## Problem (Variant A)

Implement three-tier architecture to host a PHP web application with SQL server database. In general, it must look like:

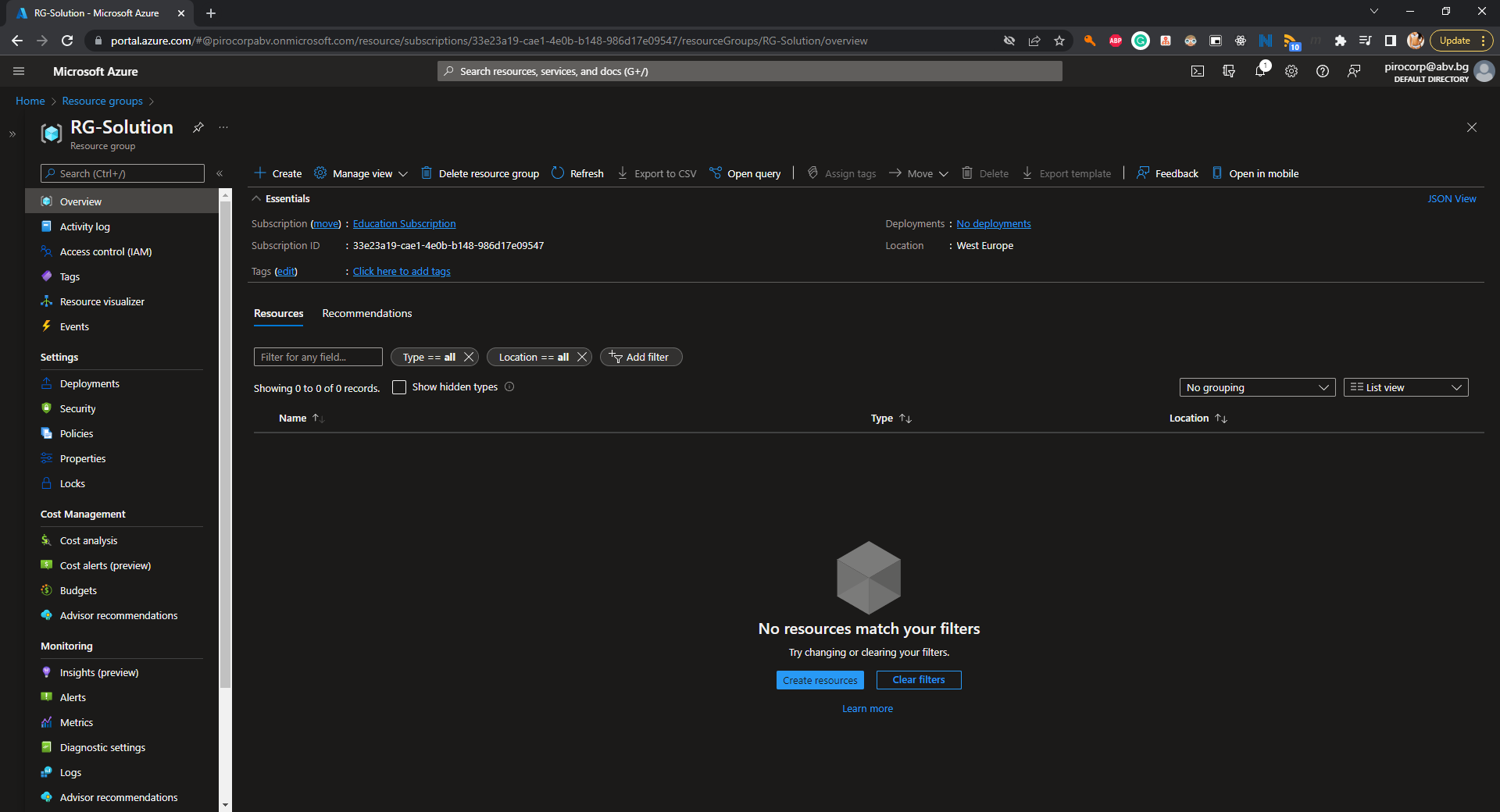
A screenshot of a cell phone

Description automatically generated

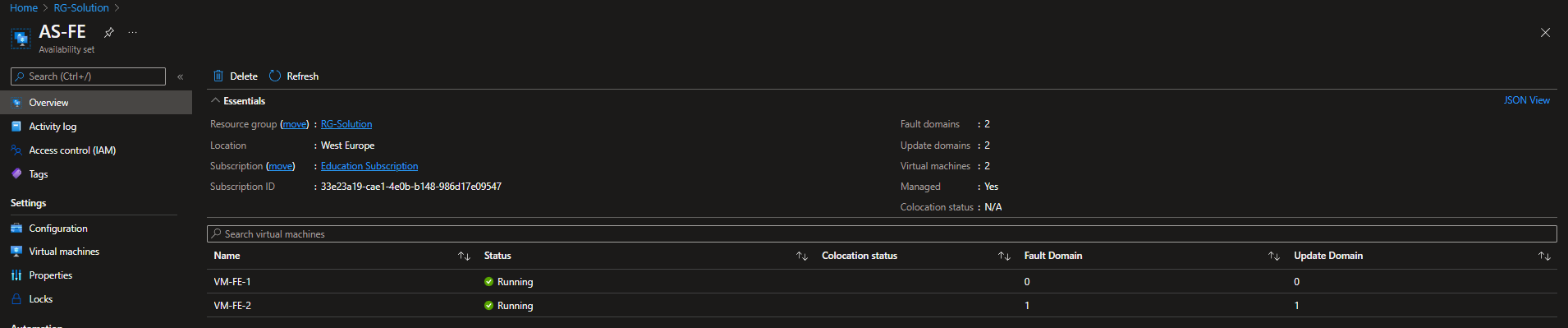
### Tasks

#### Infrastructure - 5 tasks, 15 pts

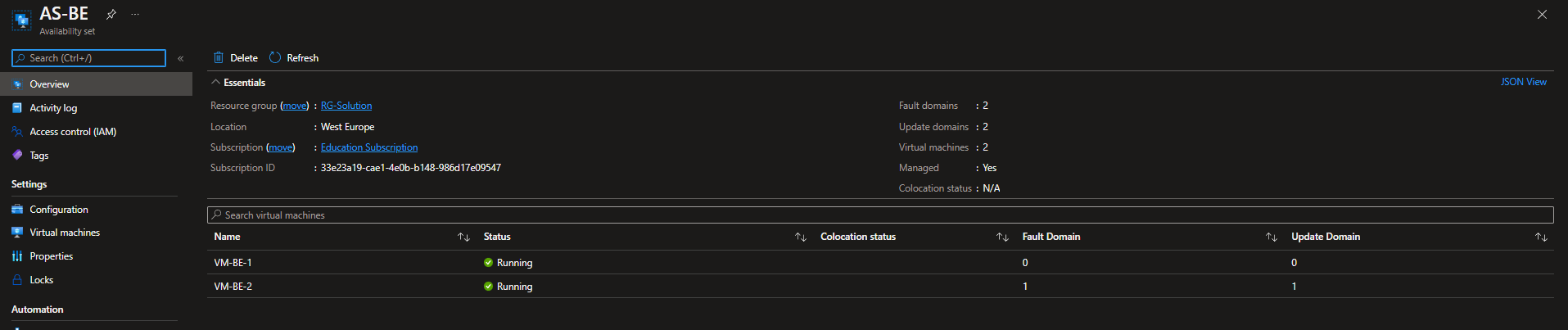
* (T101, 1 pts) Create a resource group named **RG-Solution**



* (T102, 2 pts) Create an artifact (availability set or virtual machine scale set) that provides high availability for virtual machines in the front-end group and name it **AS-FE**
* (T103, 2 pts) Create an artifact (availability set or virtual machine scale set) that provides high availability for virtual machines in the back-end group and name it **AS-BE**
* (T104, 5 pts) Create a set of **two** **Ubuntu 18.04** virtual machines in the **front-end group** each with a password set as an authentication method. If created in an availability set, name them **VM-FE-x**, where **x** is a sequence number

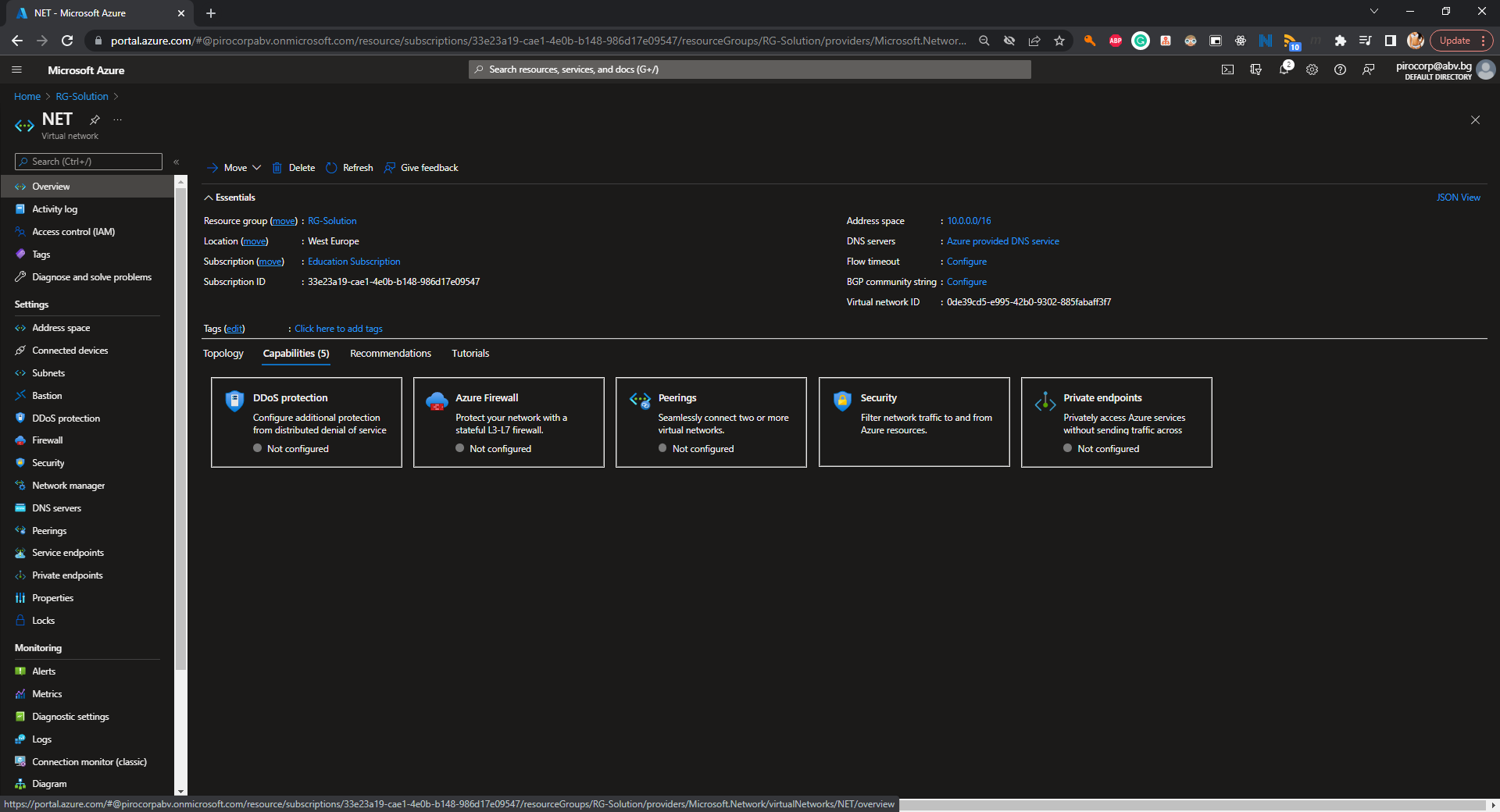


* (T105, 5 pts) Create a set of **two Ubuntu 18.04** virtual machines in the **back-end group** each with a password set as an authentication method. If created in an availability set, name them **VM-BE-x**, where **x** is a sequence number

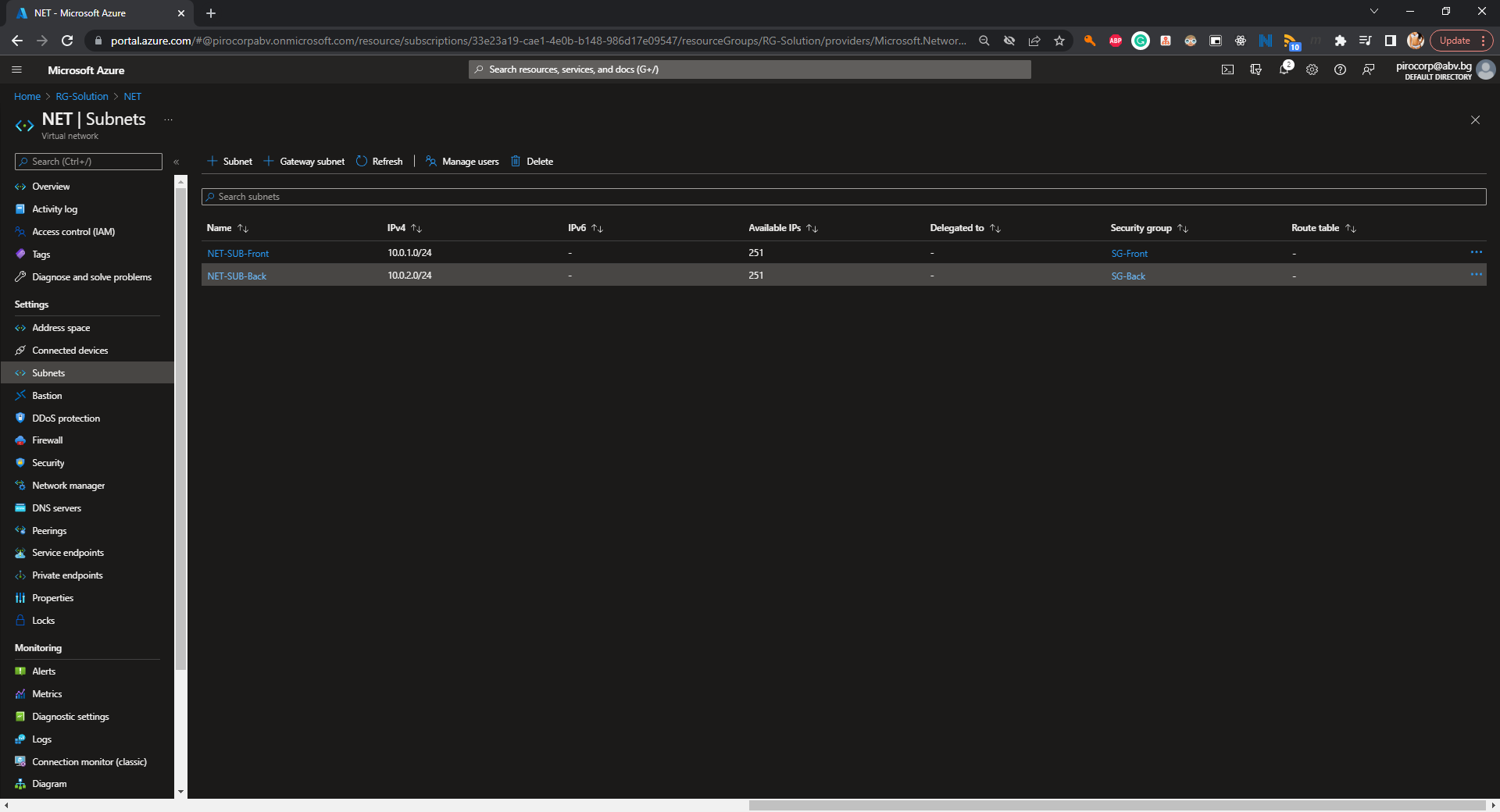


#### Networking - 7 tasks, 19 pts

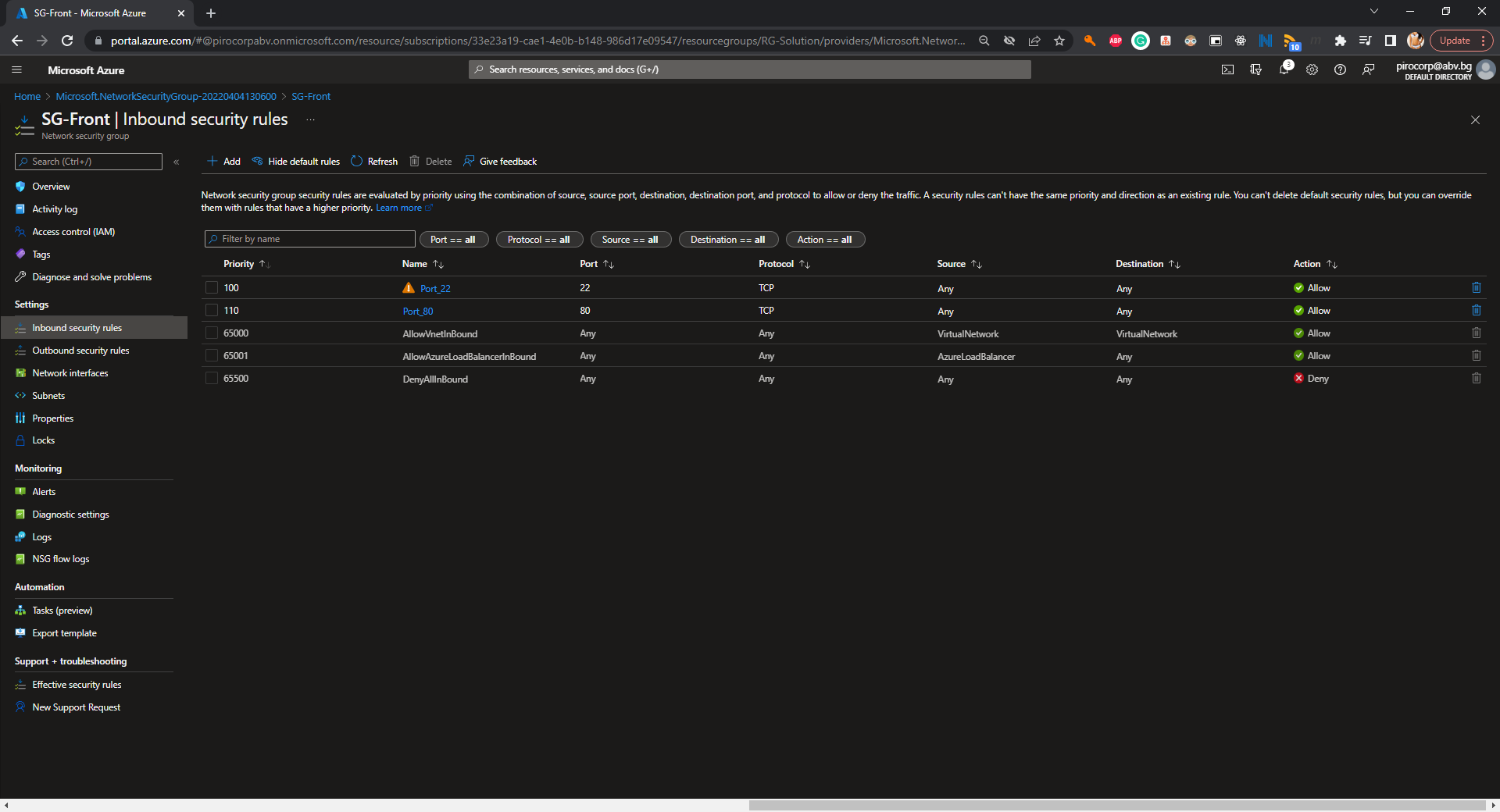
* (T201, 1 pts) Create a virtual network named **NET** with address space **10.0.0.0/16**



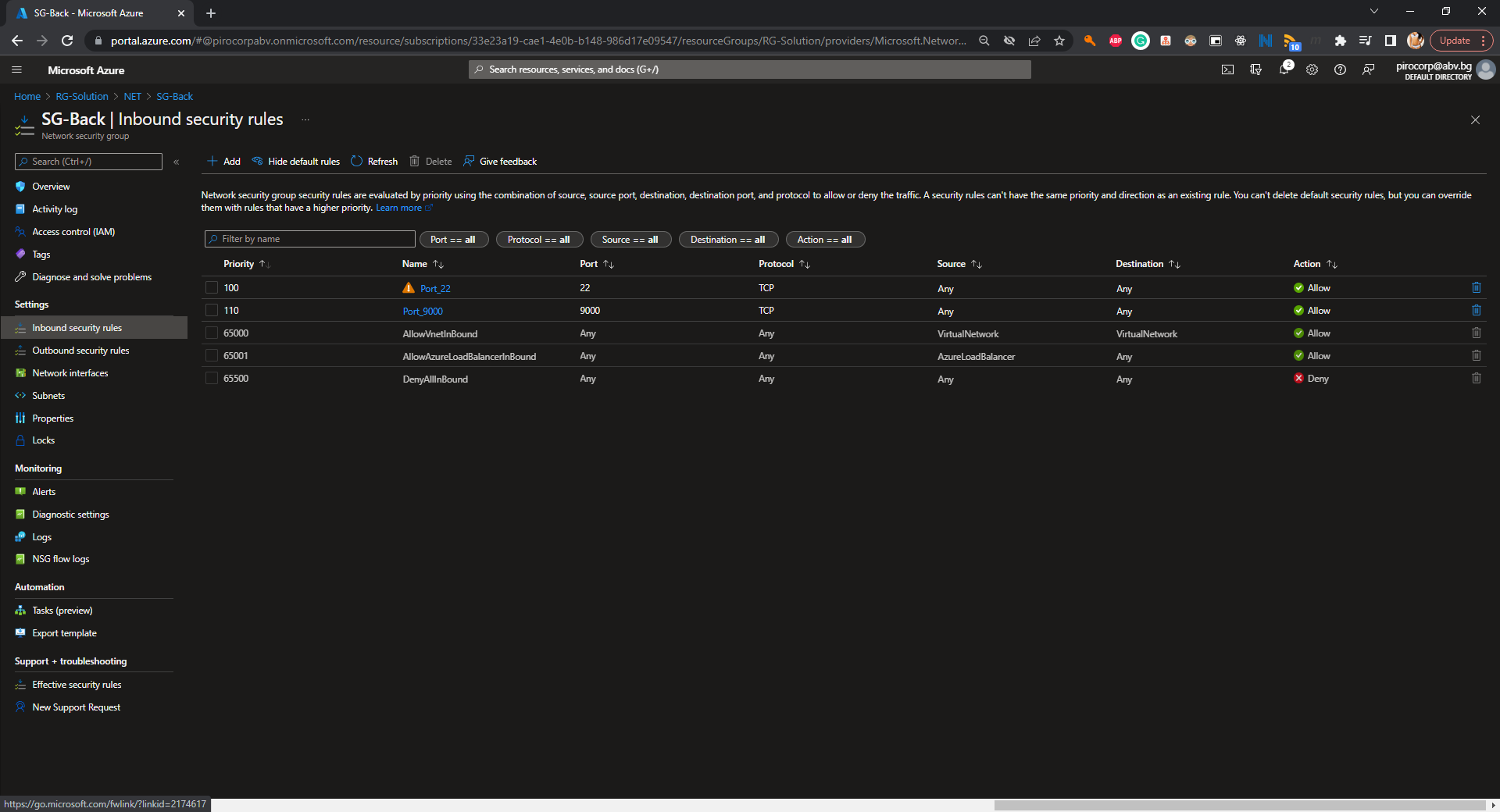
* (T202, 2 pts) Create a subnet named **NET-Sub-Front** with address space **10.0.1.0/24**
* (T203, 2 pts) Create a subnet named **NET-Sub-Back** with address space **10.0.2.0/24**



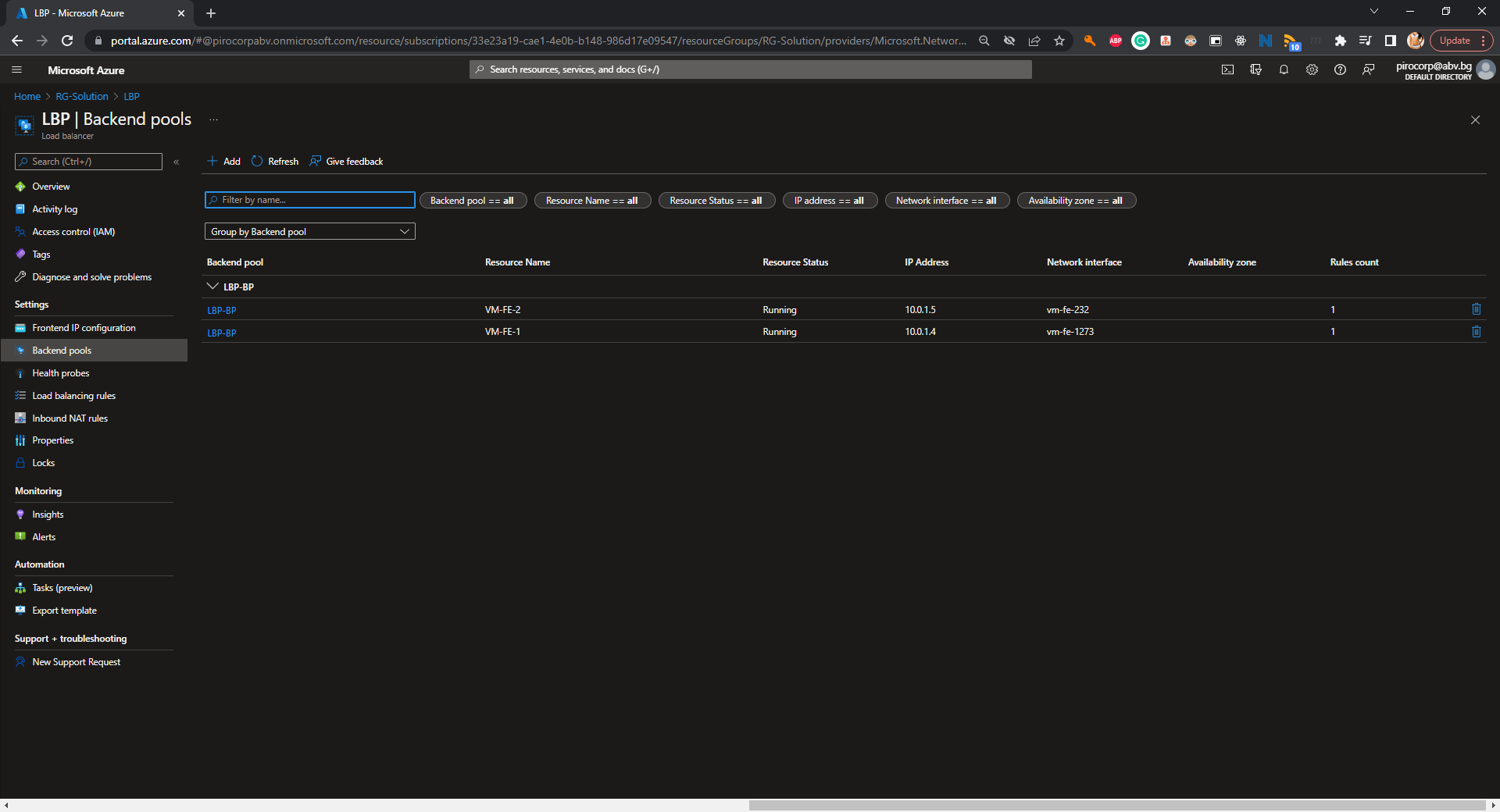
* (T204, 3 pts) Create a network security group **SG-Front**, attach it to the **NET-Sub-Front** subnet, and create two **inbound** rules – one to allow communication on port **22/tcp** and a second one to allow communication on port **80/tcp**

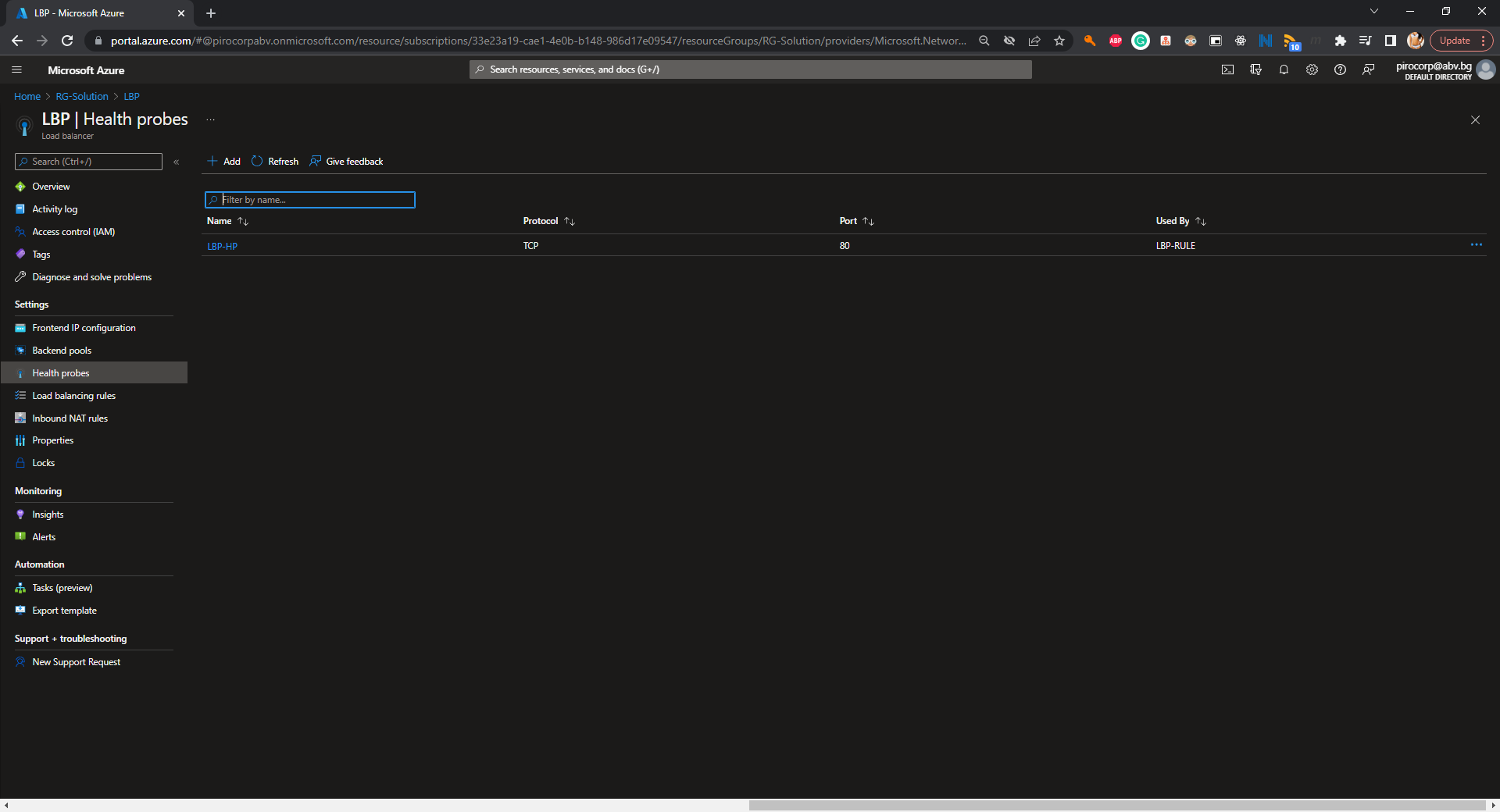


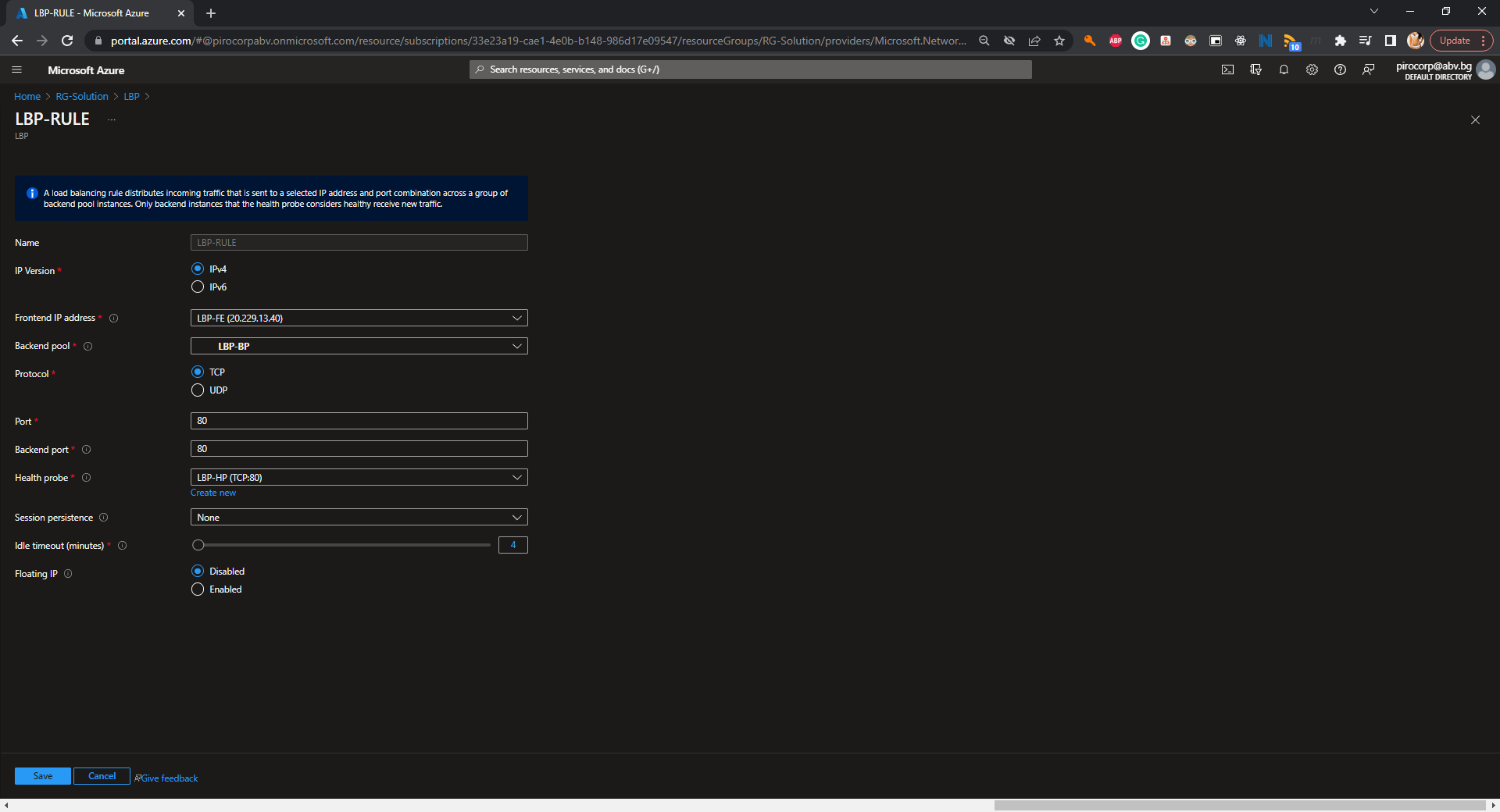
* (T205, 3 pts) Create a network security group **SG-Back**, attach it to the **NET-Sub-Back** subnet, and create two **inbound** rules – one to allow communication on port **22/tcp** and a second one to allow communication on port **9000/tcp**



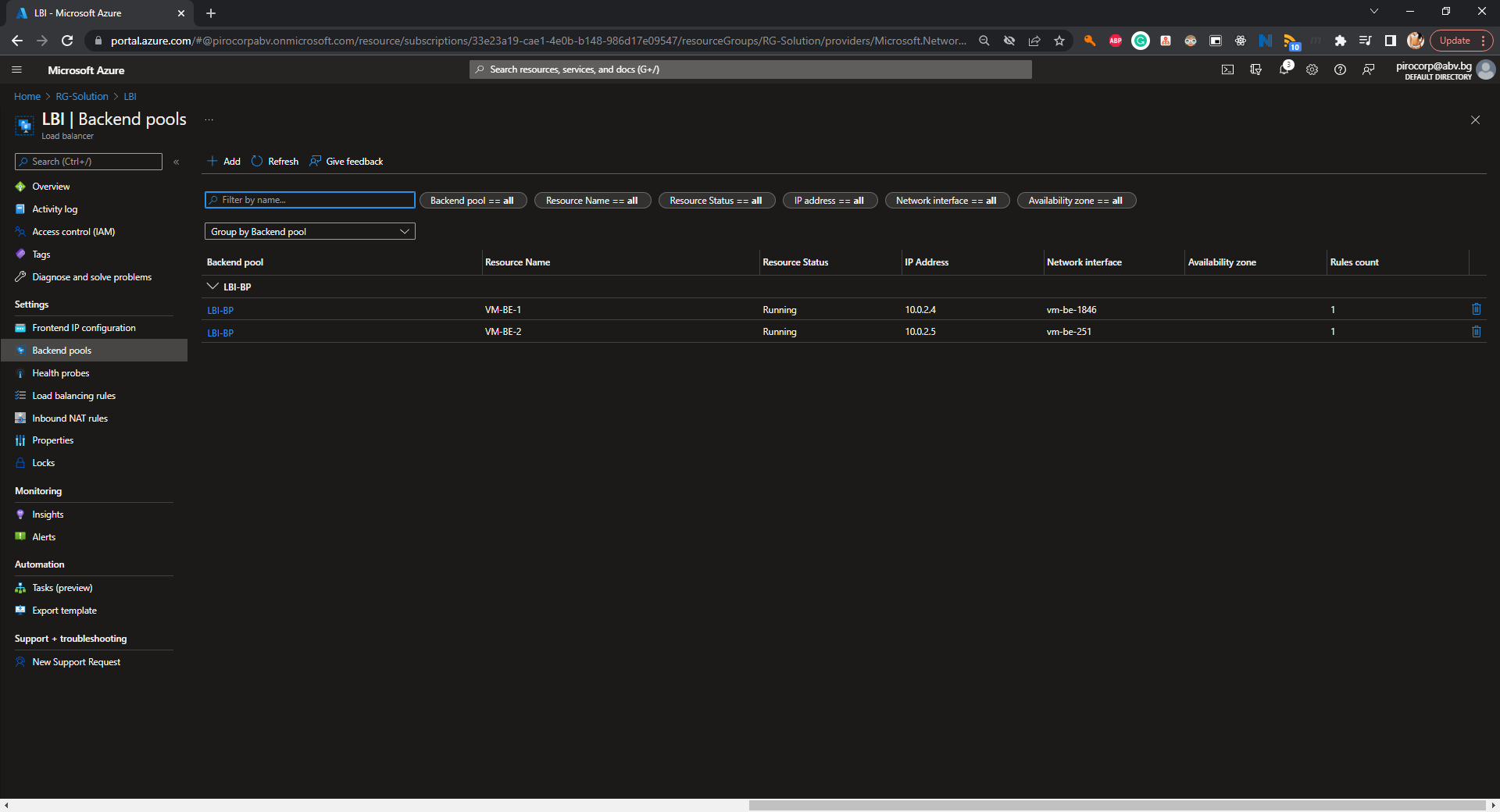
* (T206, 4 pts) Create an external load balancer named **LBP** with the corresponding set of backend pool, health probe, and load balancing rule that maps external port **80/tcp** to internal port **80/tcp**

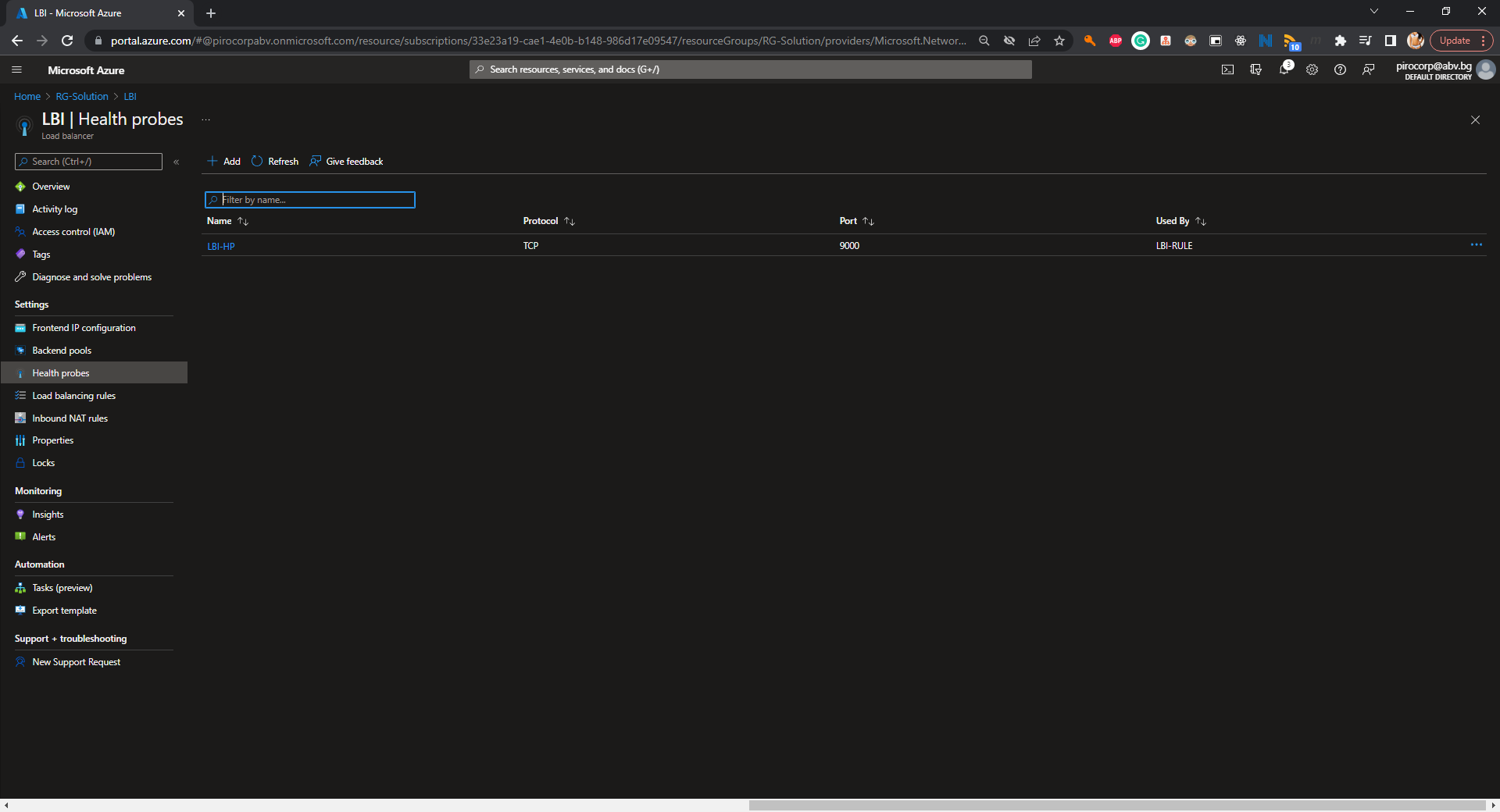


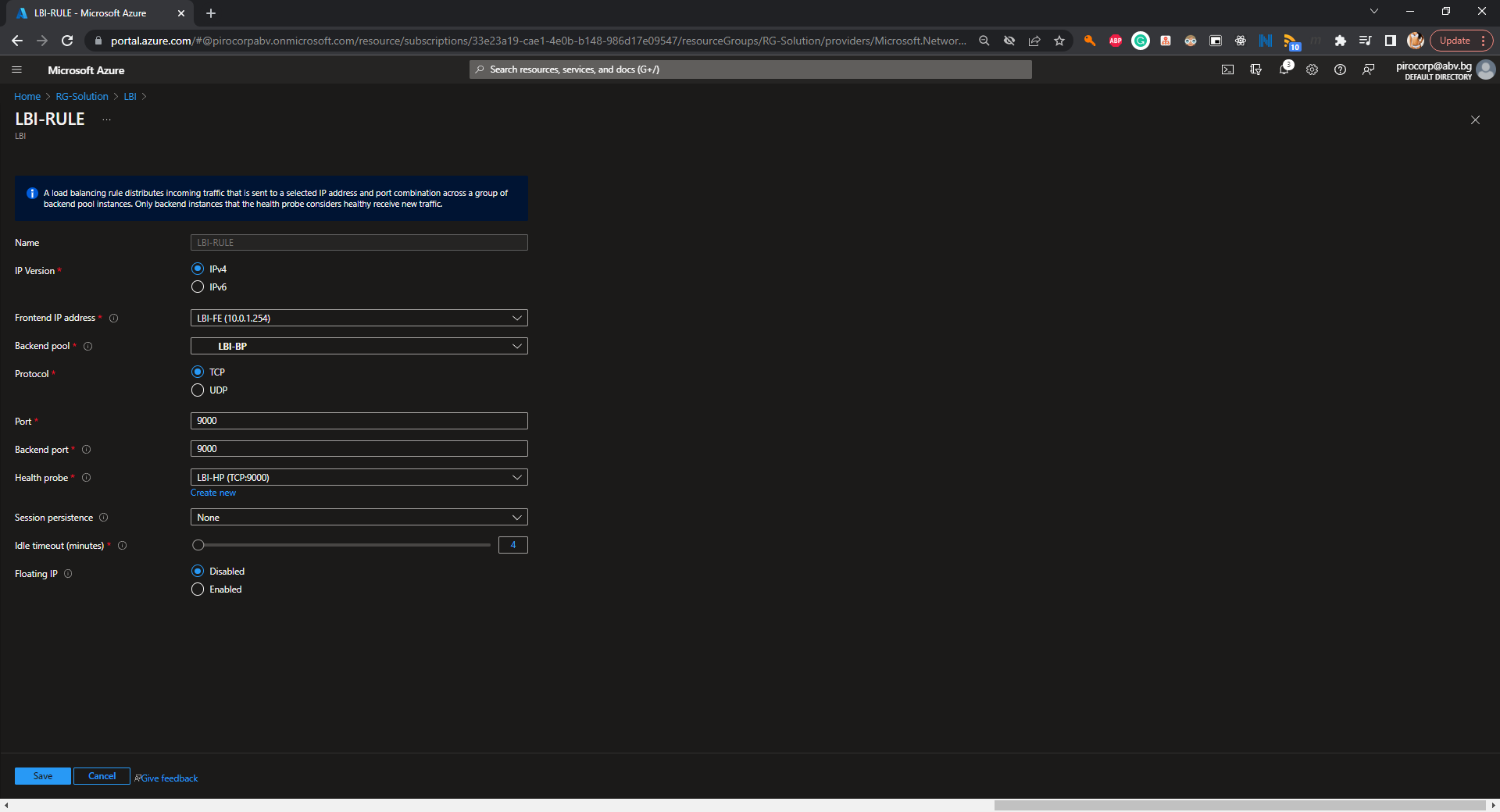




* (T207, 4 pts) Create an internal load balancer named **LBI** with the corresponding set of backend pool, health probe, and load balancing rule that maps external port **9000/tcp** to internal port **9000/tcp**

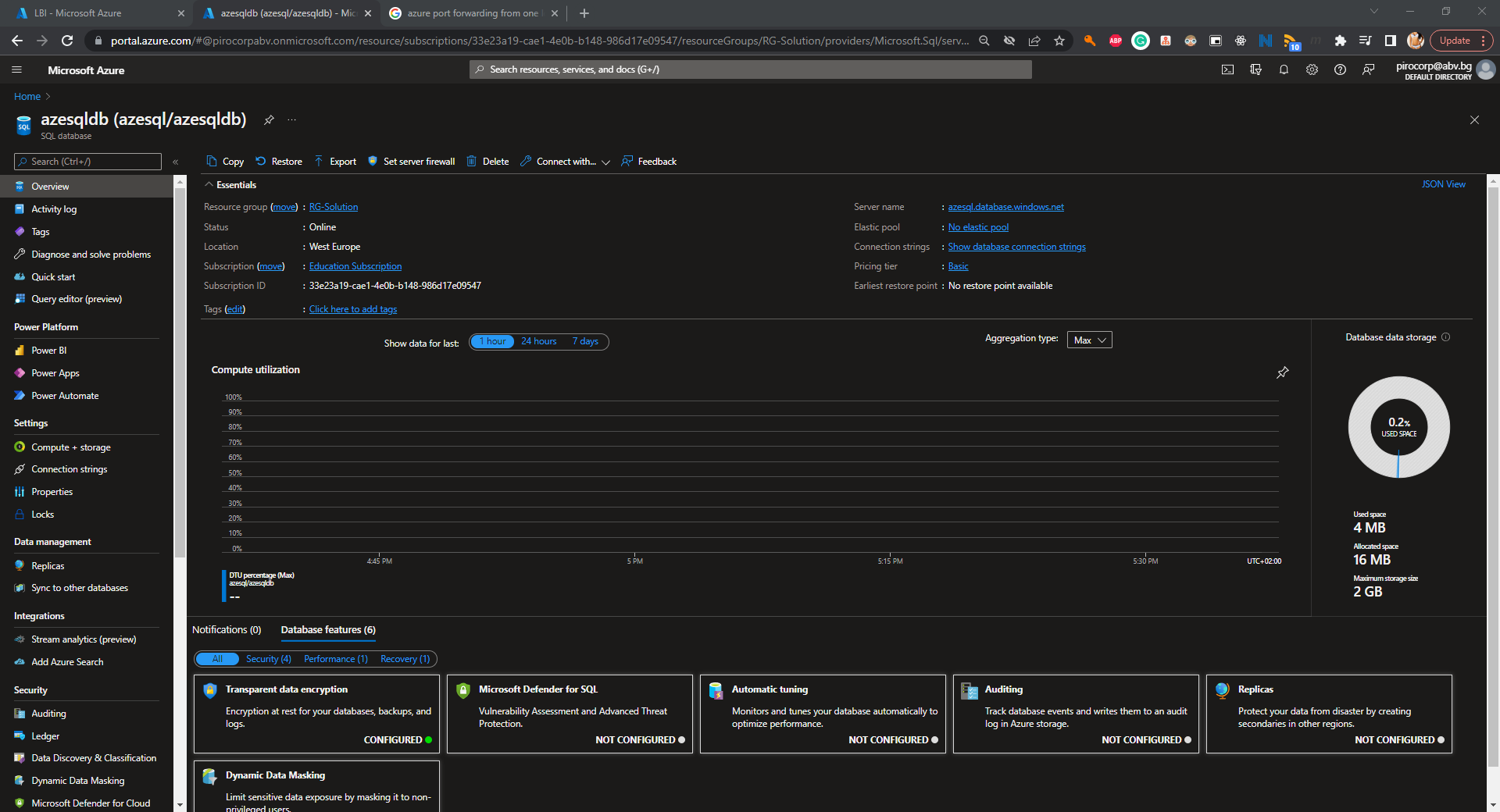




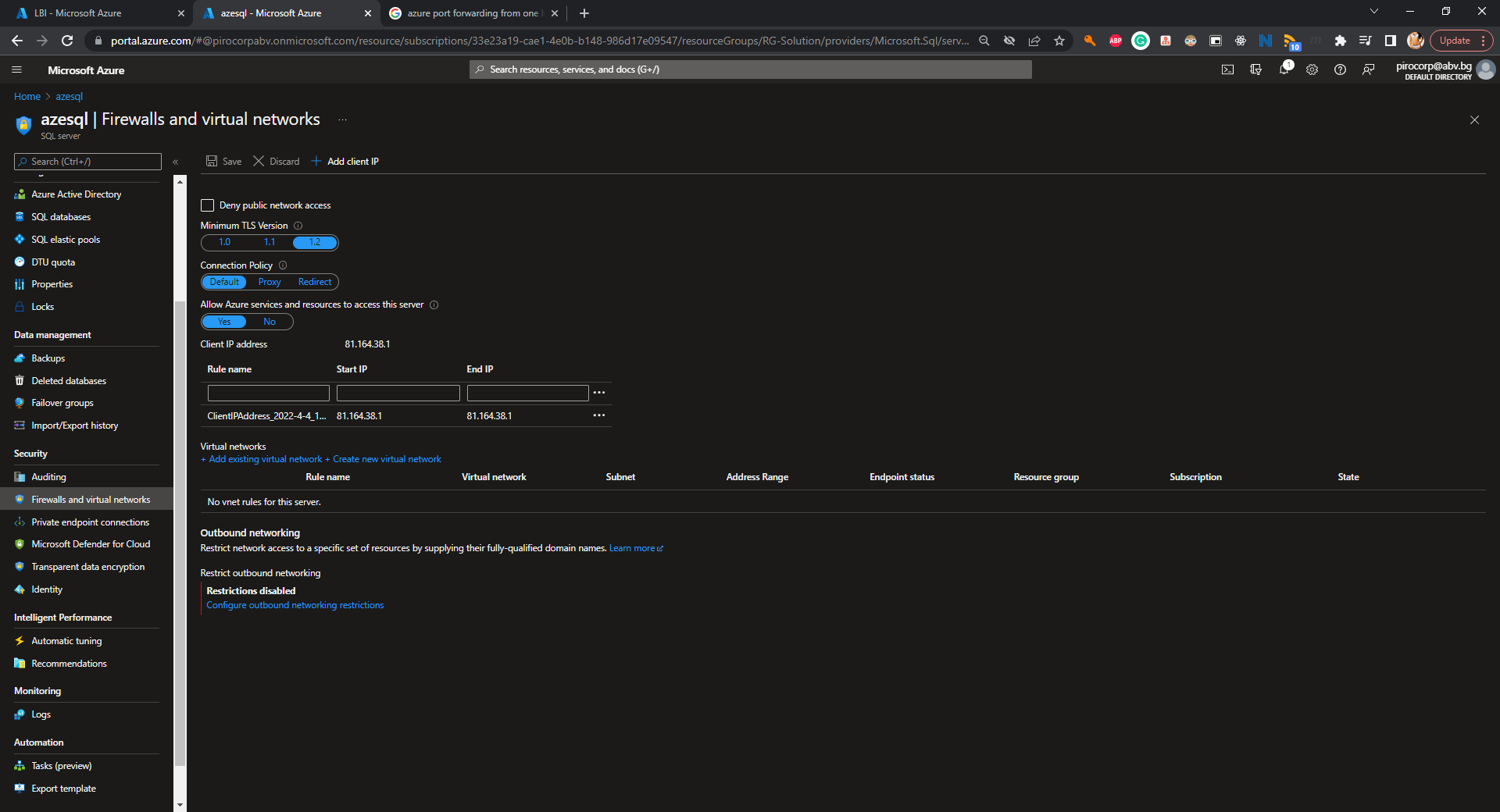


#### Databases - 3 tasks, 9 pts

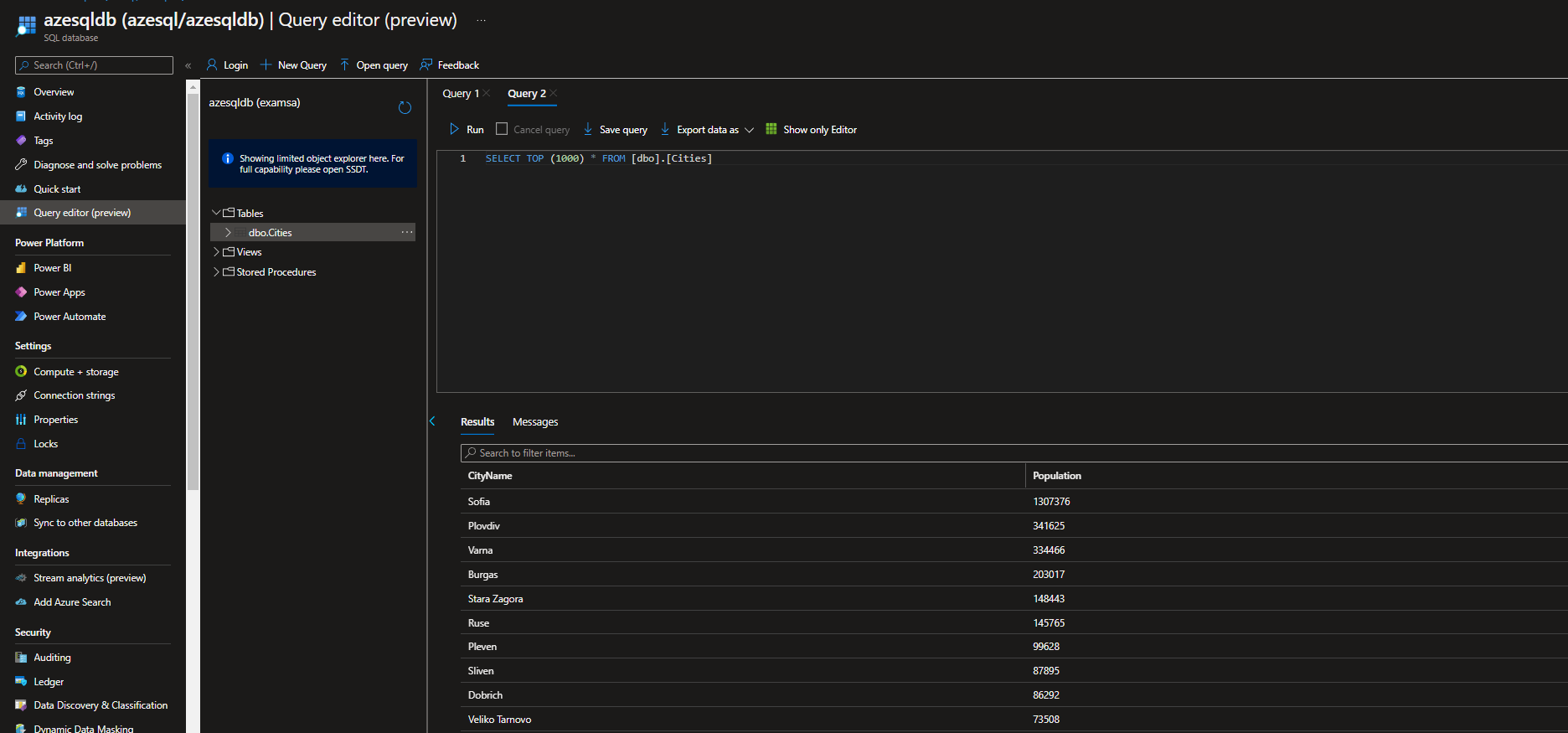
* (T301, 3 pts) Create SQL Server and a database



* (T302, 3 pts) Configure connectivity to the server



* (T303, 3 pts) Initialize the database with the help of the **load-data.sql** file part of the supporting files set

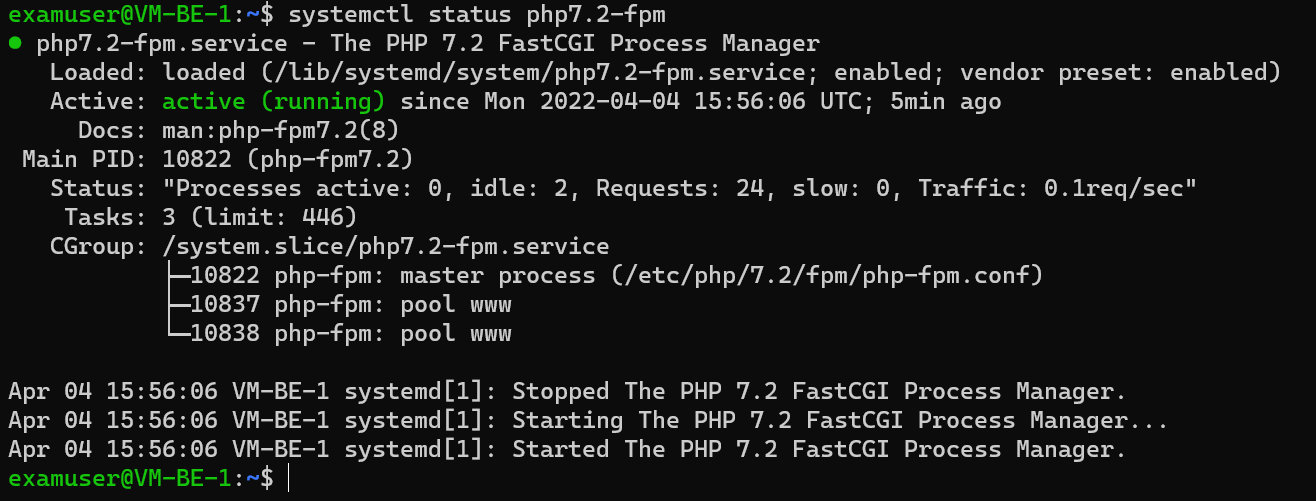


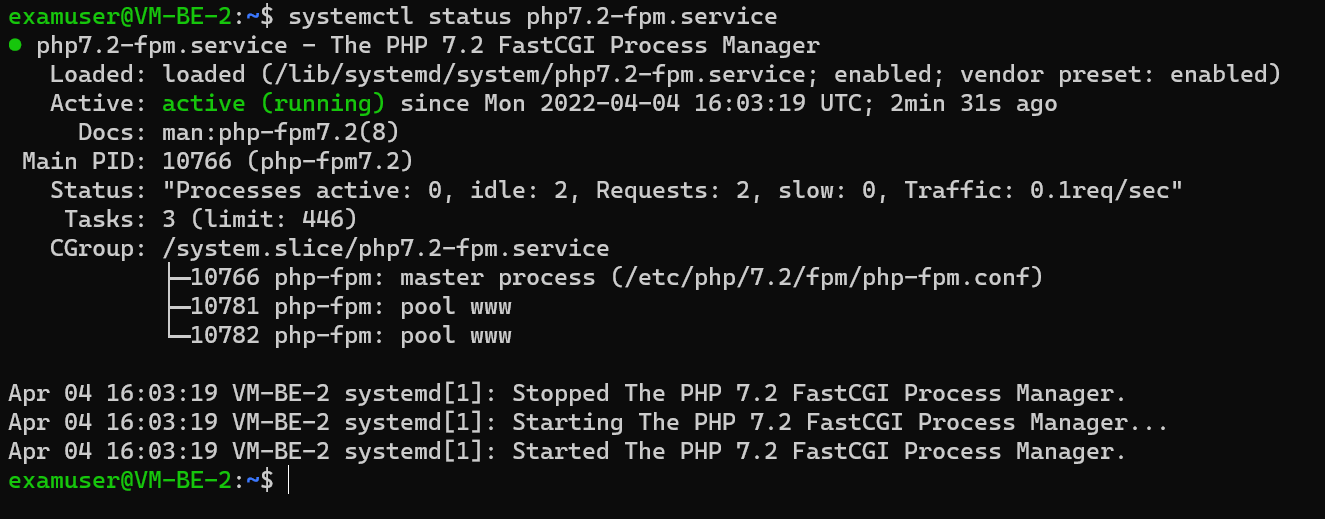
#### Software Deployment - 5 tasks, 17 pts

* (T501, 2 pts) Install **NGINX** on all **front-end** servers. For the configuration use (you are free to modify it or use your own) the **nginx-sample.conf** file part of the supporting files set



* (T502, 3 pts) Install **PHP-FPM** on all **back-end** servers. Configure it to listen on port **9000**





* (T503, 4 pts) Install **all supplementary software** on all **back-end** servers to allow them to communicate with the SQL Server database
* (T504, 5 pts) Deploy and configure (add connection string) all **php files** (part of the supporting files set) to all back-end servers
* (T505, 3 pts) Have a fully working web application

