Agenda

- Project
- Project scope – simple_net
- Project scope - LB

Project scope - simple_net

Project scope - simple_net

Develop flask application that get

```
  {
        "name": string,
        "weight_value": number,
        "time": DATE
  }
```

Example:

```
{
        "name": “Baruchi",
        "weight_value": 87,
        "time": "Wed Jun 28 18:05:52 JDT 2023"
}
```

Project scope - simple_net

Store these values in postgressql DB

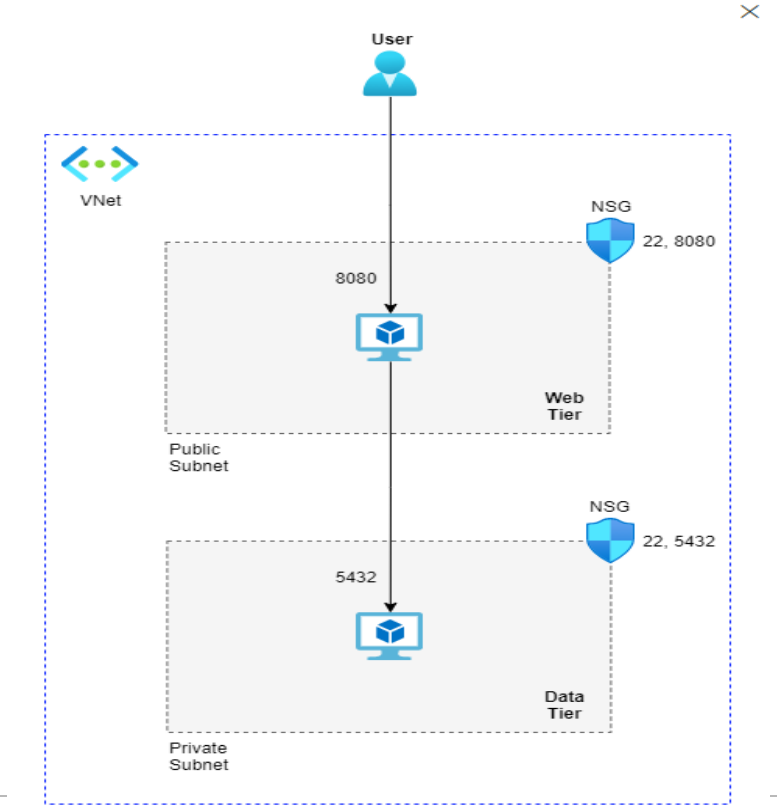
Flask application will run on “Application server” using dedicated public subnet
Postgressql will run on “DB server” using dedicated private subnet

Test the application using postman calls with GET method

Project scope - simple_net

create all the infrastructure required for the flask and db application.
All the infrastructure is not only the virtual machines where you've been deploying the application but also everything around it:
security
networking, etc.
Aspects such as organization, security, performance and cost optimization will be taken into account during the evaluation.

Project scope – simple_net - Diagram



Project scope – simple_net – Con’t

- Create the infrastructure including:

1. Create a Resource Group for your Project
1. Create a Virtual Network with two subnets: the “public” and the “private” subnets
2. Deploy the web server into the public subnet and allow users to reach the application from everywhere
3. Deploy the database server into the private subnet and ensure that there is no public access to the database (only from the application)
1. Configure the database
2. Configure the web server
3. Ensure the application is up and running (and work automatically after reboot)

Project scope – simple_net – Con't

- Considerations
- Ensure your Network Security Group (NSG) allows you to access the servers and allows communication between the web server and the database
- Make sure the database cannot be accessed from the internet (it's not publicly exposed)
- Take into account that you have a limited budget so keep servers down when not needed and use the smallest instances possible to keep costs low
- If you wish, you can use another cloud provider (but take into account that the costs will be on your own)

Project scope – simple_net – Con’t

- Expected Result

- All the infrastructure needed to deploy the Weight Tracker flask application
- The Weight Tracker flask application deployed into it and reachable from your browser
- Database server not accessible from the internet
- Ensure that your application and databases start automatically when an instance is rebooted
- Ensure that your data in the db server is persistent

Project Delivery - simple_net

- Delivery

- Run 2 postman calls: before the server reboot and after – keeps the repossesses files :
  - before_reboot.json
  - after_reboot.json
- Create Template
- Create resource group template and download the template
- Verify that templates exist on your working station !! (laptop / PC / mac / etc)
- Delete resource group
- Convert the template to Bicep files

Project Delivery - simple_net

- Push to git
  - Create public github project (name = weightTracker<your Username>)
  - Create branch feature_simple_net based on main (or master)
  - Push all the templates to feature branch (name= feature_simple_net) under folders: simple_net/templates/json and simple_net/templates/bicep accordingly
  - Push the diagram to folder: simple_net/diagram
  - Push the postman responses to simple_net/reponses/
- Send the git url

- Good Luck !!!!!

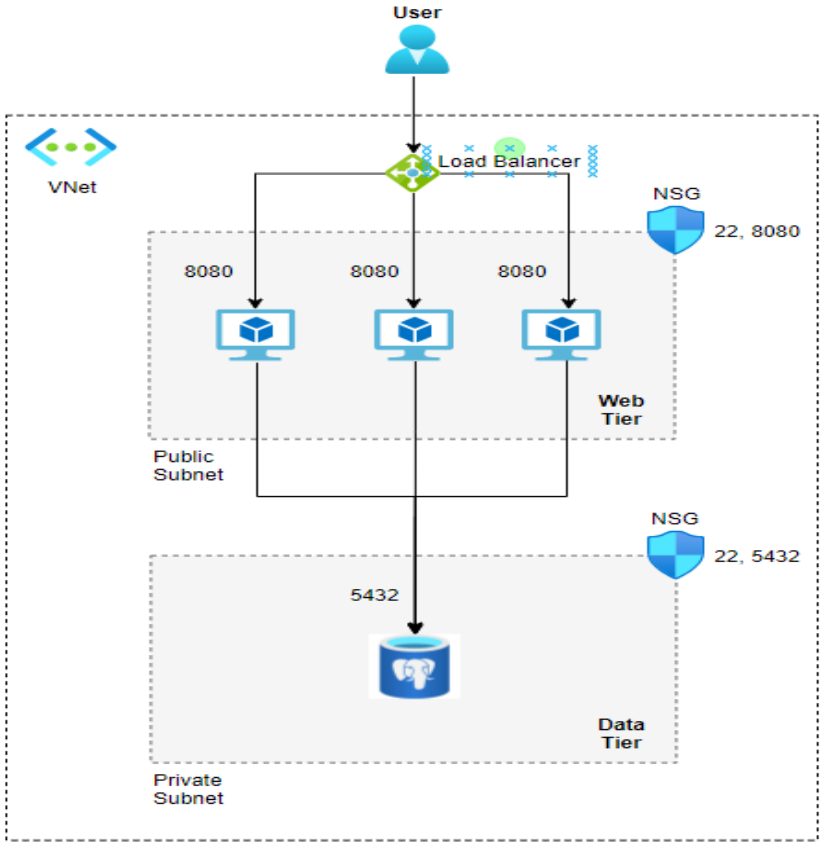
Project scope - LB

Project scope - LB

- Same scope as part simple_net with 1 difference:
- Ensure that your solution is High Available by using multiple application servers utilizing LB (as shown in the diagram below)
- NOT mandatory: Make your solution elastic by using Virtual Machine Scale Sets

- Good Luck !!!!!

Project scope - LB



Project Delivery - LB

- Delivery

- Run 2 postman calls: before the server reboot and after – keeps the repossesses files :
  - before_reboot.json
  - after_reboot.json
- Create Template
- Create resource group template and download the template
- Verify that templates exist on your working station !! (laptop / PC / mac / etc)

- Delete resource group
- Convert the template to Bicep files

Project Delivery - LB

- Push to git:
  - Create public github project (name = weightTracker<your Username>)
  - Create branch feature_lb based on feature_simple_net
  - Push all the templates to feature branch (name= feature_lb) under folders: lb/templates/json and lb/templates/bicep accordingly
  - Push the diagram to folder: lb/diagram
  - Push the postman response to lb/reponses/

- Send the git url

- Good Luck !!!!!