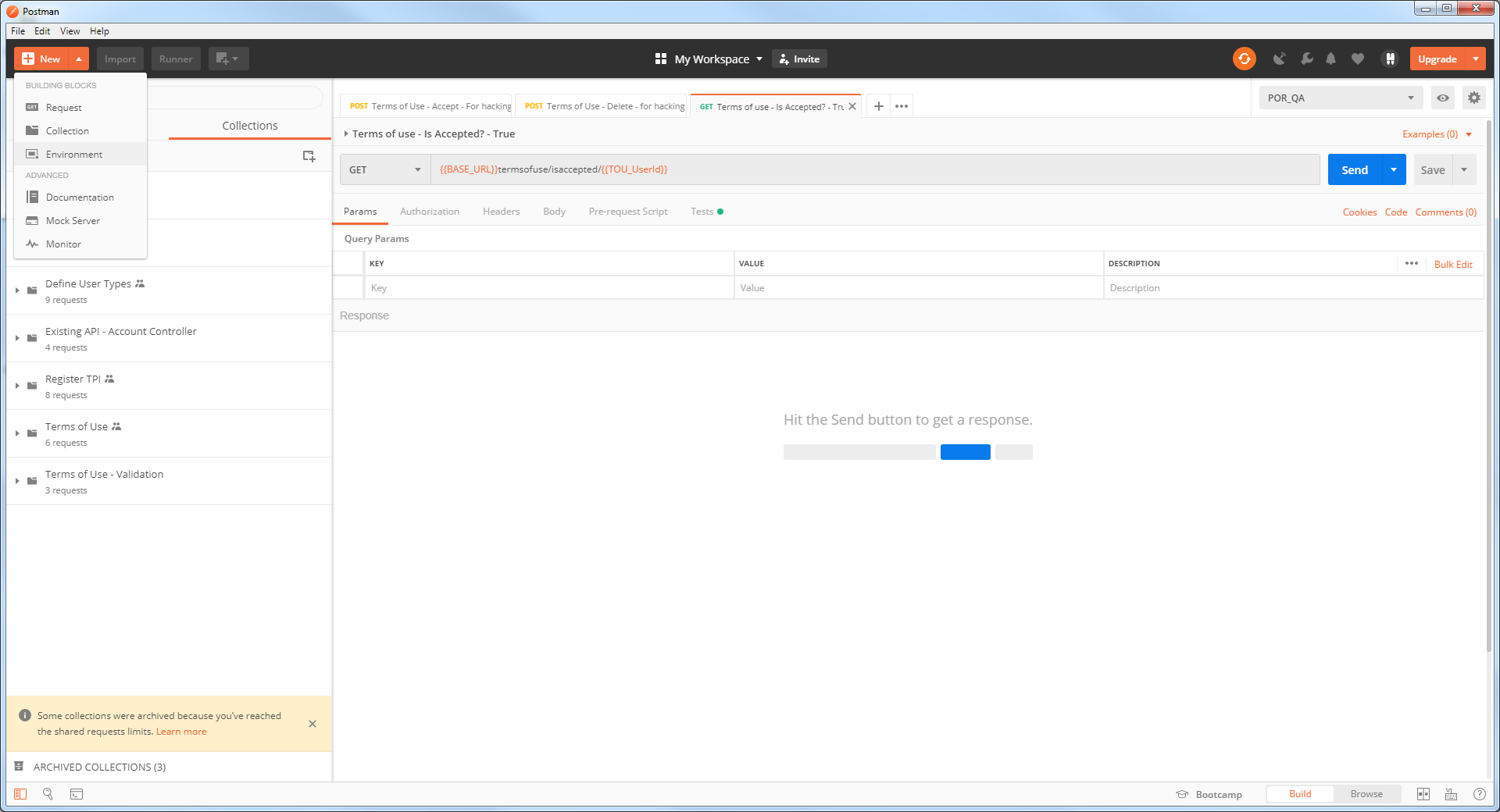
E.G.

<http://porapp01dv:81/api/account/claims/add/type1/value1>

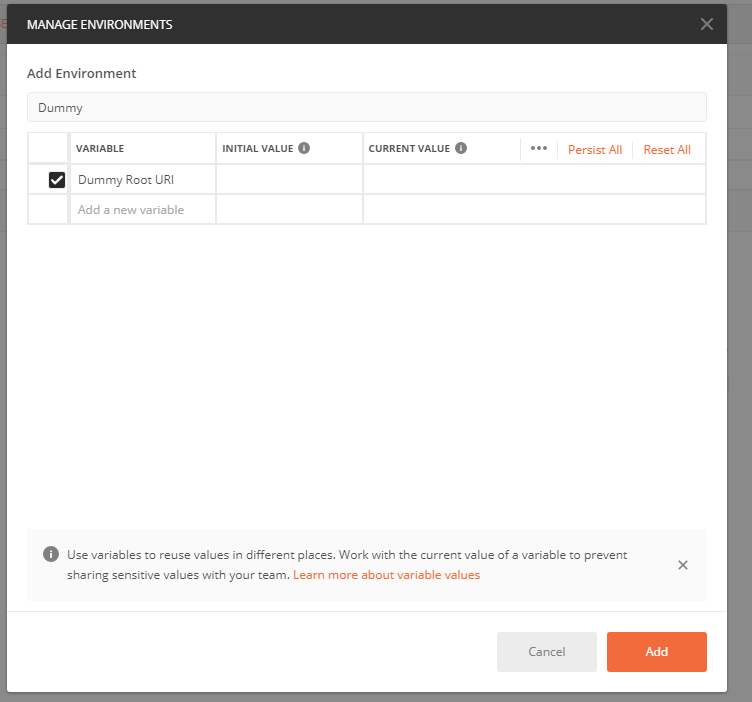
THIS IS CURRENTLY NOT WORKING. LOOK AT AGAIN LATER

## How to Create an Environment

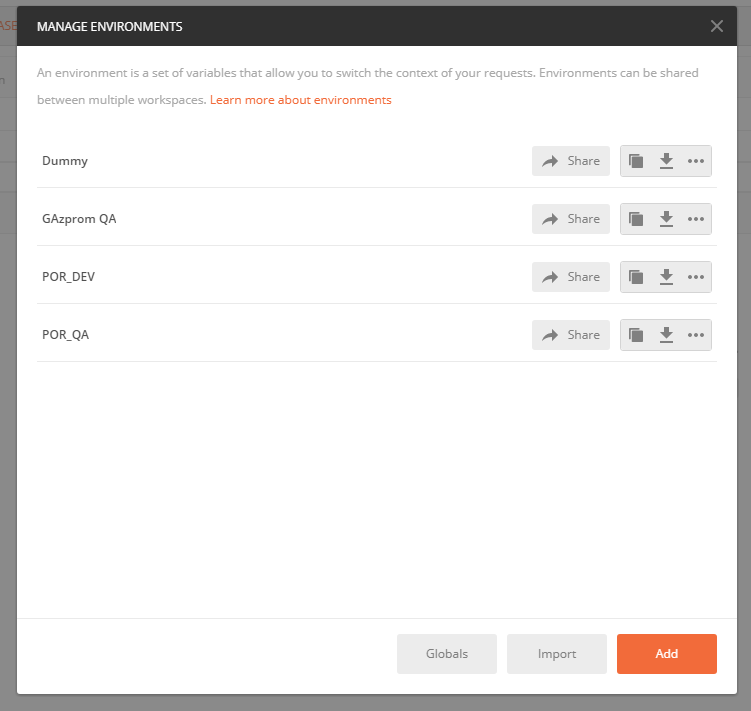
Select New Environment



Give the environment a name and set up any environment variables you need (although more can be added later)



Click on Add and the environment is created (and added to list of any other environments you may have)



Clicking on Share will allow you to share your environment (currently only available in your personal workspace) with a team workspace.

Clicking on Download will download your environment as a JSON file.

We currently save postman environments in this location:

$/UK/Portal/Main/Source/Portal.Backend.Api.PostmanTests/Environments

## Base\_URL and Different Environments

The URL for an API (e.g. termsofuse/isaccepted) will remain constant for each environment. However, the Base\_URL (e.g. http://porapp01qa:81/api/) will vary from one environment to the other.

Therefore, each environment should hold its own Base\_URL in an environment variable of the same name.

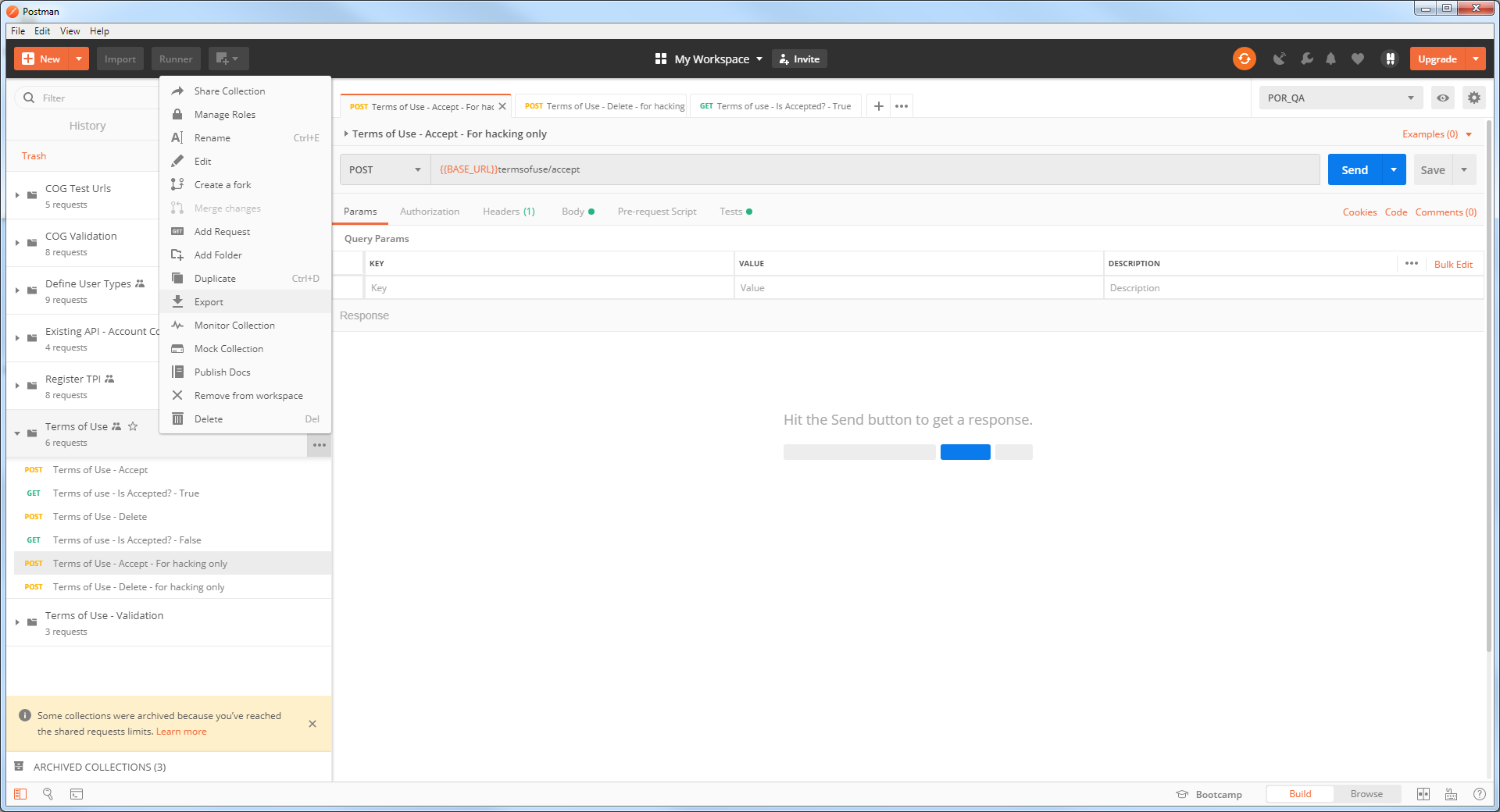
This environment is then used to create the full URL in the request.

E.G.



## Saving Collections to Source Control (in this case TFS via Visual Studio)

Click on the 3 dots beside the collection and selects Export



This will save your collection as a JSON file.

We currently save postman collections in this location:

$/UK/Portal/Main/Source/Portal.Backend.Api.PostmanTests

## Checking in Postman Environments and Collections

TRY THIS OUT / SEE TOLI ABOUT THIS.

The easiest way is to use Windows File Explorer.

Add your collections here. If you are just updating your collection, then overwrite the existing one here.

C:\Users\gmacdona\Source\Workspaces\UK\Portal\Main\Source\Portal.Backend.Api.PostmanTests

Add your environments here.

C:\Users\gmacdona\Source\Workspaces\UK\Portal\Main\Source\Portal.Backend.Api.PostmanTests\Environments

### To get Visual Studio to pick up the changes.

For files that you have changed in file explorer, it will alert you to this as soon as you click on the file in VS

For files that you have just added, click on the Show All Files icon at top of Solution Explorer. Right click on the file and select Include in Project. . It will now be shown in Pending Changes

For files you have deleted via File Explorer: find the file in Solution Explorer and delete it. It will be shown in Pending Changes with a line through it.

It will be shown in Pending Changes.

Then check in via Team Explorer -> Pending Changes as normal.

## Using Newman Command Line Interface (CLI)

Useful links:

Uploading files

<https://www.youtube.com/watch?v=c07IsbSNqfI&feature=youtu.be>

Example CLI command:

newman run NewmanFielsBase.json - d invoicesList.json -e POR\_QAenv.json

-d Specify a data file to use either json or csv

-e Specify a Postman environment as a JSON [file]

Got this working for files that are in source control with this CLI command:

newman run Portal.API.postman\_collection.json -d invoicesList.json -e \Users\gmacdona\Source\Workspaces\UK\Portal\Main\Source\Portal.Backend.Api.PostmanTests\Environments\POR\_QA.postman\_environment.json

Breaking it down to make it easier to read:

newman run Portal.API.postman\_collection.json

-d invoicesList.json

-e \Users\gmacdona\Source\Workspaces\UK\Portal\Main\Source\Portal.Backend.Api.PostmanTests\Environments\POR\_QA.postman\_environment.json

Only problem is that will iterate the WHOLE collection for each line in the input file, insted of just interating the finel request for each line.

Perhaps need to split into 2 collections?