DOTNET CORE-AZURE MINI PROJECT

Create a **Web API Project** to store Product Information. Use Entity Framework to store the product information in the database. The user should be able to perform all the CRUD Operations. Configure **GET, POST, PUT and DELETE**.

The Product Entity should have the following properties:

* ProductID
* ProductName
* Price
* Brand
* ManufactureDate
* ExpirationDate

Use Data Annotations to

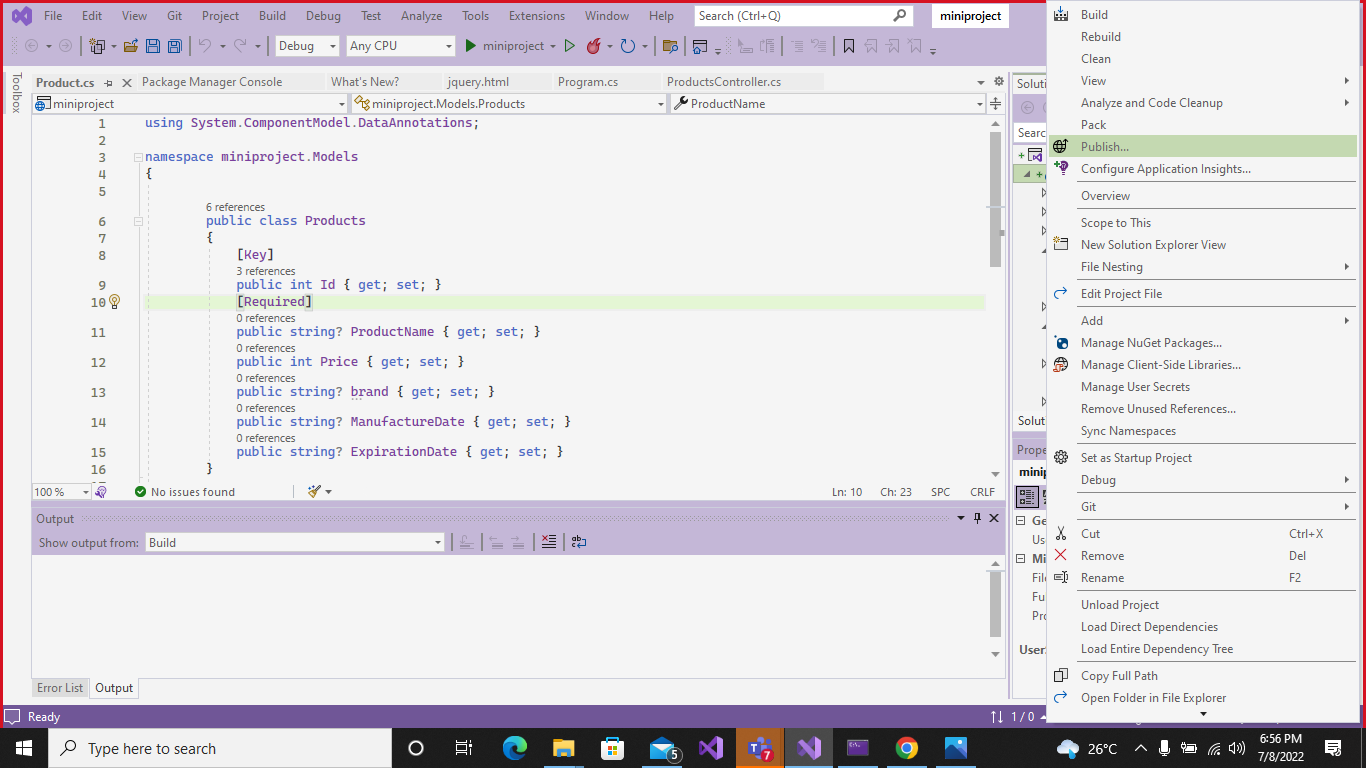
* Mark the Primary Key
* Make ProductName Mandatory
* Make Price a Number

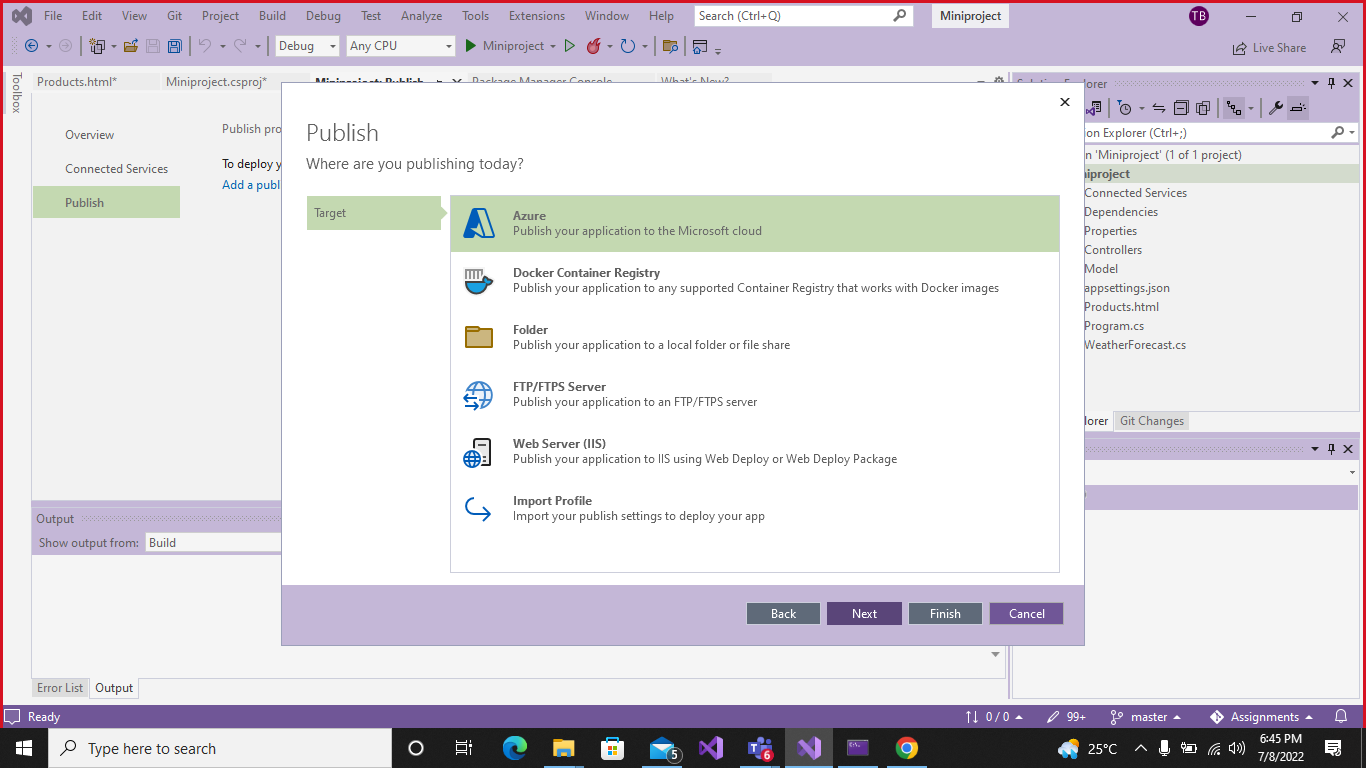
Create a JQuery and AJAX Client to consume the Web API and show the result.

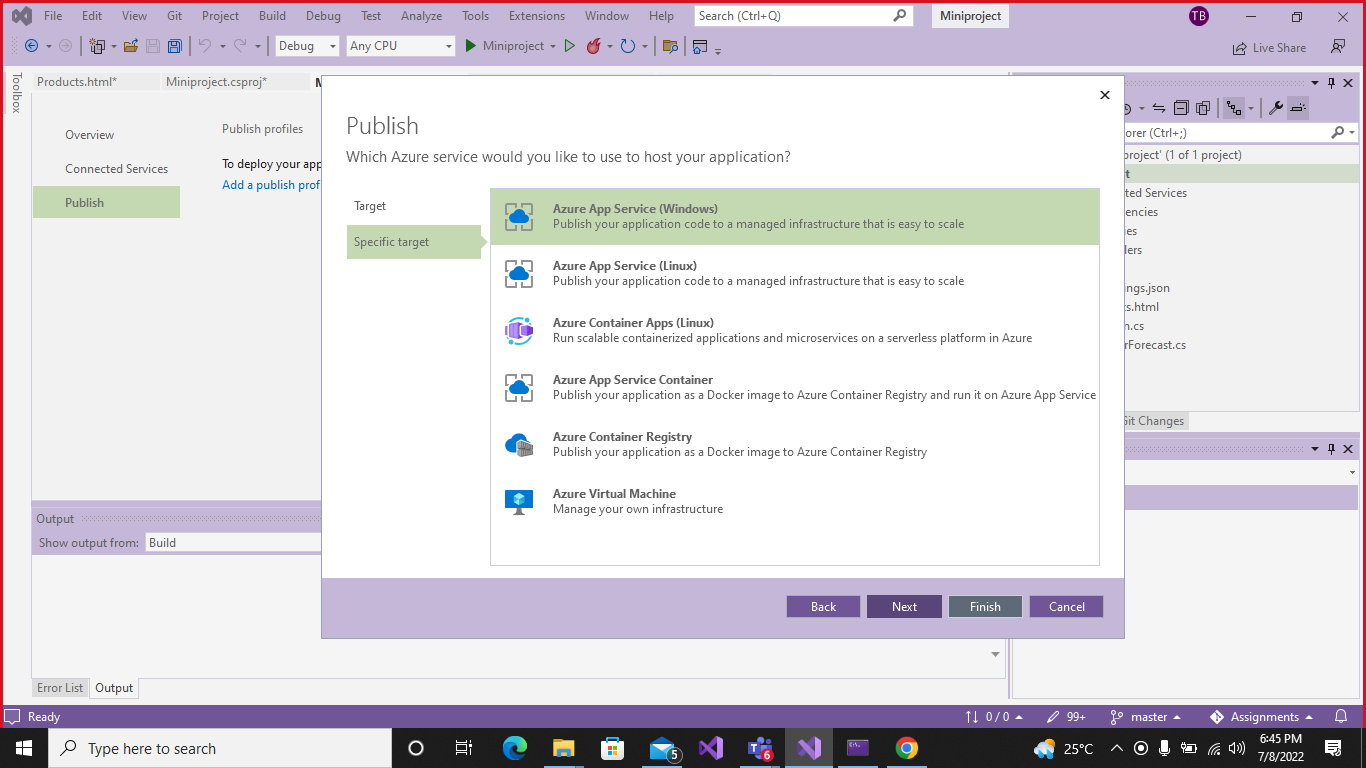
**Azure Hosting:**

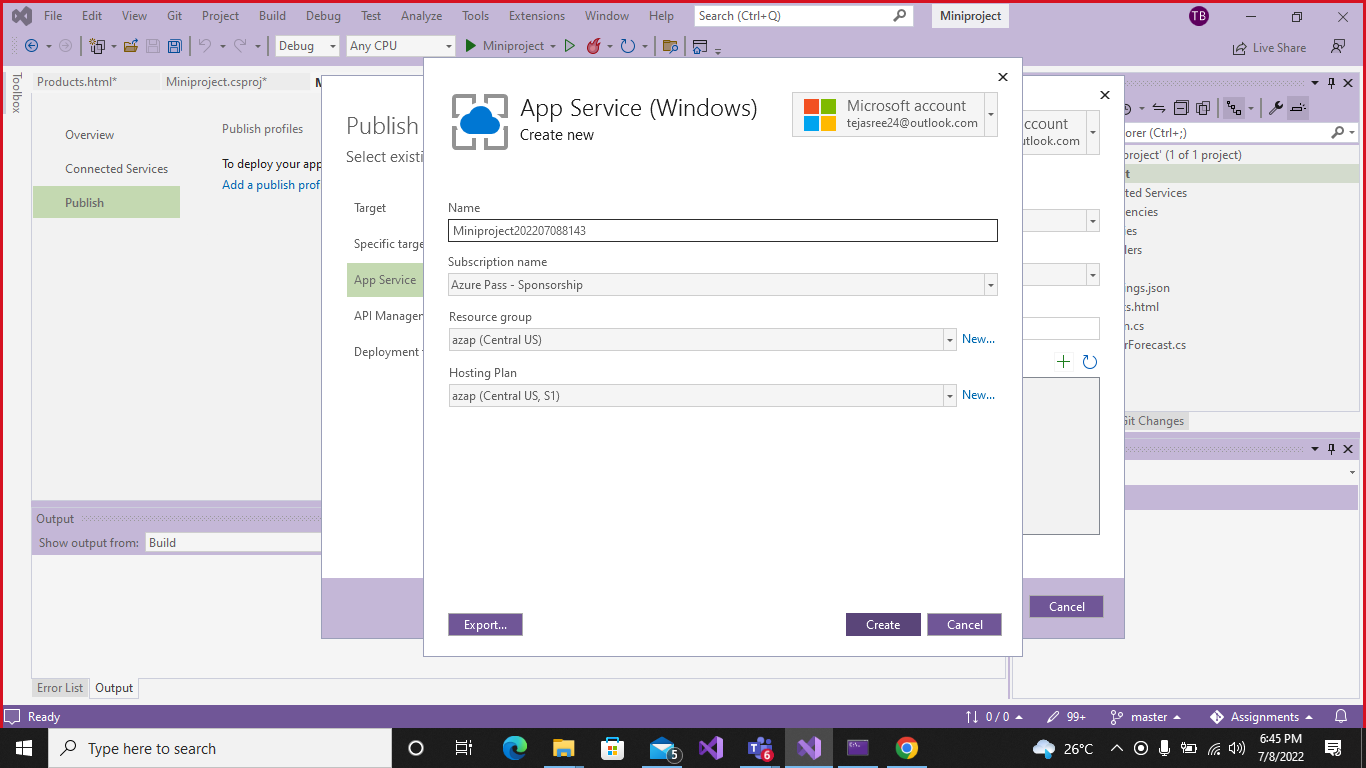
* Host the web api in azure and consume the same using JQuery Client.
* Configure Scale out by adding rules for custom scaling.
* Configure Deployment slots for staging and production.
* Configure Application Insights for the project.
* Configure Swagger for the api.
* Work with Log Analytics with the sample logs available.

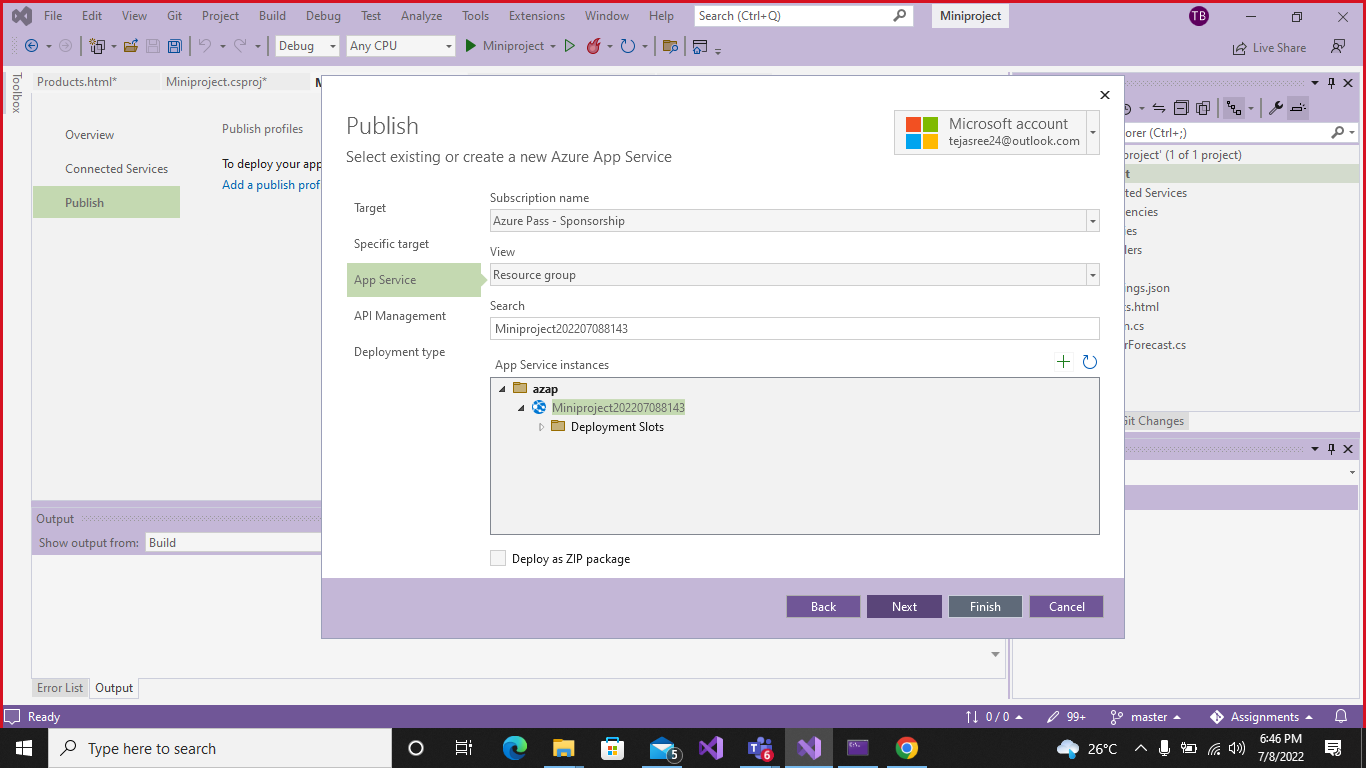
**1.Host the web api in azure and consume the same using JQuery Client.**

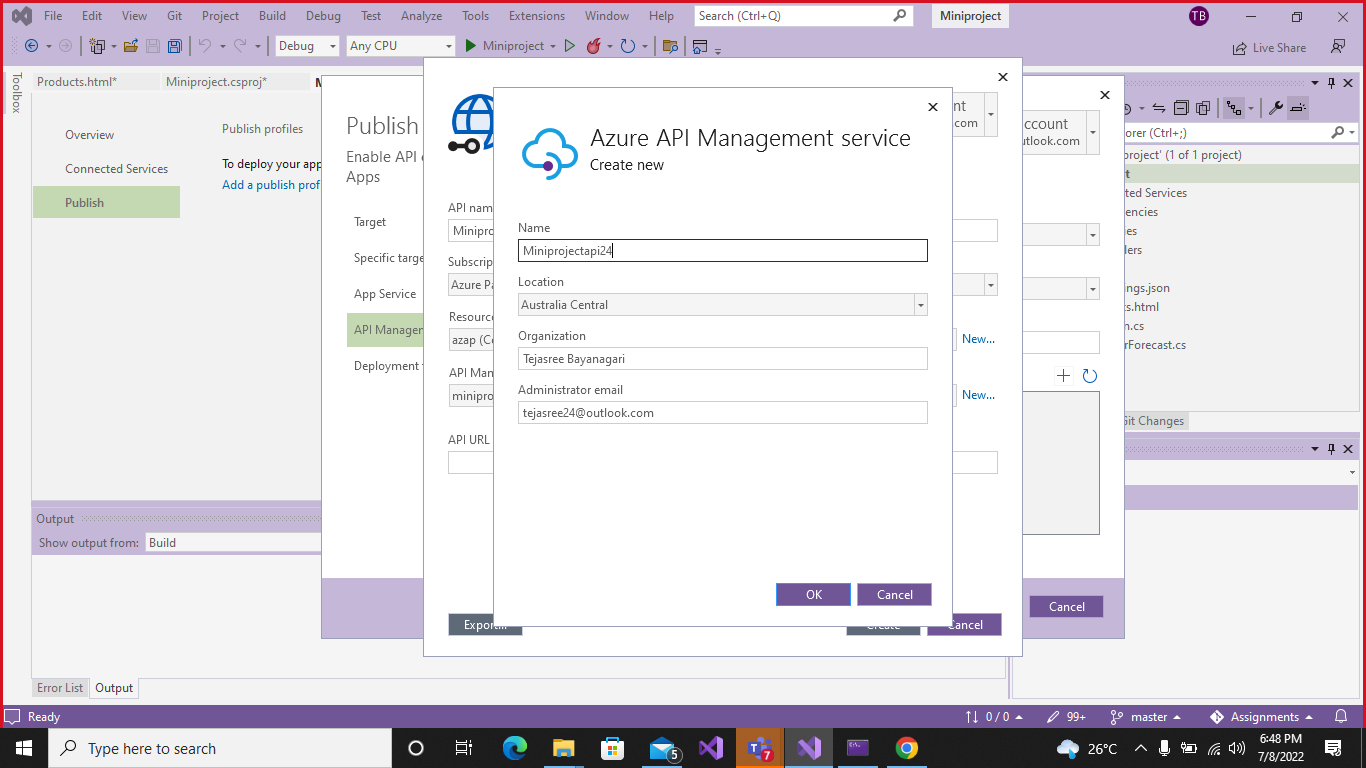
 Give a right click on project and publish

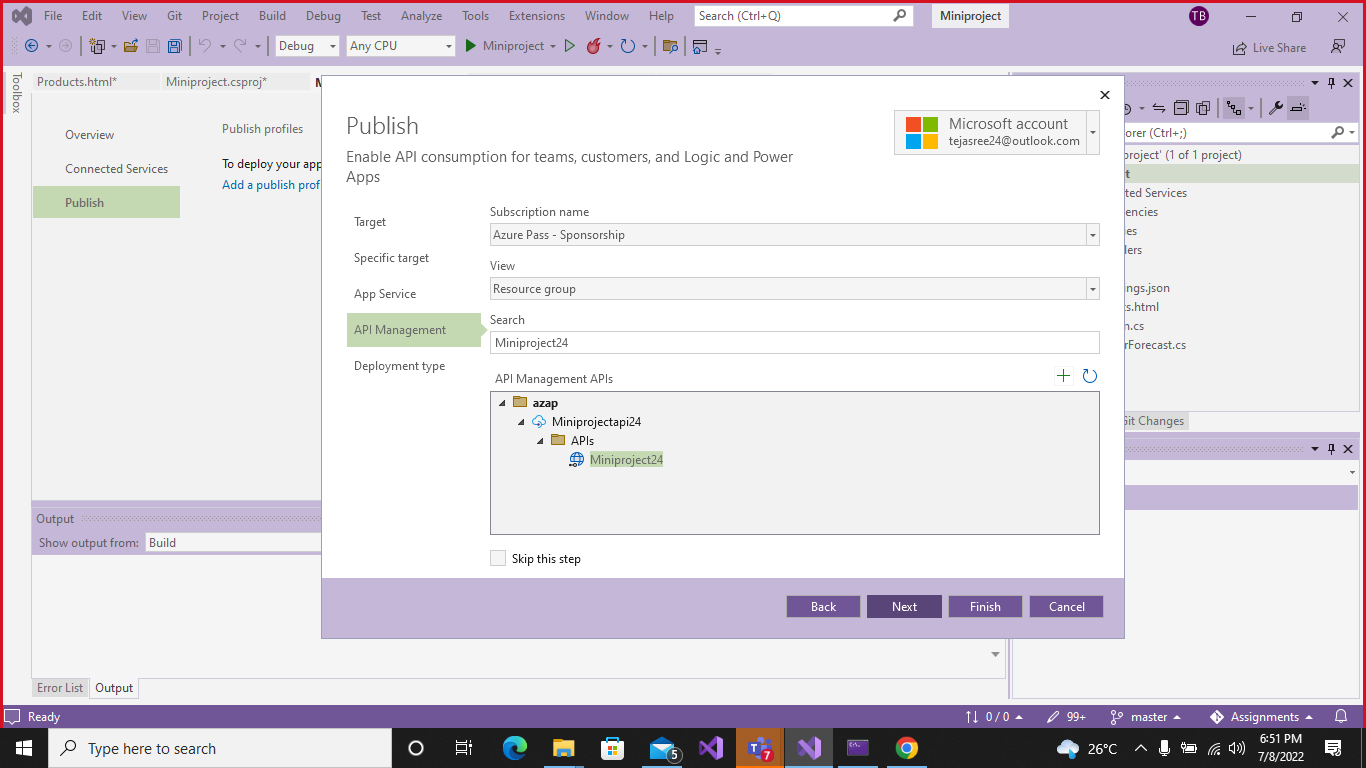
Select Azure and click on next.

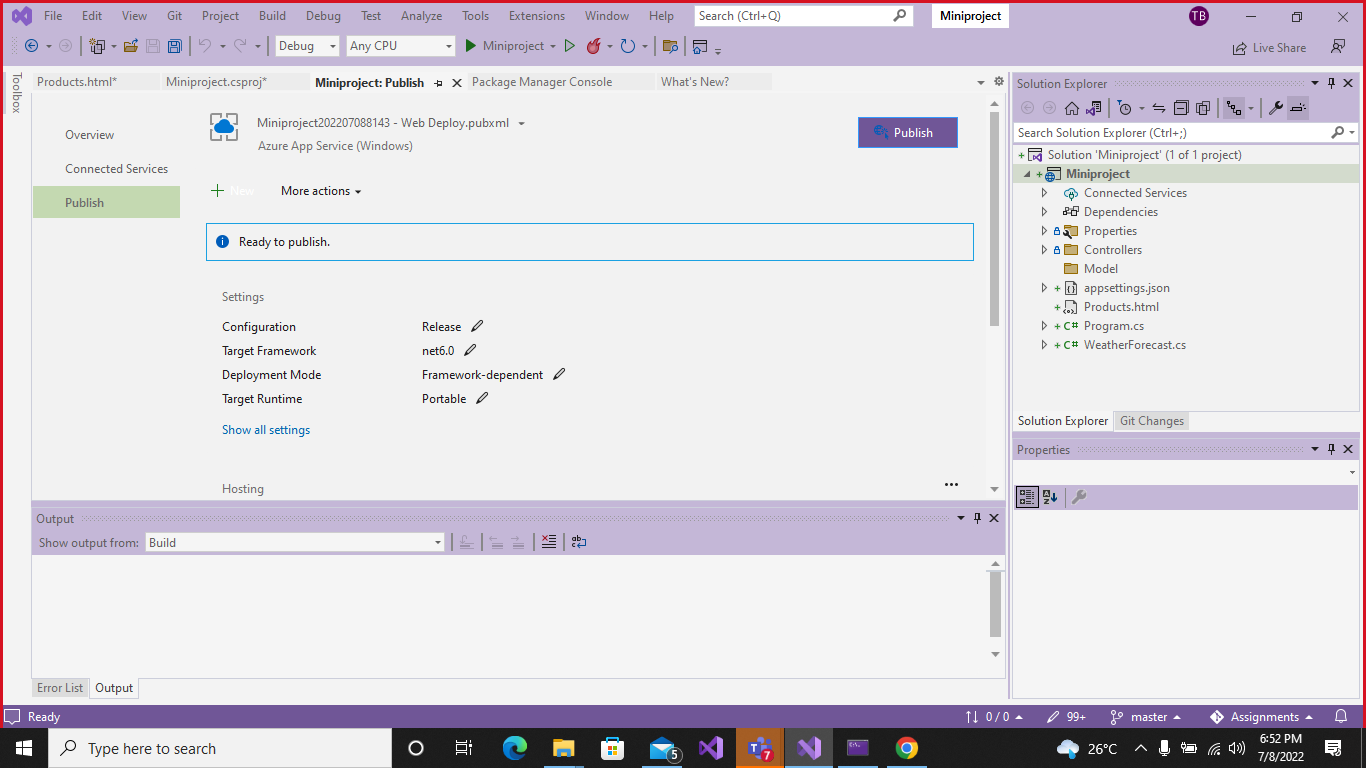
Select Azure App Service (windows) and click on next.

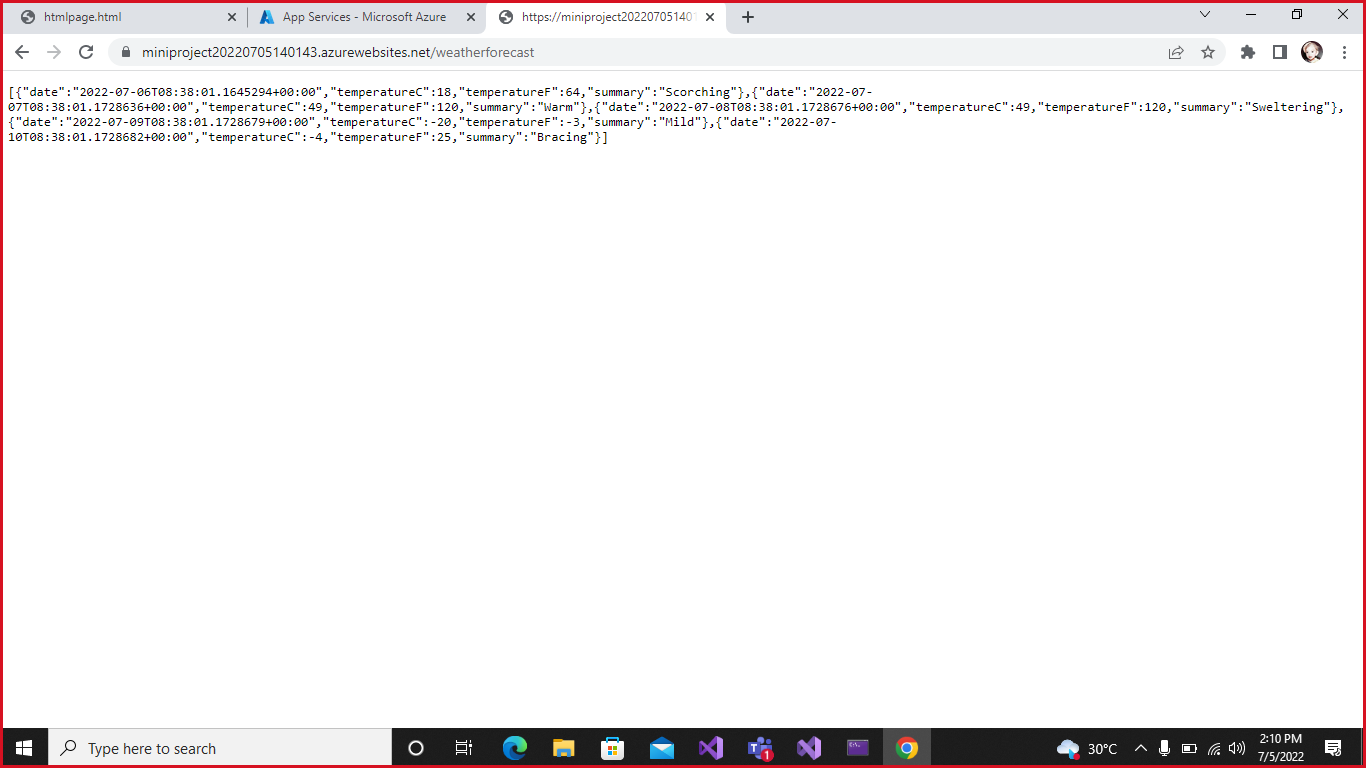
Enter app name, resource group and app service and click on create.

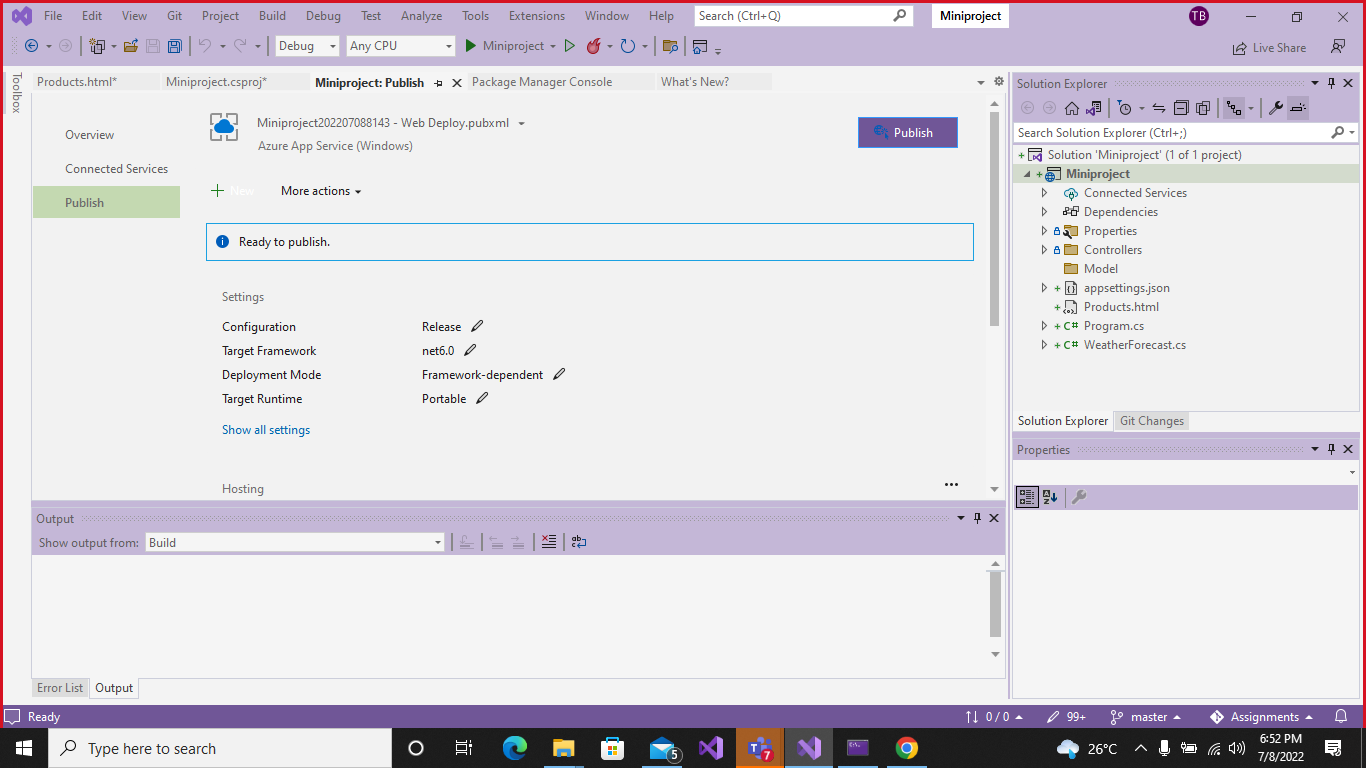
Select created instance project.

Create the azure API management service.

Select the API management APIs instance.

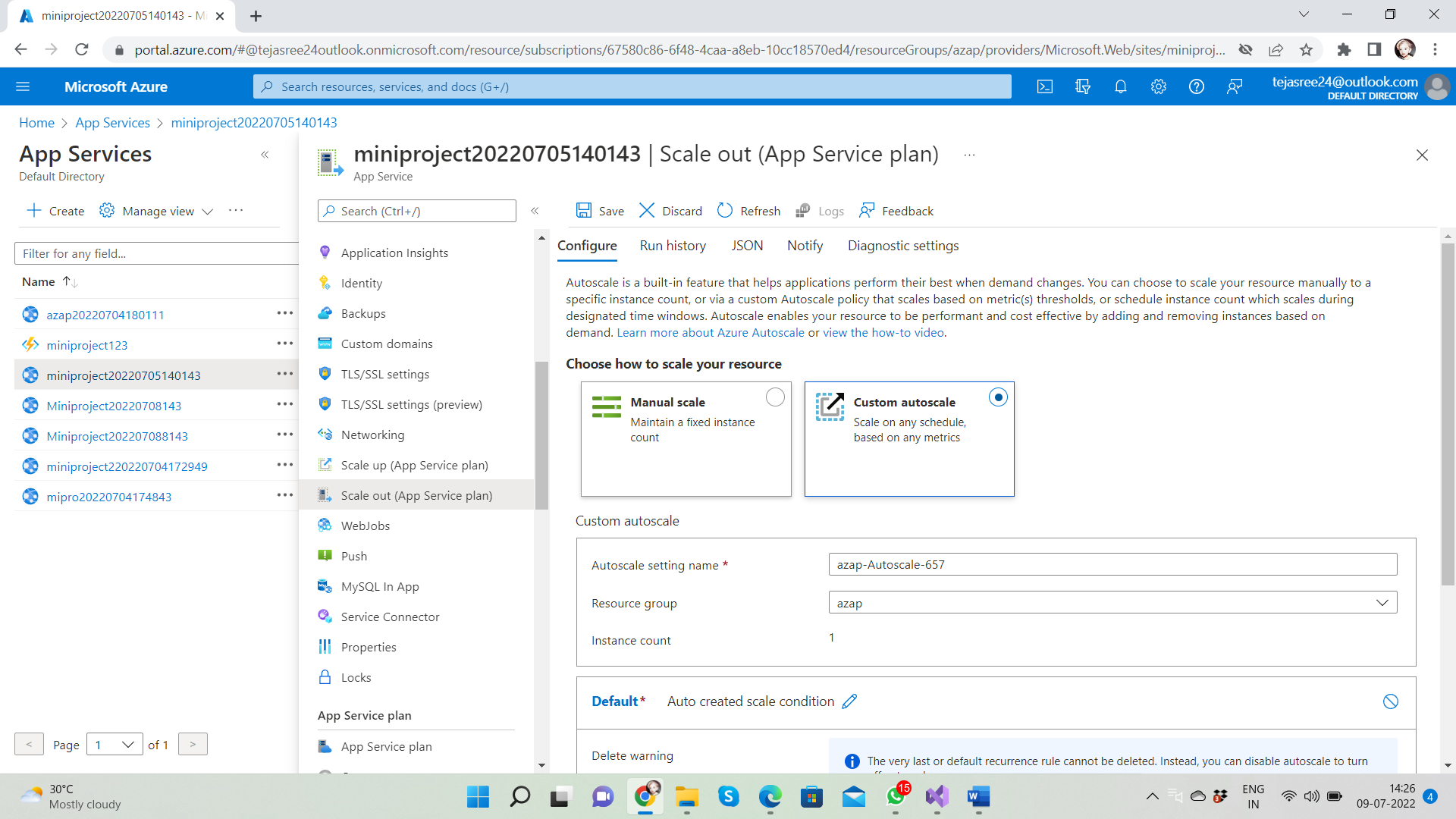
Click on the publish.

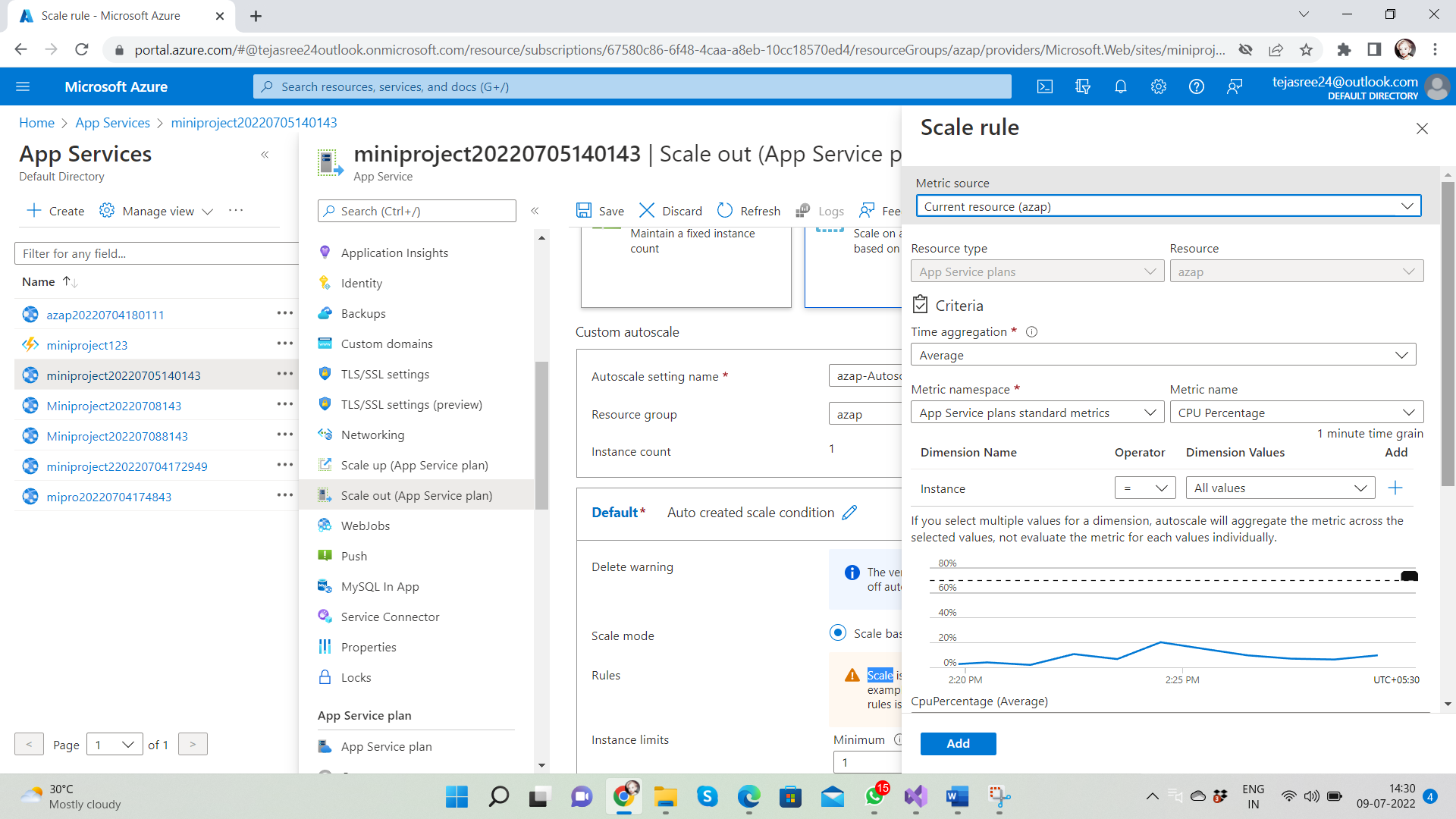
The web API will publish and the output open in web browser.

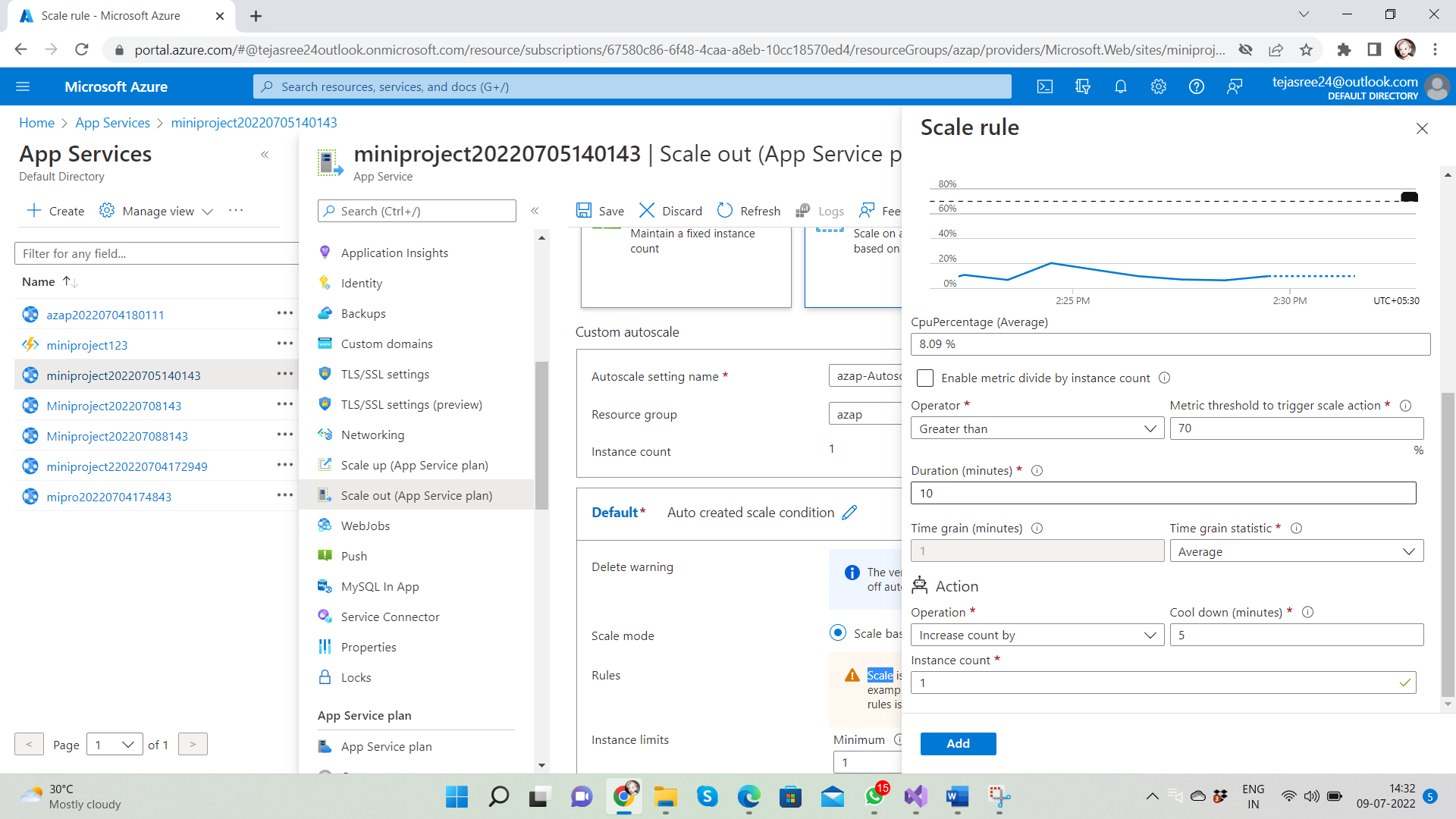
Now click on the Ready to Publish.

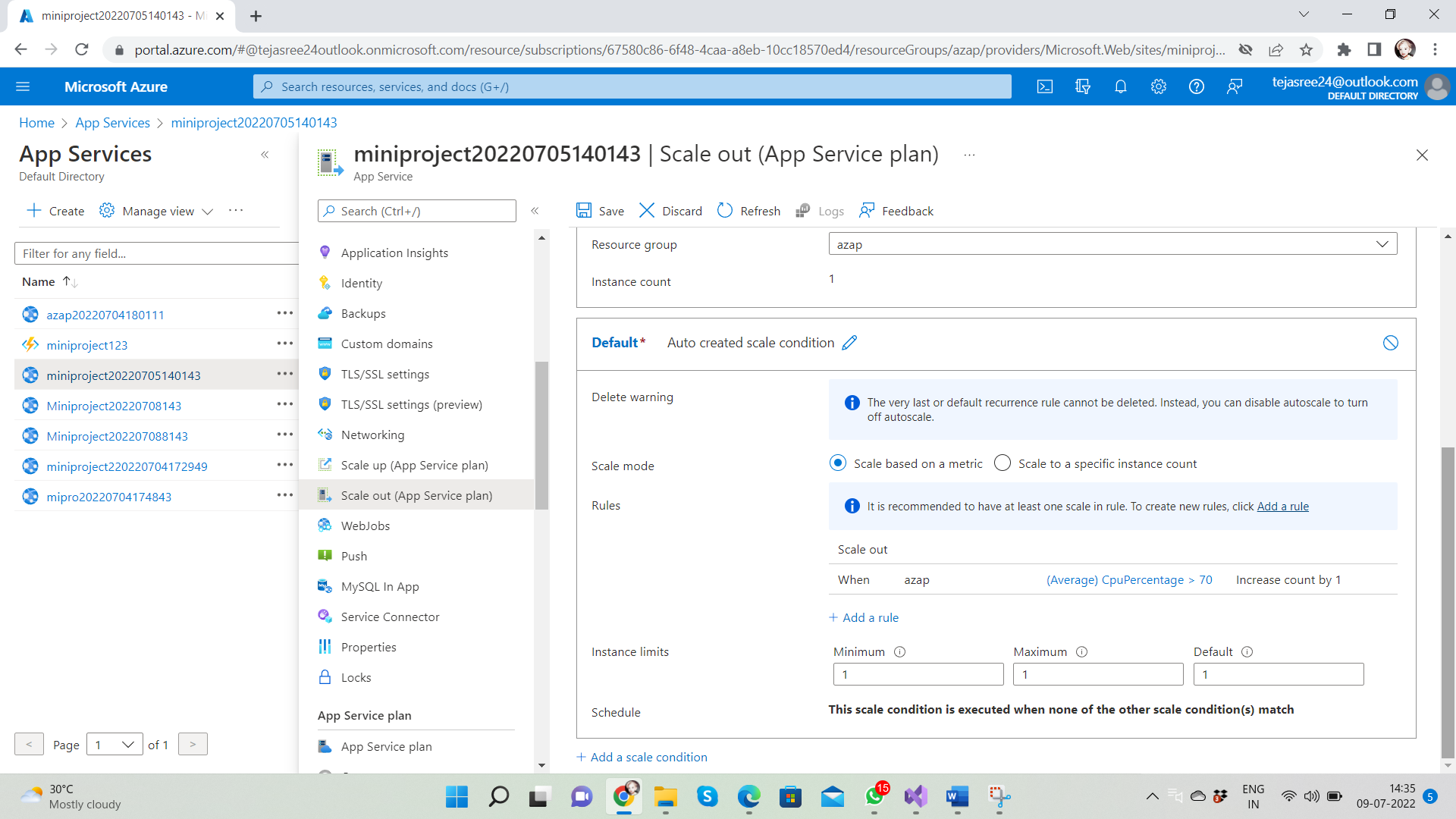
**2.Configure Scale out by adding rules for custom scaling.**

Switch to Azure portal, Select the API we created and following step to scale out.

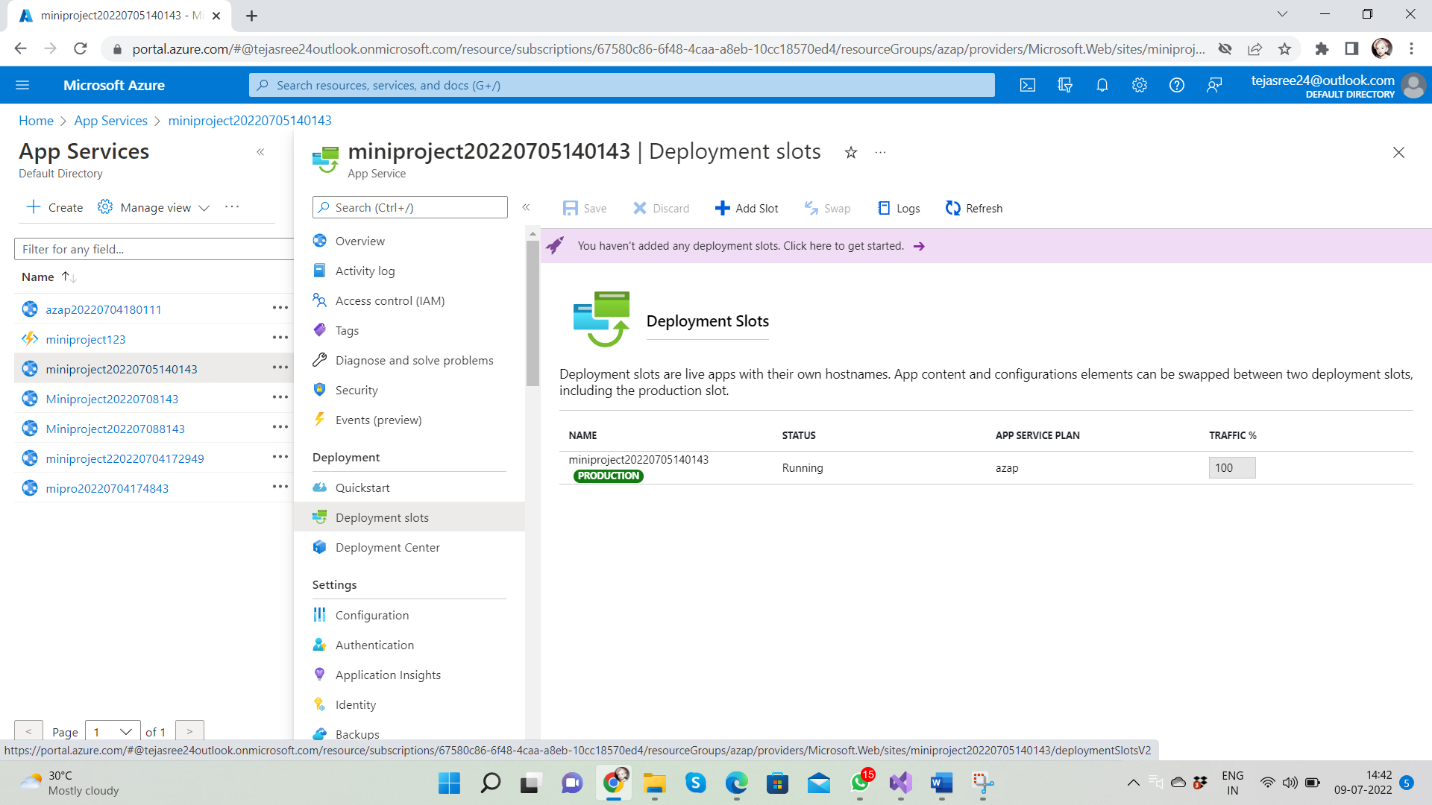
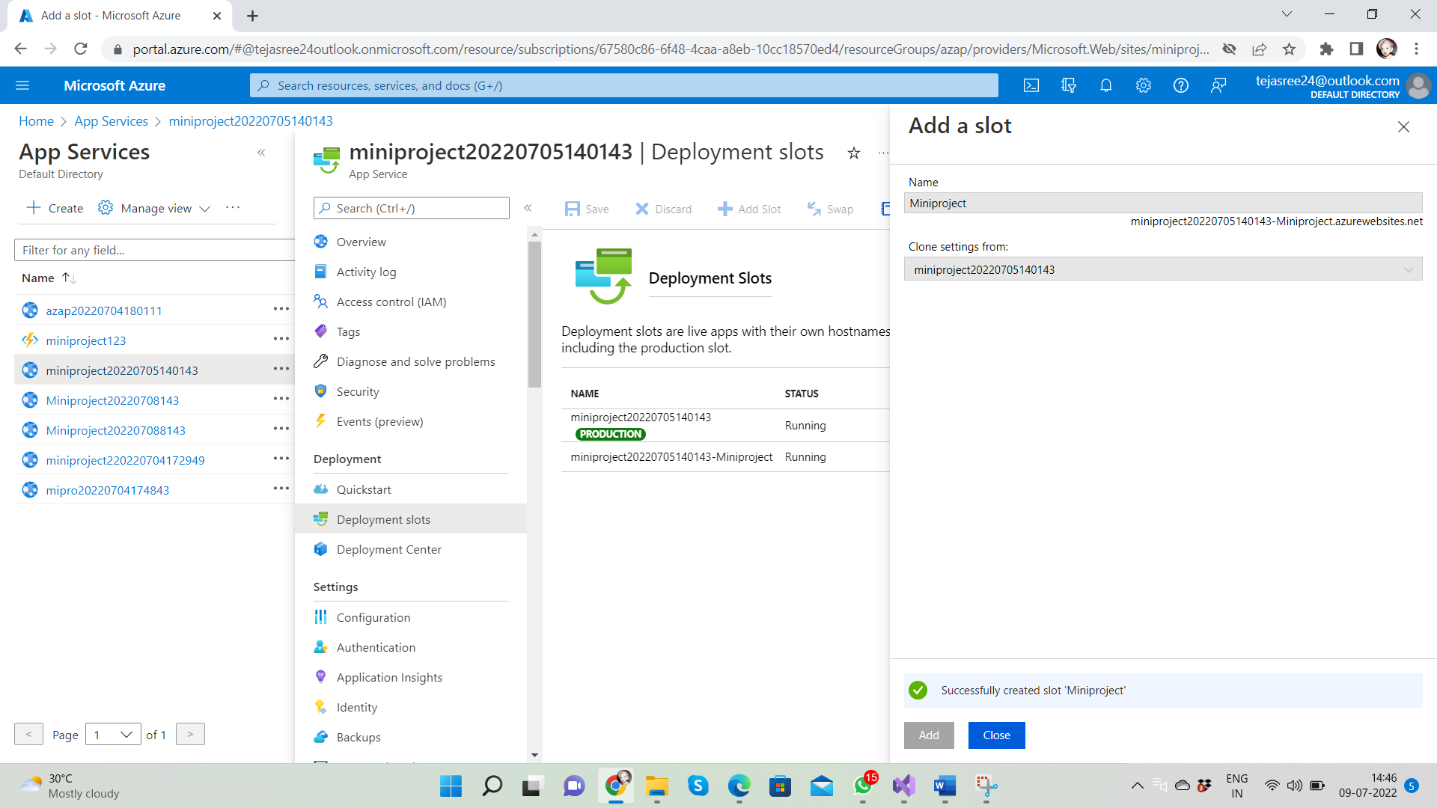




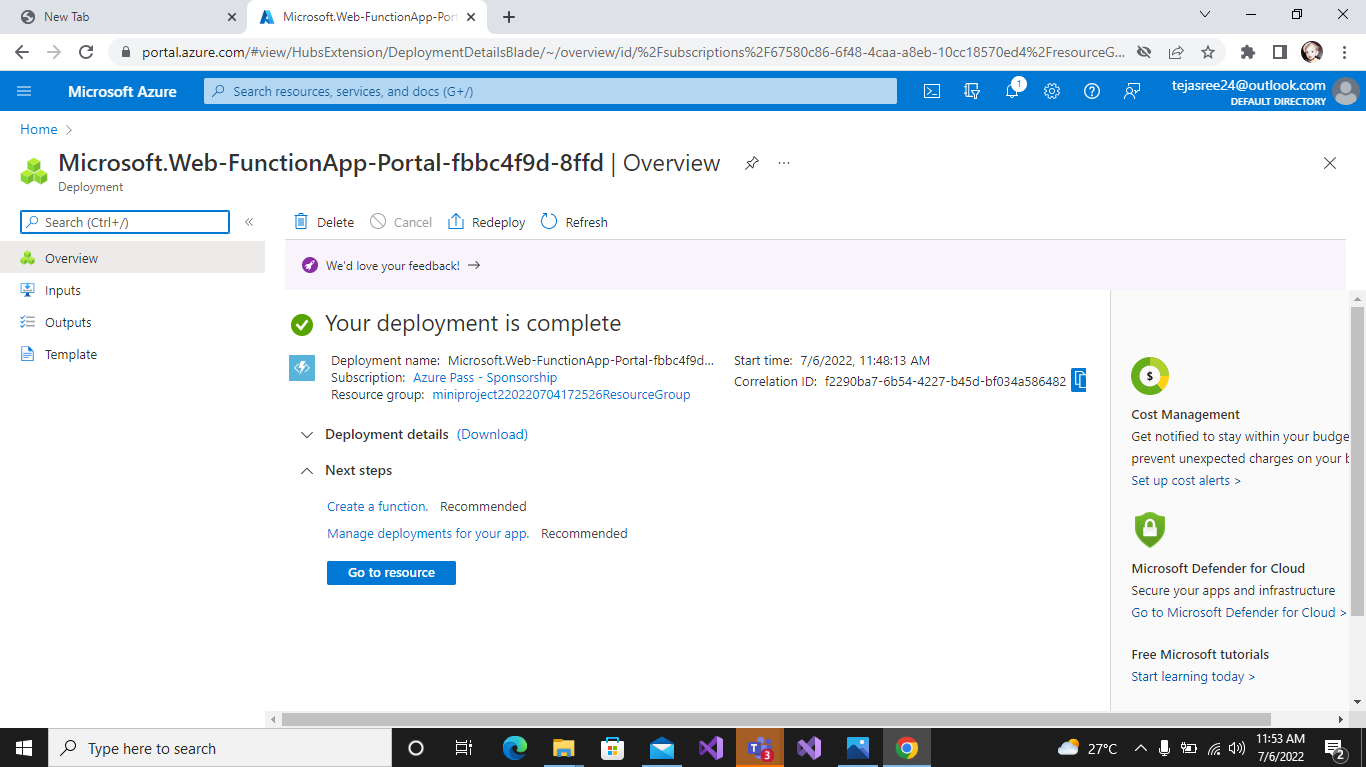


After adding the scale it show rules.

**3.Configure Deployment slots for staging and production.**

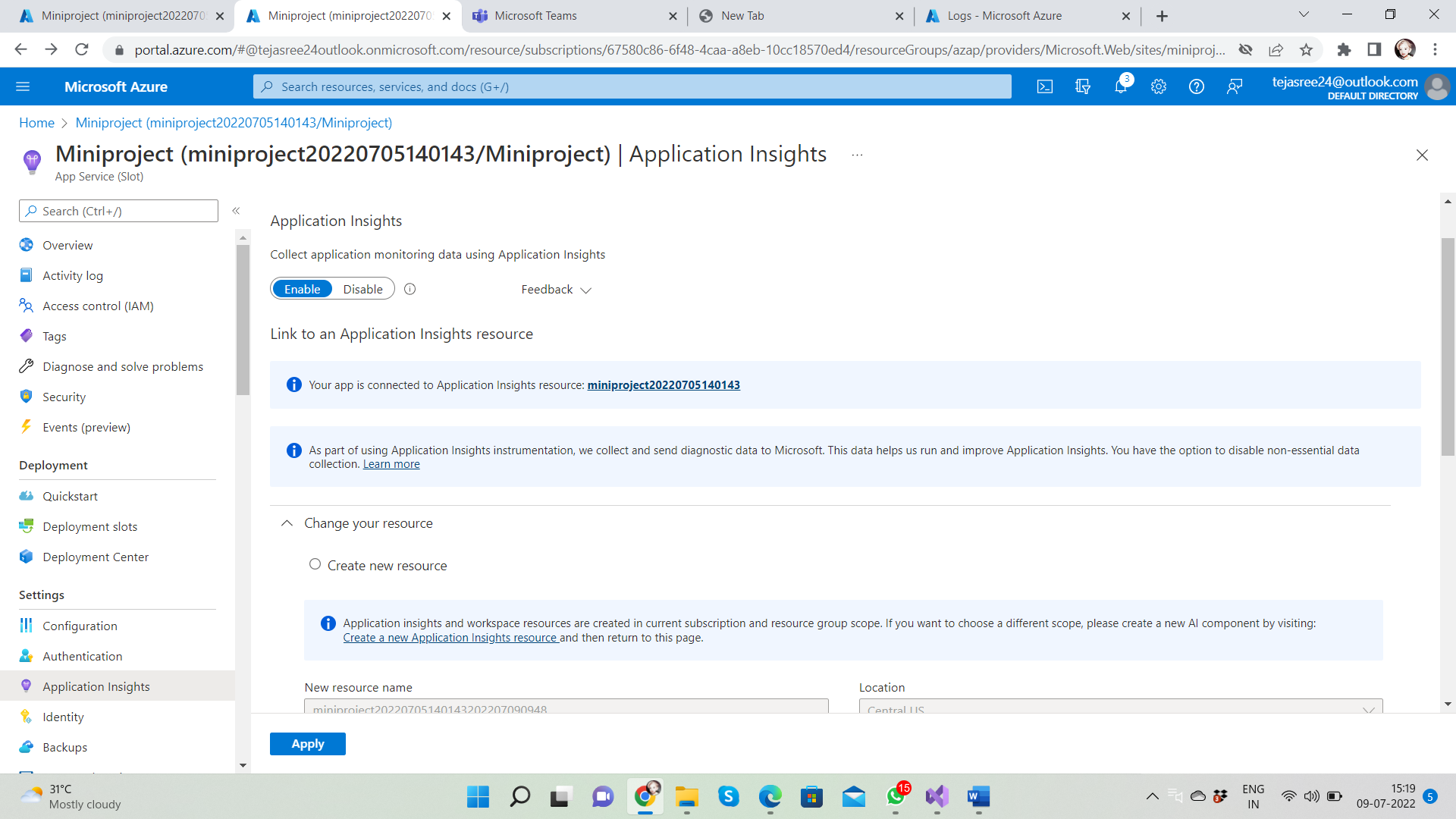
Azure Functions deployment slots allow your function app to run different instances called "slots". Slots are different environments exposed via a publicly available endpoint. One app instance is always mapped to the production slot, and you can swap instances assigned to a slot on demand. Function apps running under the Apps Service plan may have multiple slots, while under the Consumption plan only one slot is allowed.

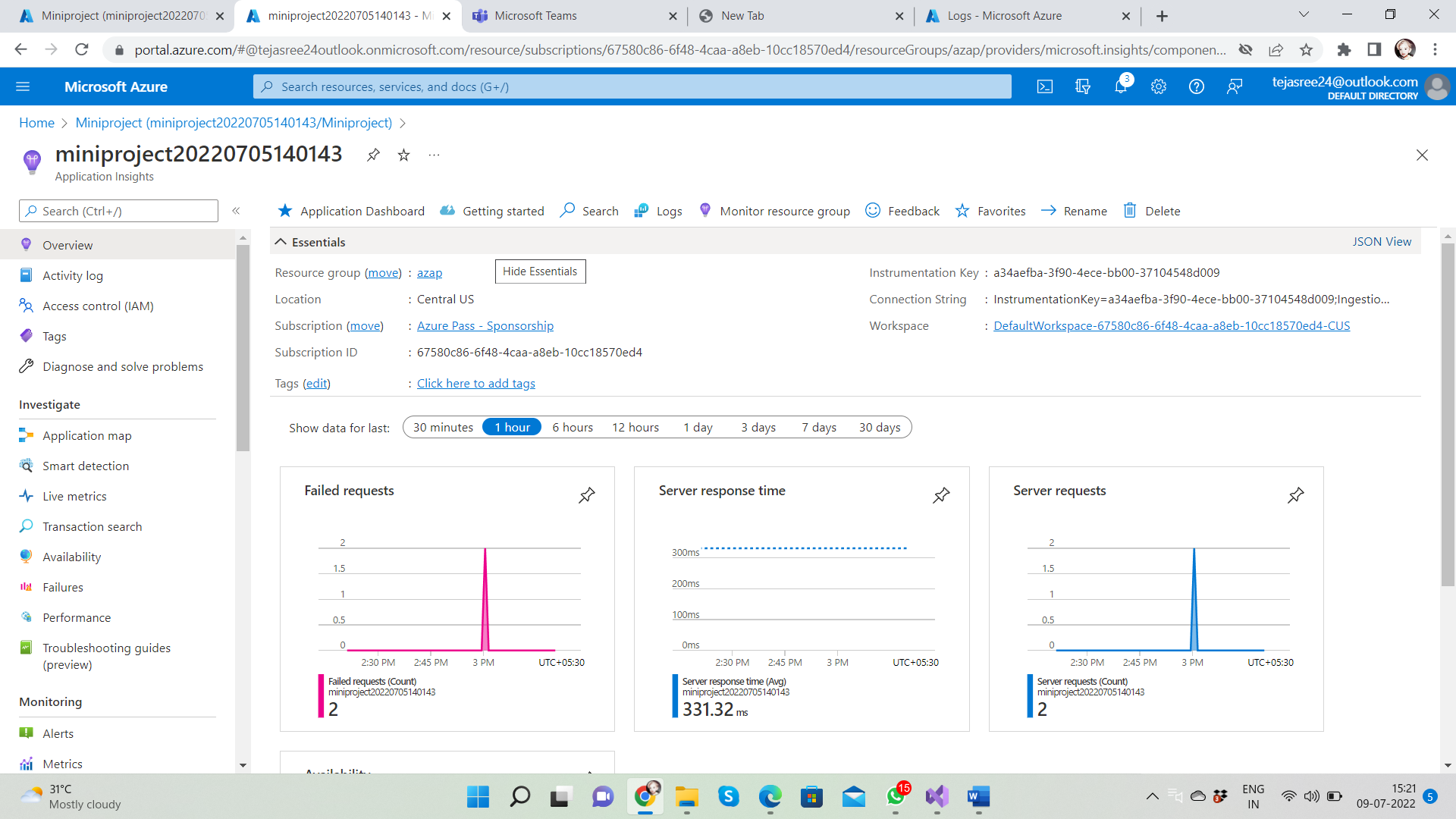
After add a slot Successful both the production and staging ,then after swaping deployment will complete.



**4. Configure Application Insights for the project**

Select the Application Insights.





**5. Configure Swagger for the API**

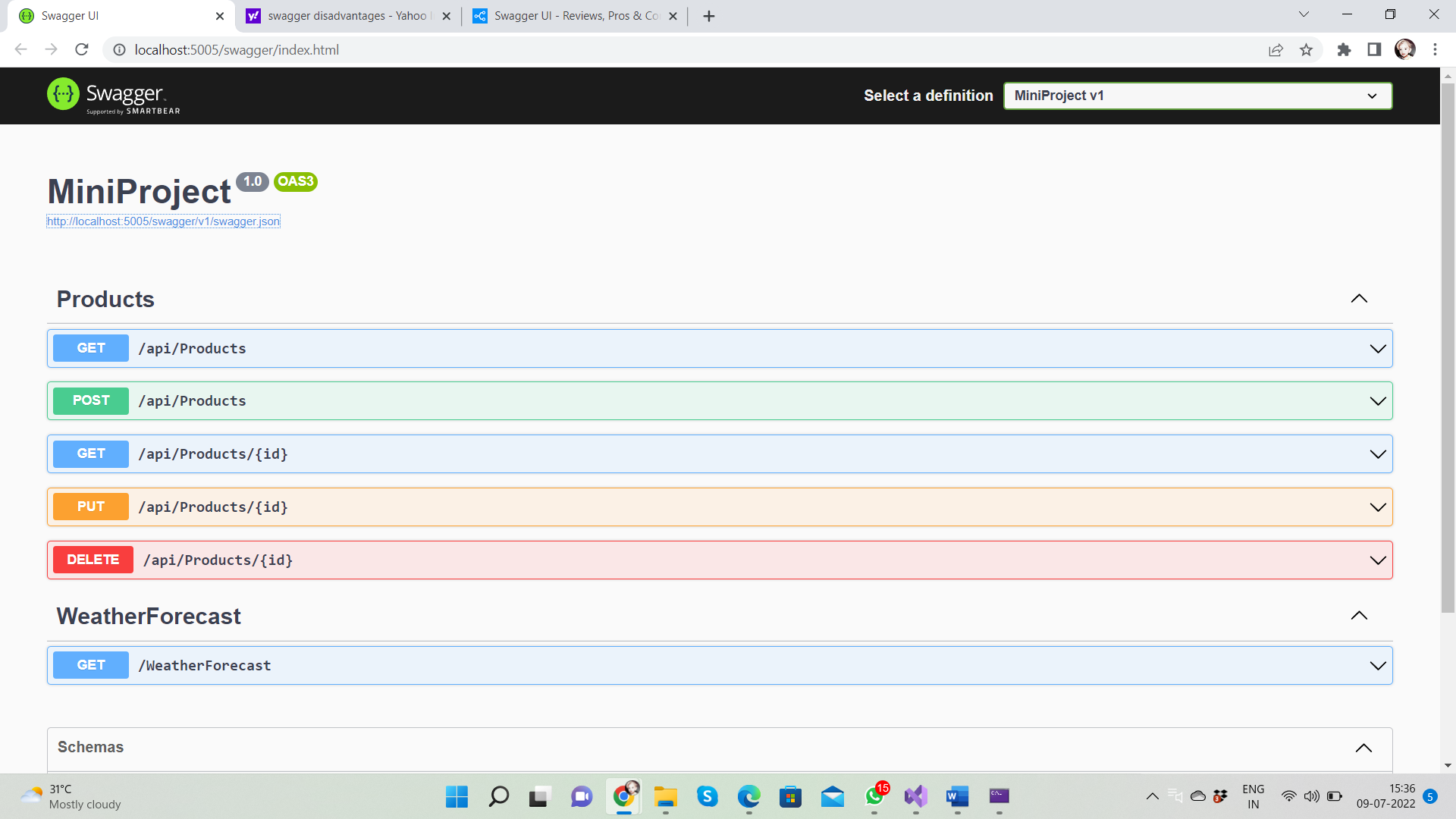
Swagger UI allows anyone be it your development team or your end consumers to visualize and interact with the API’s resources without having any of the implementation logic in place. It’s automatically generated from your Open API (formerly known as Swagger) Specification, with the visual documentation making it easy for back end implementation and client side consumption.

Advantages:

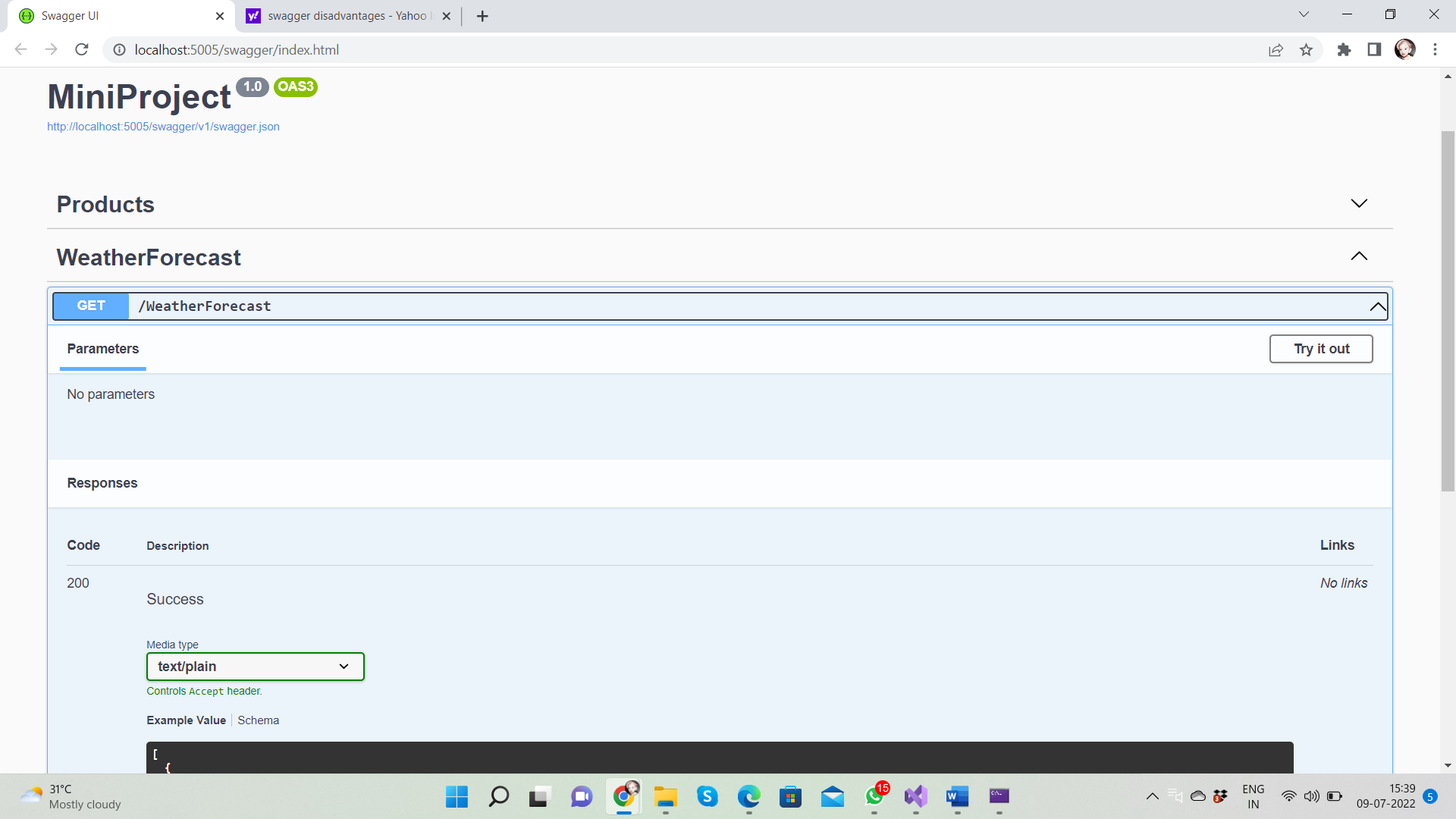
* Dependency Free - The UI works in any development environment, be it locally or in the web
* Human Friendly - Allow end developers to effortlessly interact and try out every single operation your API exposes for easy consumption
* Easy to Navigate - Quickly find and work with resources and endpoints with neatly categorized documentation
* All Browser Support - Cater to every possible scenario with Swagger UI working in all major browsers.
* Fully Customizable - Style and tweak your Swagger UI the way you want with full source code access.
* Complete OAS Support - Visualize APIs defined in Swagger 2.0 or OAS 3.0

Swagger UI's Features

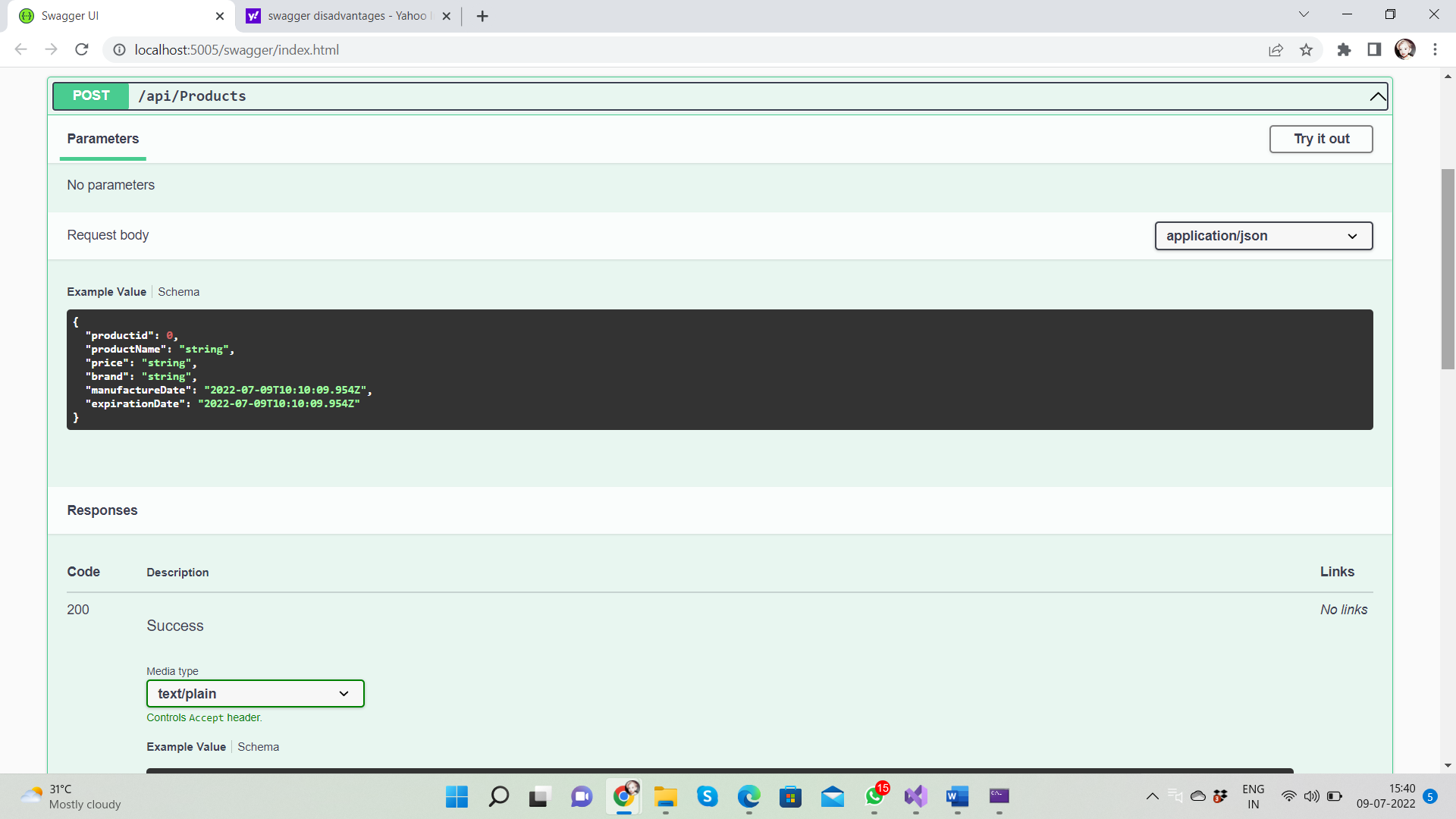
* The UI works in any development environment, be it locally or in the web
* Allow end developers to effortlessly interact and try out every single operation your API exposes for easy consumption
* Quickly find and work with resources and endpoints with neatly categorized documentation
* Cater to every possible scenario with Swagger UI working in all major browsers

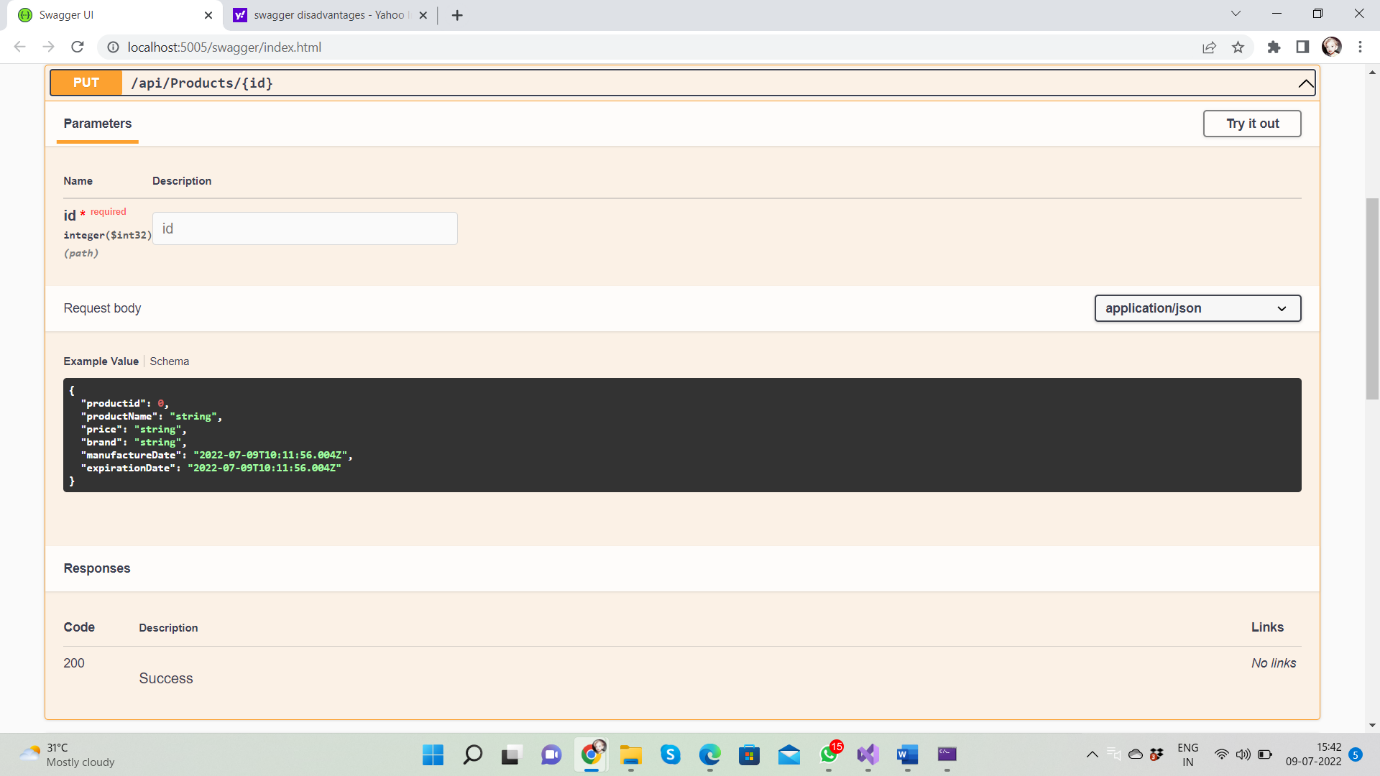


**GET**

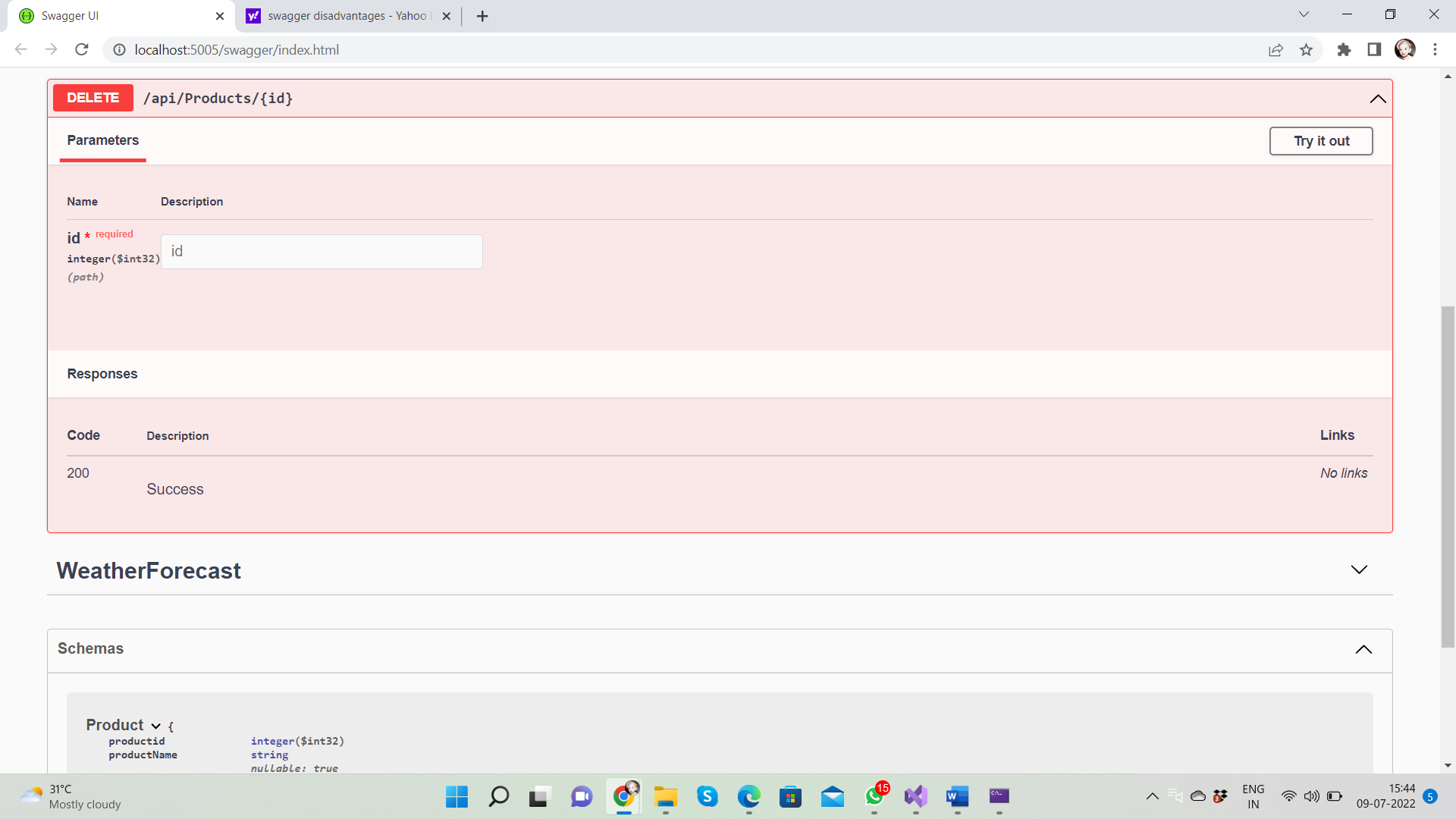


**POST**



**PUT**

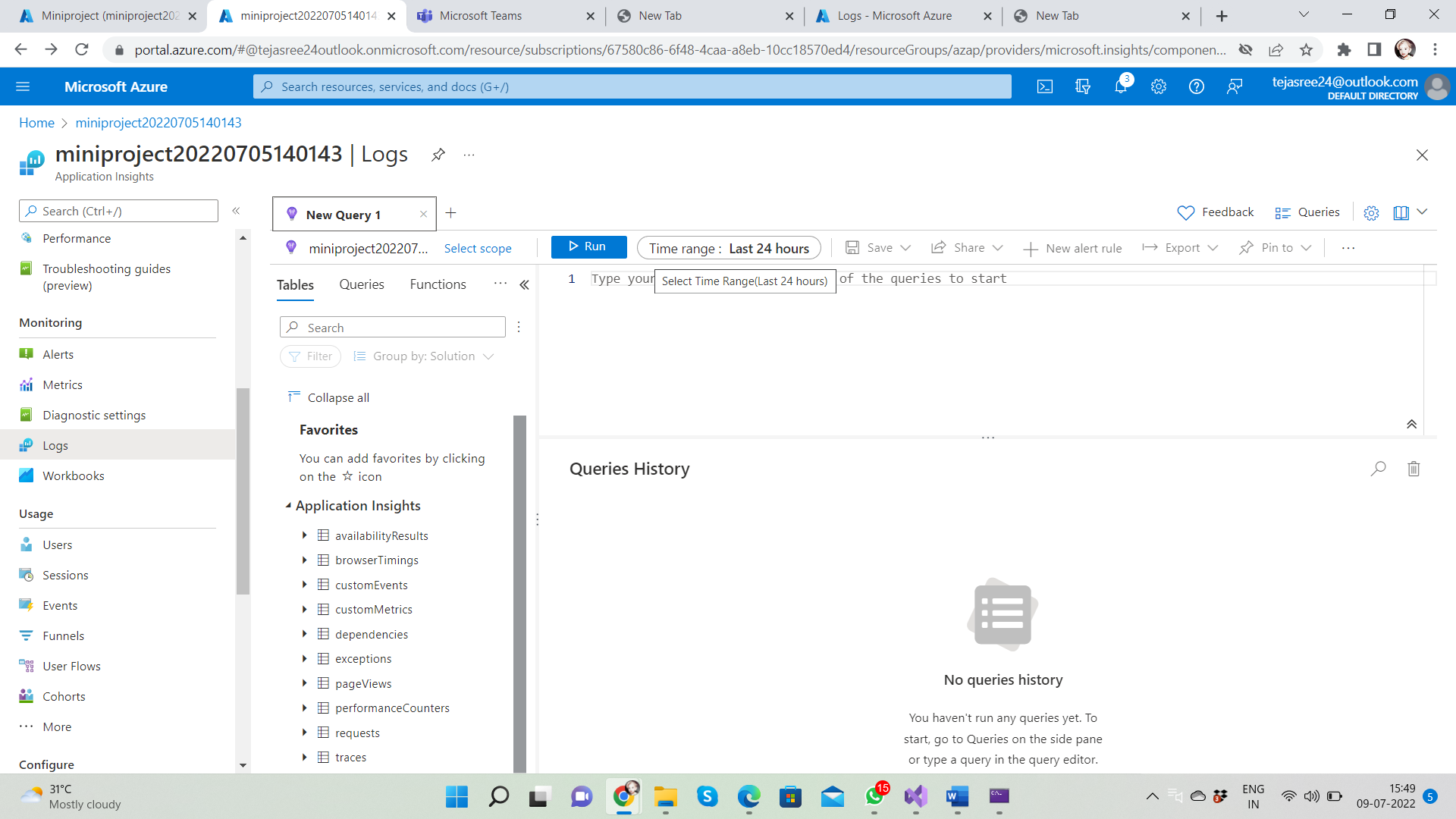
**DELETE**



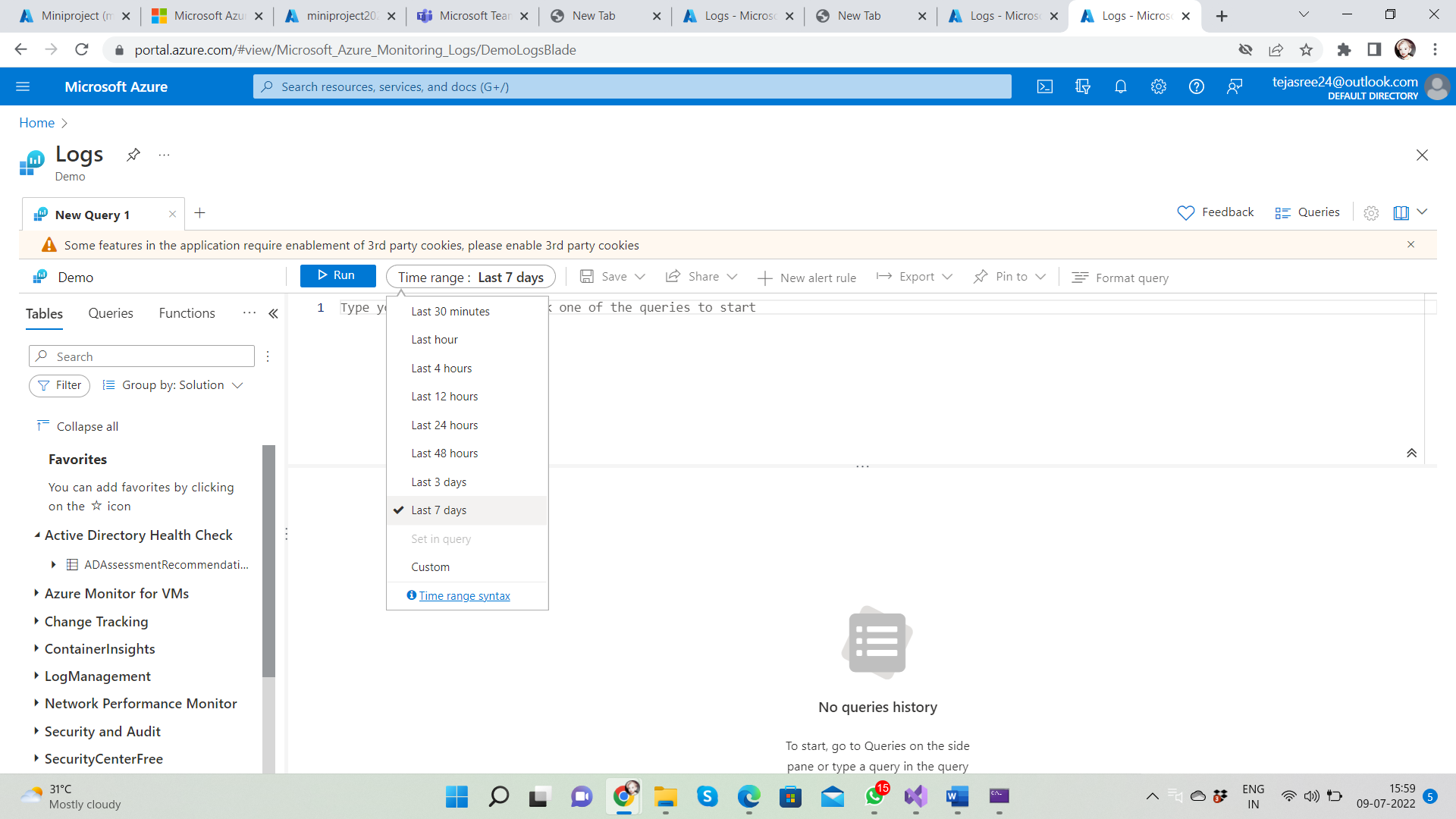
**6.Work with Log Analytics with the sample logs available**

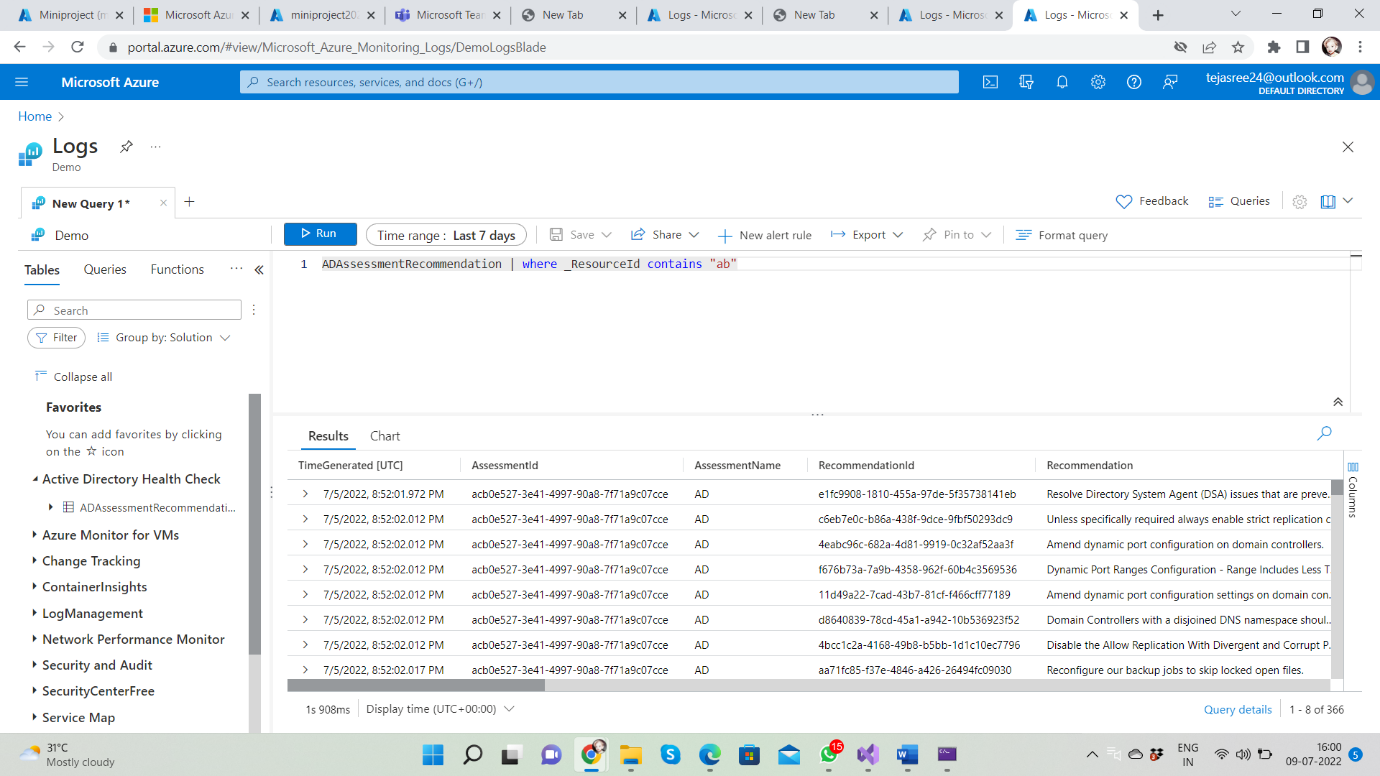
Log Analytics is a tool in the Azure portal to edit and run log queries from data collected by Azure Monitor logs and interactively analyse their results. You can use Log Analytics queries to retrieve records that match particular criteria, identify trends, analyse patterns, and provide various insights into your data.

Select the logs in Azure Portal.



Select Logs from Azure menu



The query AD Assessment Recommendation | where \_ Resource Id contains "ab" returns result to get records.