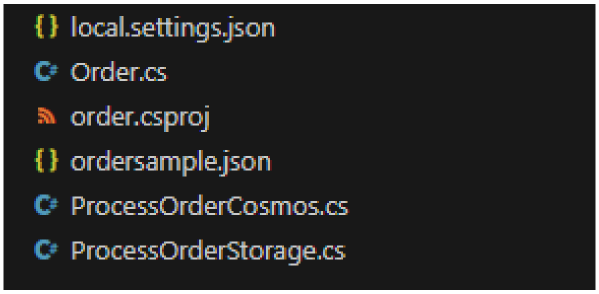
Azure section 16.01 - App function Hands on part 1

First step is to download azure function core tools from net and install it.

> 💡 Azure function core tools are the runtime environment of the app function. Without this we wont be able to run our code which is associated with the app functions.
> We will be needing this at the time of execution of the the code on the local machine.

1. Download the order module code from the repository. Extract it and open it in VS code.
2. Then for the app function we need to download the app extension in the VS code.
3. Once the extension is download we can see it in the azure section but under that we wont be able to see any working function because we haven't deployed any.
4. Next step is to extract the order code and paste it at the same location where we have pasted the catalog, inventory, weather module of the read it app.
5. Remember for the catalog and weather we have used VMs and then for the inventory we have used containers. For the order module we will be using app functions.
6. Once the order module is copied to the code section then we need to open it in the VS code and we need to run it locally.
7. For that we first need to download the postman tool. So now we will also download the postman tool.

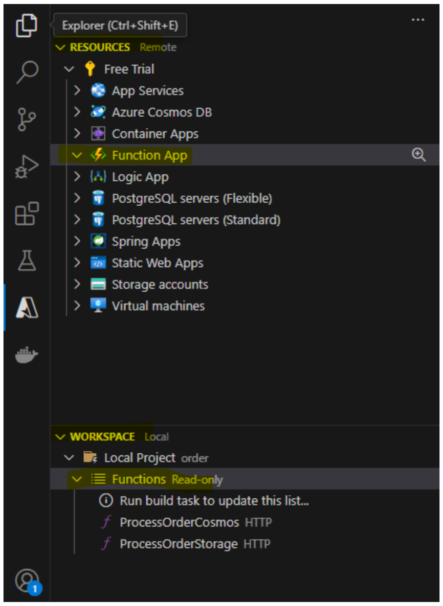
8. We will use the concept of trigger and binding in this order module to making it work. So whenever the user will pic an order then it will trigger a http event. This HTTP event will get stored in the storage class or cosmos DB. And we will connect the app service to the recourse so that the data read write flow is done.



local.settings.json
Order.cs
order.csproj
ordersample.json
ProcessOrderCosmos.cs
ProcessOrderStorage.cs

1. local settings is the configuration file of how the order module will behave.
2. Order.css is the css file of the order page and all the actions are mentioned in this file.
3. Order sample file contains the values of the sample order which we will be using for the testing purpose.

Process order cosmos/storage are the two different files which will get deployed at the app functions.



Once the extension is downloaded we can see the app function inside the resources tab.
But there is no any function app present.

This means that there is only the service available but there is nothing created which can consume this service of the app function.

Below the workspace is the local workspace.

It has both the files of the function which we are going to deploy on the app function.

We can see these two files is because it is present in the order module file which we fetch in the VS code at the start of the hands on.

If we see the both function code then we can figure three things:

1. the trigger is made with the HTTP URL.

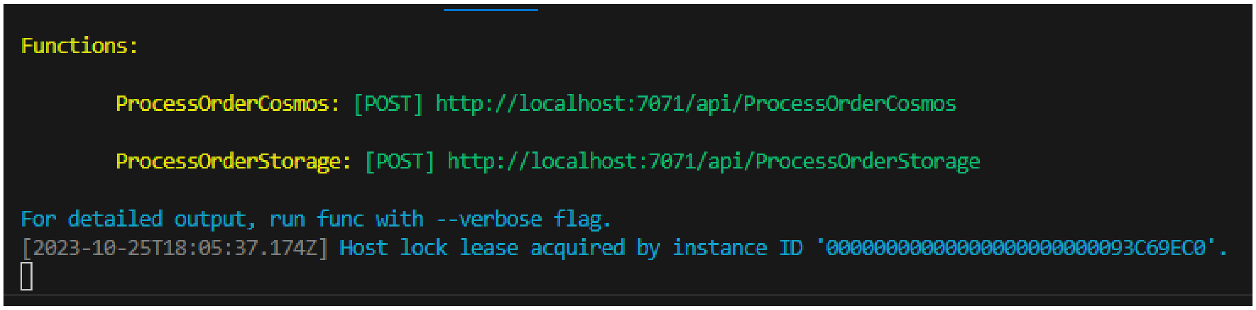
2. The order details is the string and it is stored in the container when the order is triggered with the HTTP request.
3. The container and the app function are binded together.

Hands on Step 2:

Now open the terminal in the Vs code and there we need to first execute the command **dotnet restore**

This command will just refresh all the dotnet extensions.

Now we will run the code with hitting f5 and after the code is build then we have two links in the terminal.



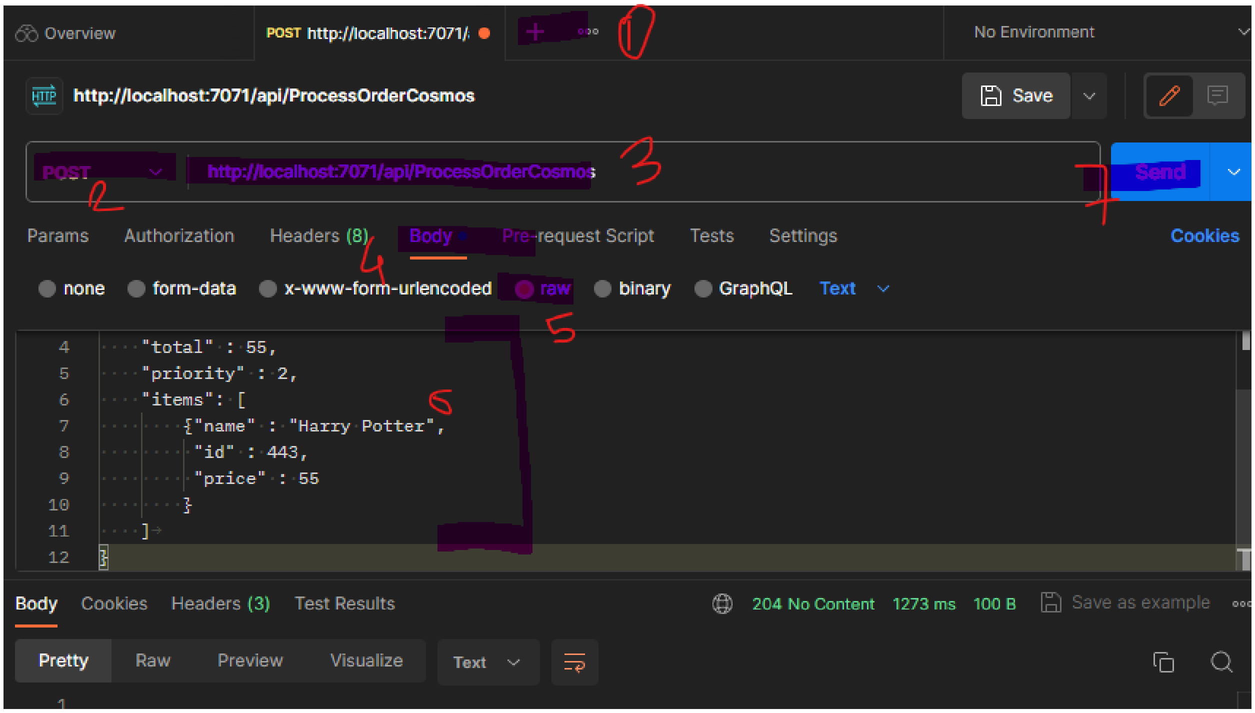
```
Functions:

        ProcessOrderCosmos: [POST] http://localhost:7071/api/ProcessOrderCosmos

        ProcessOrderStorage: [POST] http://localhost:7071/api/ProcessOrderStorage

For detailed output, run func with --verbose flag.
[2023-10-25T18:05:37.174Z] Host lock lease acquired by instance ID '00000000000000000000000093C69EC0'.
```

1. After the postman is installed we need to sign in to it and use the above mentioned link to establish connection.
2. So go to the postman and sign in. There we will be seeing a plus sign which will open a new page. There we need to enter the link which we got after running the code in the VScode.
3. Before pasting the link we need to choose the request type and set it to “POST ” request type.
4. Below that choose body and then click on raw. Copy the sample order details to the area under the raw and send the request.



http://localhost:7071/api/ProcessOrderCosmos

POST http://localhost:7071/api/ProcessOrderCosmos Send

Params | Authorization | Headers (8) | Body | Pre-request Script | Tests | Settings | Cookies

○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL Text

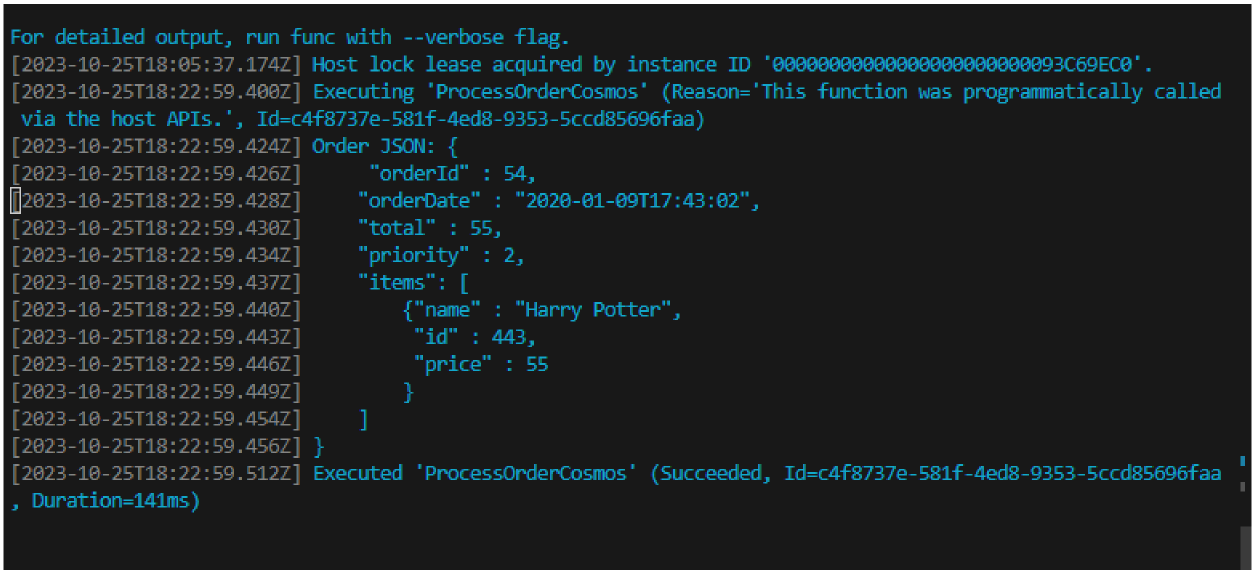
```
    "total" : 55,
    "priority" : 2,
    "items": [
        {"name" : "Harry Potter",
         "id" : 443,
         "price" : 55
        }
    ]
}
```

Body | Cookies | Headers (3) | Test Results — 204 No Content 1273 ms 100 B Save as example

Pretty | Raw | Preview | Visualize | Text

After this is done we can see a message on the post man page 204 no content in the green color. This means that the request is sent from our end **successfully** but there is no response in return.

If we go to the VS code then we can see that there is the request receive log in the terminal.



```
For detailed output, run func with --verbose flag.
[2023-10-25T18:05:37.174Z] Host lock lease acquired by instance ID '00000000000000000000000093C69EC0'.
[2023-10-25T18:22:59.400Z] Executing 'ProcessOrderCosmos' (Reason='This function was programmatically called
 via the host APIs.', Id=c4f8737e-581f-4ed8-9353-5ccd85696faa)
[2023-10-25T18:22:59.424Z] Order JSON: {
[2023-10-25T18:22:59.426Z]      "orderId" : 54,
[2023-10-25T18:22:59.428Z]     "orderDate" : "2020-01-09T17:43:02",
[2023-10-25T18:22:59.430Z]     "total" : 55,
[2023-10-25T18:22:59.434Z]     "priority" : 2,
[2023-10-25T18:22:59.437Z]     "items": [
[2023-10-25T18:22:59.440Z]         {"name" : "Harry Potter",
[2023-10-25T18:22:59.443Z]          "id" : 443,
[2023-10-25T18:22:59.446Z]          "price" : 55
[2023-10-25T18:22:59.449Z]         }
[2023-10-25T18:22:59.454Z]     ]
[2023-10-25T18:22:59.456Z] }
[2023-10-25T18:22:59.512Z] Executed 'ProcessOrderCosmos' (Succeeded, Id=c4f8737e-581f-4ed8-9353-5ccd85696faa
, Duration=141ms)
```

Here we have successfully ran the code on the local machine. Now in next part we will be deploying it to the app functions.