Agenda

- Project scope – simple_net

Project scope - simple_net

Project scope - simple_net

Develop flask application that get

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

Example:

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

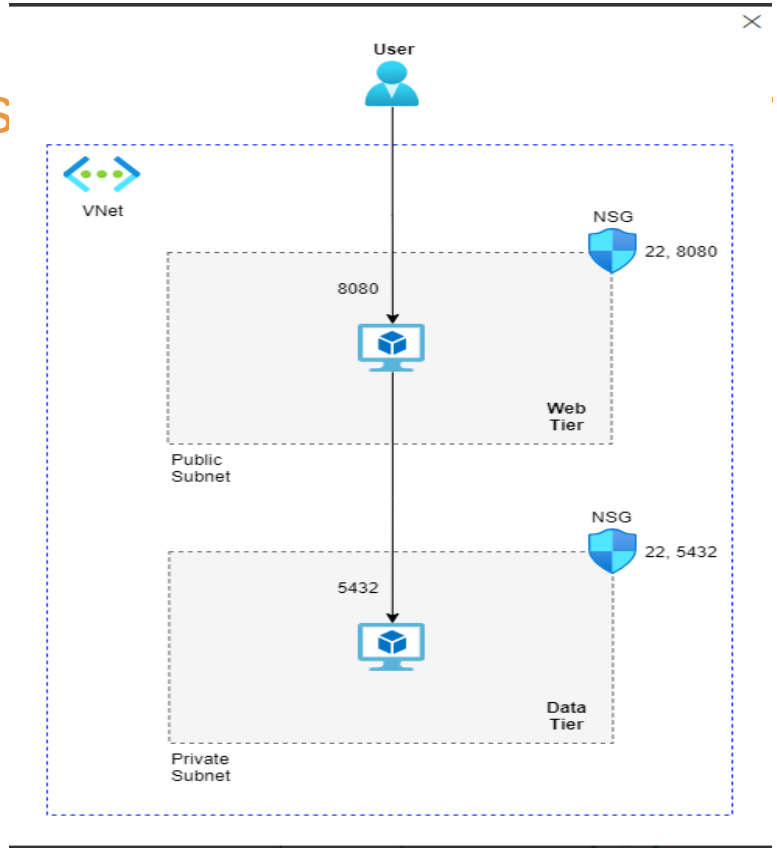
Project scope - simple_net

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 s am



Project scope – simple_net – Details

- Create the infrastructure using terraform templates :

1. Resource Group for your Project
1. 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 – Considerations

- Considerations
- Make sure no passwords exist in variable file
- 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 – Considerations

Terraform considerations

1. Variables (use project-name as a variable)
2. Naming convention
3. Plan -> apply -> destroy -> plan -> apply

Project scope – simple_net – Bonus

Terraform

- Use modules
- Use service principle to authenticate

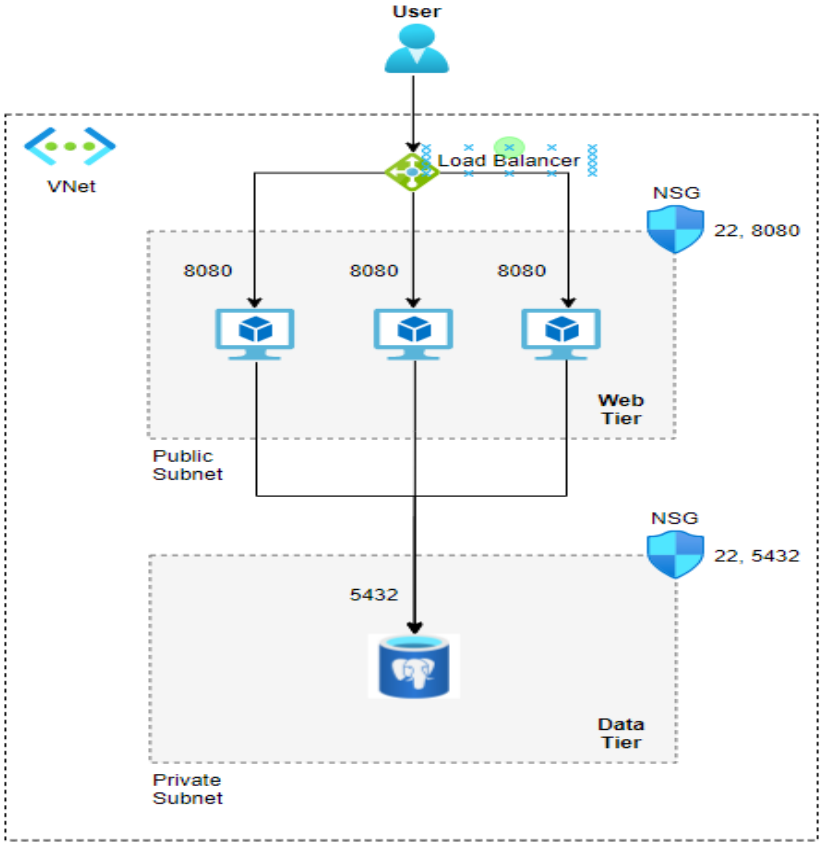
Project Architecture : USE LB – diagram in next slide

- 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

Project Arcitechtire : USE DB service

- Make sure you can connect ONLY from web application vm

Project scope - LB



Project scope – simple_net – Expected Result

- Expected Result

- All the infrastructure needed to deploy your flask application are up and running
- Your 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

- Under git student folder:

- Create Terraform private project
- Application folder – All application folder
- Infrastructure - Terraform templates files
- README file to summarize all project