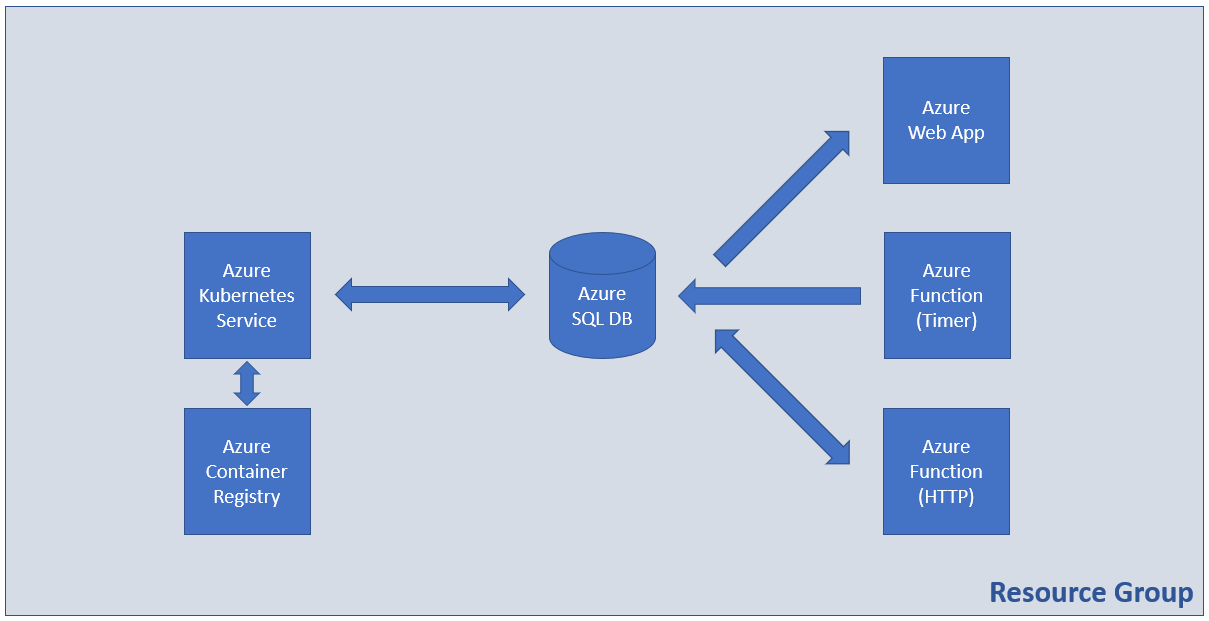
# Exam: AzE (2022.02) – 2022.04.09 - Variant #2

### Rules

* Be sure to follow exactly the **IP addresses**, **resource names, file names, etc.** as requested in the tasks
* Tasks execution order should not be derived from the order in which they are listed below
* Тhere are tasks that depend on the successful completion of one or more other tasks
* All resources must be created in **one region** (it can be any, for example ***West Europe***)
* For this exam try to use the smallest and cheapest possible resource option that will do the job
* You are free to setup public IP addresses or other additional options to virtual machines or other resources for debugging purposes
* **The aim is** not to build a production-ready solution but a **working one**

### Goal

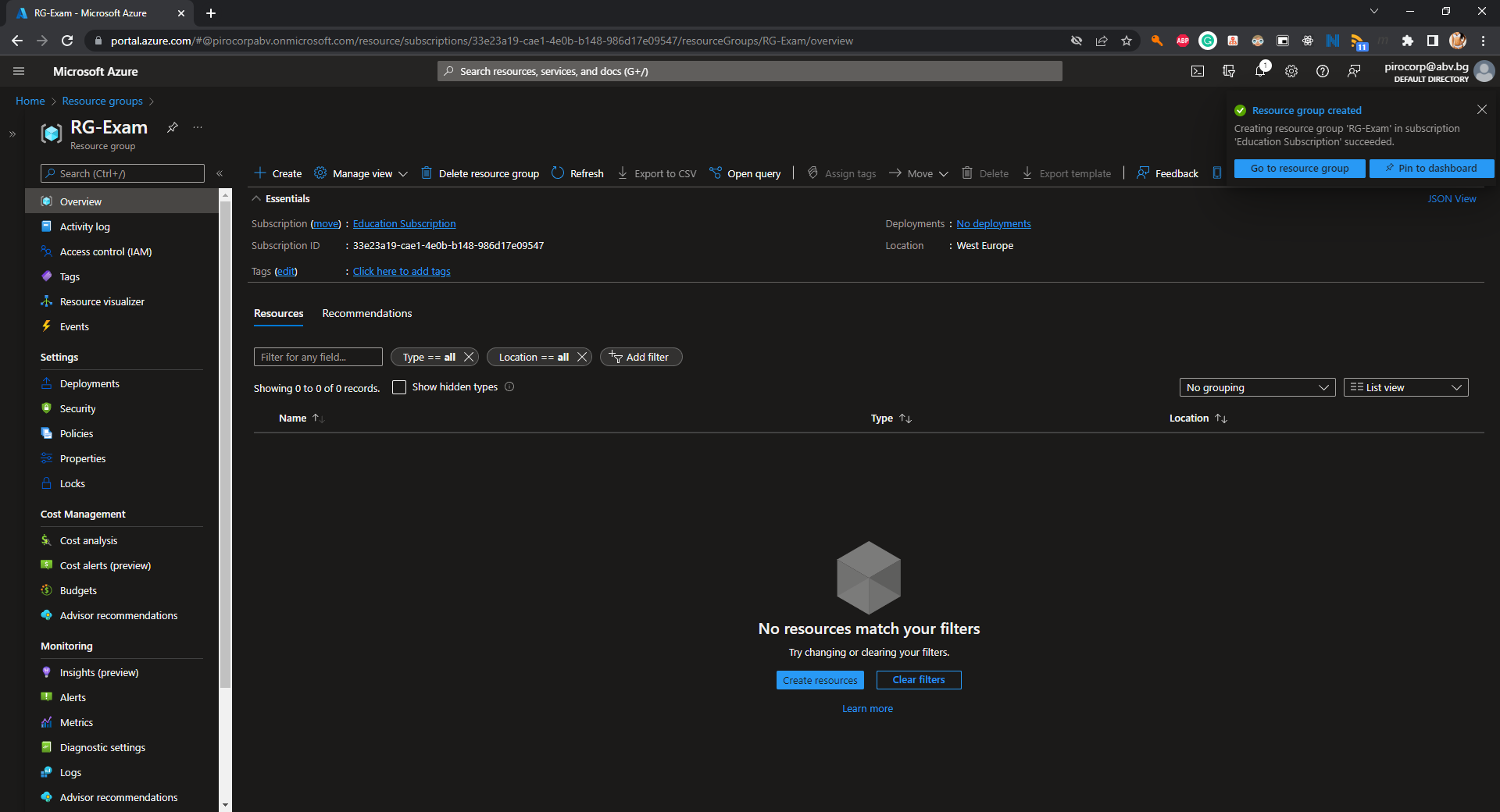


Supporting files set can be downloaded from [**https://zahariev.pro/q/aze-exam-20220409-variant-2.zip**](https://zahariev.pro/q/aze-exam-20220409-variant-2.zip) or from the **Regular Exam** section of the course site

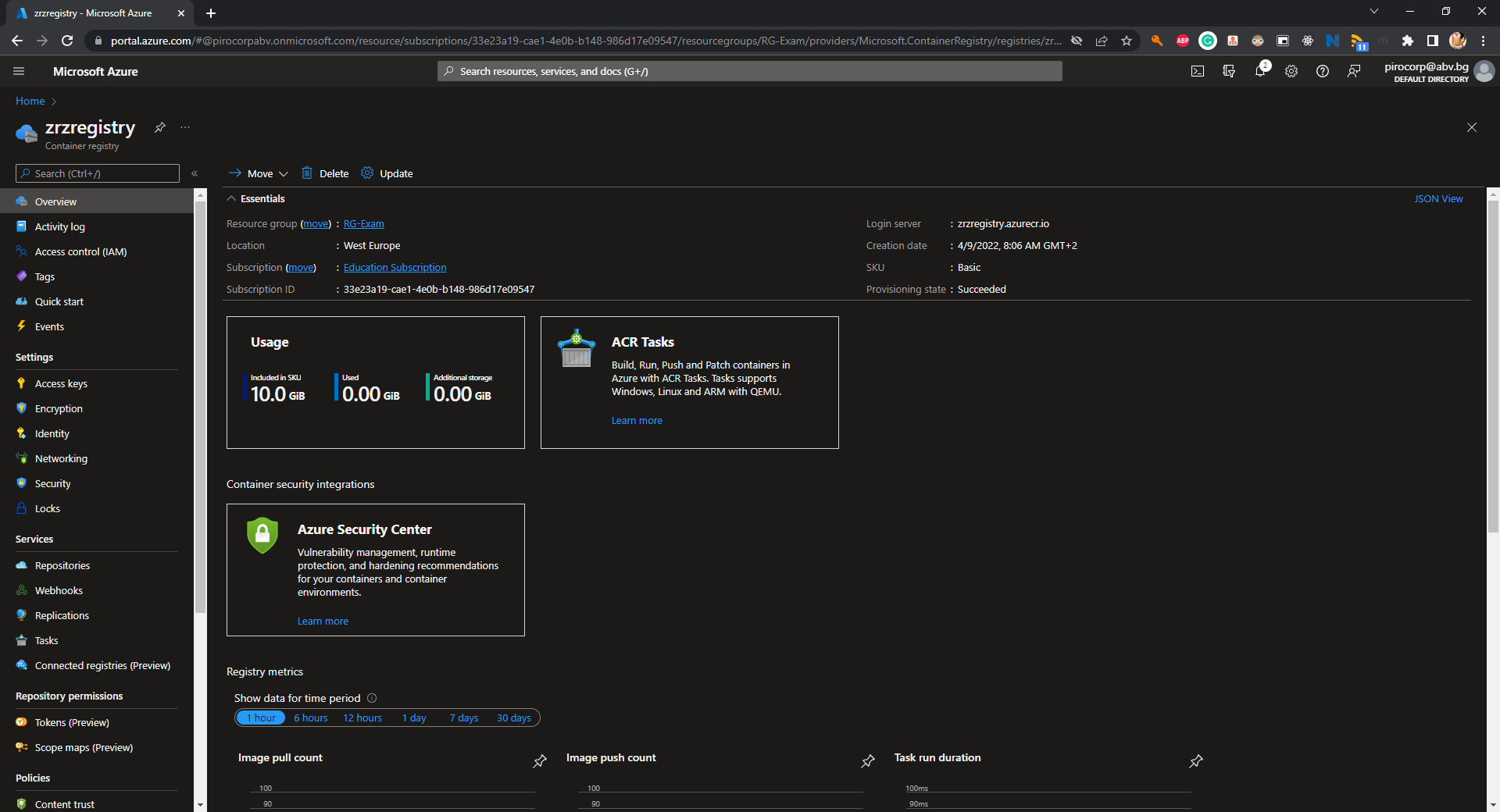
### Tasks

#### Infrastructure - 5 tasks, 11 pts

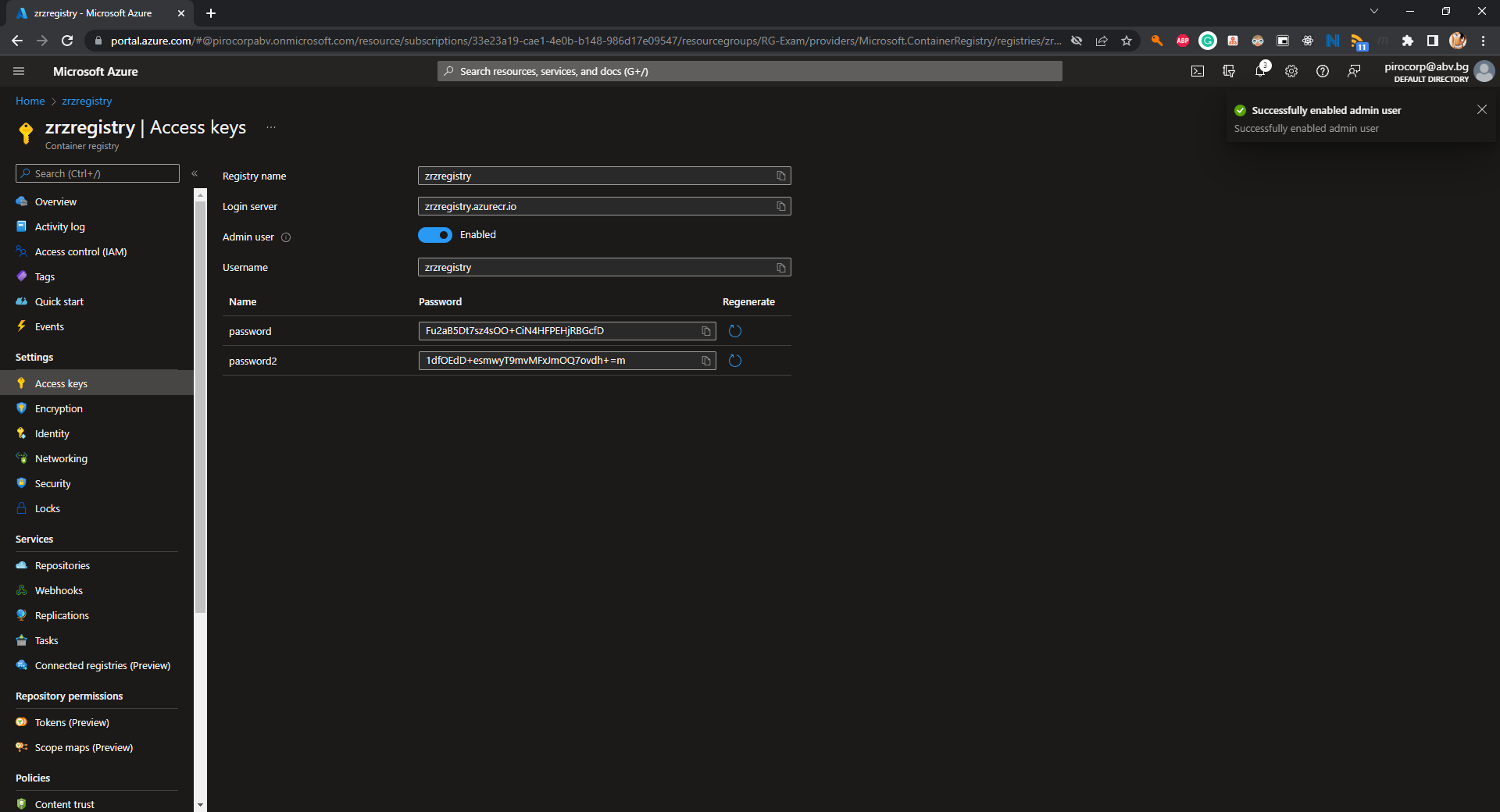
* (T101, 1 pts) Create a resource group named **RG-Exam**



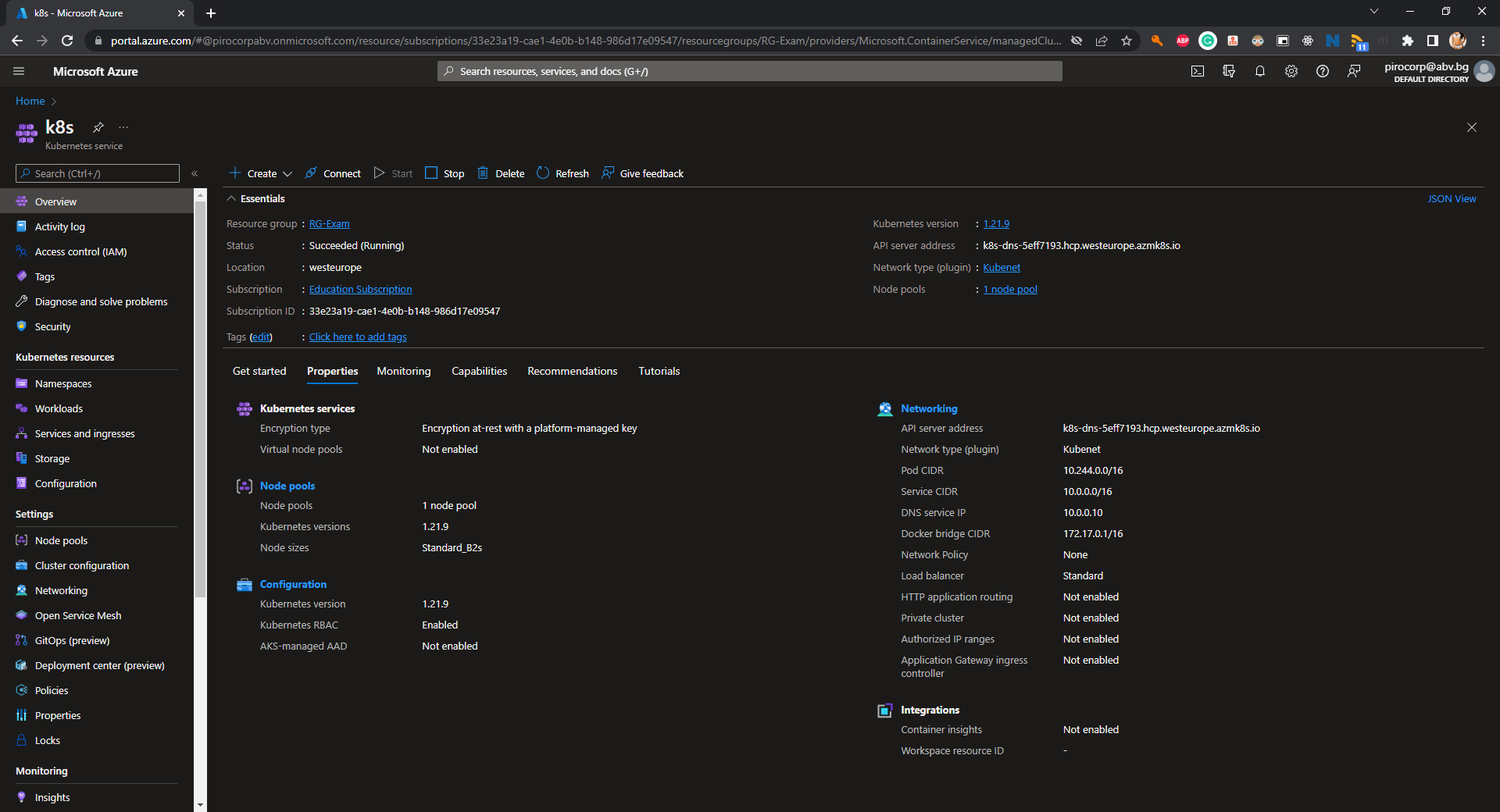
* (T102, 2 pts) Create a container registry with Basic SKU

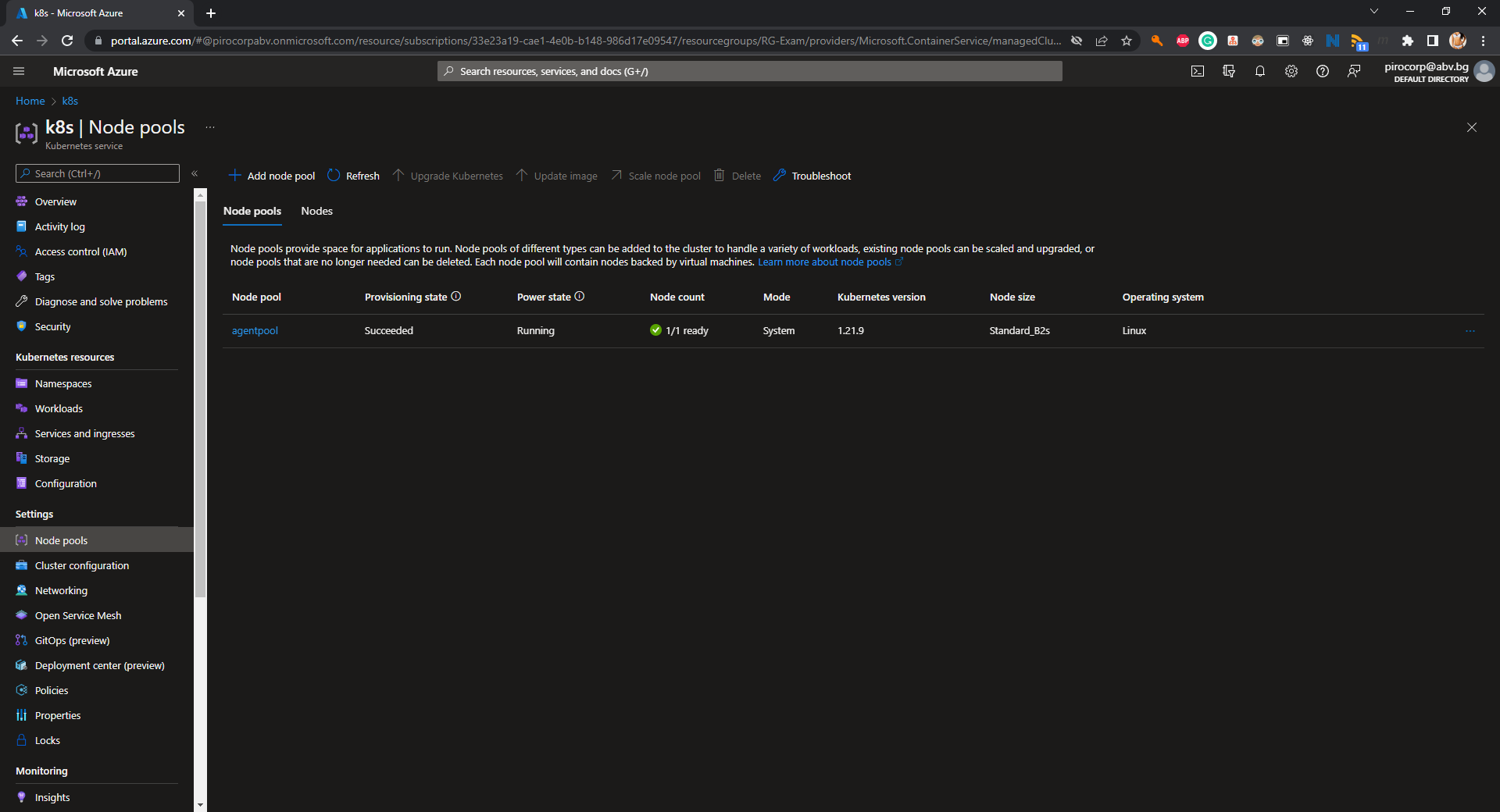


* (T103, 2 pts) Enable the Admin user

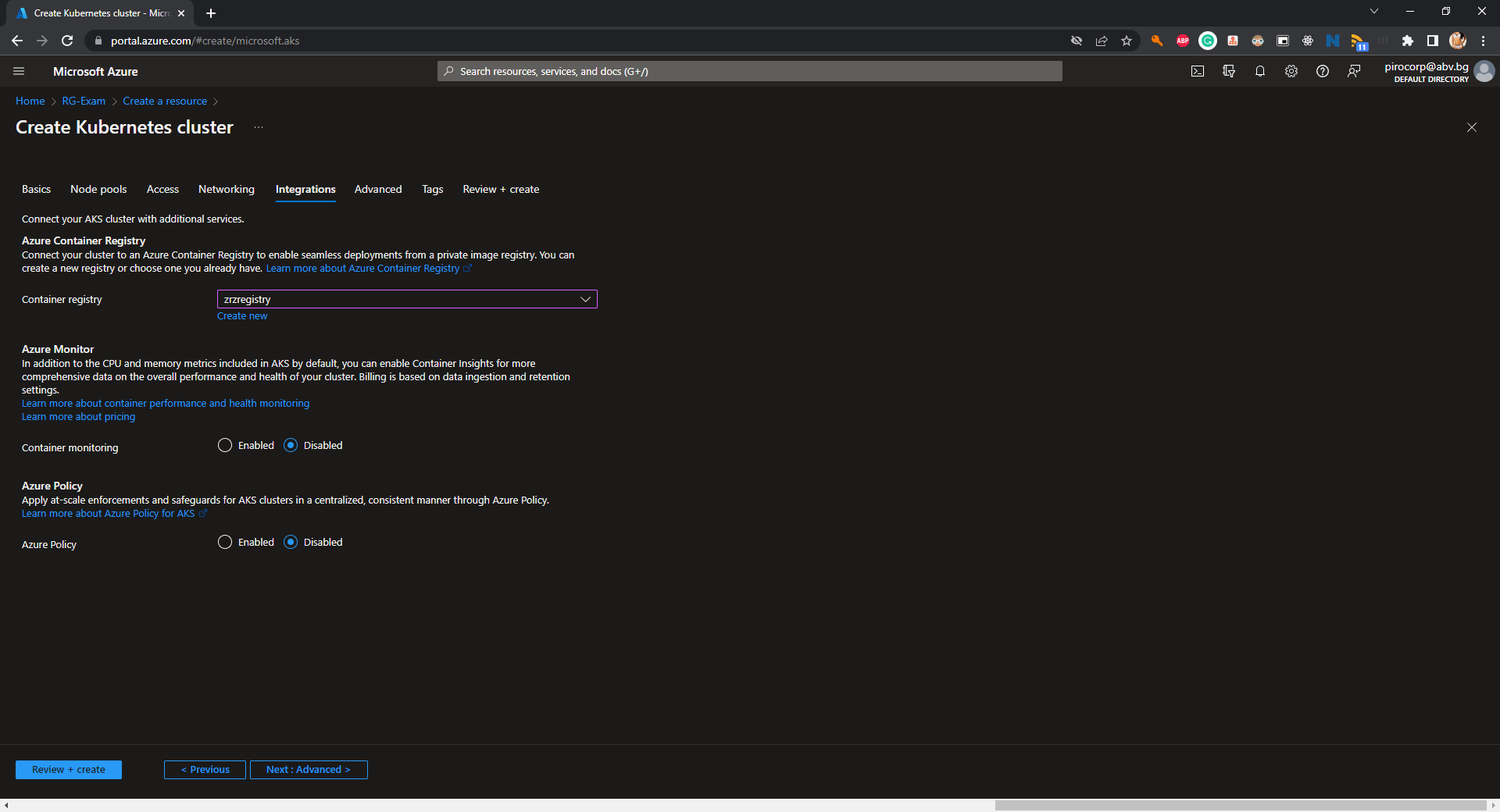


* (T104, 4 pts) Create an Azure Kubernetes Service resource with one node



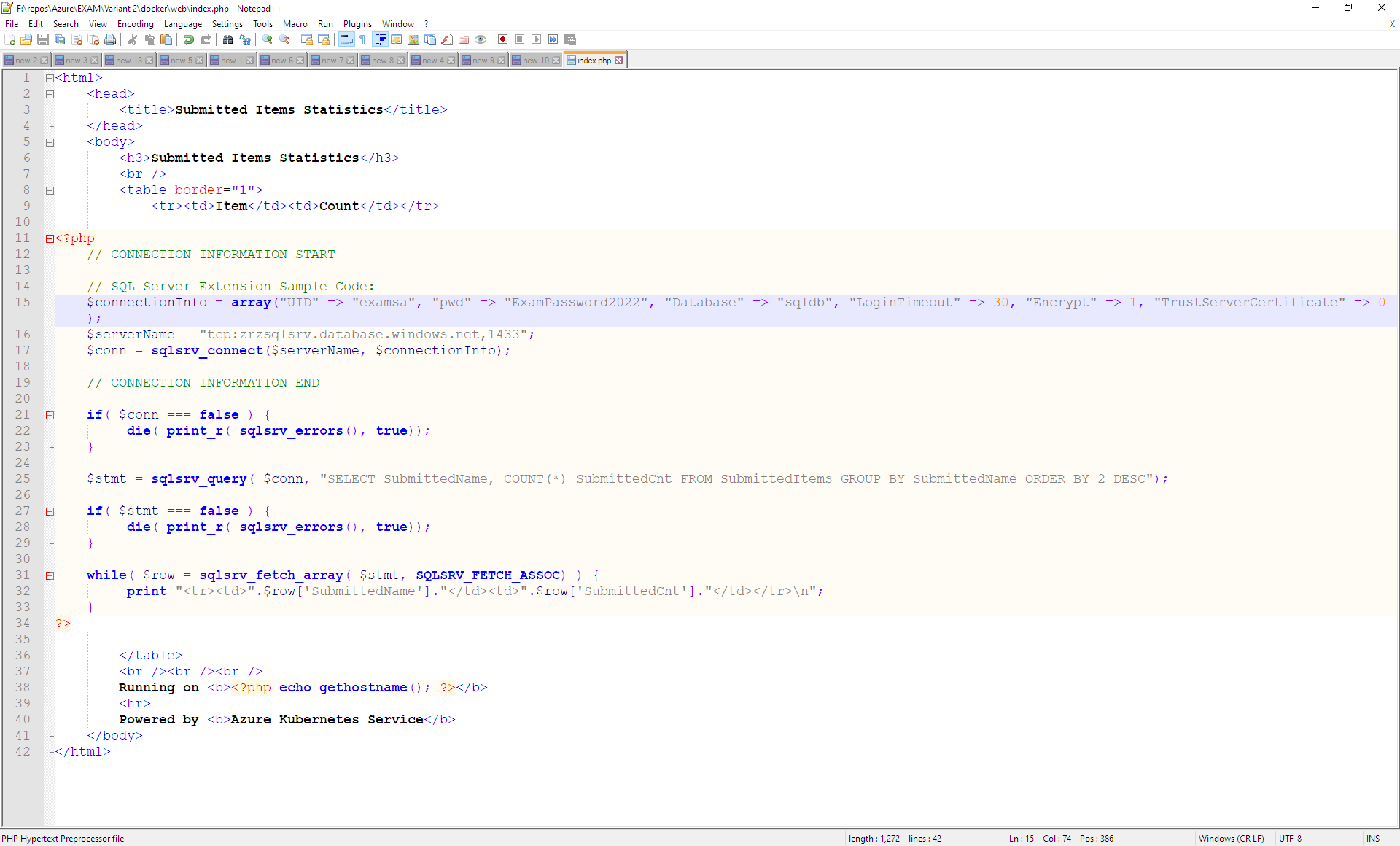


* (T105, 2 pts) Link the ACR to the AKS



#### Containers and Images - 7 tasks, 13 pts

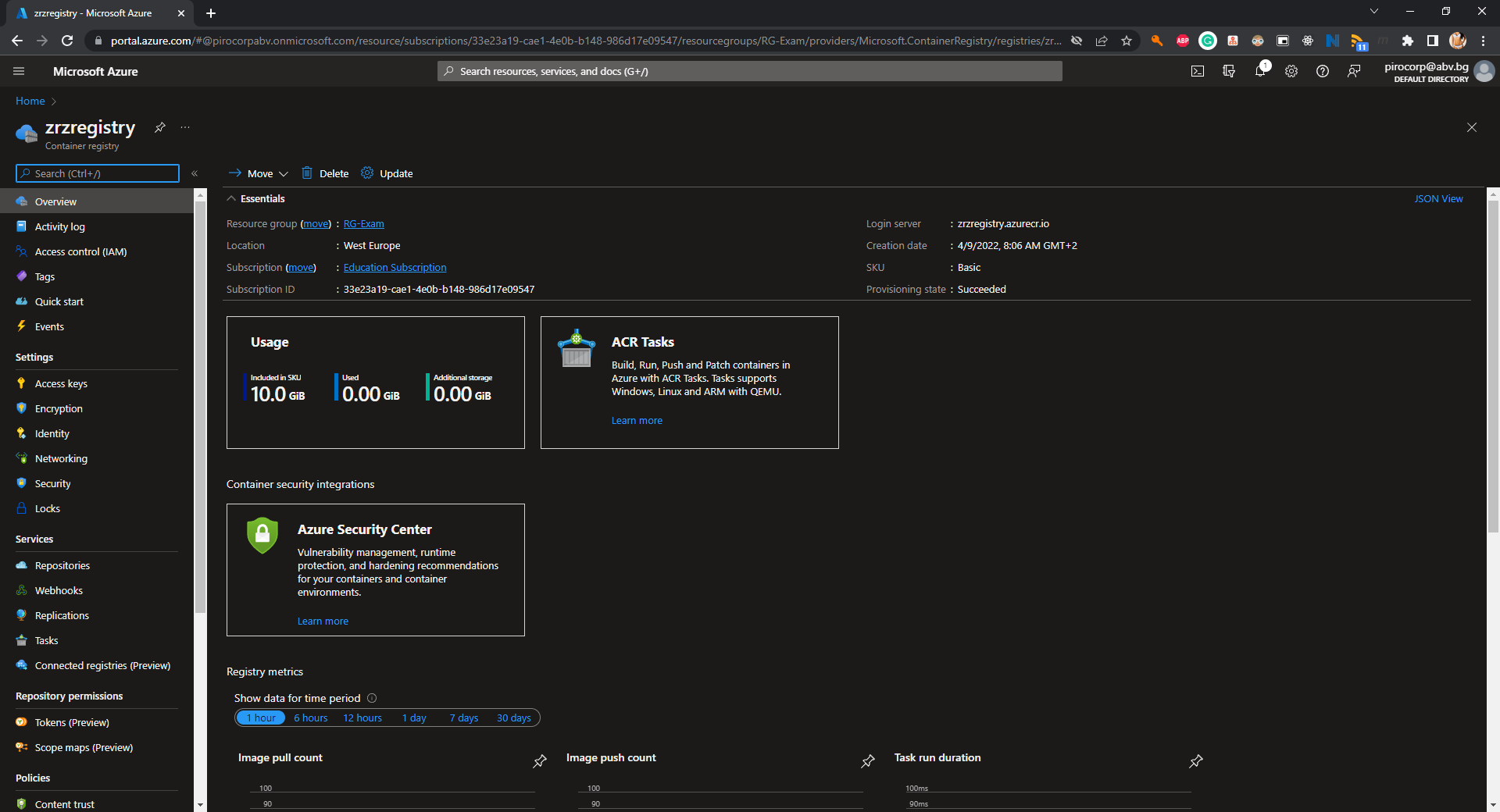
* (T201, 2 pts) Add the SQL connection string to the **index.php** file in the **docker/web** folder

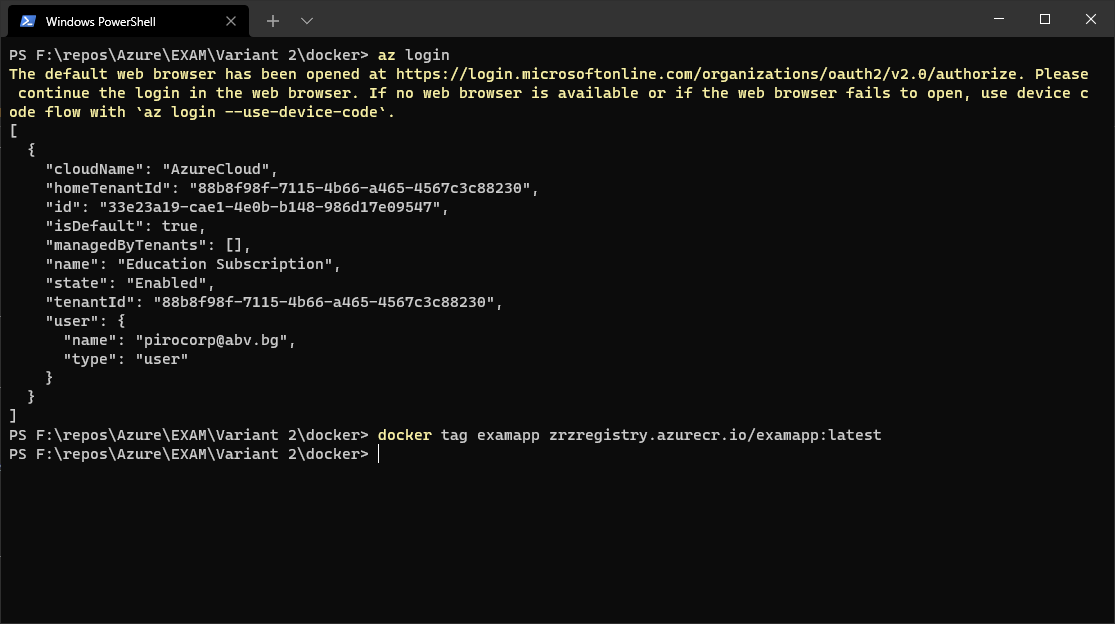


* (T202, 2 pts) Build the Docker image from the **Dockerfile** that is in the **docker/** folder

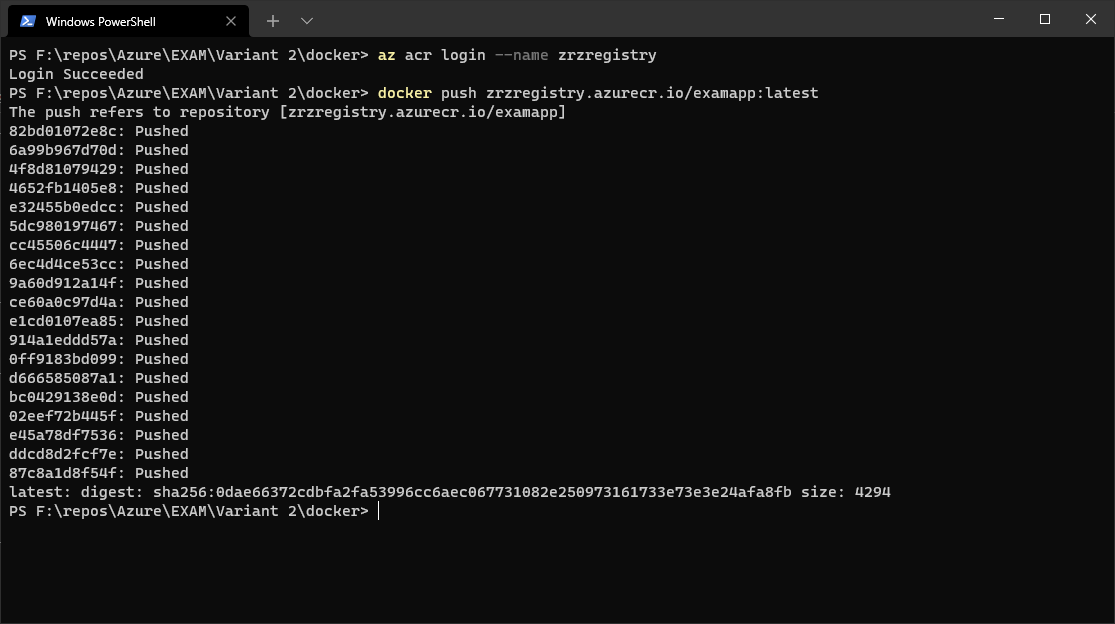


* (T203, 1 pts) Tag the Docker image for the Azure Container Registry

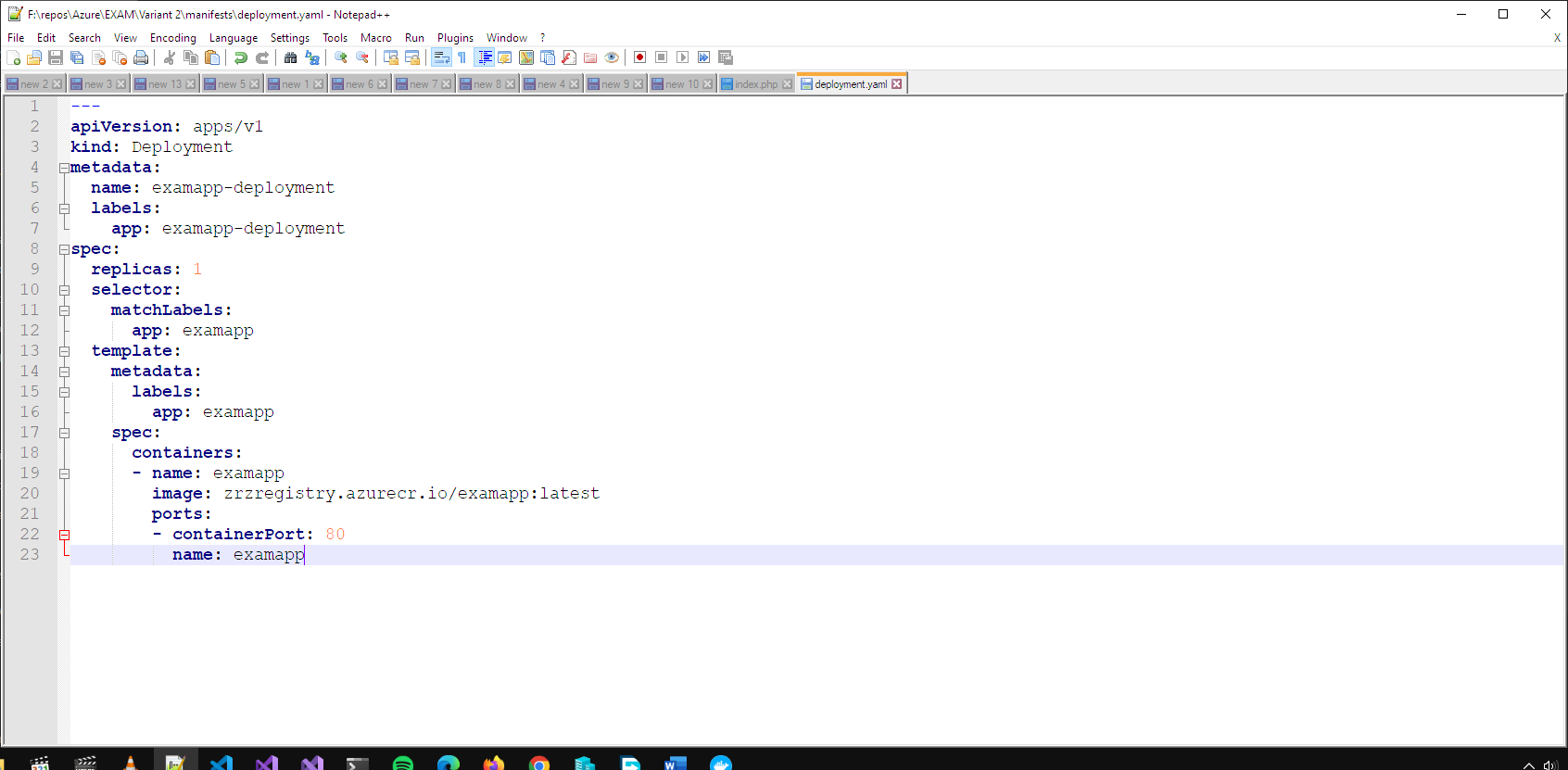




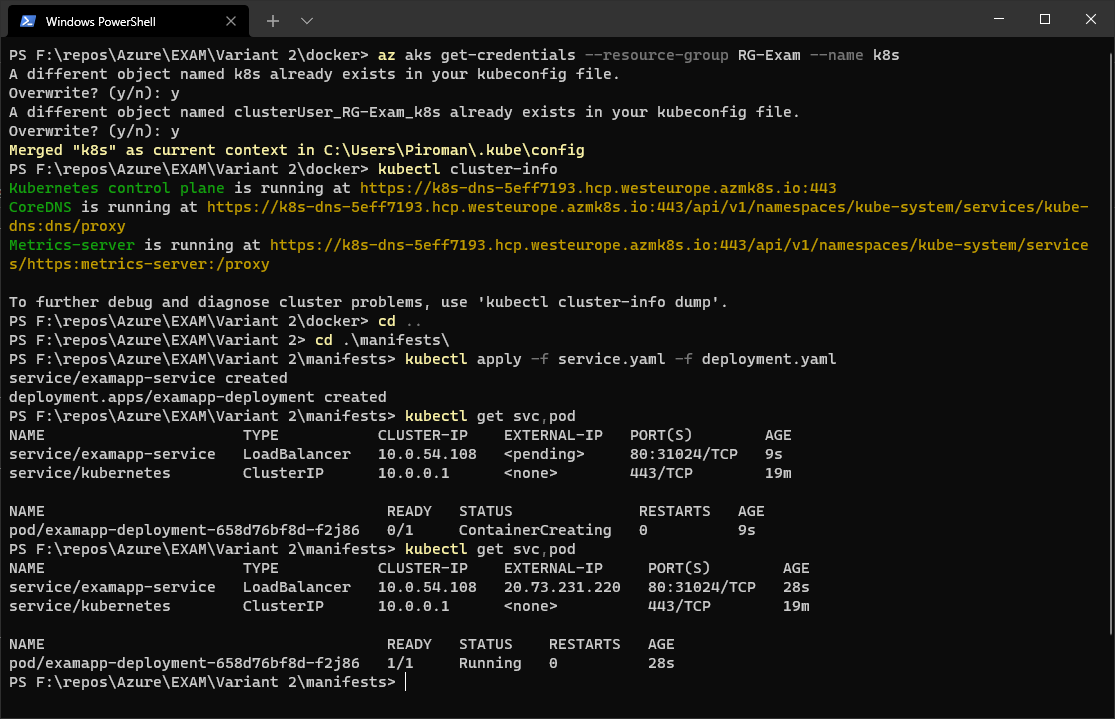
* (T204, 1 pts) Publish the Docker image to the Azure Container Registry



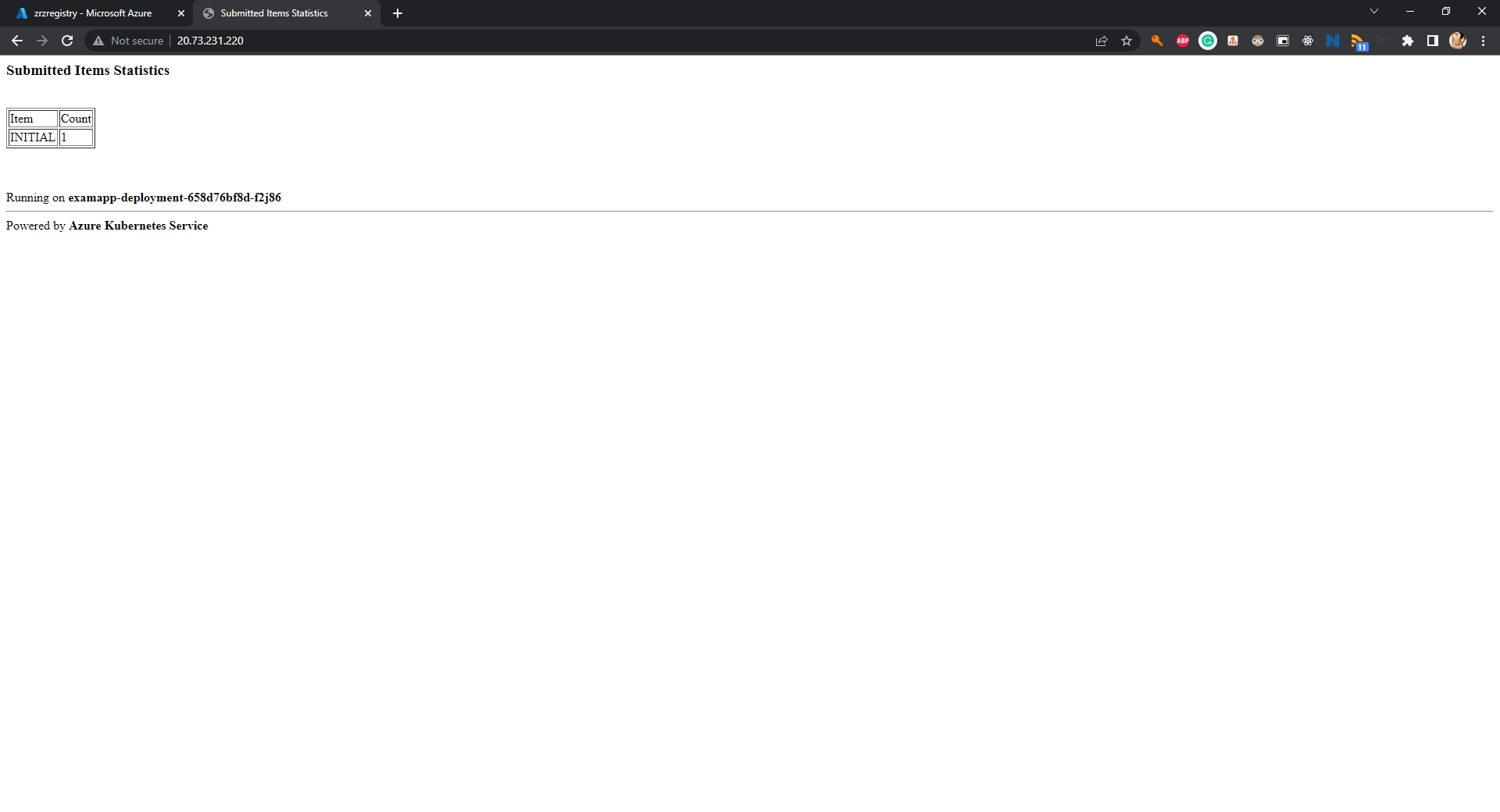
* (T205, 3 pts) Adjust the **deployment.yaml** file in the **manifests/** folder to point to the published Docker image



* (T206, 2 pts) Publish the manifests to the Kubernetes cluster (Azure Kubernetes Service)

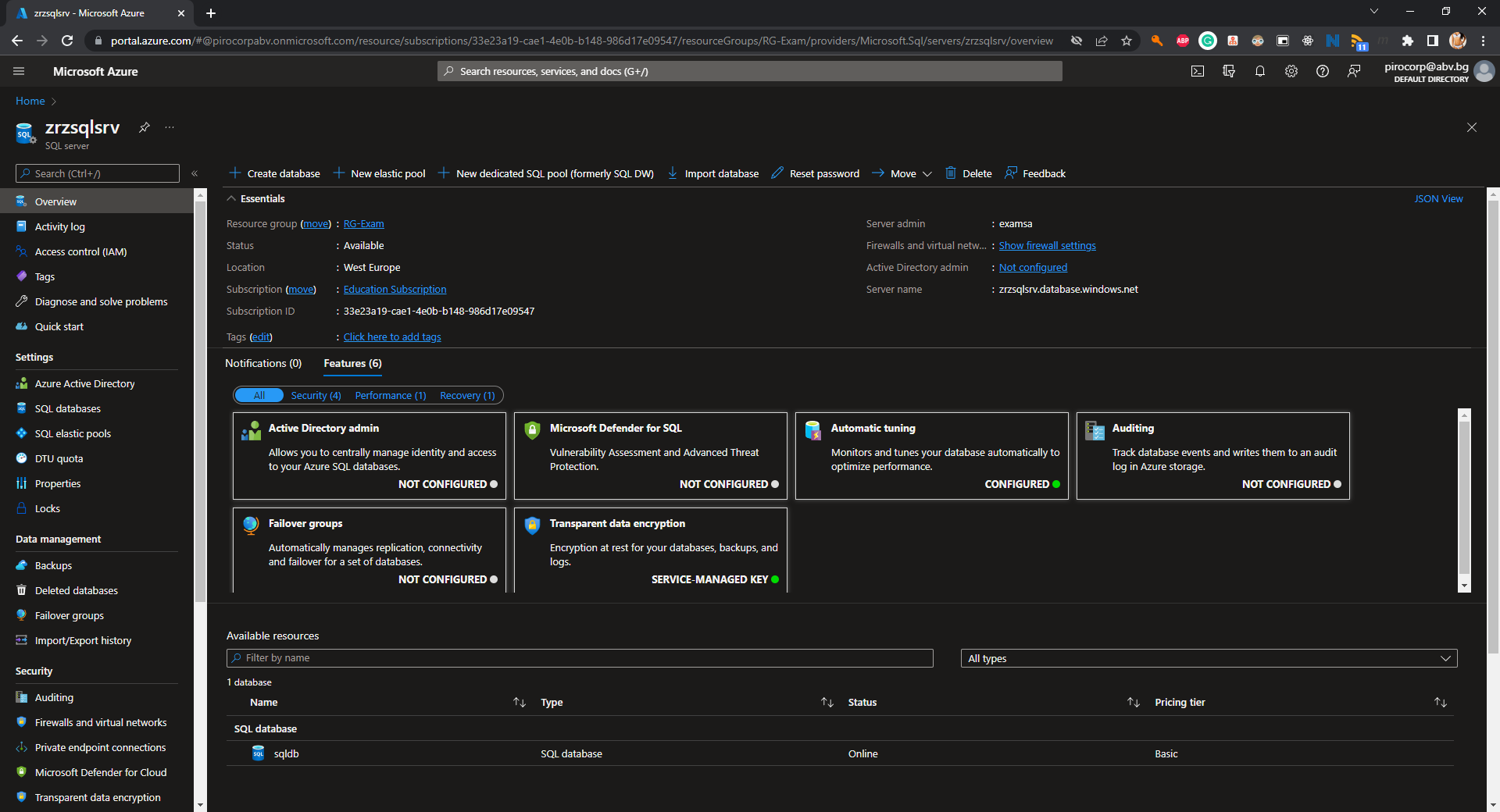


* (T207, 2 pts) Make sure that the app is working and showing correct results



#### Databases - 3 tasks, 7 pts

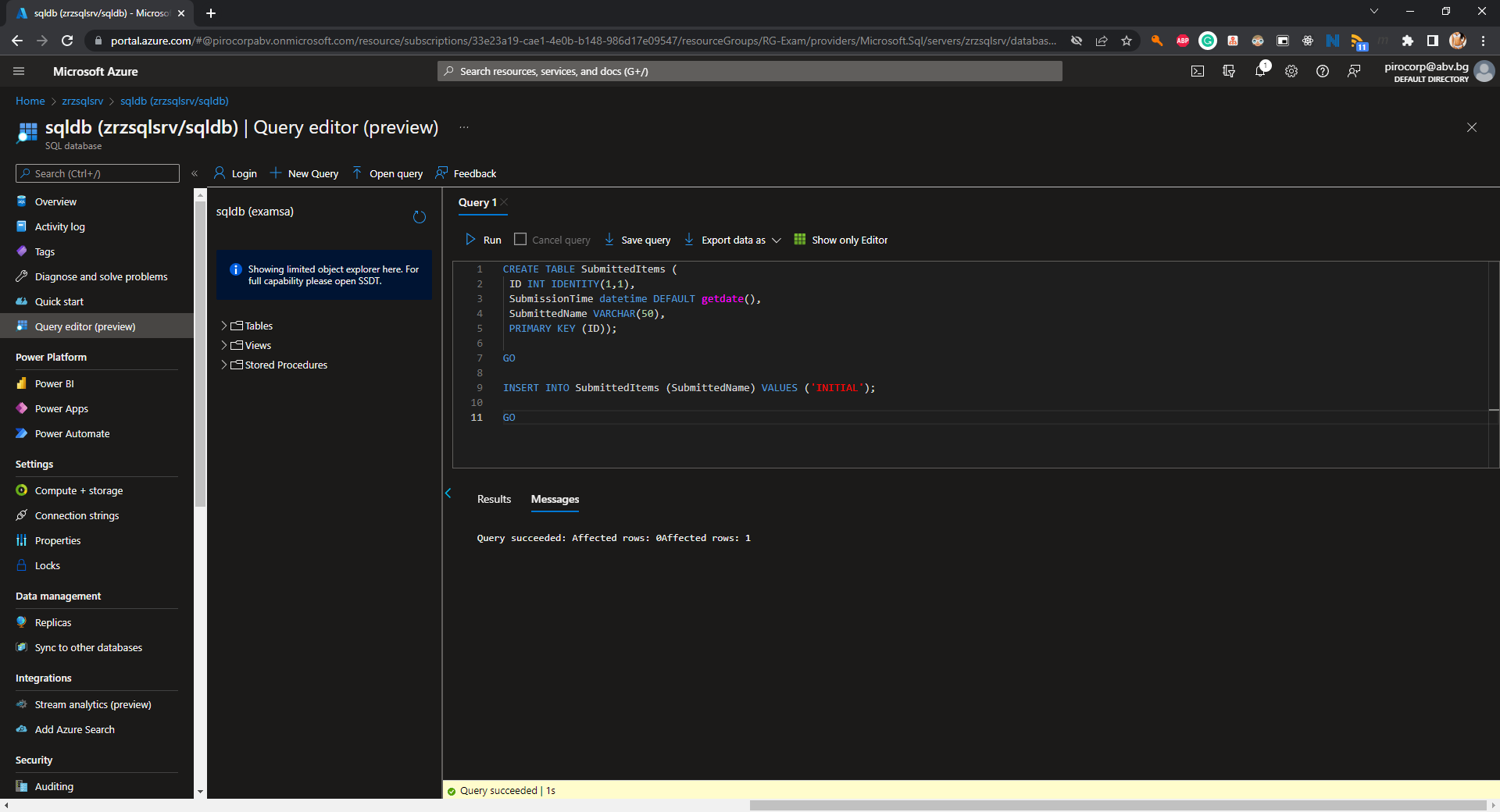
* (T301, 3 pts) Create an Azure SQL Database resource (incl. a server)

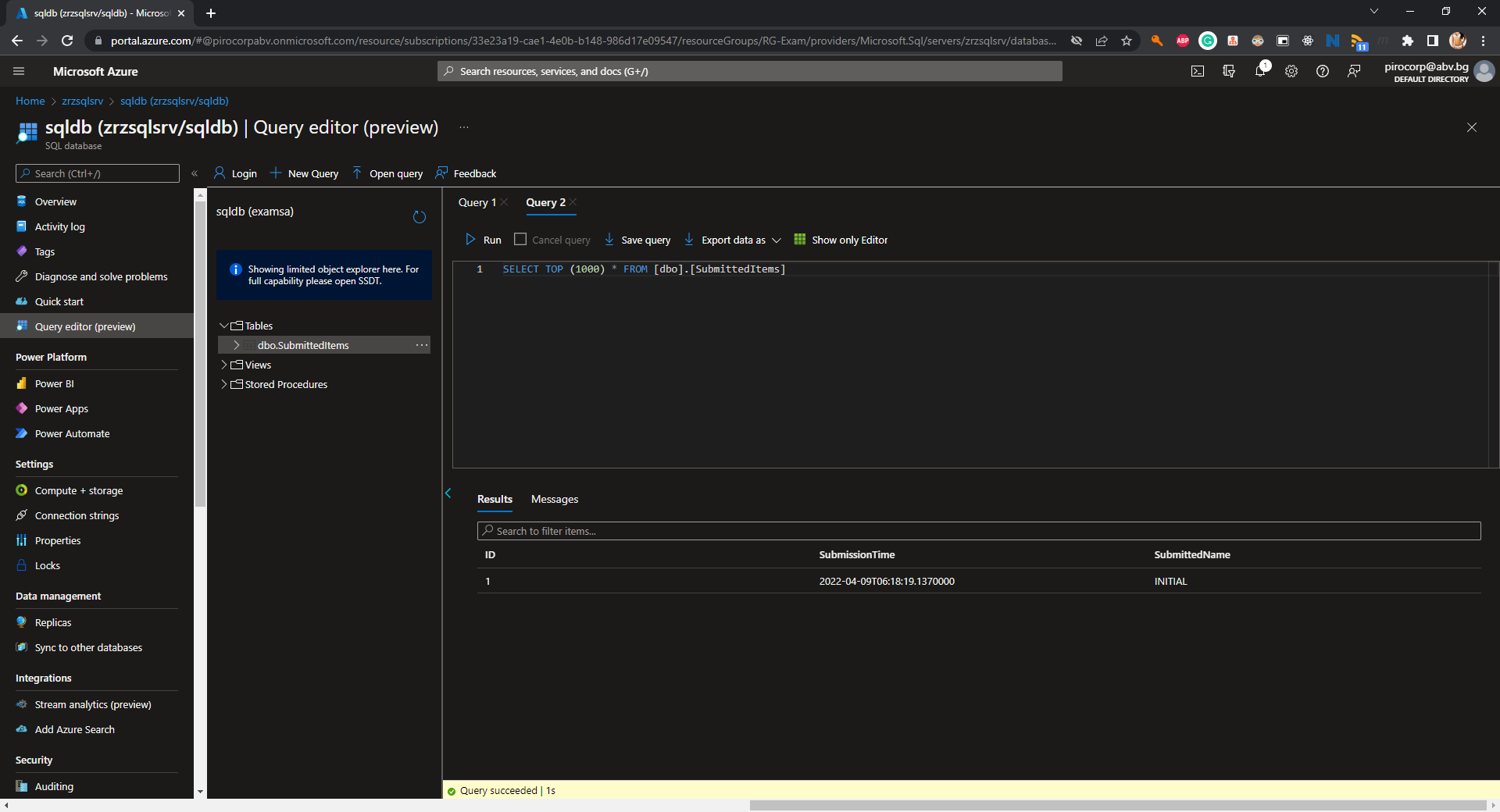


* (T302, 2 pts) Configure the connectivity to the server



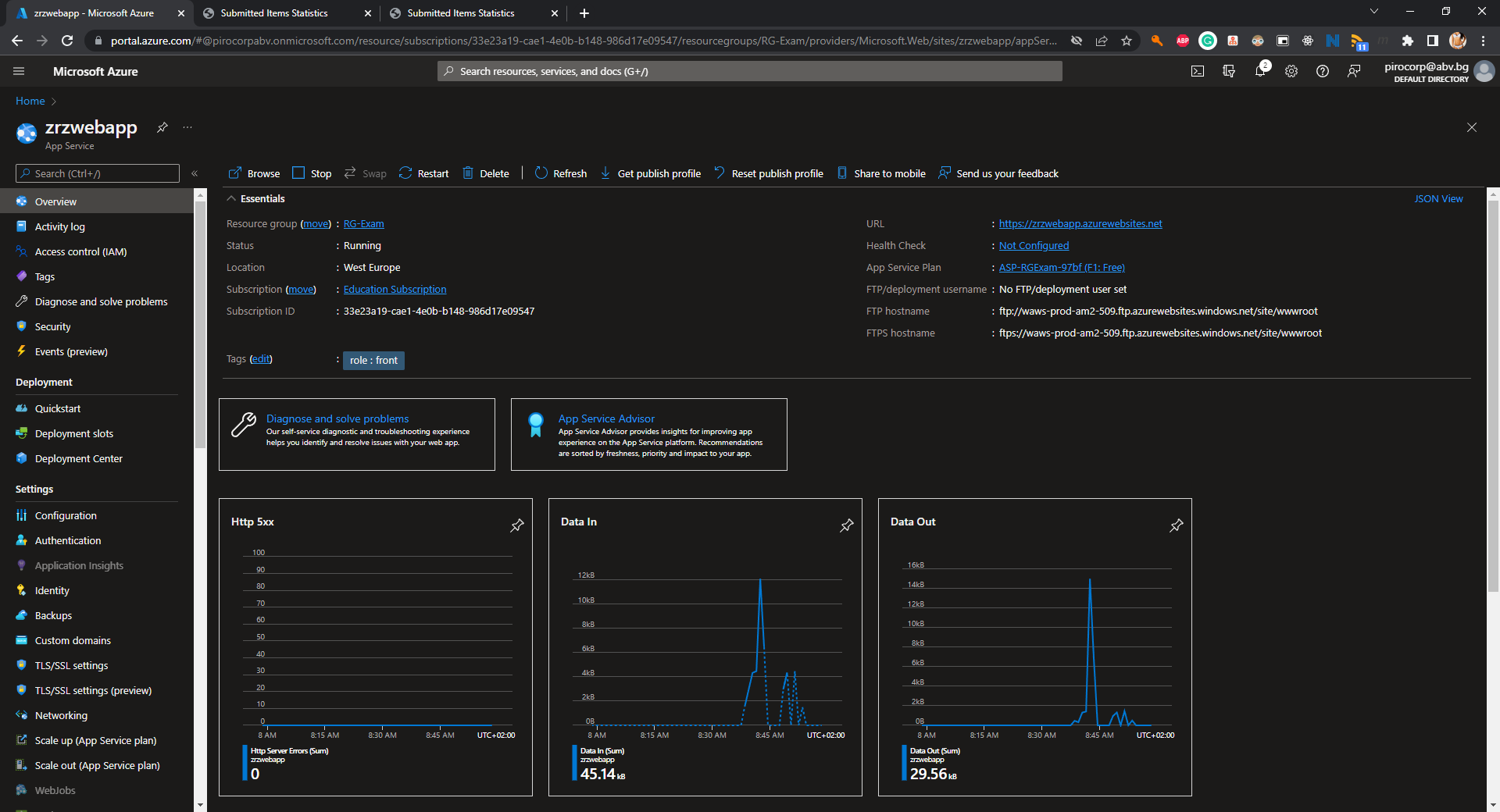
* (T303, 2 pts) Initialize the database with the help of the **sql/create-structures.sql** file (part of the supporting files set)

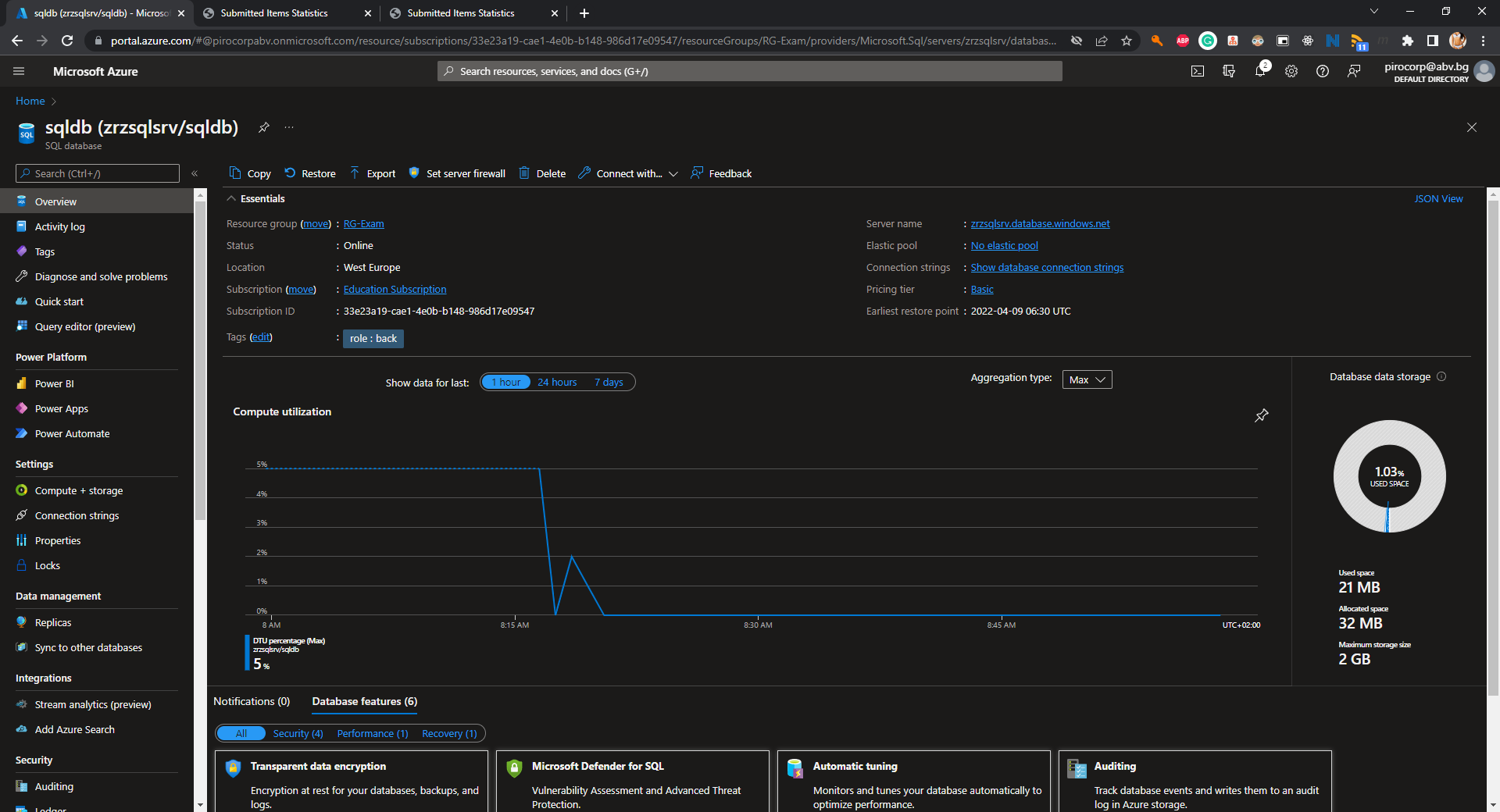




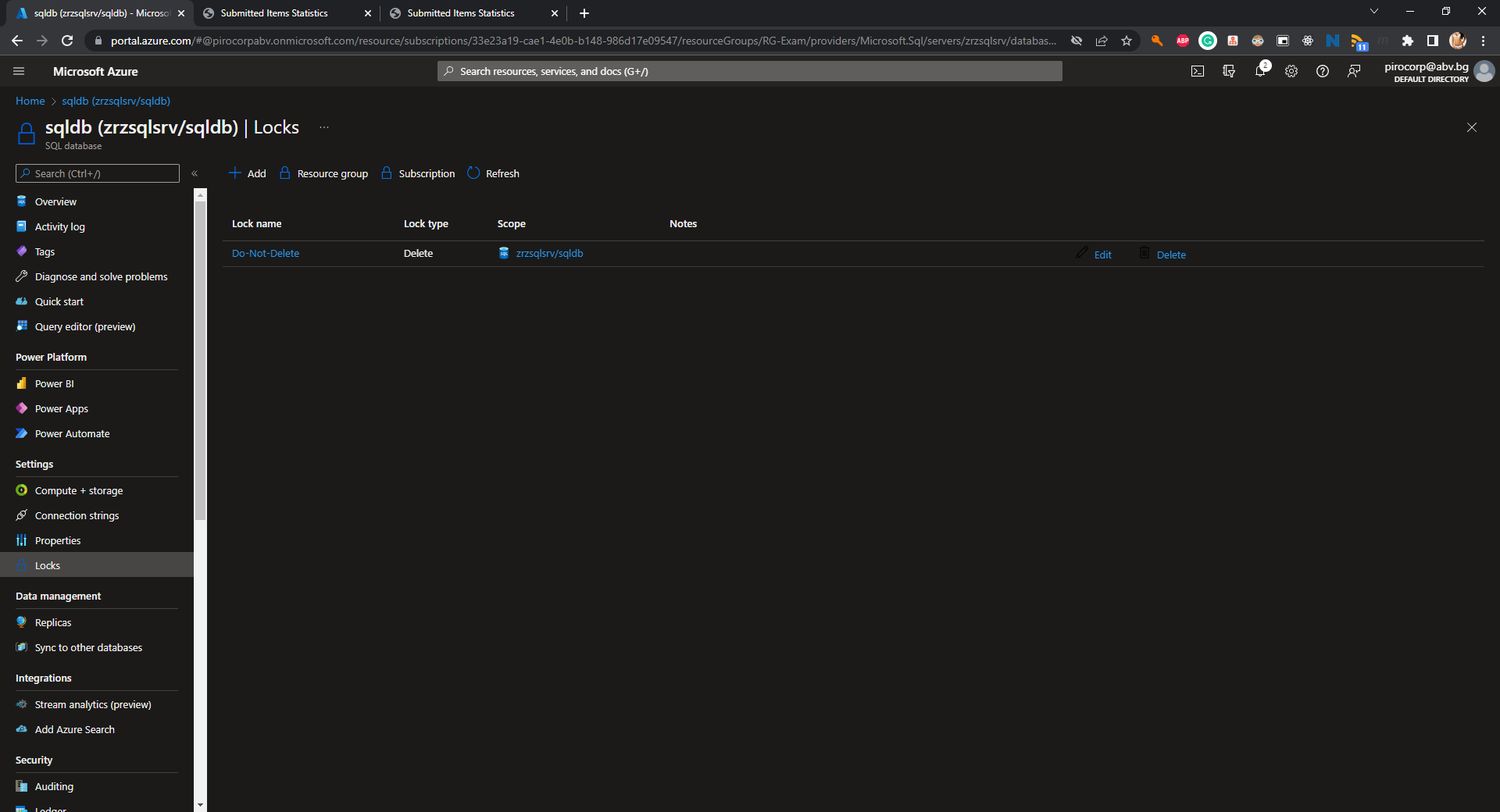
#### Organizational - 4 tasks, 7 pts

* (T401, 2 pts) Apply a tag **role** with value of **front** on the web app and with value of **back** on the database

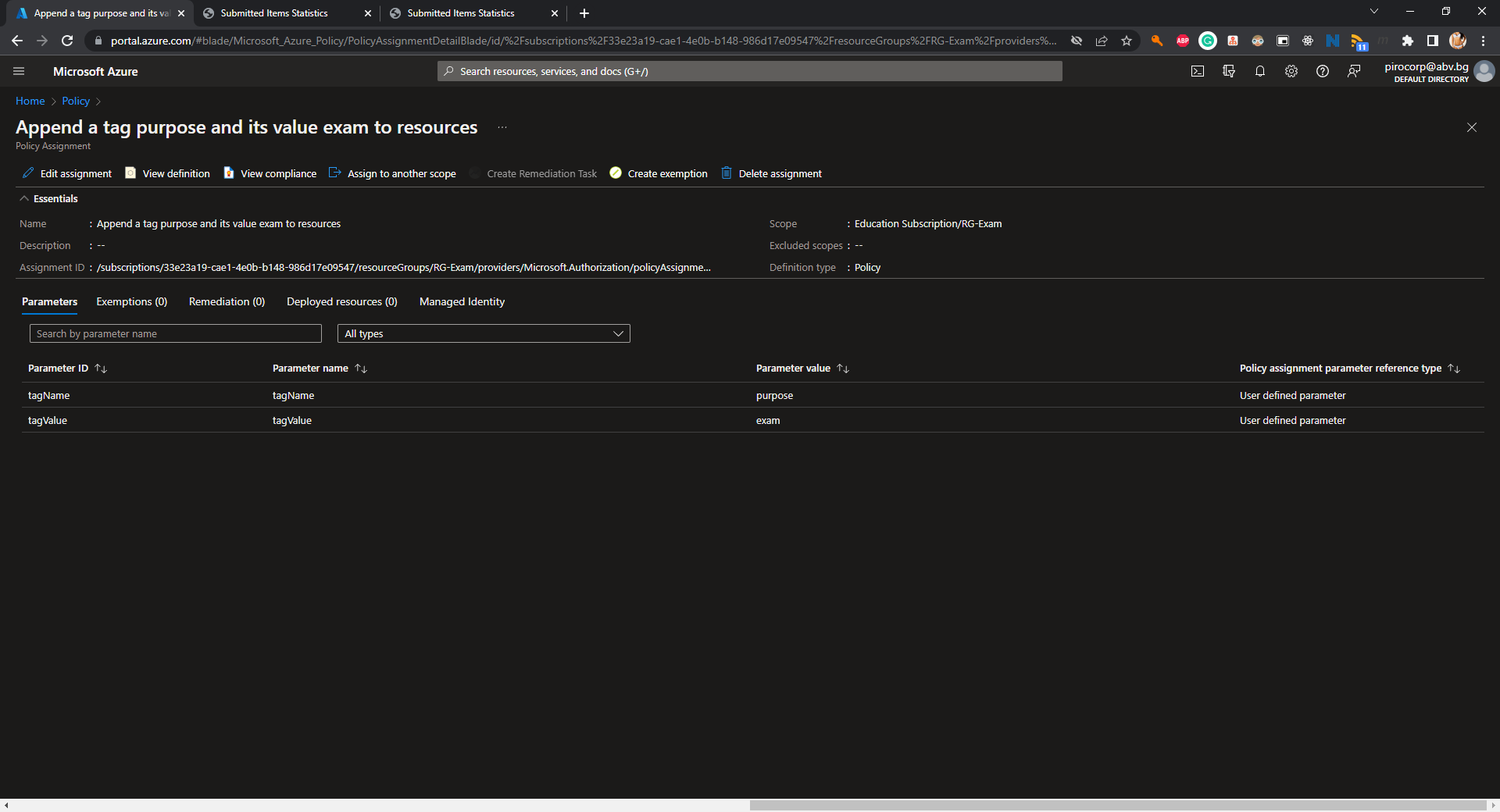




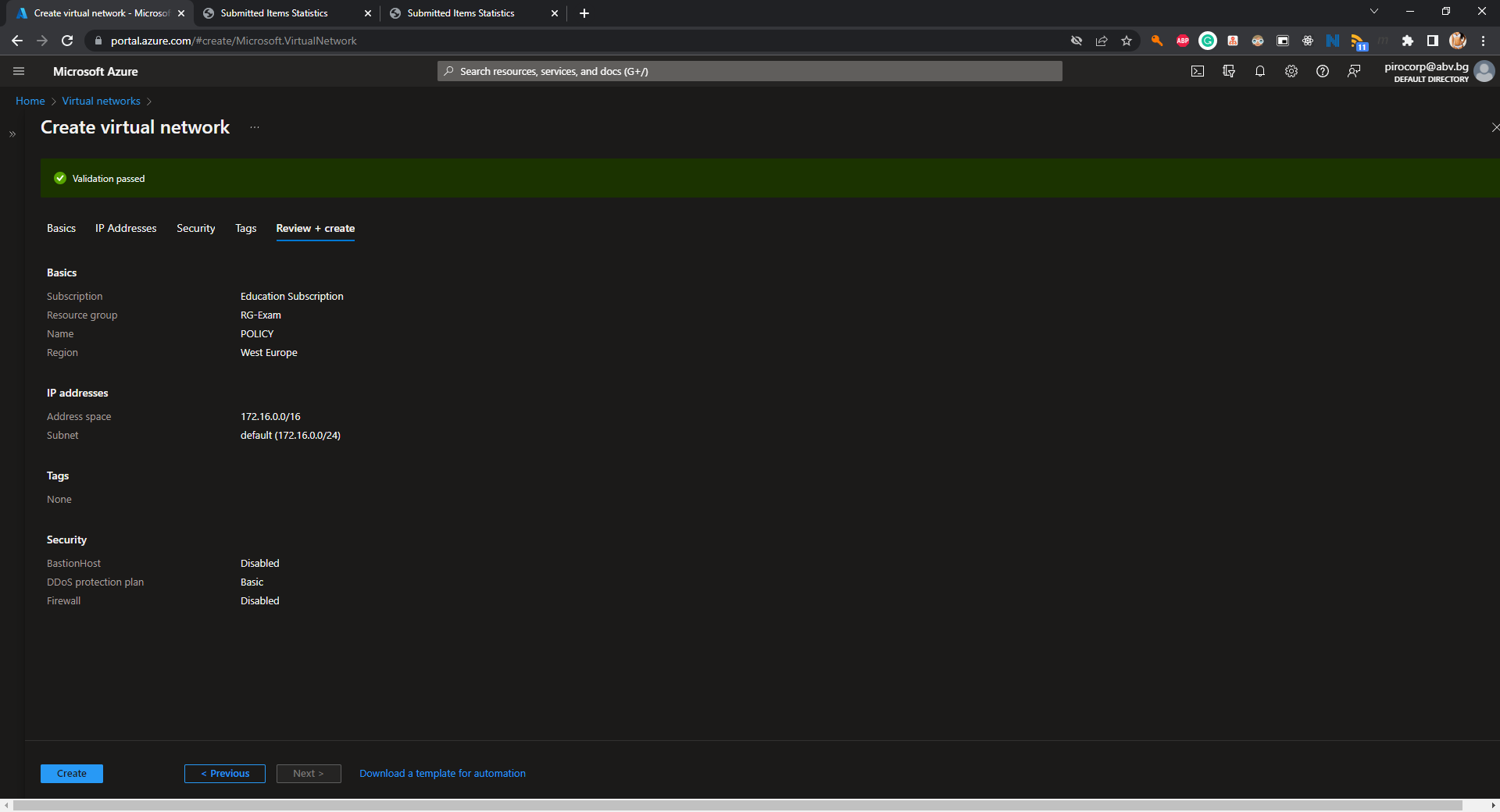
* (T402, 1 pts) Put a resource lock of type **Delete** on the database

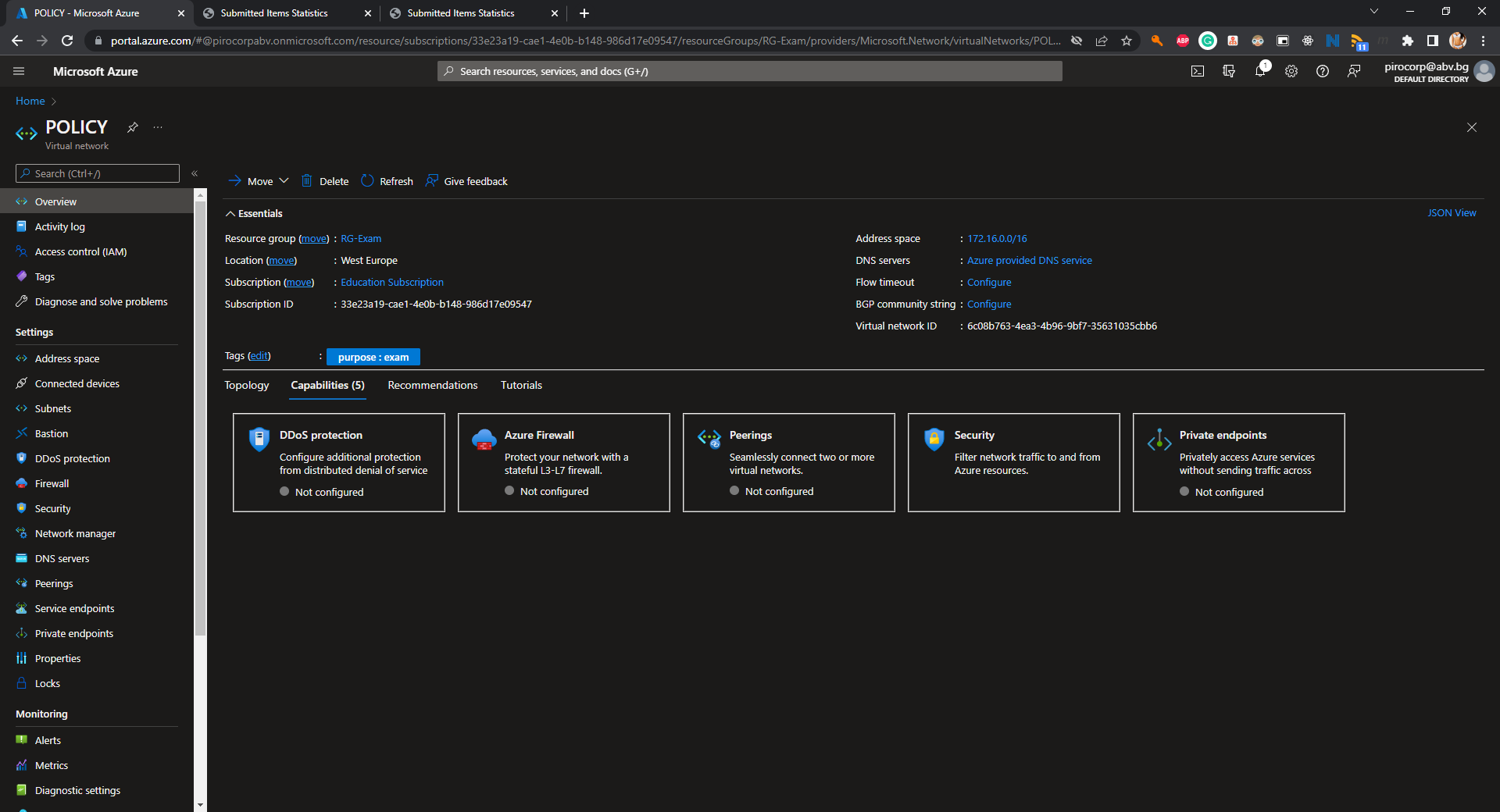


* (T403, 3 pts) Define (if needed) and assign a policy that enforces tag **purpose** with value **exam** on every new or updated resource in the resource group



* (T404, 1 pts) Prove that the policy is working by adding a resource of your choice with name **POLICY** (this can be anything, for example a virtual network, network security group, etc.)

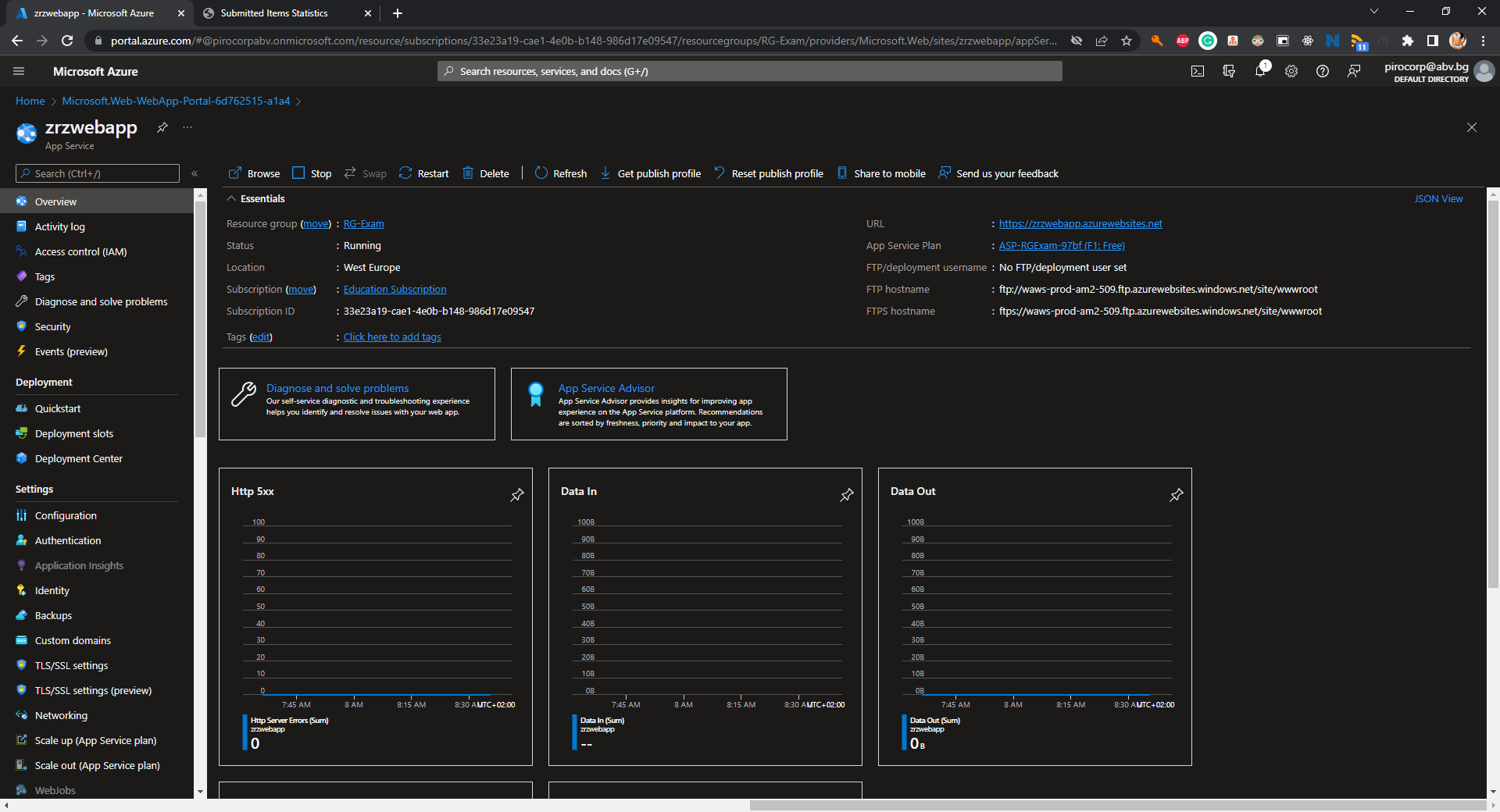




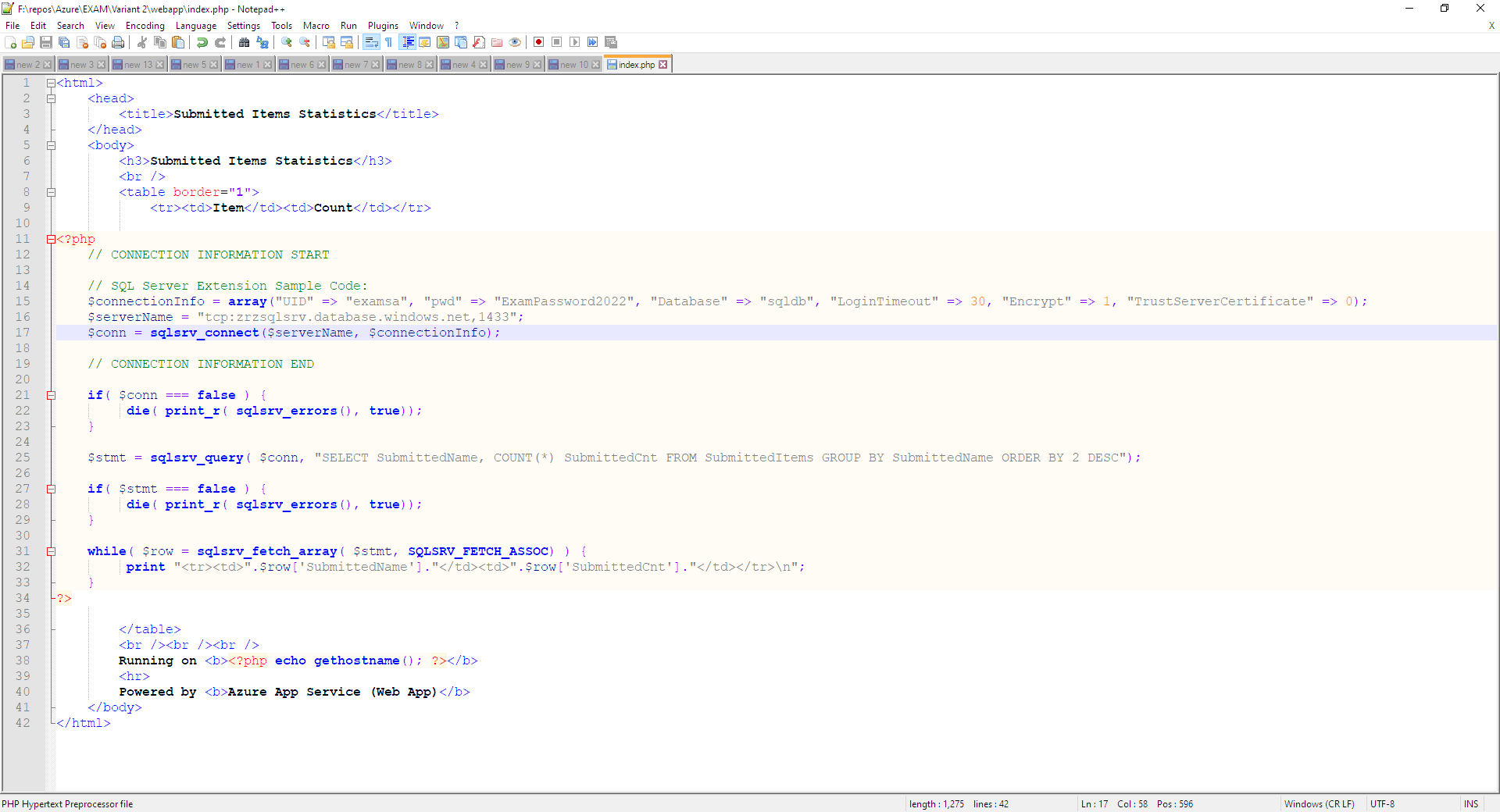
#### Web Apps and Functions - 8 tasks, 22 pts

##### Web Application

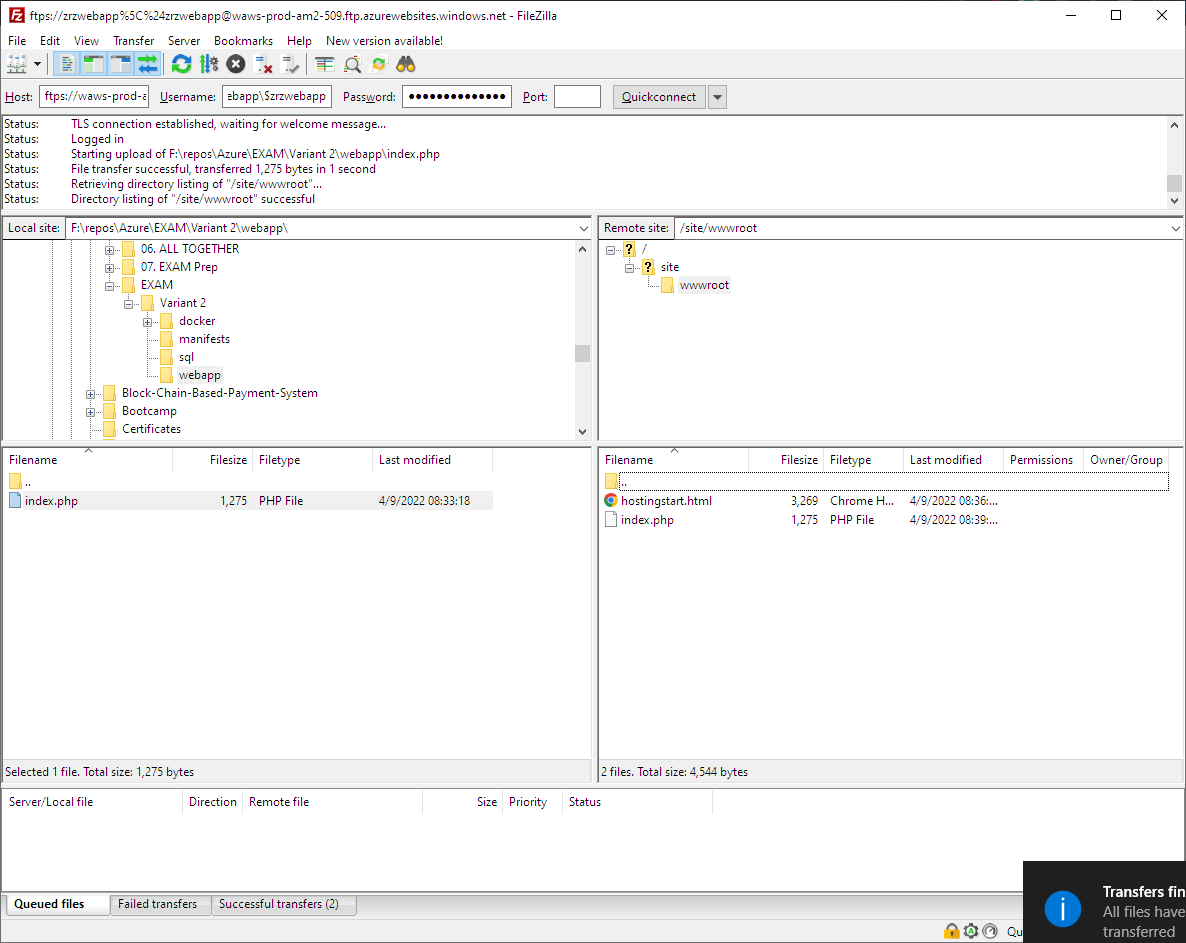
* (T501, 3 pts) Create a PHP code-based (not container-based) web application (App Service) \*



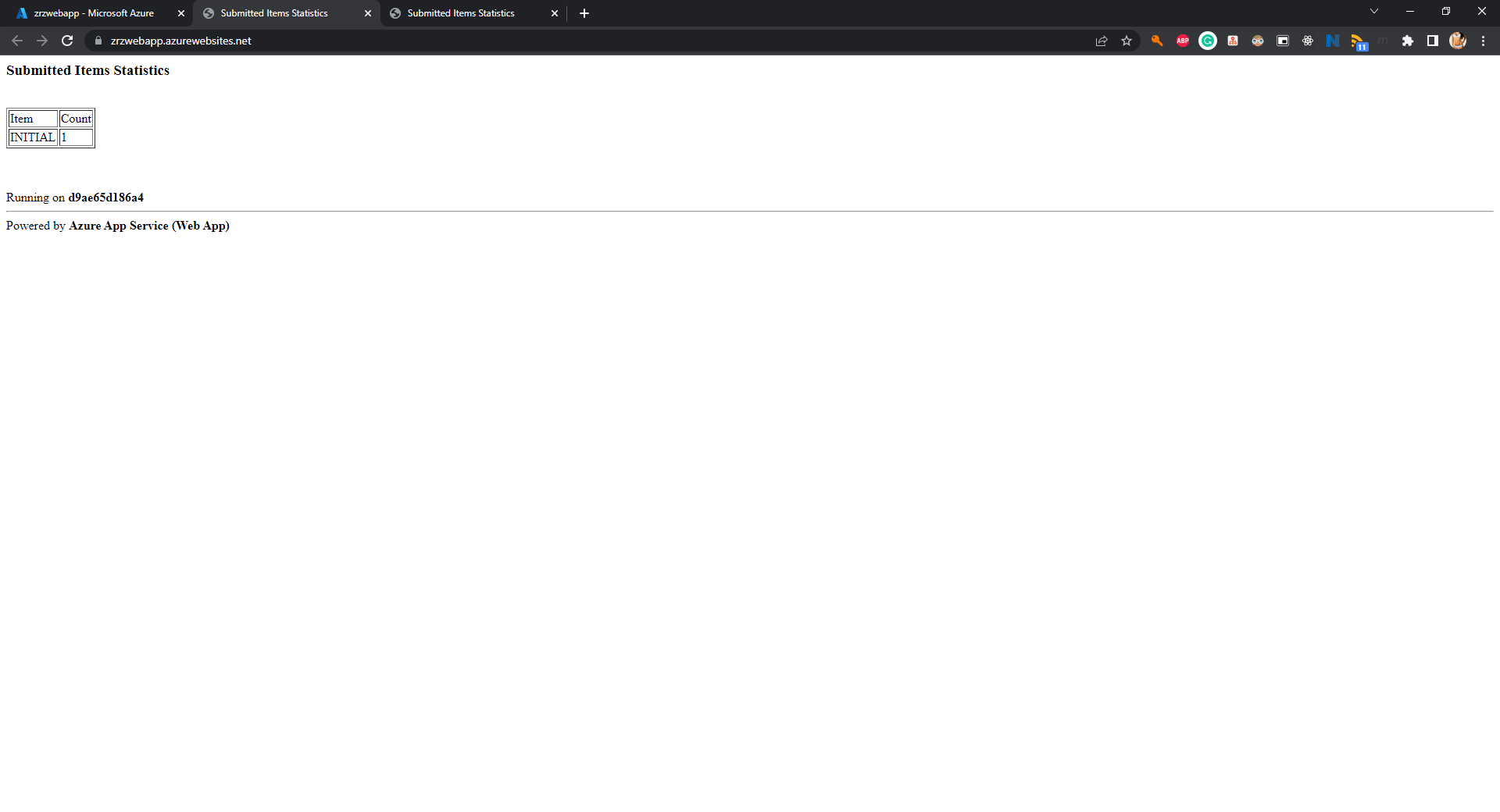
* (T502, 2 pts) Add the SQL connection string to the **index.php** file in the **webapp/** folder



* (T503, 2 pts) Deploy the web application code to Azure and make sure it is working

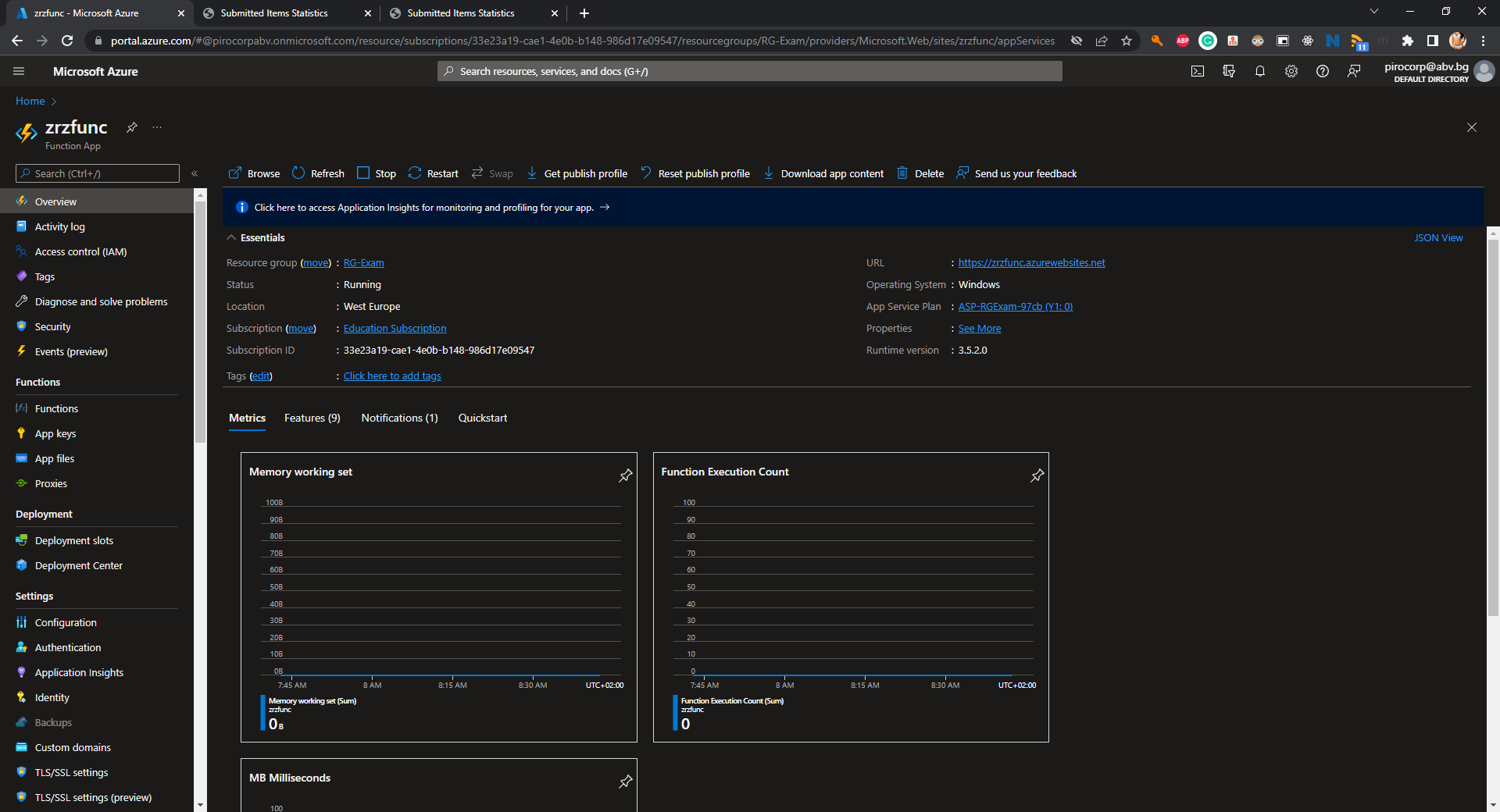


* (T504, 2 pts) Make sure that the web app is working and showing correct results

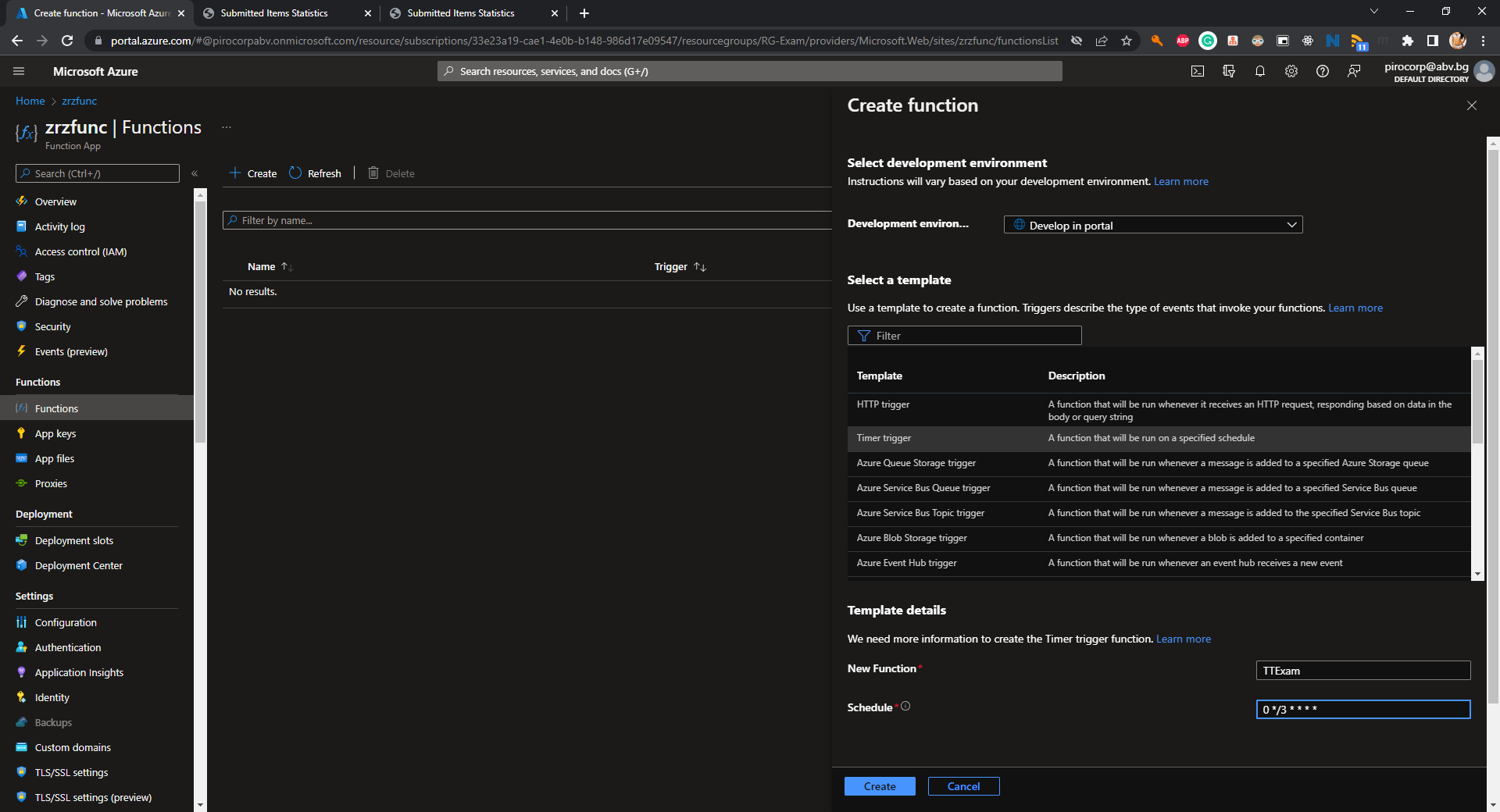


##### Function App

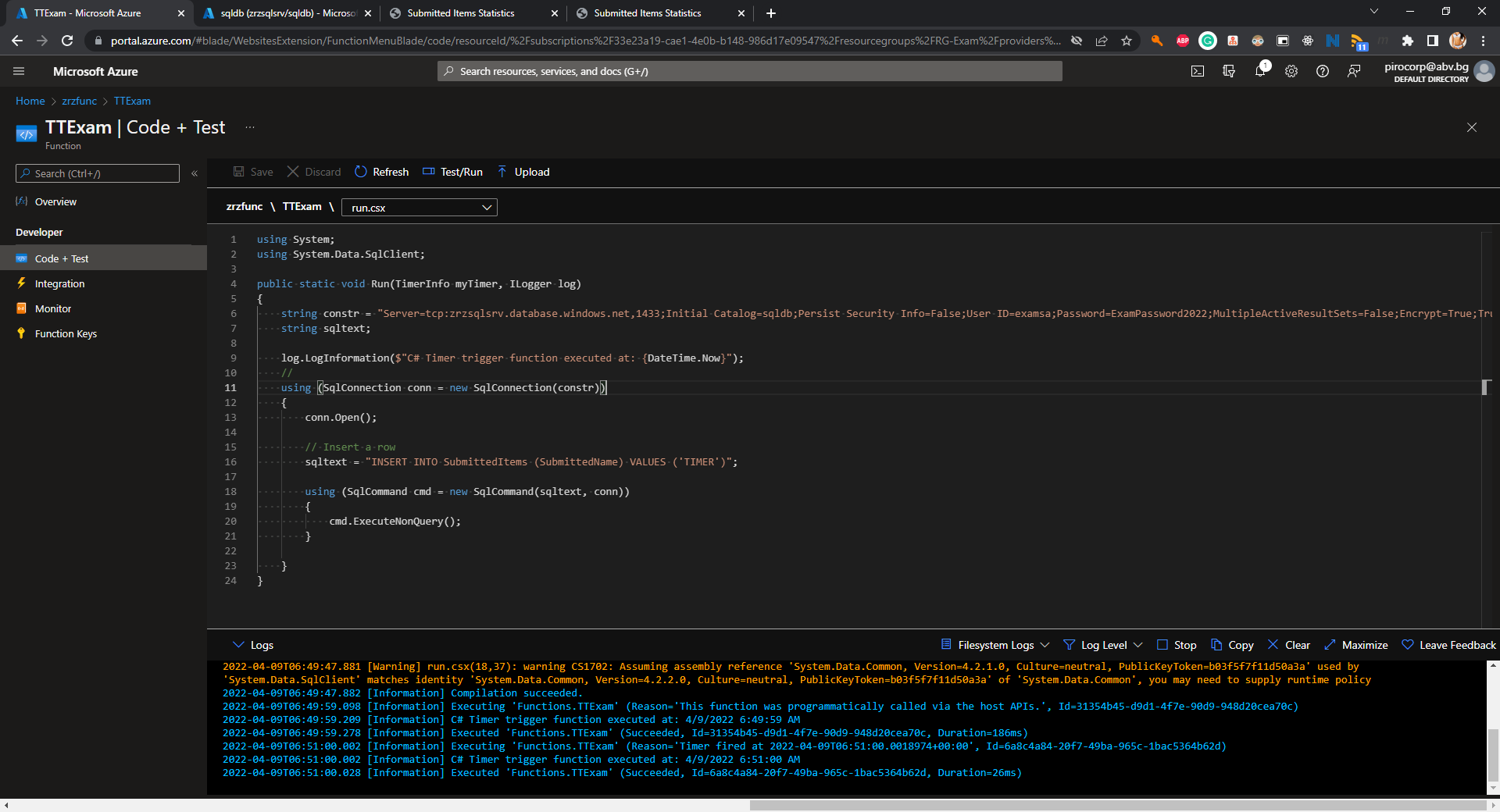
* (T505, 2 pts) Create a code-based **Function App** with **.NET Core** as runtime \*

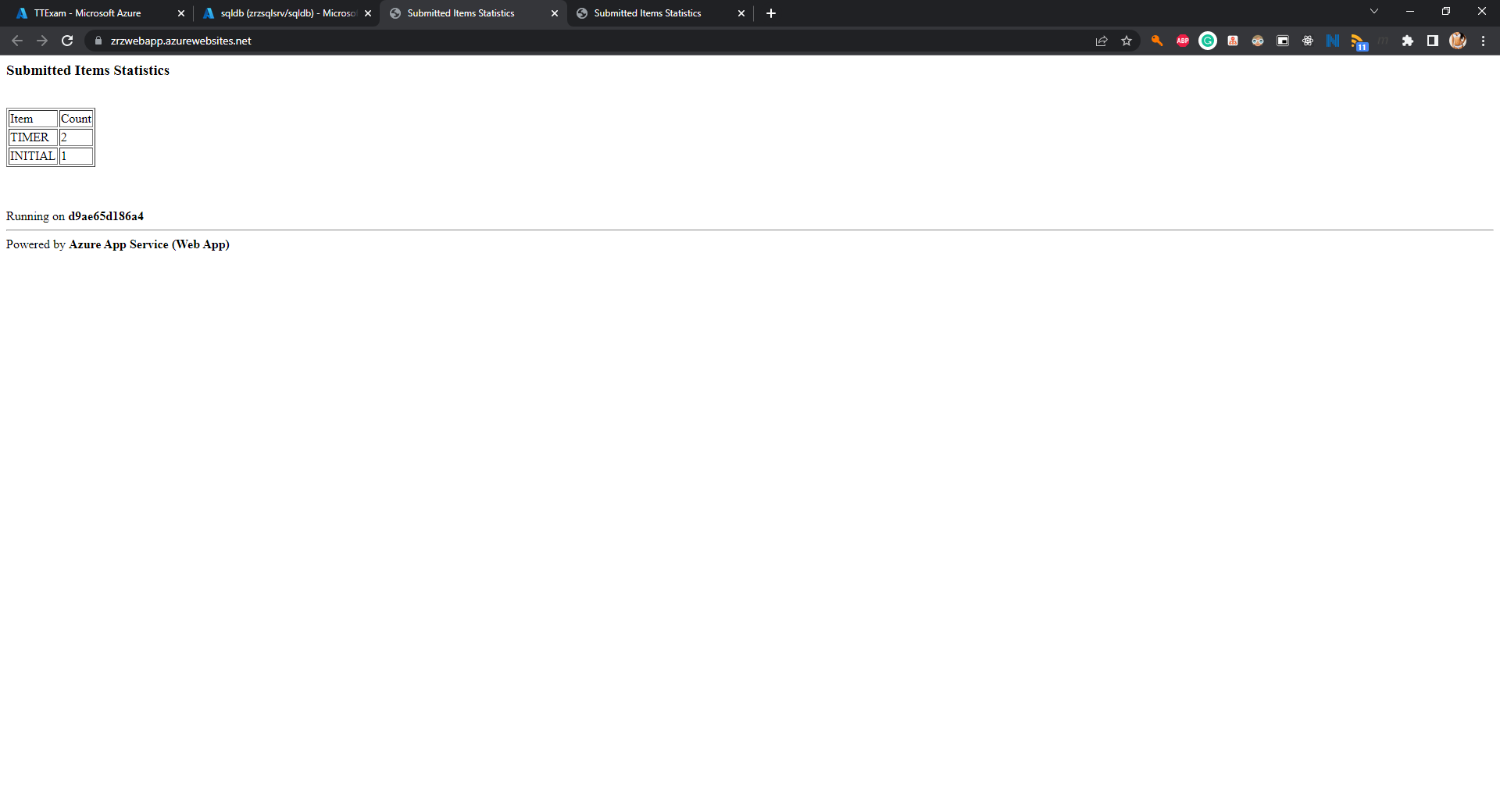


* (T506, 1 pts) Create a **Timer triggered** function named **TTExam** that
  + (T506.1, 1 pts) executes **every three minutes**

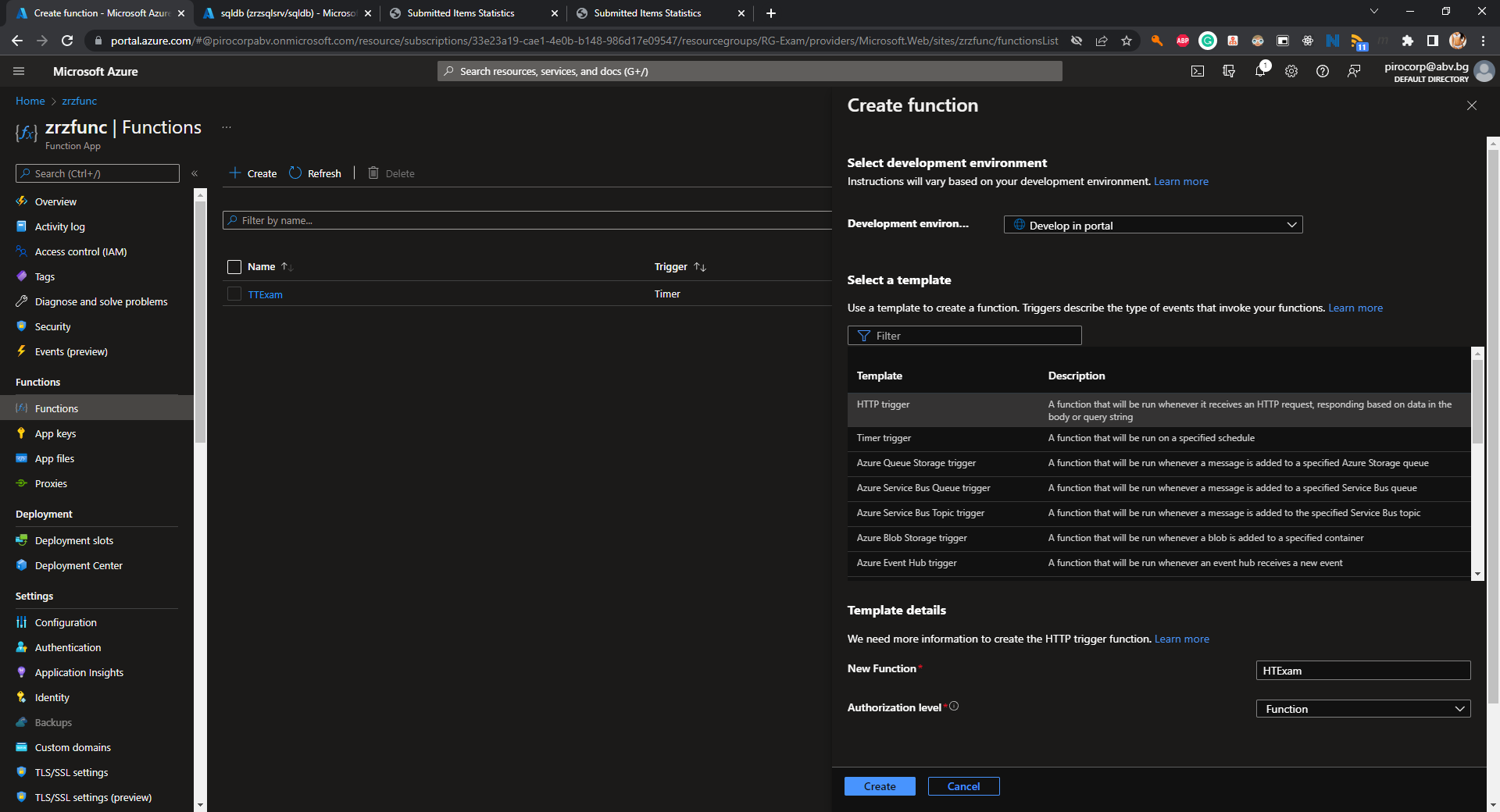


* + (T506.2, 2 pts) and inserts a row with **SubmittedName=TIMER** in the database (table **SubmittedItems**)

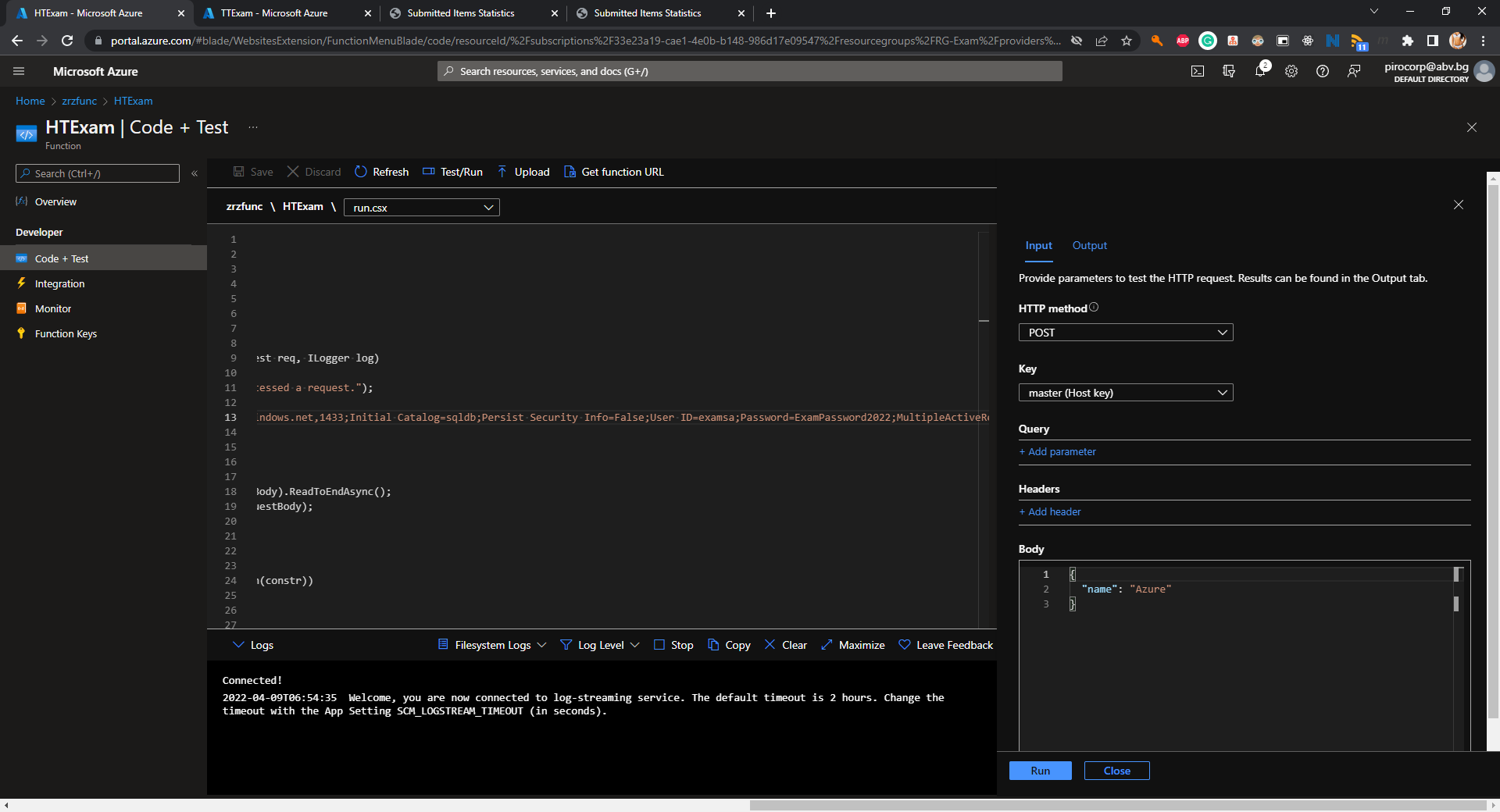




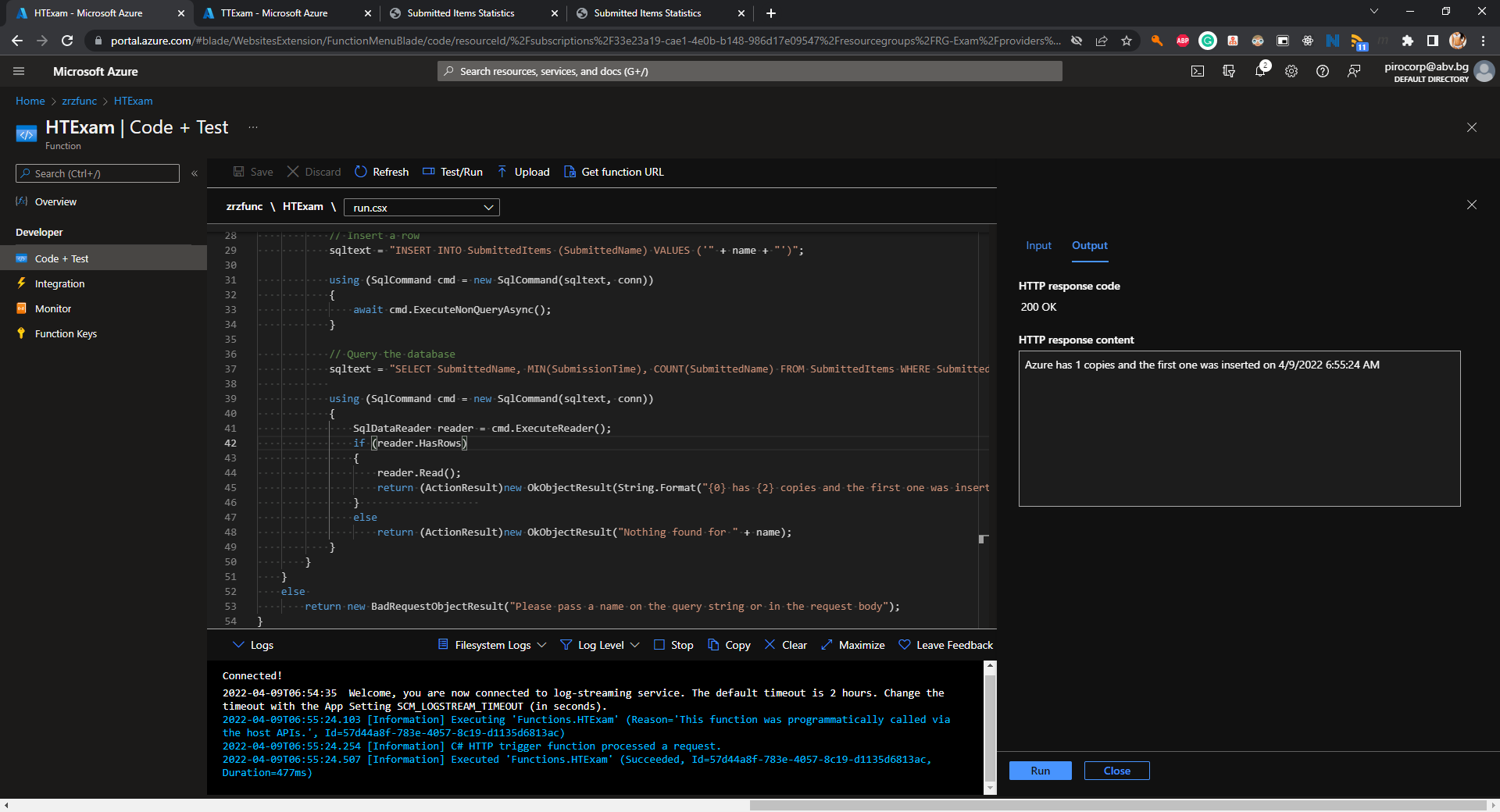
* (T507, 1 pts) Create a **HTTP triggered** function named **HTExam** that



* + (T507.1, 1 pts) when executed it must accept a single parameter (**name**)

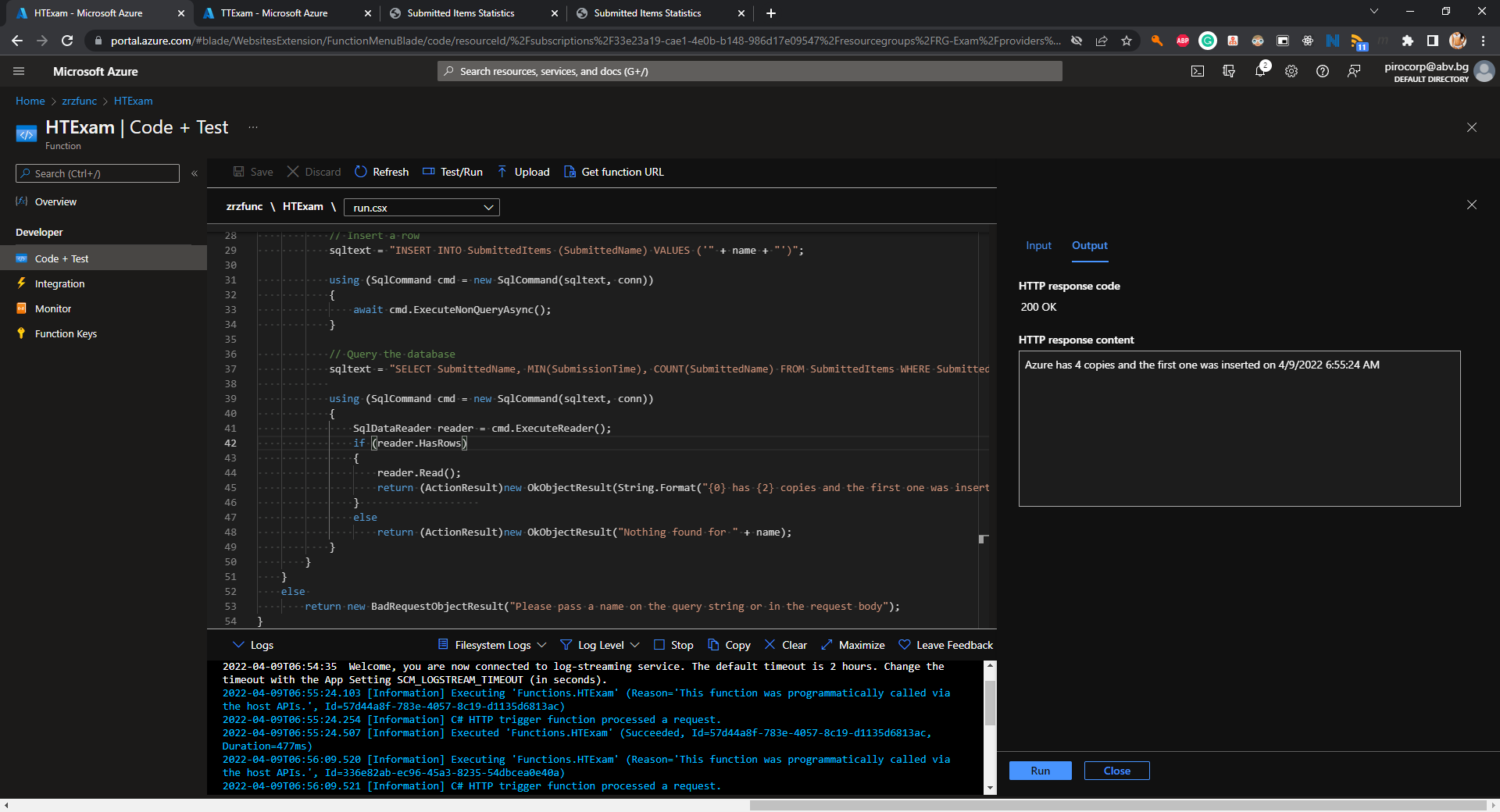


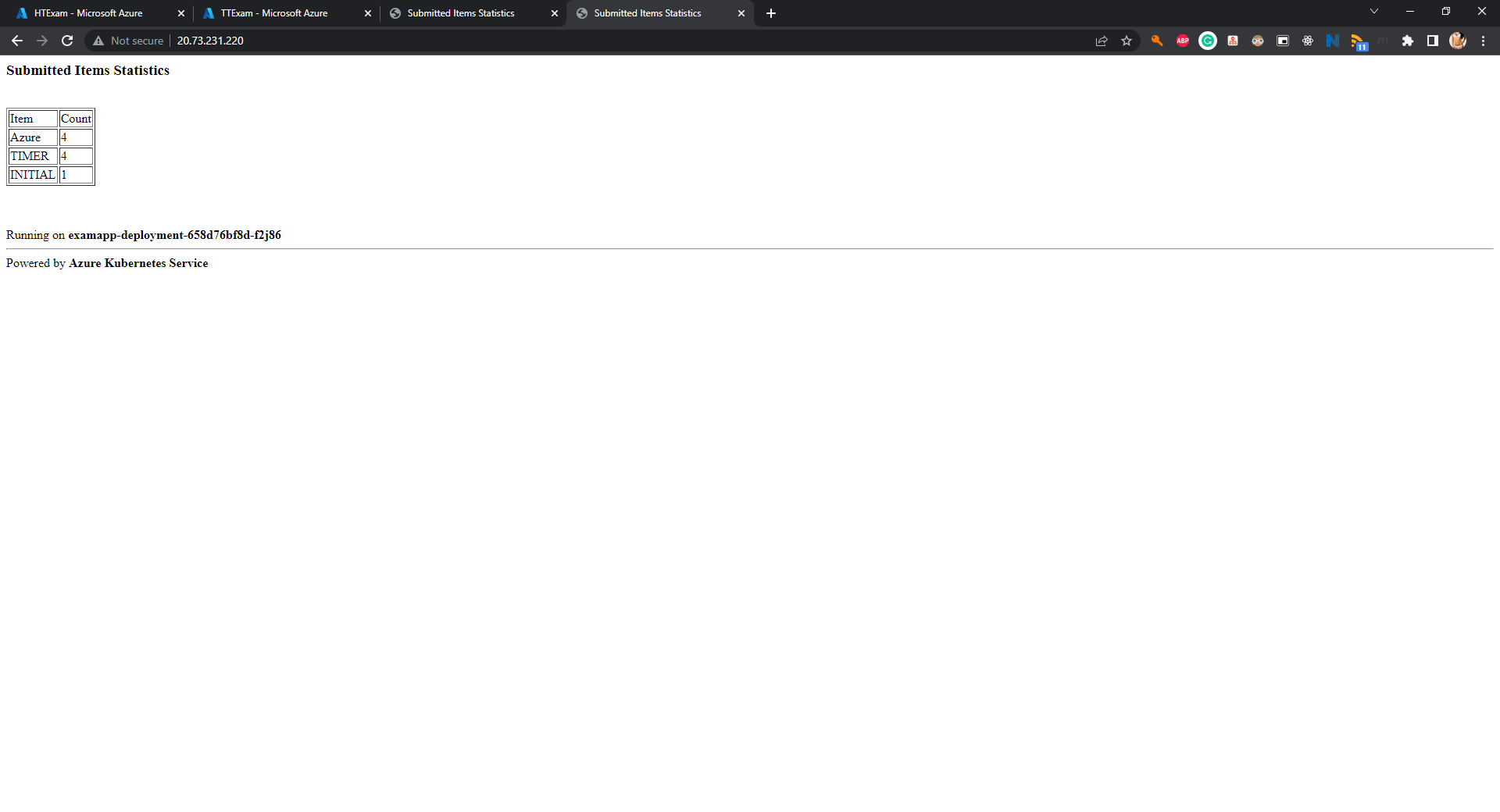
* + (T507.2, 2 pts) stores the value in the **SubmittedName** column of table **SubmittedItems** in the database
  + (T507.3, 1 pts) and returns how many times the value has been inserted and when was the first time. The format should be **VALUE has N copies and the first one was inserted on TIME**



*For example, if the function was called 5 times with the text* ***Exam****, and the first execution was on* ***09.04.2022 09:30****, then it should return* ***Exam has 5 copies and the first one was inserted on 09.04.2022 09:30****. Please note that the format of the time is not important and may not match the example*

* (T508, 2 pts) Make sure that you have executed the HTTP triggered function successfully several times





*\* Note that you may need to create additional resource group(s) or they may be created automatically*

