FINAL-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

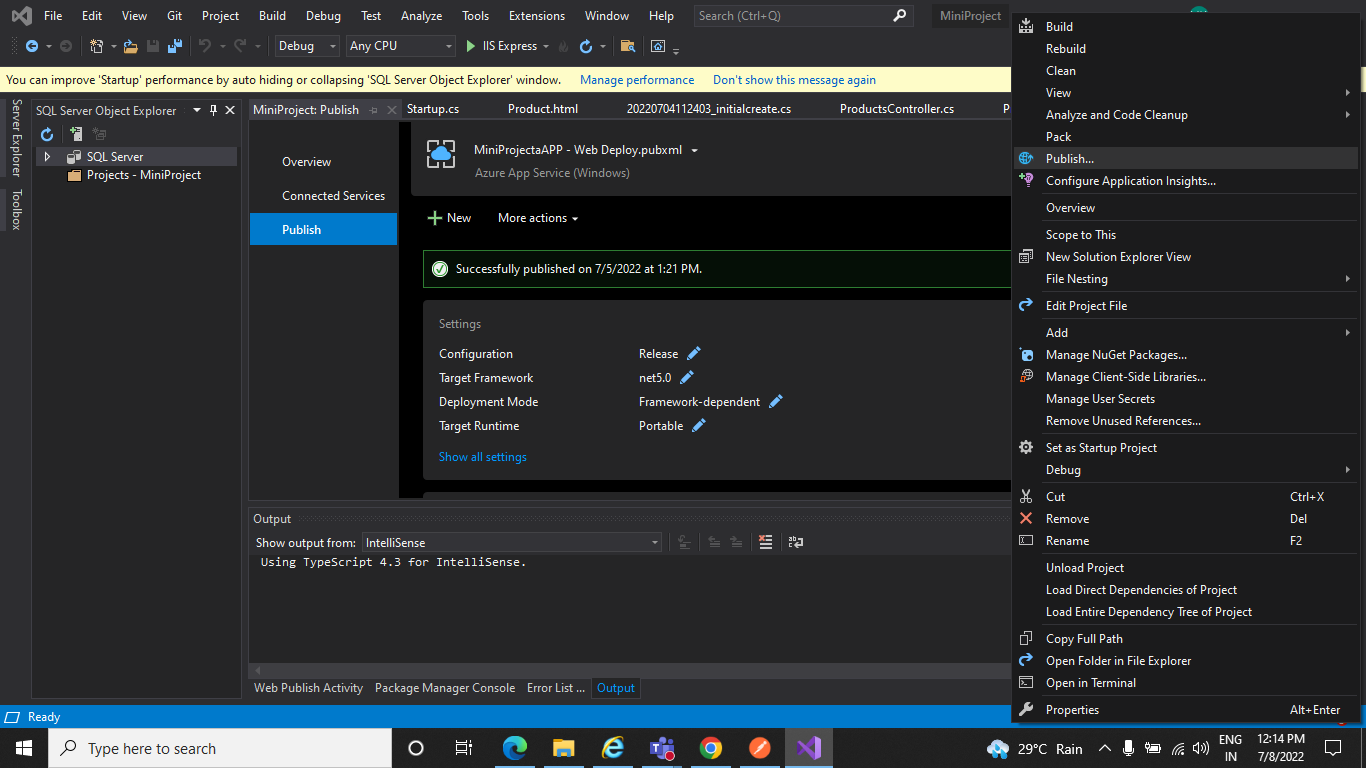
Create a JQuery and AJAX Client to consume the Web API and show the result.

Azure Hosting:

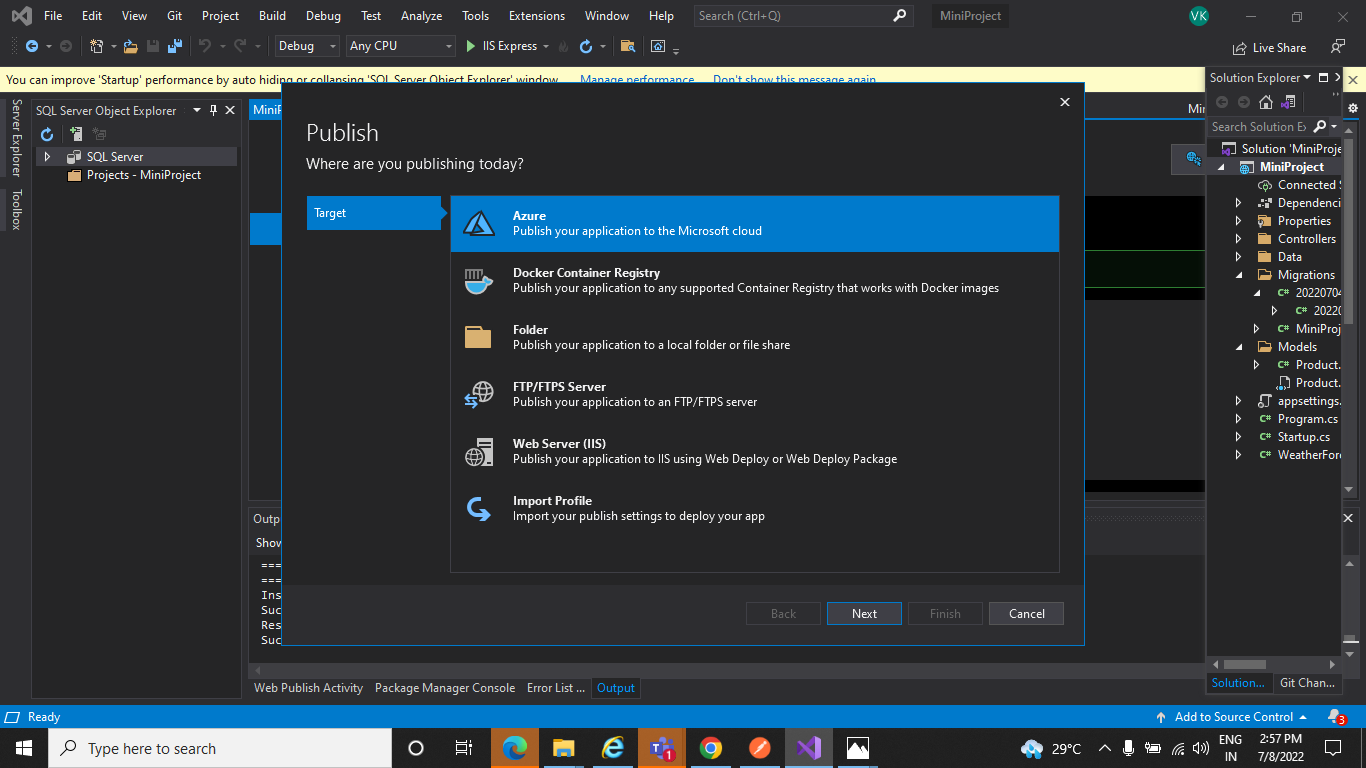
* Host the web api in azure and consume the same using JQuery Client.
* Configure Scale out by adding rules for custom scaling
* Configure Deployment slots for staging and production
* Configure Application Insights for the project
* Configure Swagger for the api
* Work with Log Analytics with the sample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

PUBLISHING API ON AZURE:-

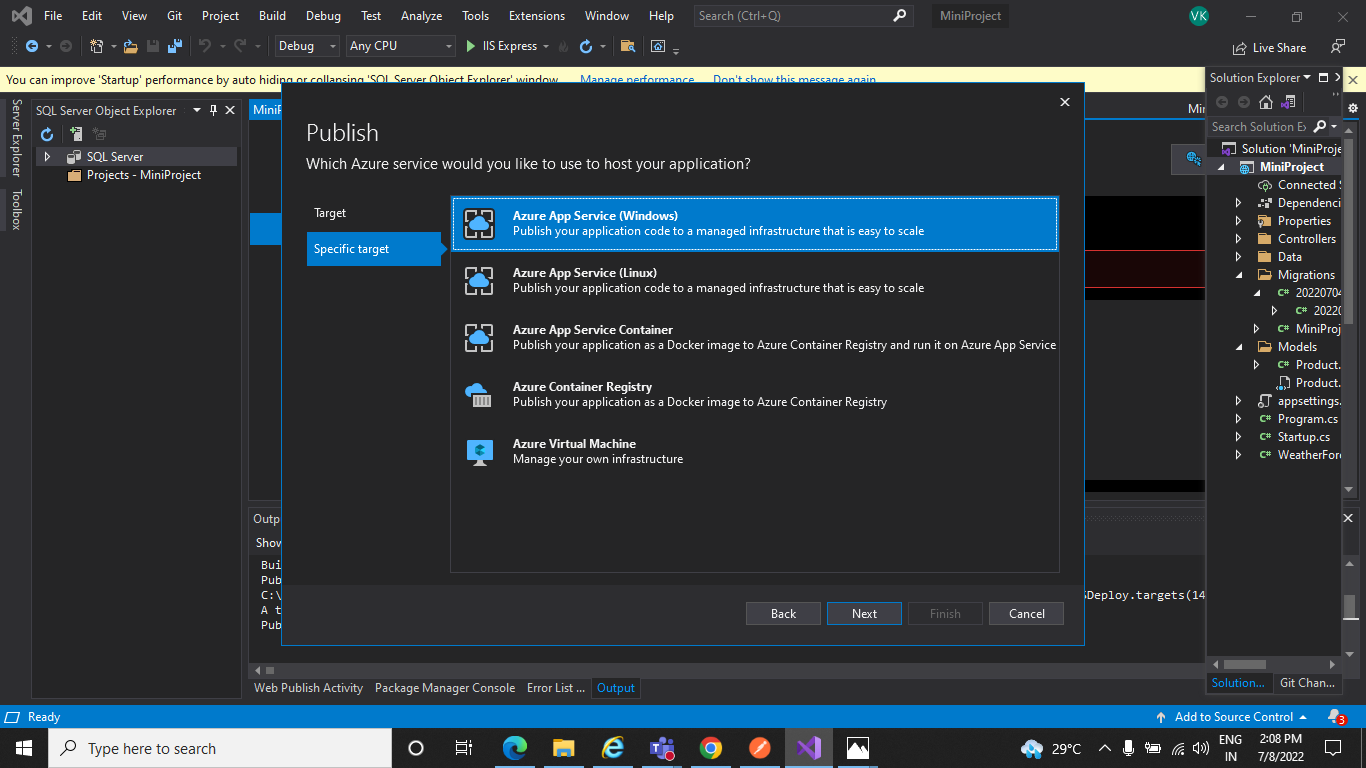
. Go to solution explorer ,right click on project and select publish



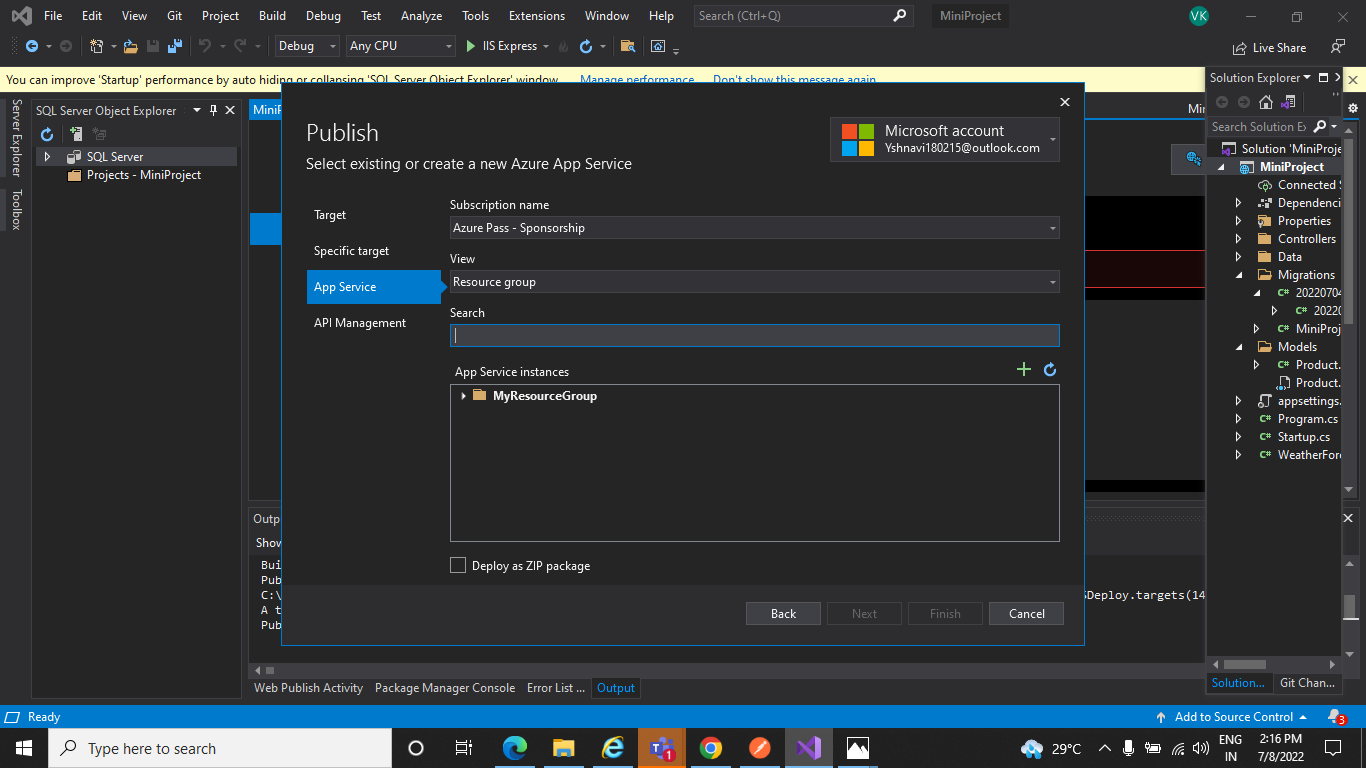
. In the Publish dialog, select Azure and select the Next button



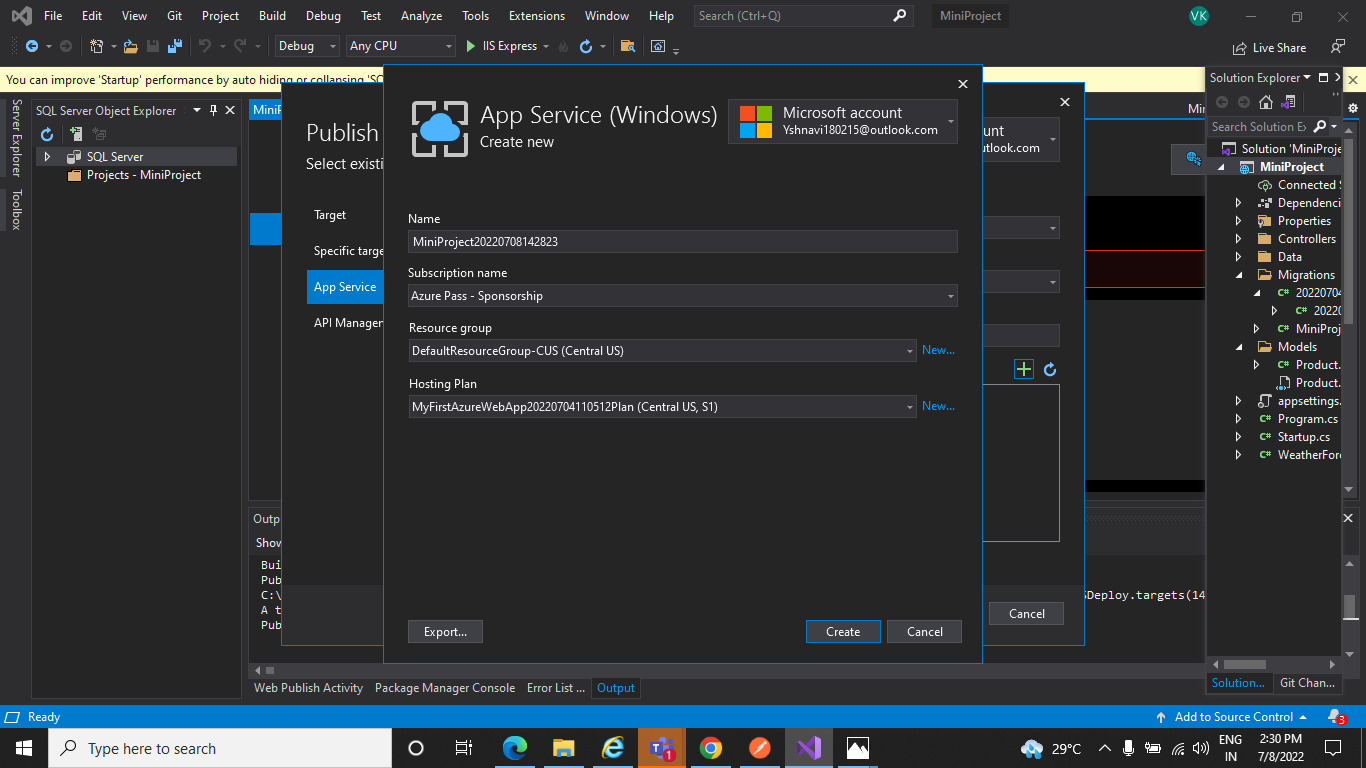
. Select Azure App Service (Windows) and select the Next button



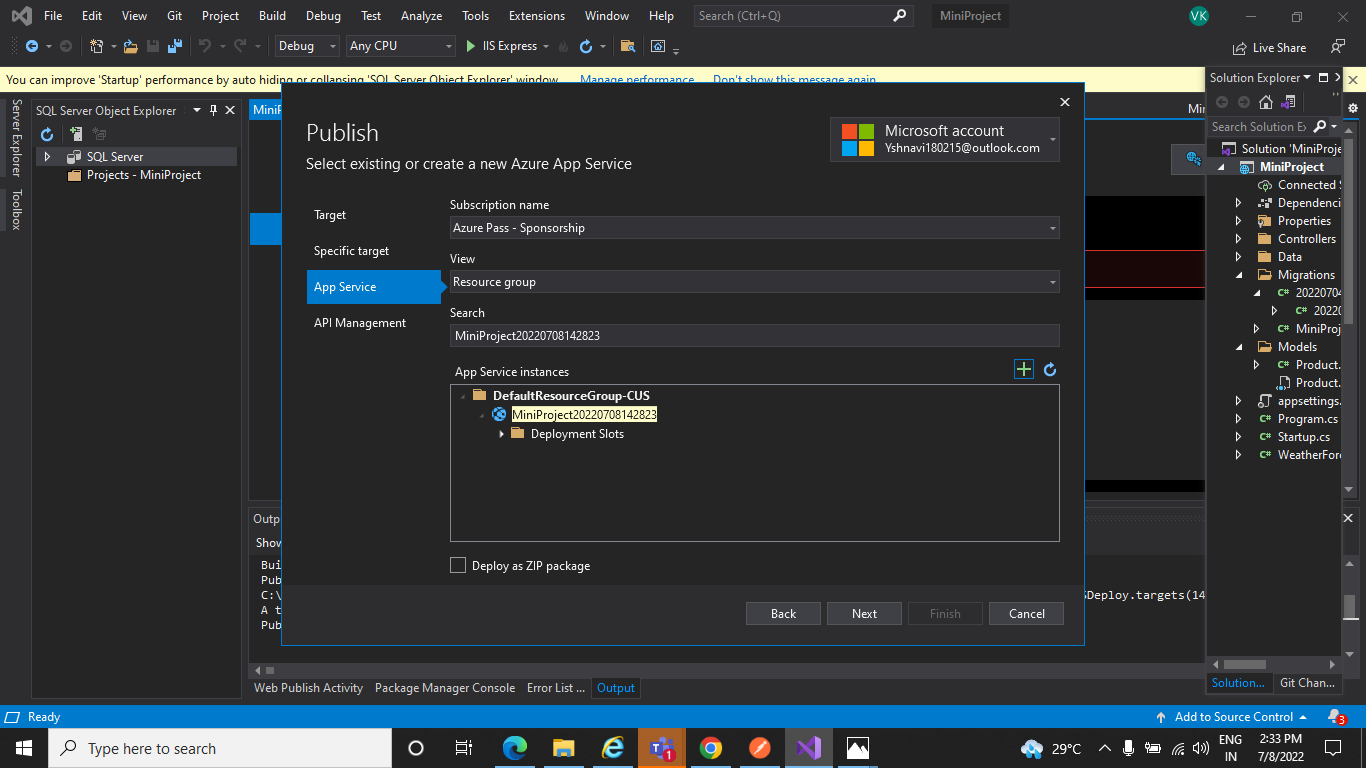
Select Create a new Azure App Service



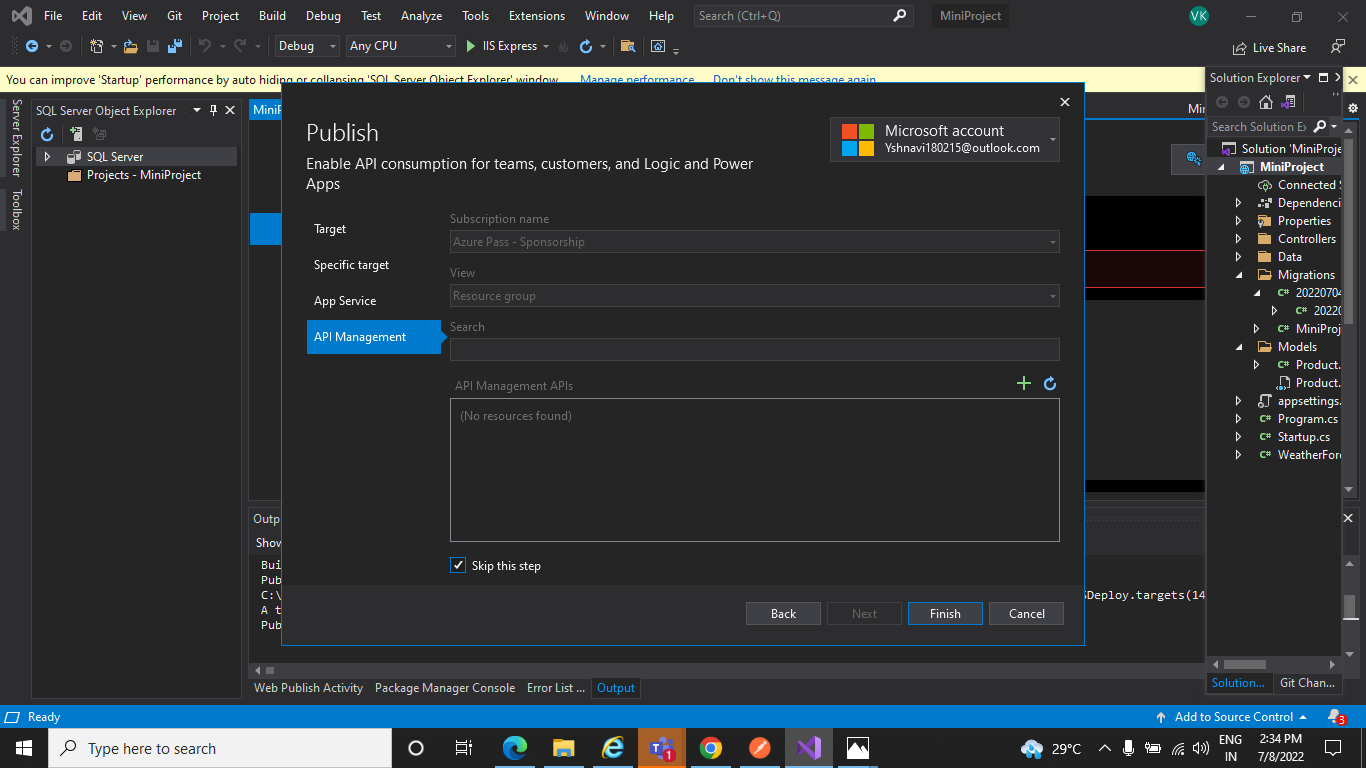
.Create App Service dialog appears. The App Name, Resource Group and App Service plan entry fields are populated. Select Create button.



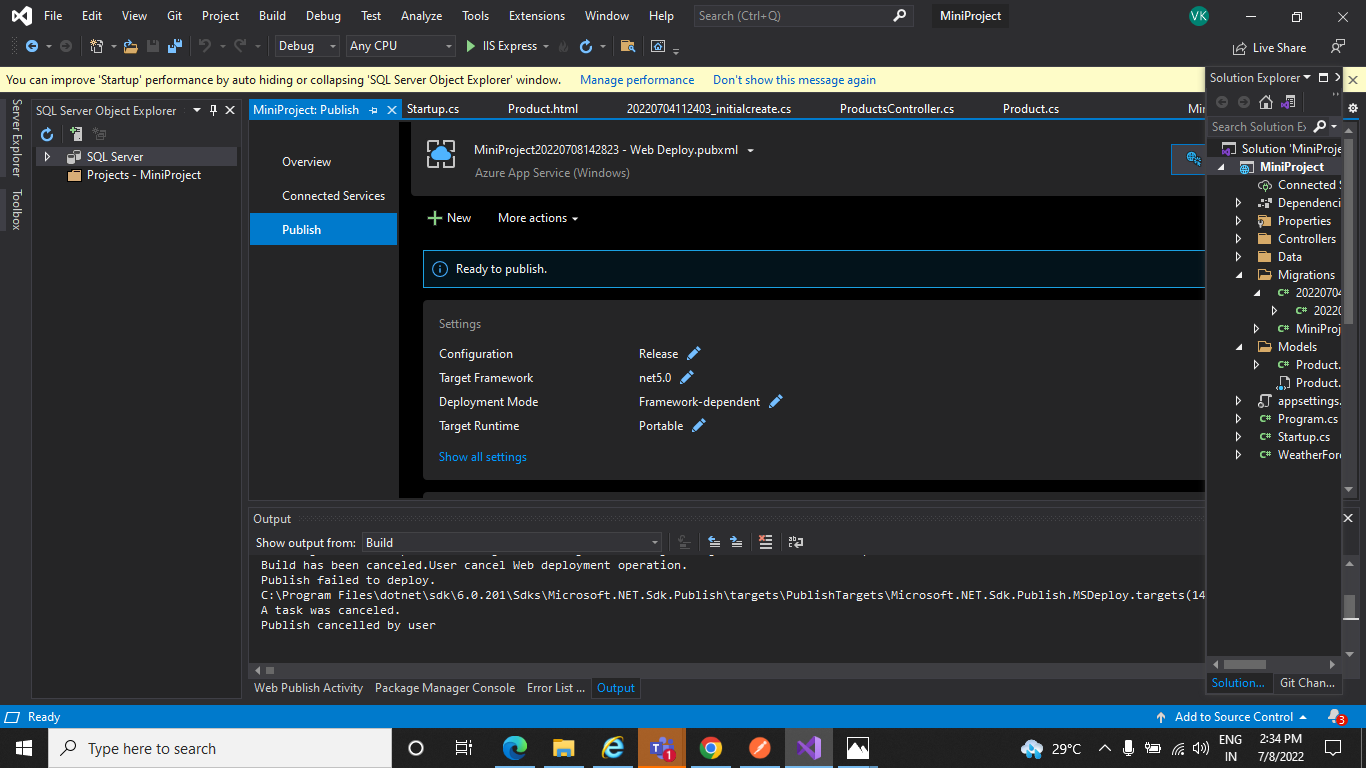
. Click on Next



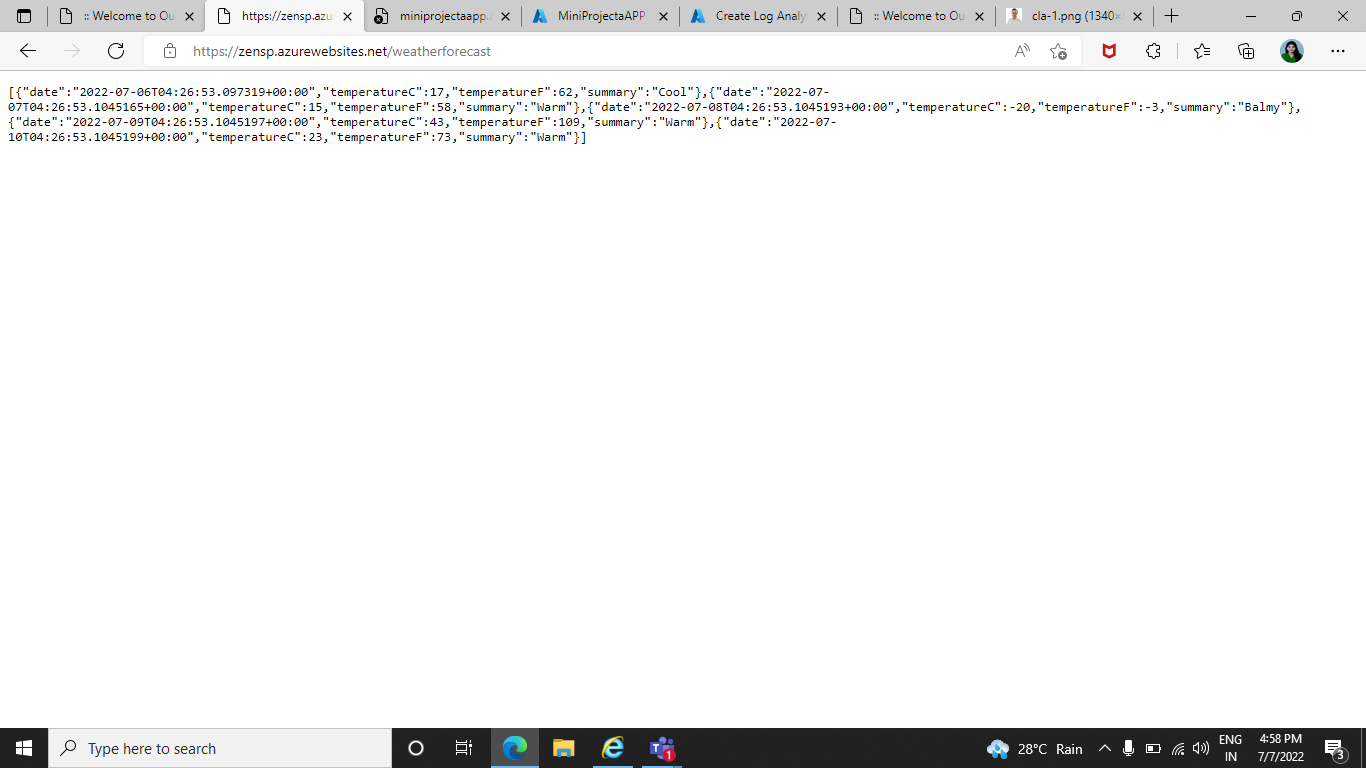
. click on Finish



. After completion of creation automatically the dialog is closed and the Publish dialog focus again.The instance that was created is automatically selected.

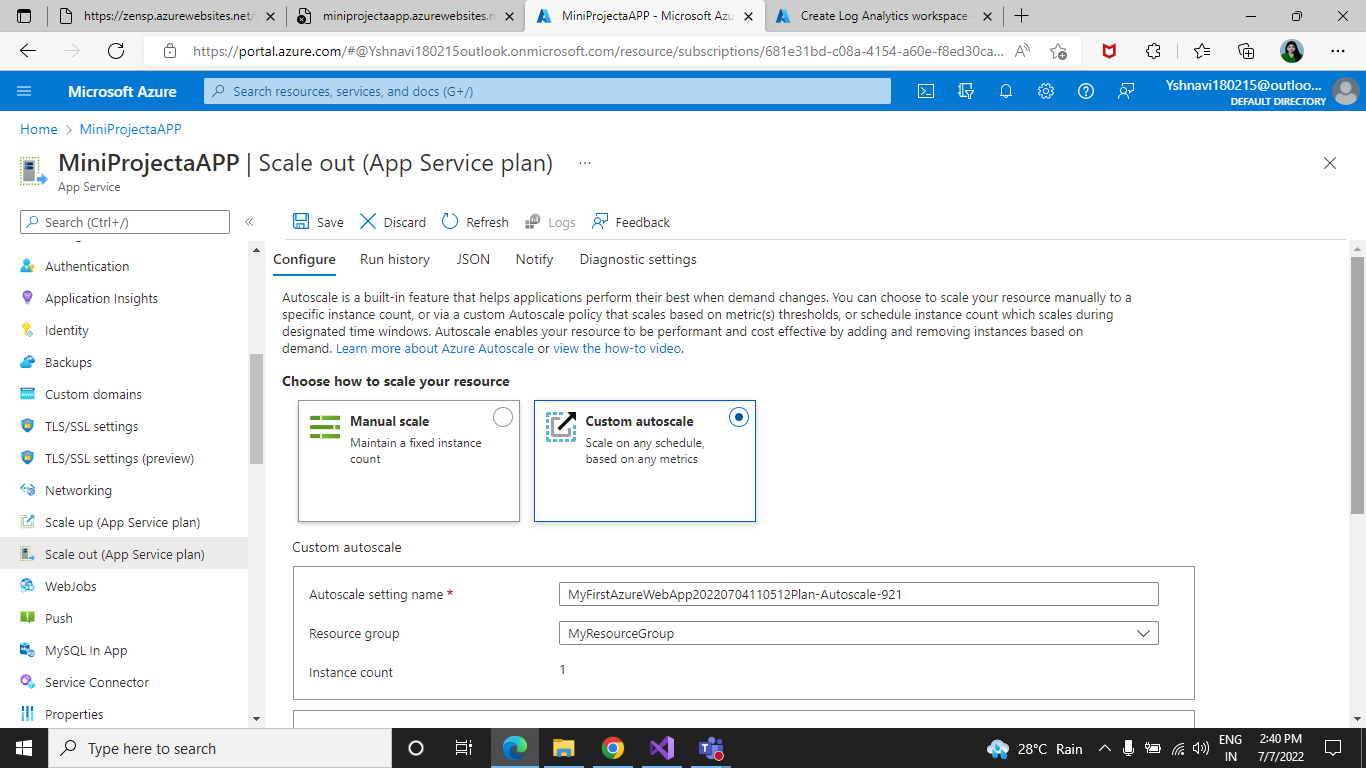


. Then if you click on Publish button a new browser window will appear and You will get JSON result.

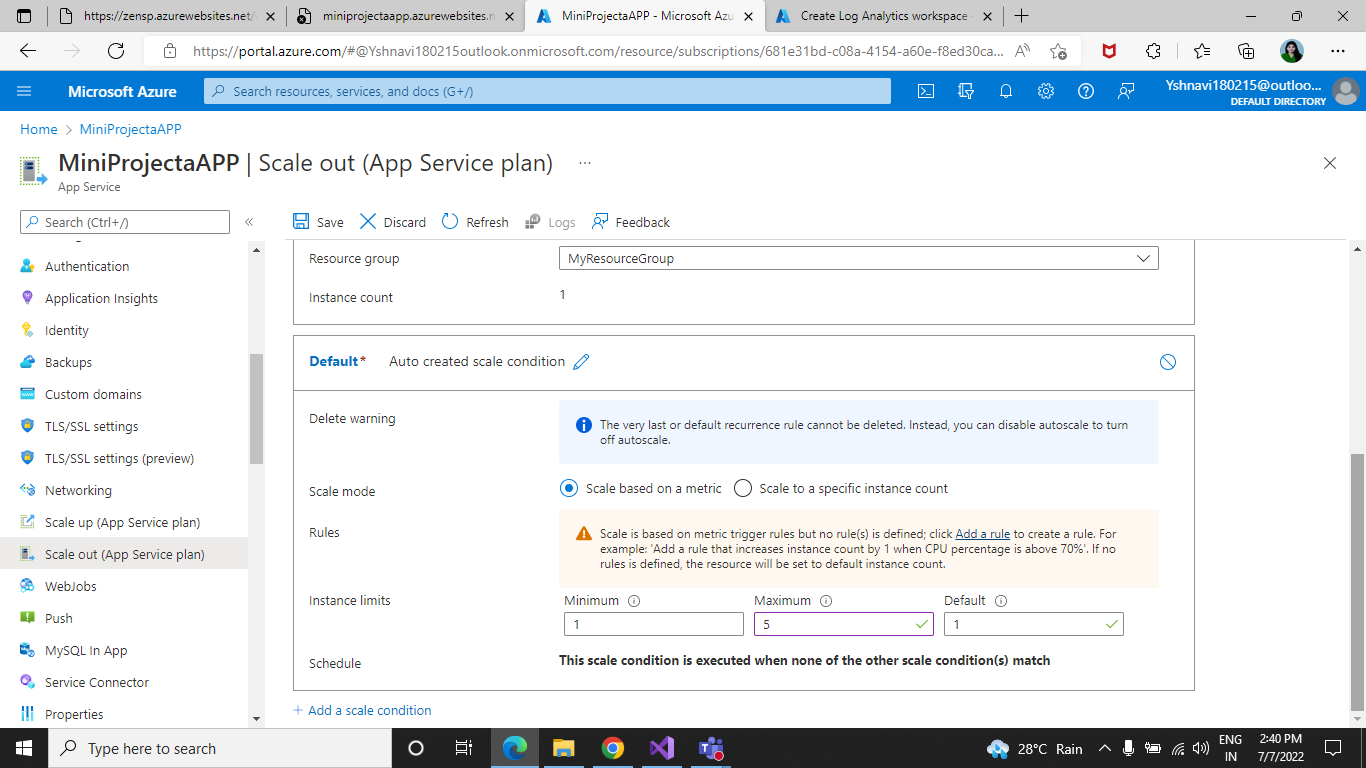


Configure Scale out by adding rules for custom scaling :-

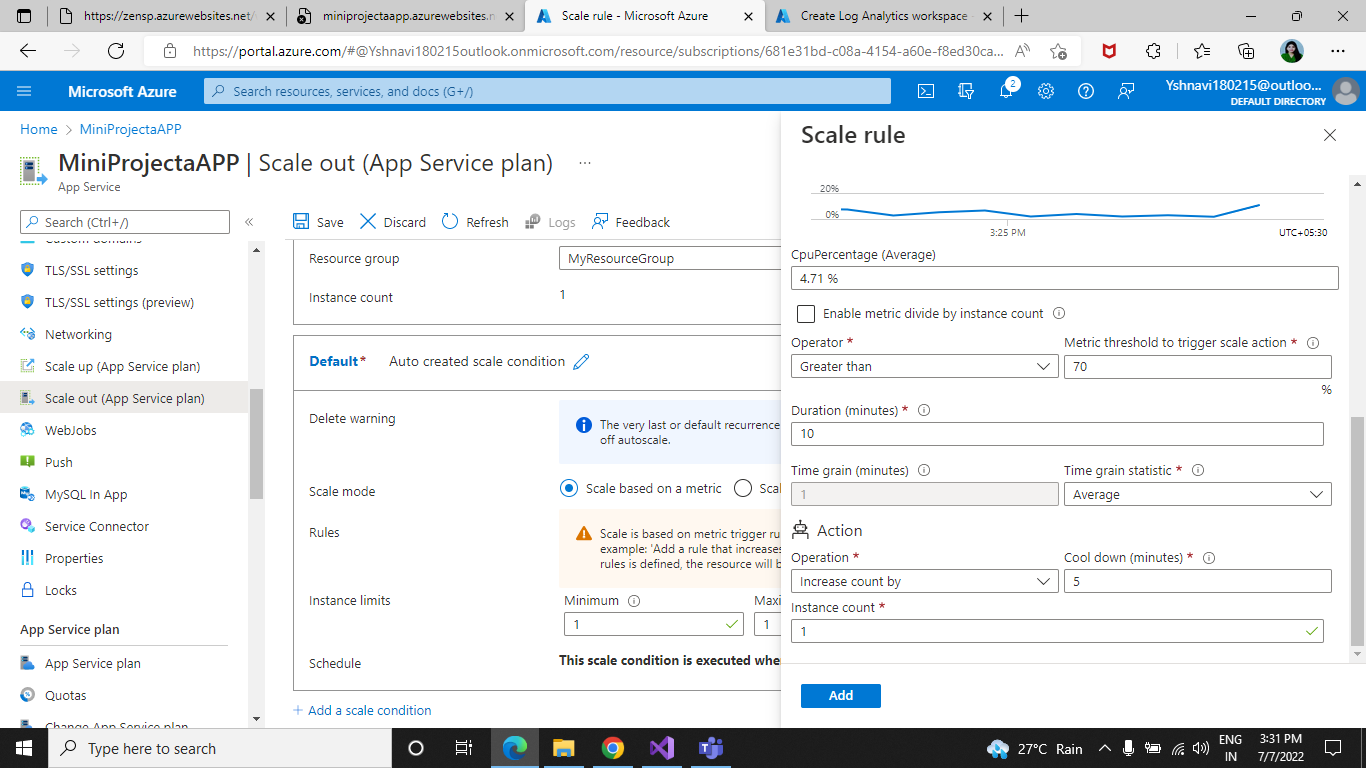
. Open Azure portal and open App Service ,go to settings and select scale out (App Service Plan), Select Custom autoscale.



. And then click Add a rule. This opens as a context pane on the right side.



By default, this sets the option to scale your instance count by 1 if the CPU percentage of the resource exceeds 70 percent. Leave it at its default values and click Add.

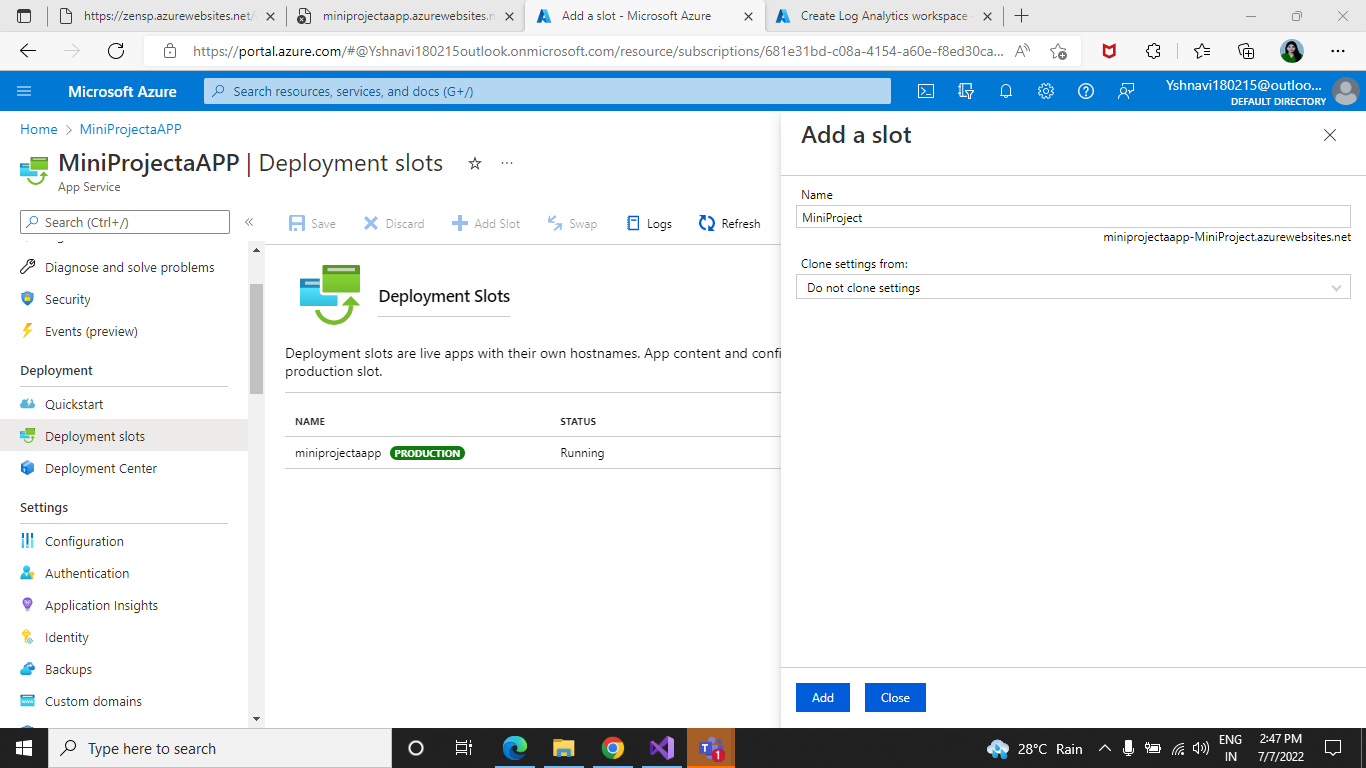


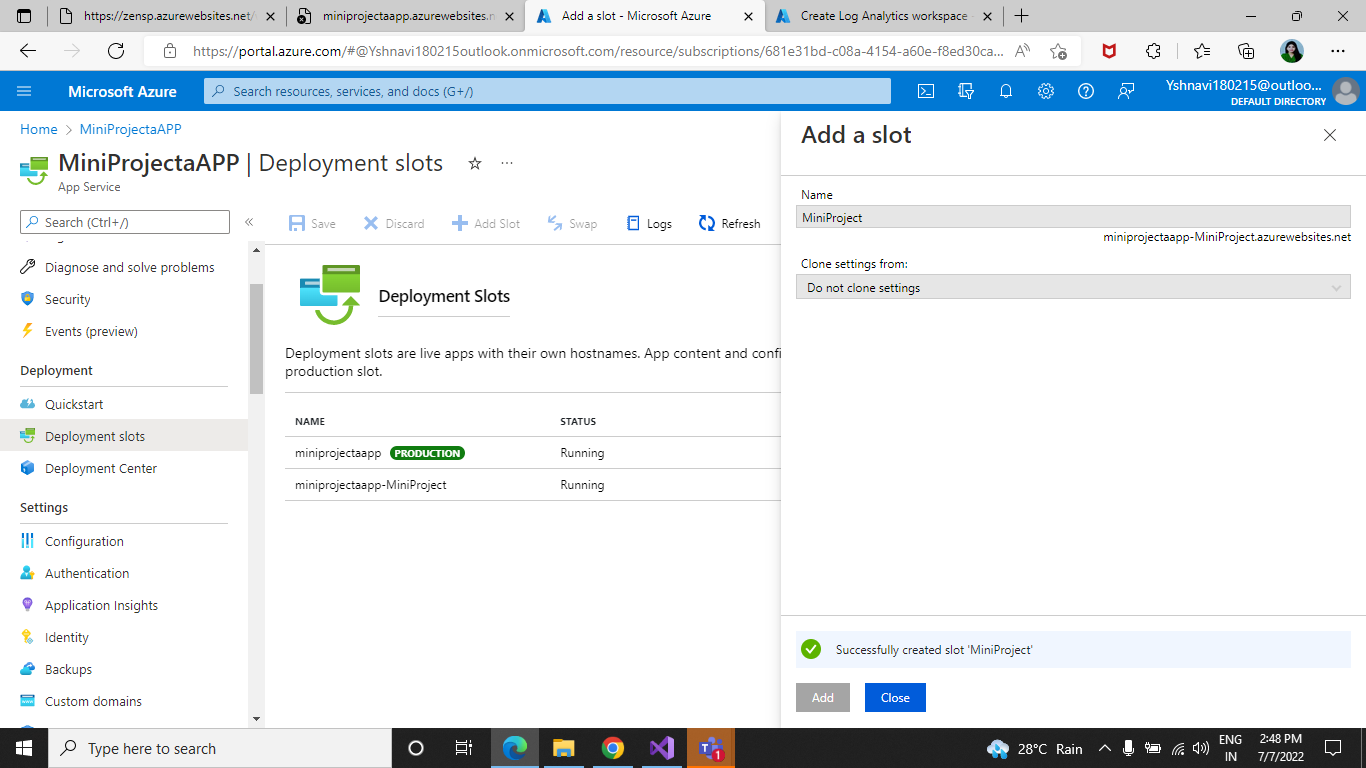
Scale Out:-

Autoscale is a built-in feature that helps applications perform their best when demand changes. You can choose to scale your resource manually to a specific instance count, or via a custom Autoscale policy that scales based on metric(s) thresholds, or schedule instance count which scales during designated time windows. Autoscale enables your resource to be performant and cost-effective by adding and removing instances based on demand in this way scale out can be done of your App service.

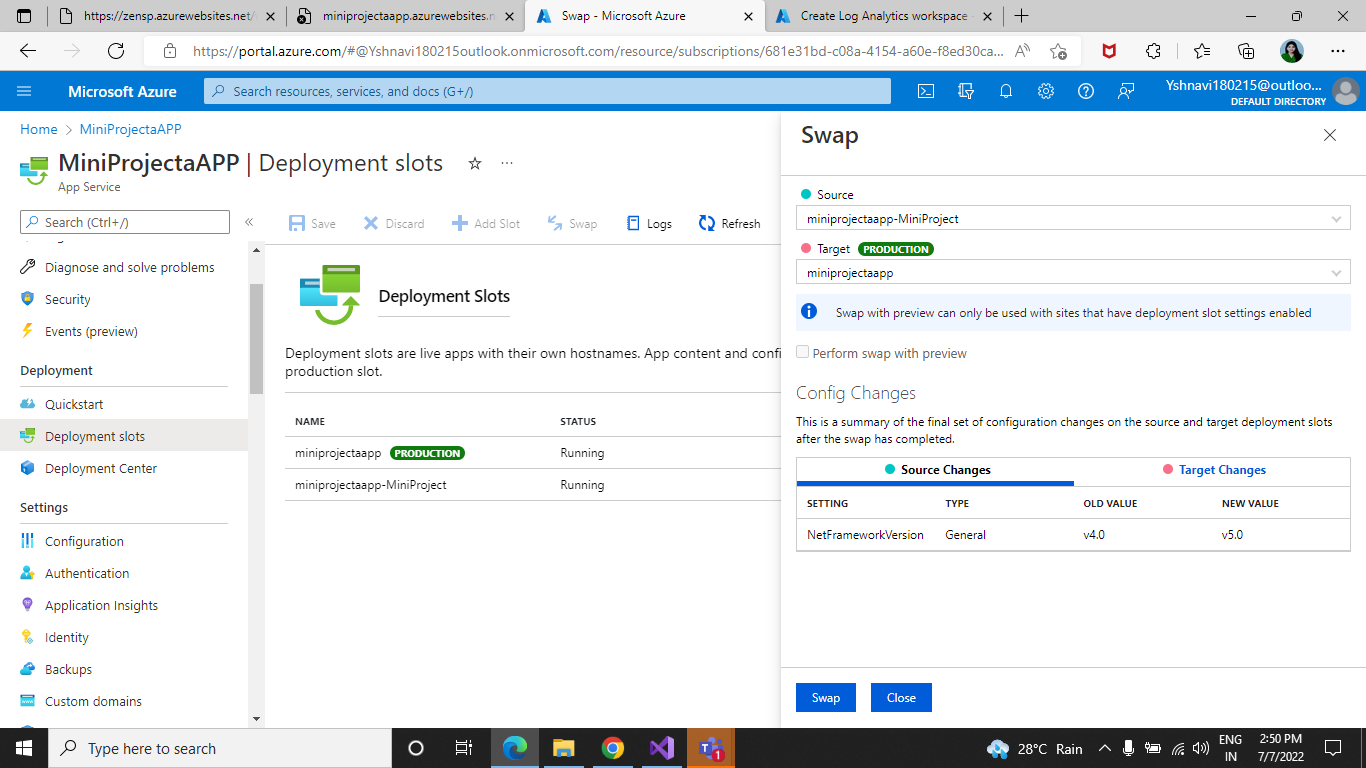
Configuration of Deployment slots:-

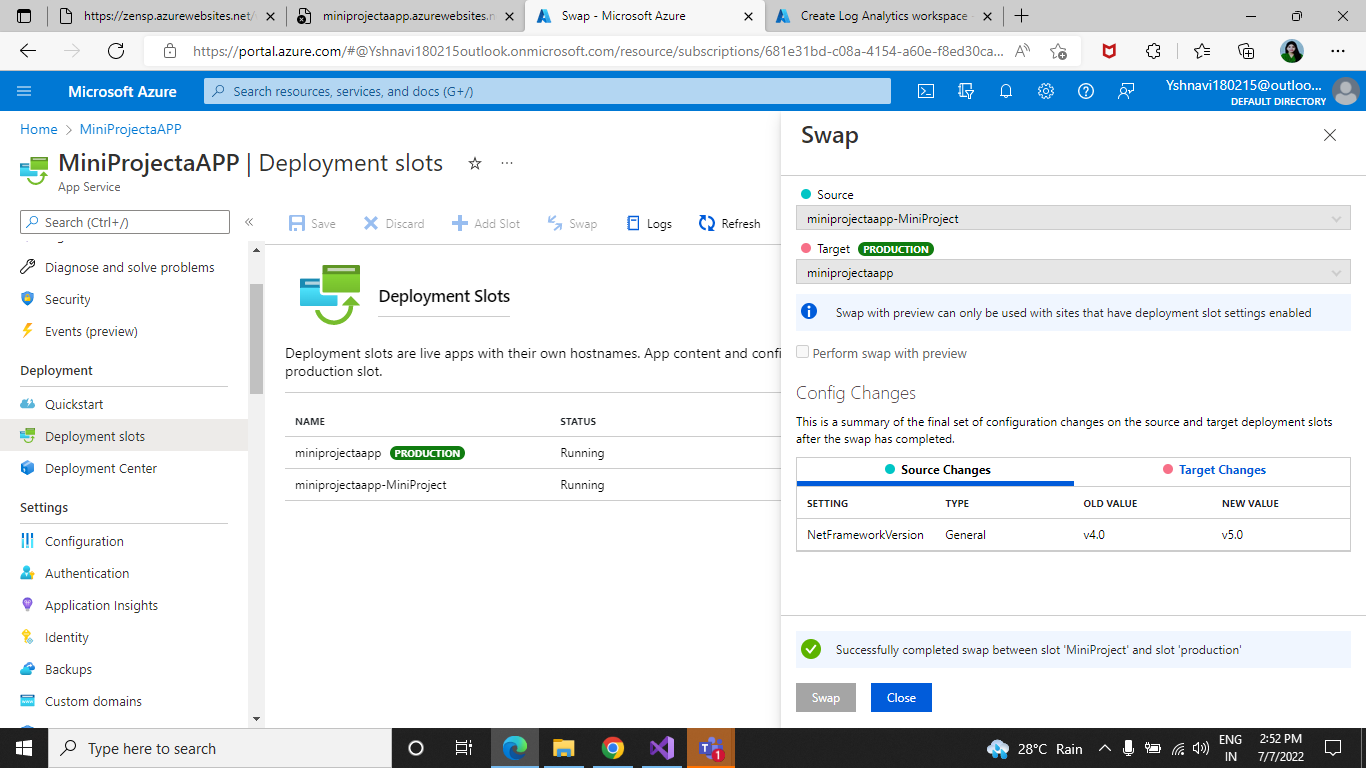
. Go to Deployment slots then Select + Add slot





. Verify the configuration settings for your swap and select Swap





Deployment slots:-

Azure functions deployment slots allow your function app to run different instances called “slots”. These 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 consumption plan only one slot is allowed.

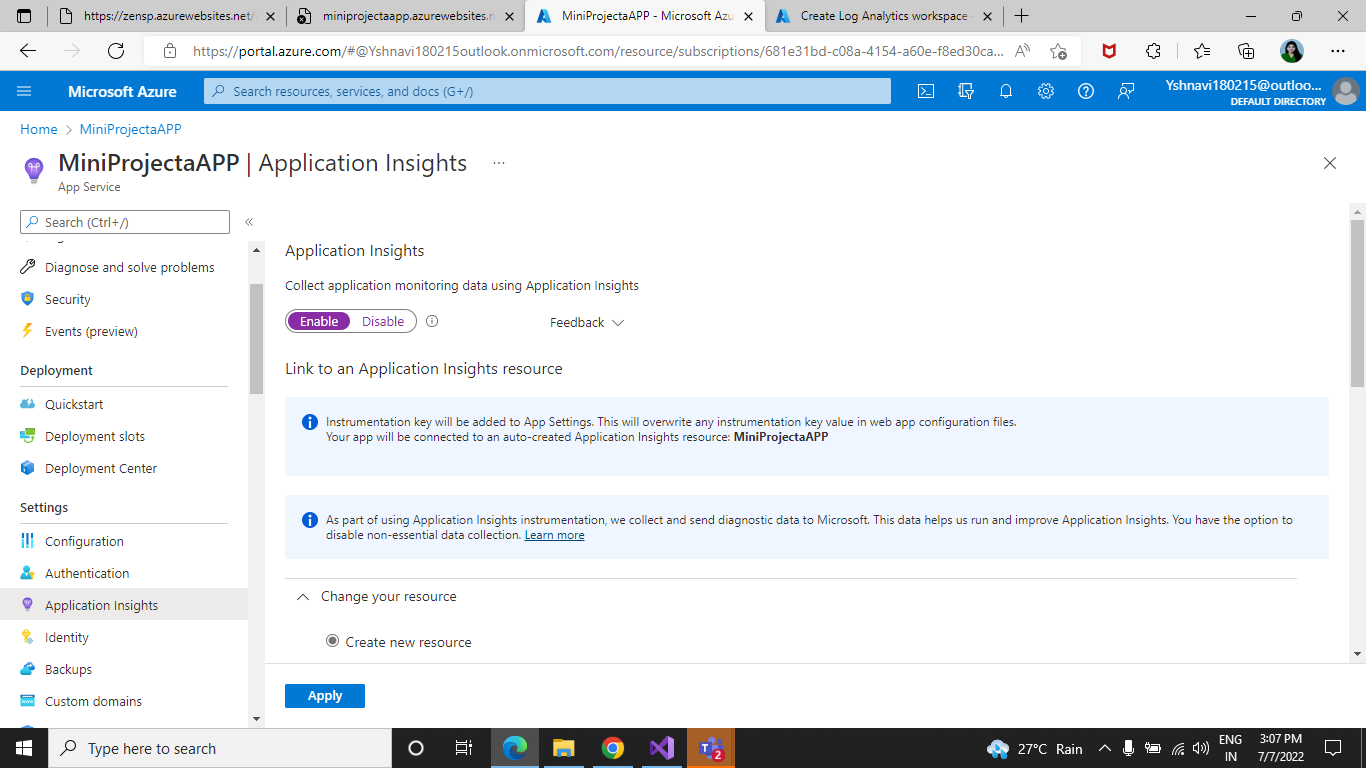
The following reflect how functions are affected by swapping slots:

-Traffic redirection is seamless; no requests are dropped because of a swap. This seamless behavior is a result of the next function triggers being routed to the swapped slot.

- Currently executing function are terminated during the swap. Please review Improve the performance and reliability of Azure Functions to learn how to write stateless and defensive functions.

Configure Application Insights for the project :-

. Select Application Insights

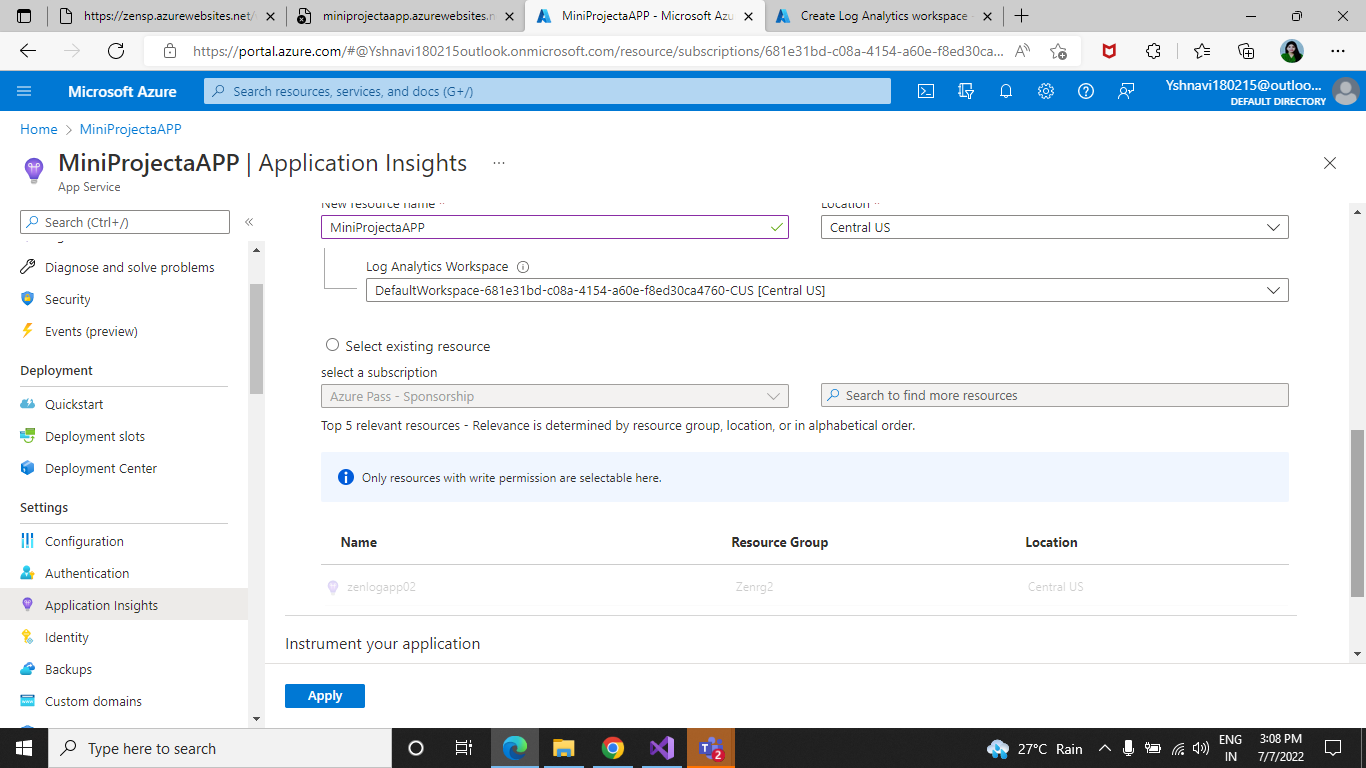


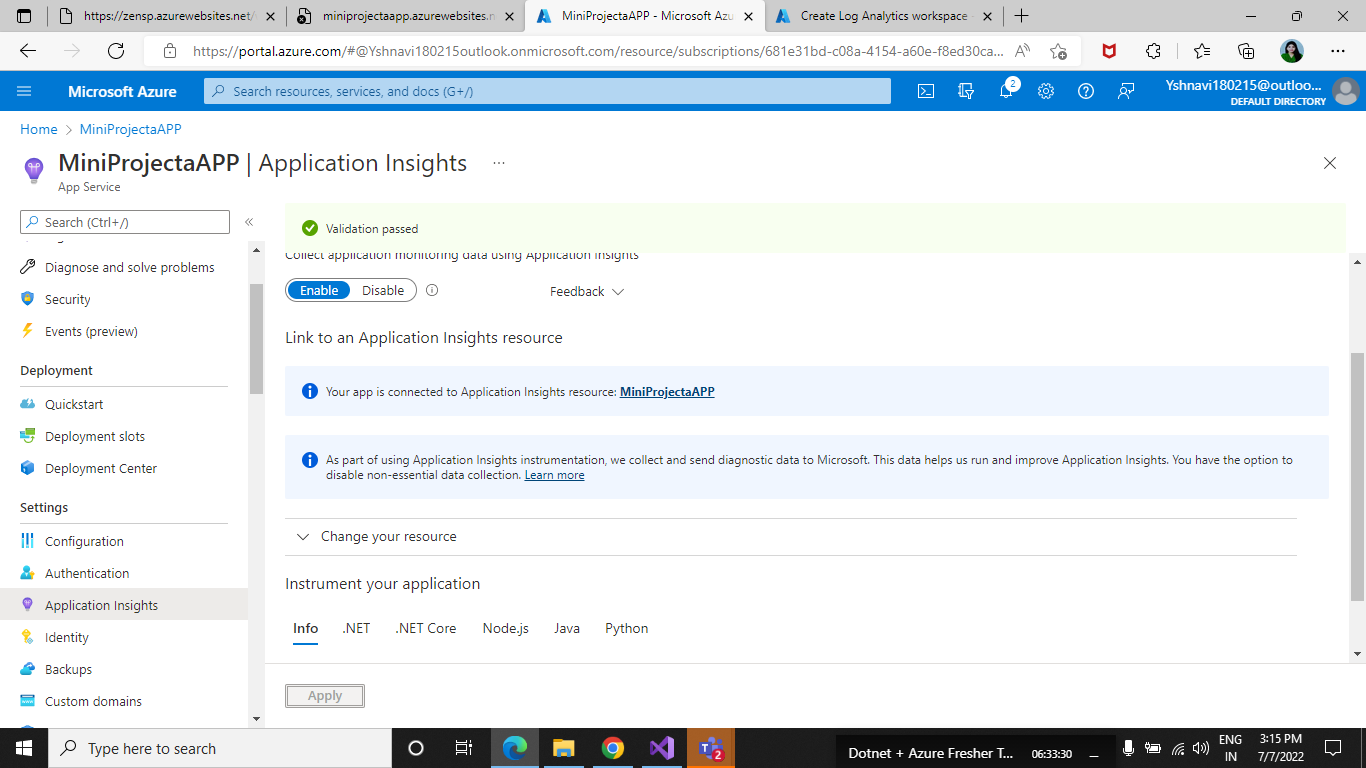
Application Insights:-

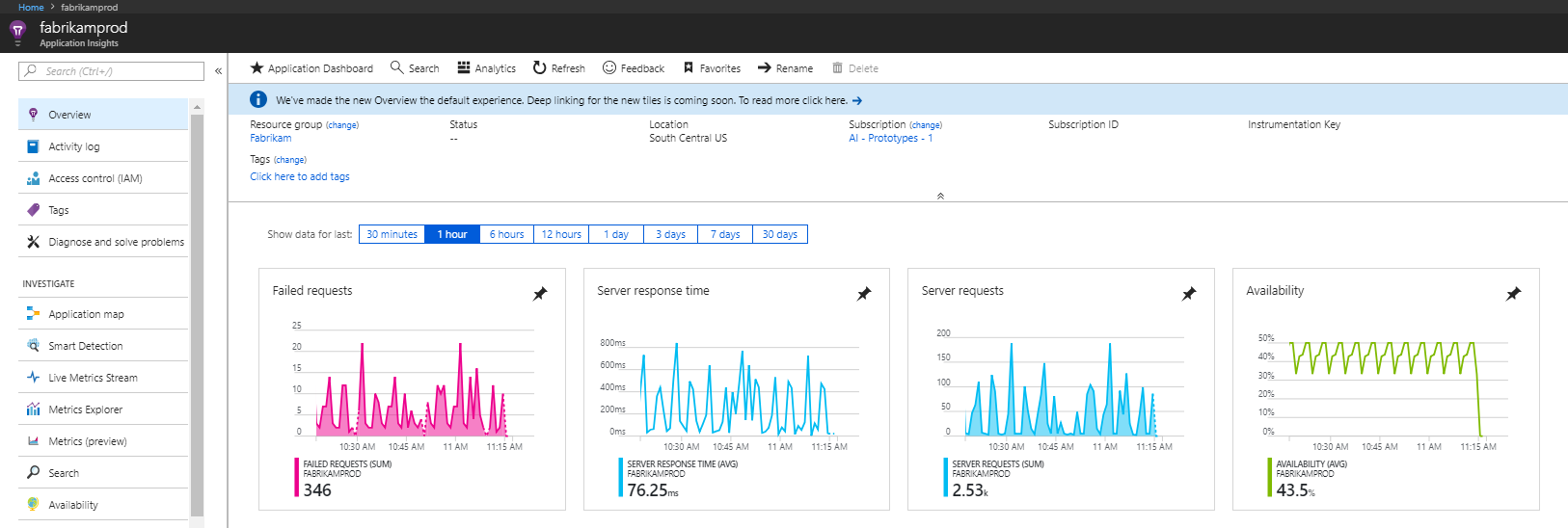
Application Insights can monitor Azure cloud service apps for availability ,performance failures and live web apps.Developers and DevOps professionals can ususge by combining data from Application insights SDKs with Azure Diagnostics data from your cloud services .

--Application Insights:-

* Supports a wide variety of platforms, including .NET, Node.js, Java, and Python.
* Works for apps hosted on-premises, hybrid, or on any public cloud.
* Integrates with DevOps processes.
* Has connection points to many development tools.
* Can monitor and analyze telemetry from mobile apps by integrating with Visual Studio App Cente.







Application Insights includes the following design considerations for cost optimization:

* Consider using sampling to reduce the amount of data that's sent:
* Sampling is a feature in Application Insights. It's a recommended way to reduce data traffic, data, and storage costs. Refer to [Sampling in Application Insights](https://docs.microsoft.com/en-us/azure/azure-monitor/app/sampling).
* Consider turning off collection for unneeded modules:
* On configuration files, you can enable or disable data modules and initializers for tracking data from your applications. Refer to [Application Insights for web pages](https://docs.microsoft.com/en-us/azure/azure-monitor/app/javascript#configuration).
* Consider limiting Asynchronous JavaScript and XML (AJAX) call tracing:
* AJAX calls can be limited to reduce costs. Refer to [Application Insights for web pages](https://docs.microsoft.com/en-us/azure/azure-monitor/app/javascript#configuration), which explains the fields and its configurations.

Configuration Swagger API :-

Swagger:-

Swagger allows you to describe the structure of your APIs so that machines can read them.

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.

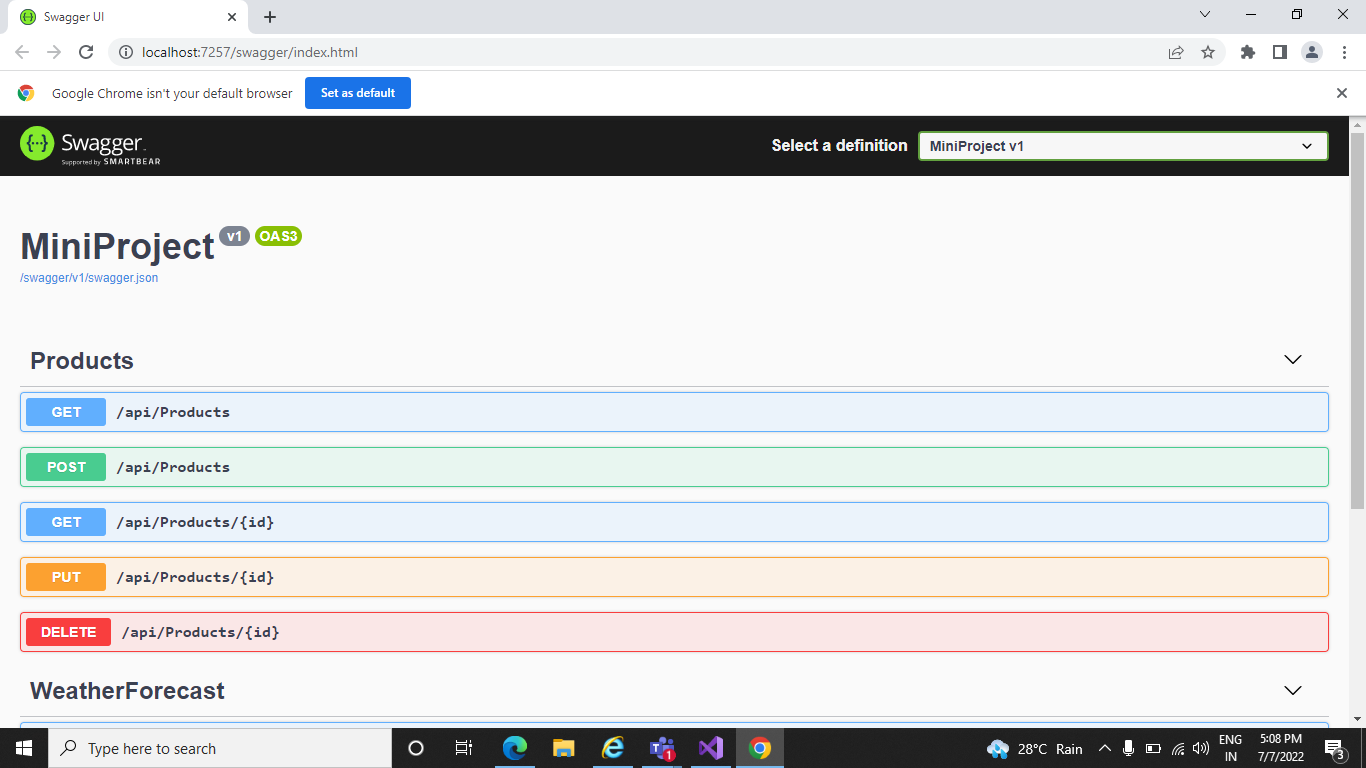
There are a few ways in which Swagger can help drive your API development further:-

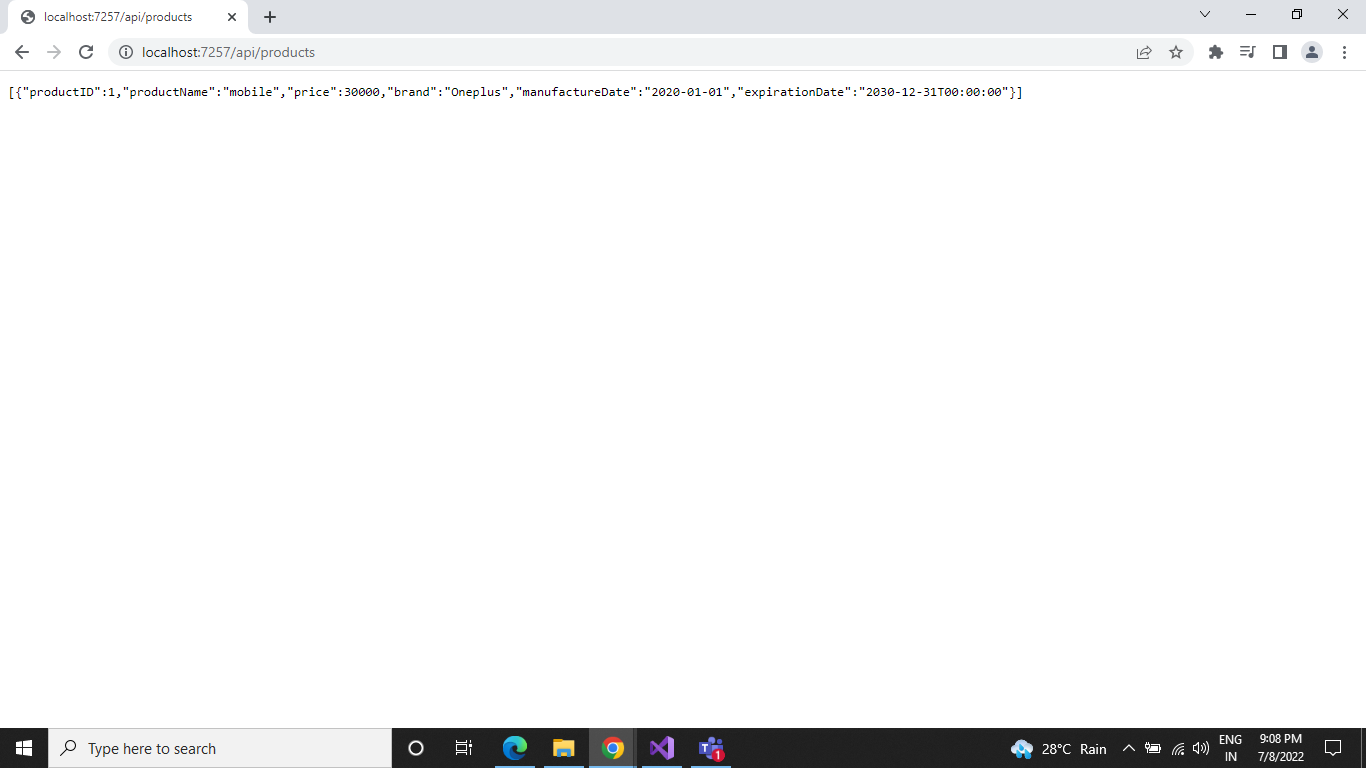
-Design-first users: Use Swagger Codegen to generate a server stub for your API. The only thing left is to implement the server logic – and your API is ready to go live!

-Use Swagger Codegen to generate client libraries for your API in over 40 languages.

-Use Swagger UI to generate interactive API documentation that lets your users try out the API calls directly in the browser.

-Use the spec to connect API-related tools to your API. For example, import the spec to SoapUI to create automated tests for your API.



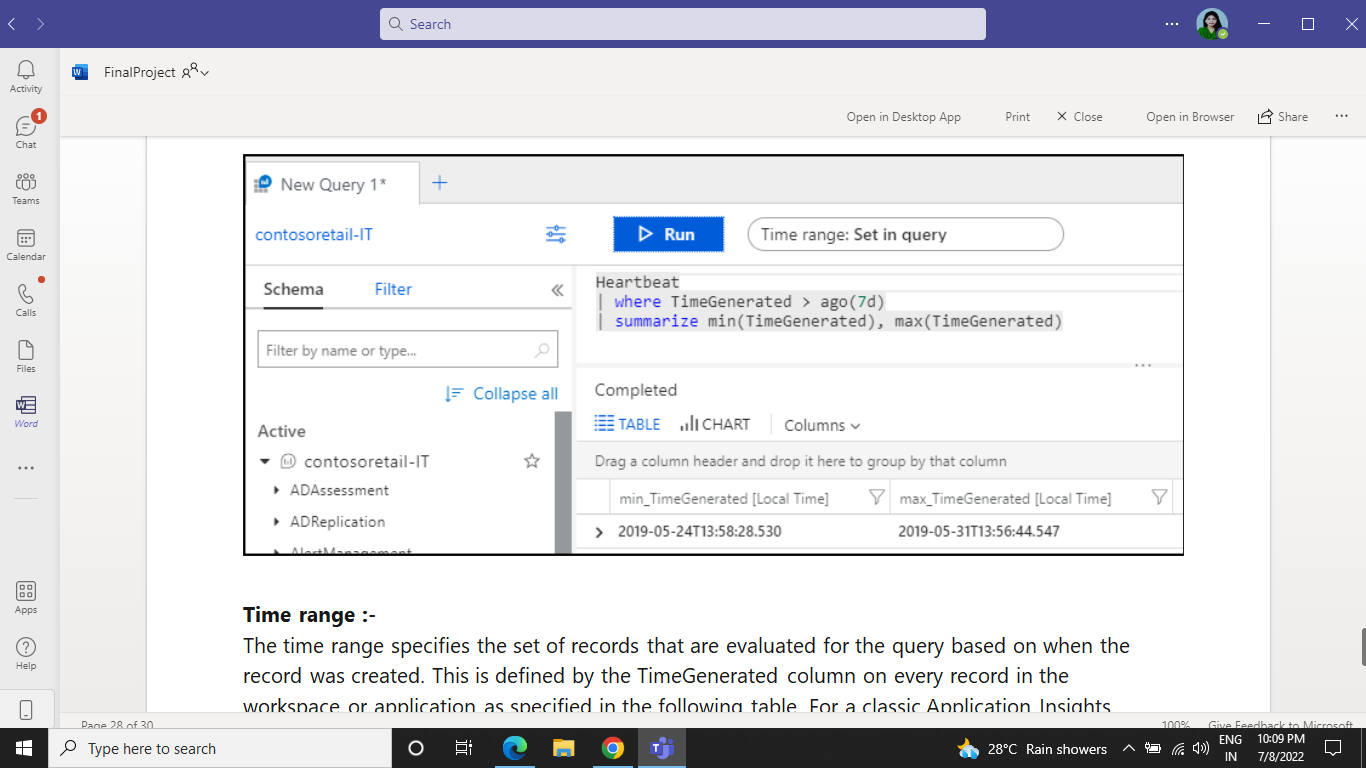


Log Analytics:-

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

Query scope :-

The query scope defines the records that are evaluated by the query. This will usually include all records in a single Log Analytics workspace or Application Insights application. Log Analytics also allows you to set a scope for a particular monitored Azure resource. This allows a resource owner to focus only on their data, even if that resource writes to multiple workspaces.



Time Range:-

The time range specifies the set of records that are evaluated for the query based on when the record was created. This is defined by the TimeGenerated column on every record in the workspace or application as specified in the following table. For a classic Application Insights application, the timestamp column is used for the time range.

Set the time range by selecting it from the time picker at the top of the Log Analytics window. You can select a predefined period or select Custom to specify a specific time range.

