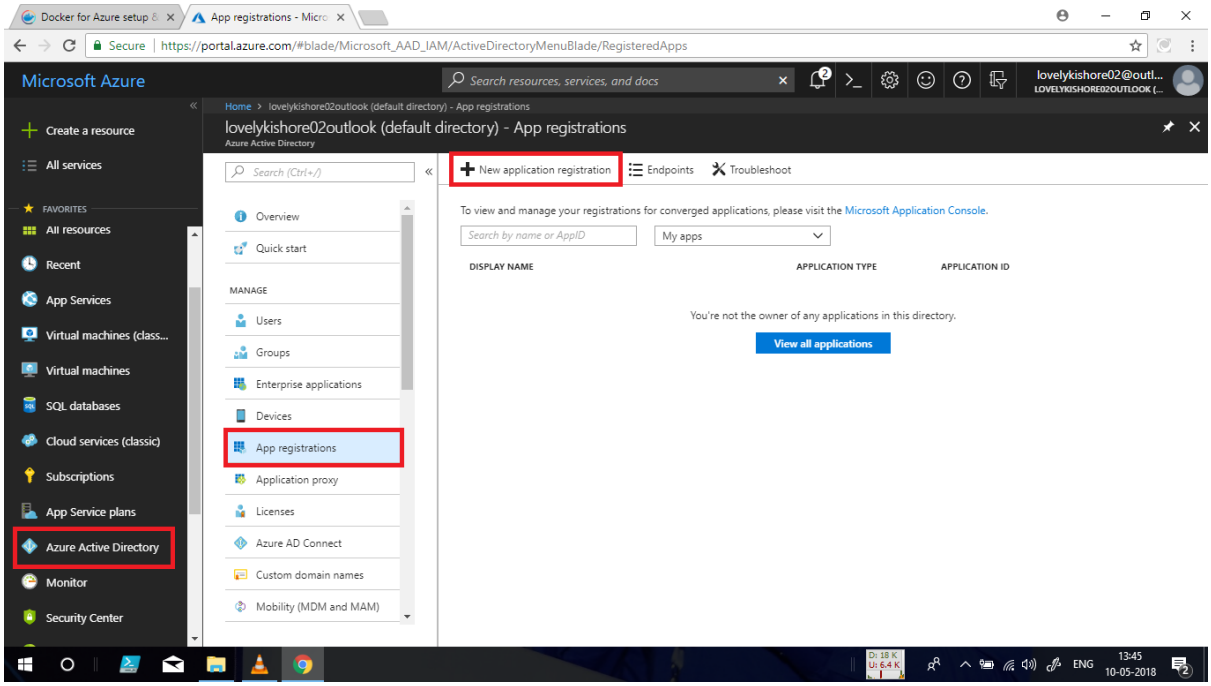
Docker Swarm

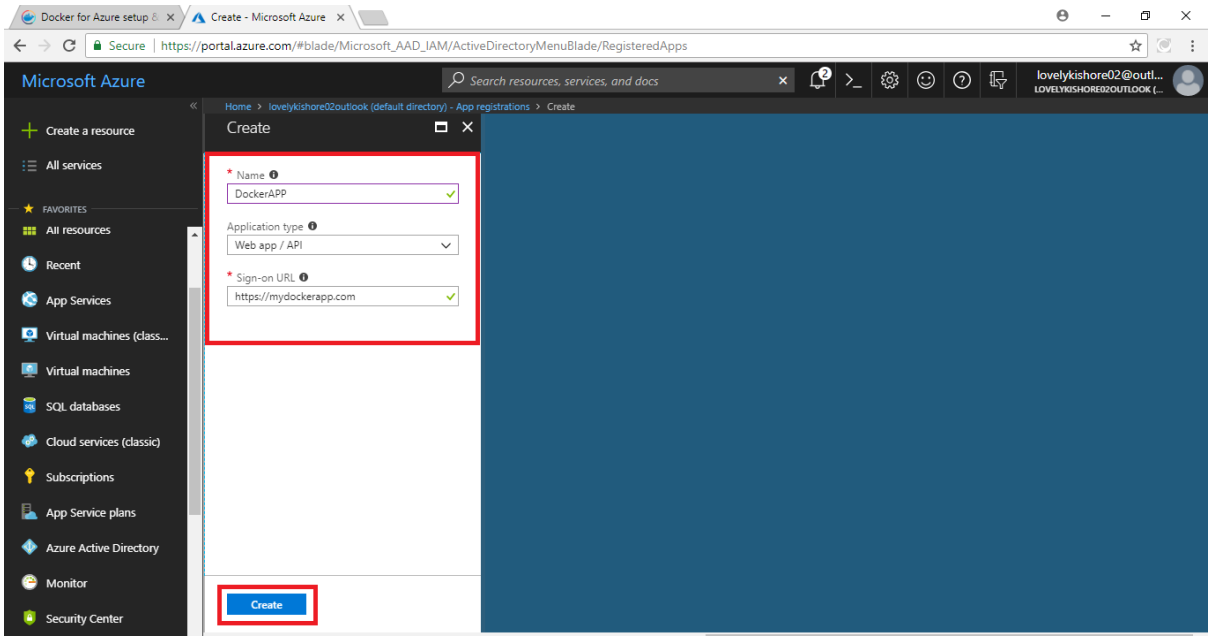
This HOL will guide you in deploying Docker Swarm along and hosting NGINX web server. Followed by the same, load balancing and scaling the number of instances will be shown.

Creating Web API app for role:

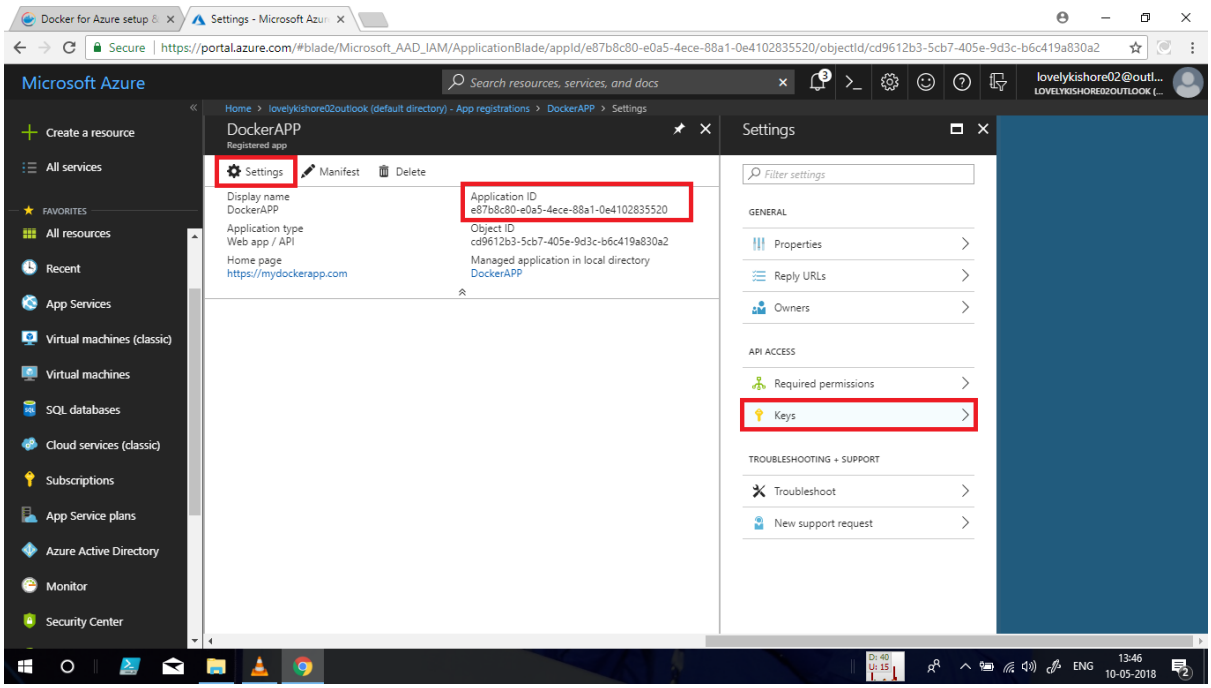
Login into your azure portal and go to **Active Directory->App Registrations->+New application registration**.



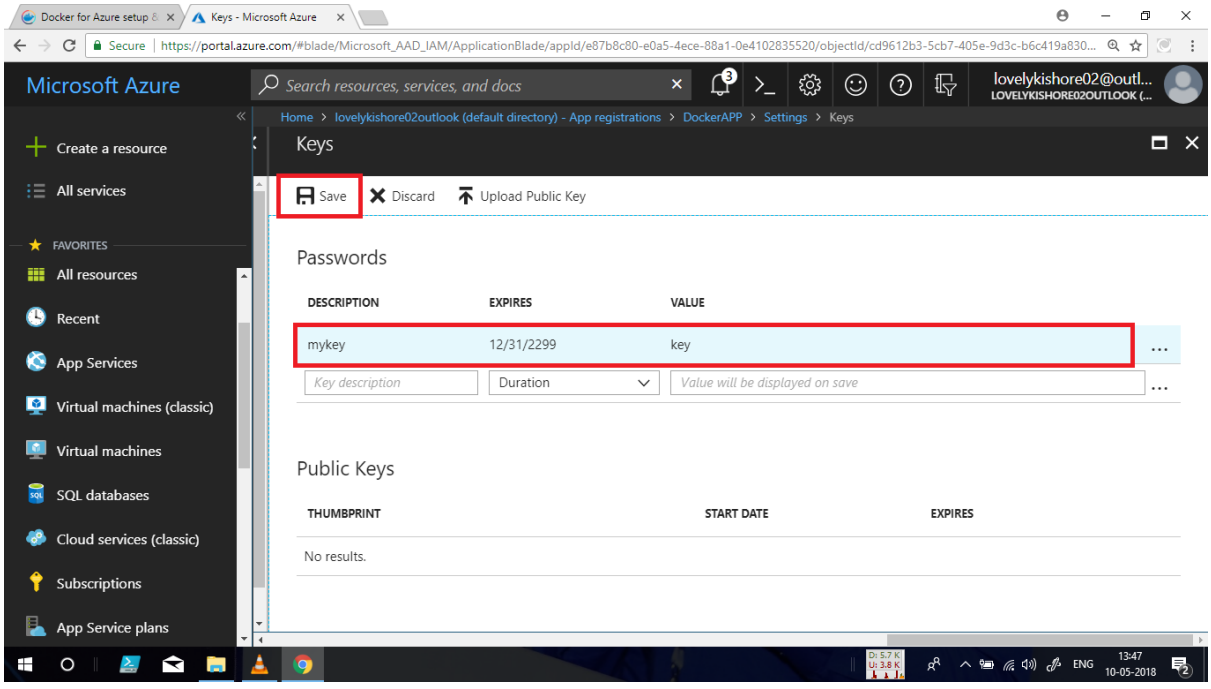
Give a name to your app and choose settings as denoted below. Finally, click on create to deploy an application.



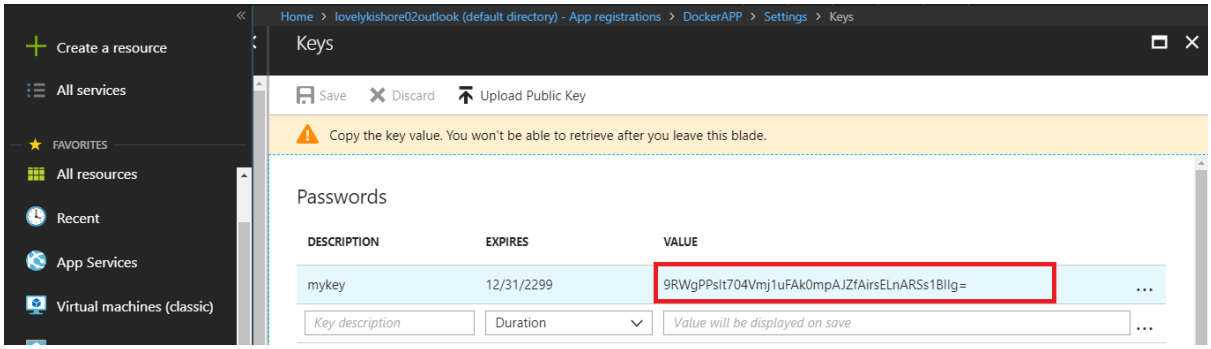
After the app gets created, make note of your **Application ID** and click on settings and go to **keys**.



Give a name for your password and choose an expiry date and click on save button to get your key.

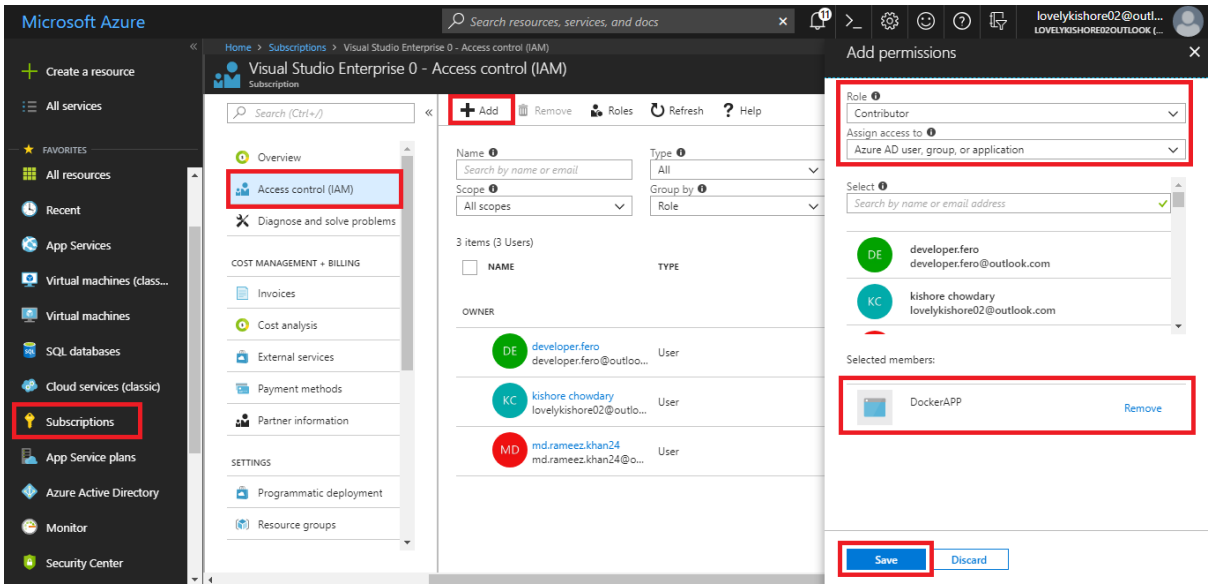


Note the key as well. It is called as app secret and you will be using it in further steps.



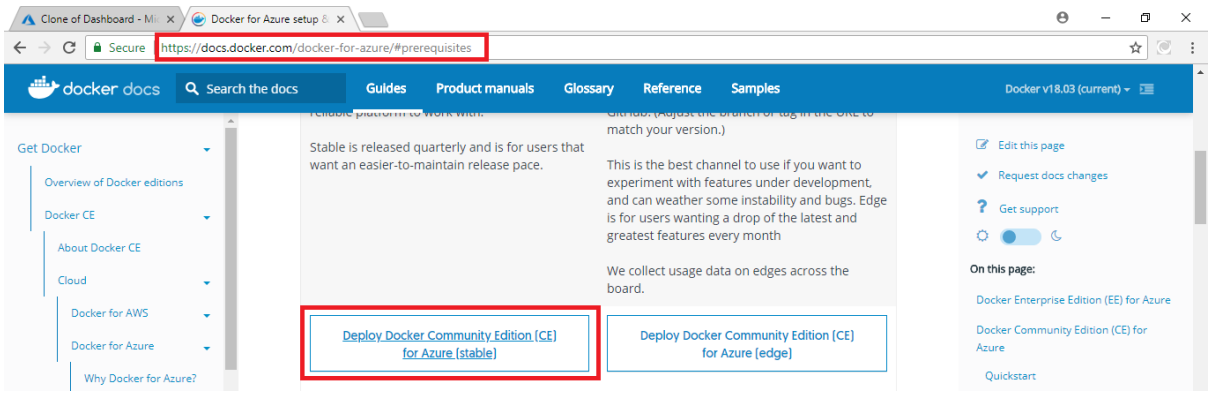
Providing access to Subscription:

Now, let us grant access to subscription for the app that we have created now. For this, go to **subscription->Access Control->Add**. Here, choose the role as contributor and select the app that you created now. Finally, click on **Save** button.

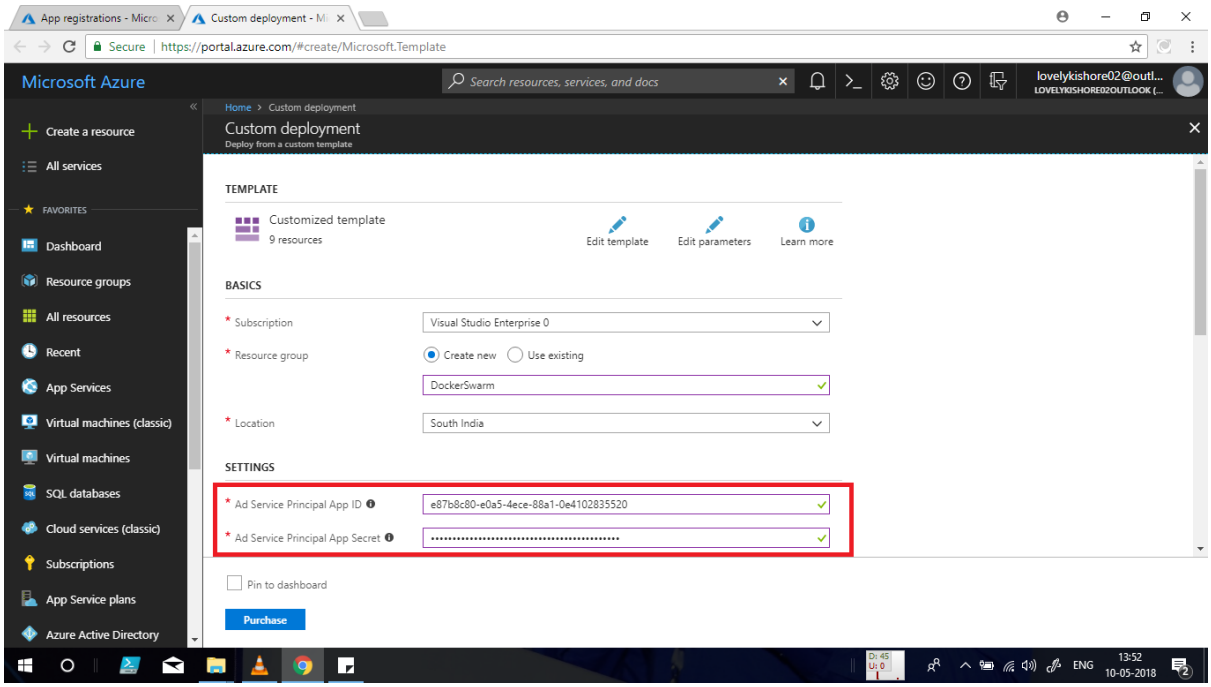


Creating Docker Swarm:

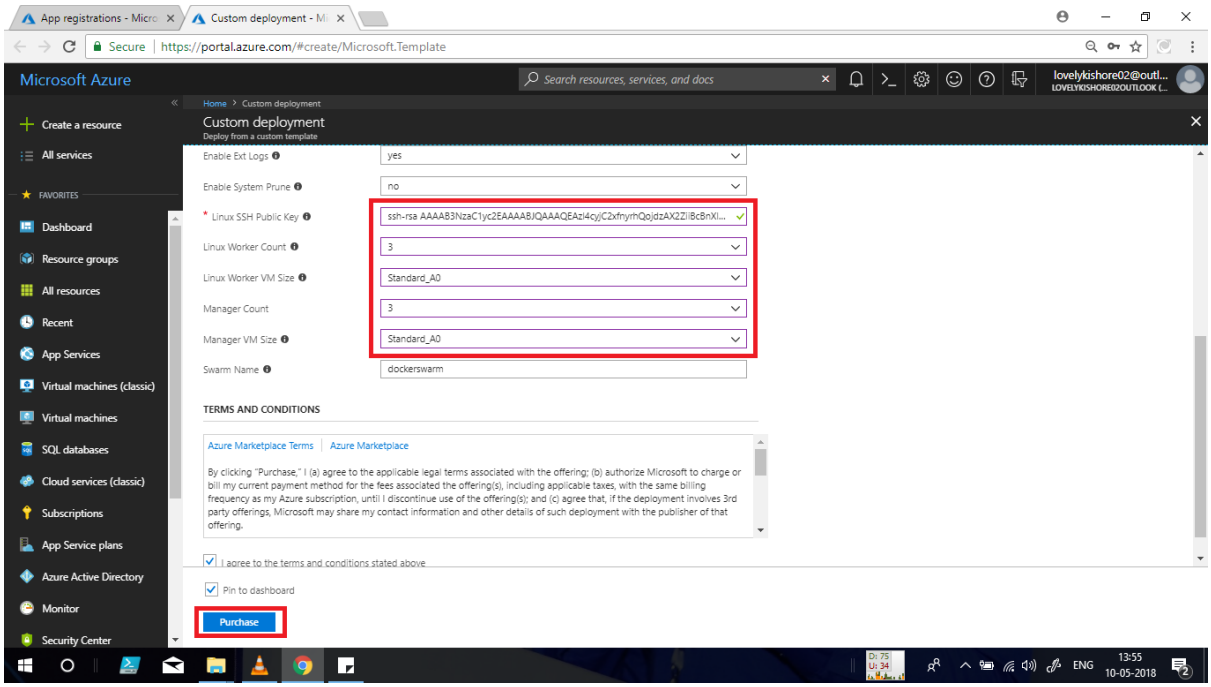
Go to the URL - https://docs.docker.com/docker-for-azure/#quickstart and click on the below denoted menu. This will take you into azure portal again.



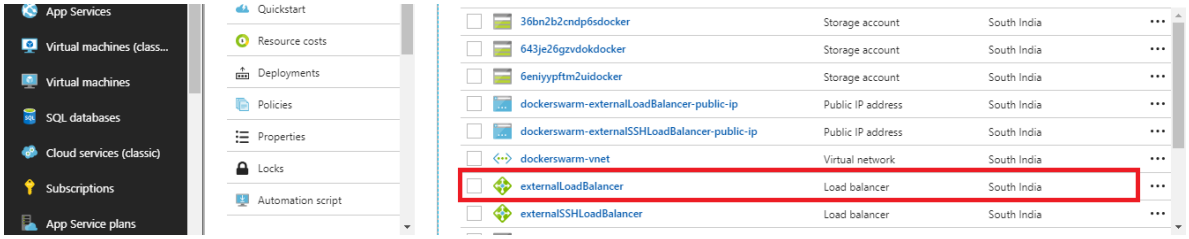
Choose or create a resource group and select a suitable location. Then fill the **App ID** and **App Key** that you copied in the previous step. This is why we registered an application in AD.



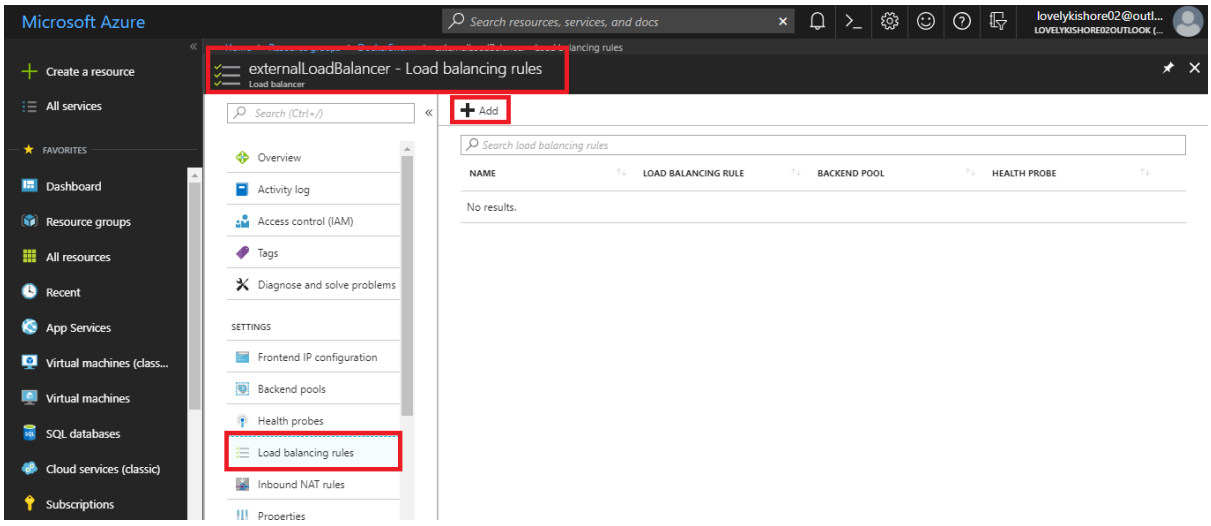
Generate a public key for your Linux environment and paste it in the **Public key** box. Choose the required number of worker count, manager count and VM size. Fill the needed details, agree the terms and click on **Purchase** button. The deployment will take a minimum of 5 minutes depending on the SKUs that you choose.



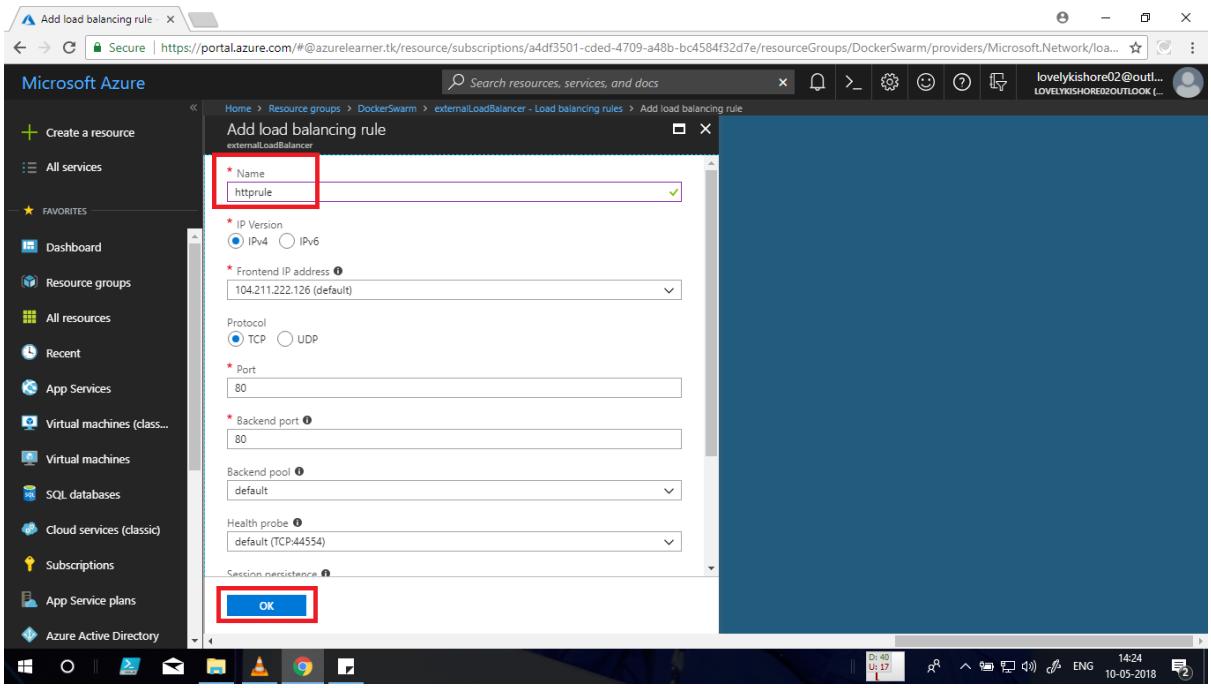
After all the resources get deployed, click on the **externalLoadBalancer**.



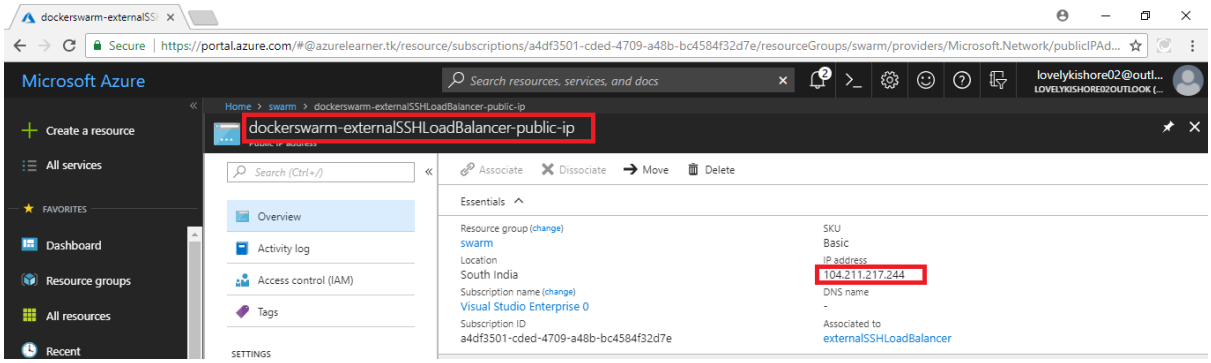
In the left side menu of **externalLoadBalancer** click **on Load Balancing Rules** and add a rule.



Give a name to the rule and click on **OK** without changing any other settings.



Now, copy the public IP address of **externalSSLoadBalancer**. Using the IP address, connect to the Linux VM using Putty.



Make sure to use **Docker** as the user name.

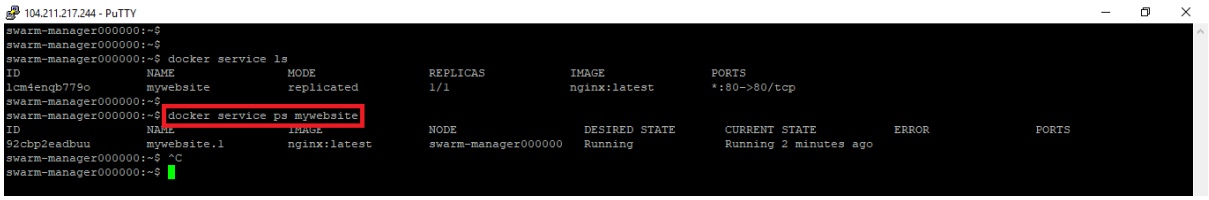
Run the command **docker node ls .** This will show you all the nodes.

Hosting a NGINX server:

Run this command **docker service create --name mywebsite -p 80:80 nginx** to host NGINX server in your container.

Now, run the command **docker service ls** to view the website that you hosted now.

Run the command **docker service ps my website** to see the number of replications of your site.

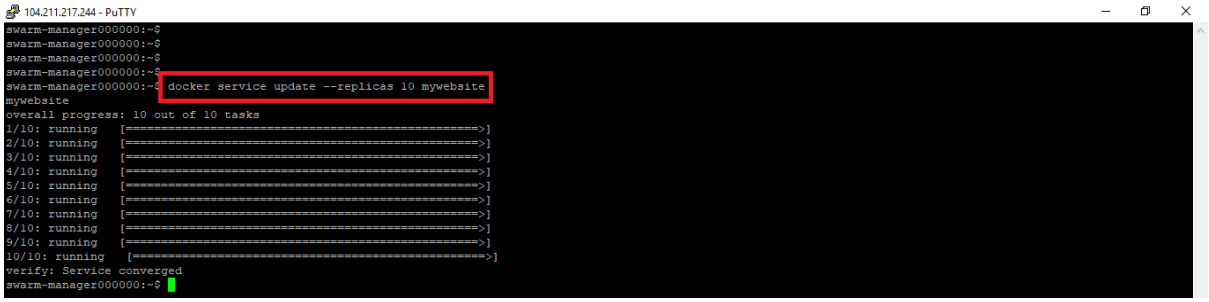


Ping the IP of your **externalSSLoadBalancer** in the browser. You will get response from the NGINX server.

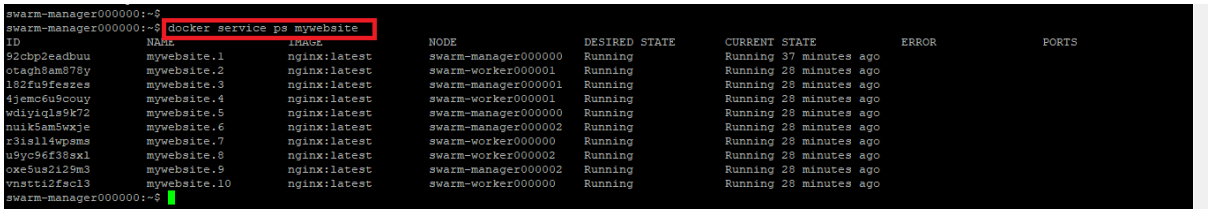
Replicating multiple instances of website:

Run this command to get multiple replicas for your website.

docker service update --replicas 10 mywebsite

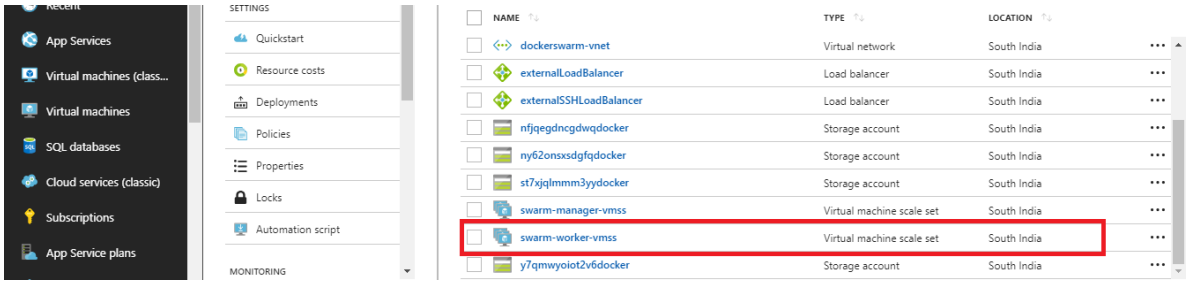


Now, run the command **docker service ps my website .** This will show you all the 10 replicas.

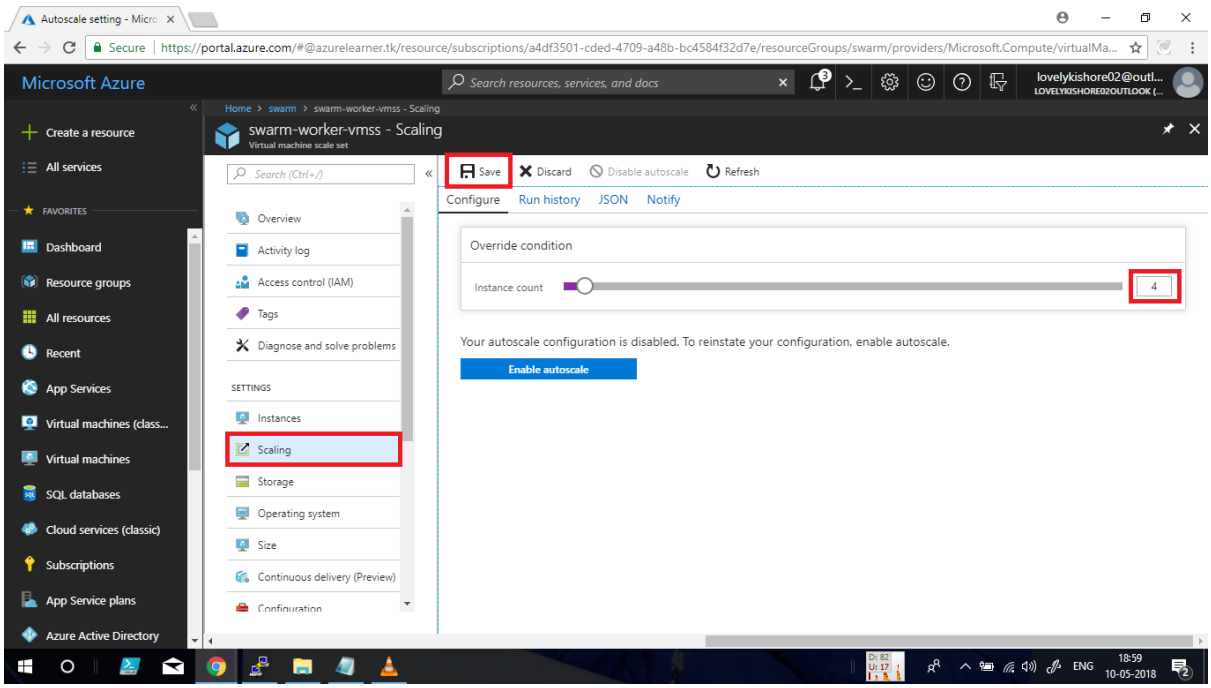


Scaling Instances of Worker VMSS:

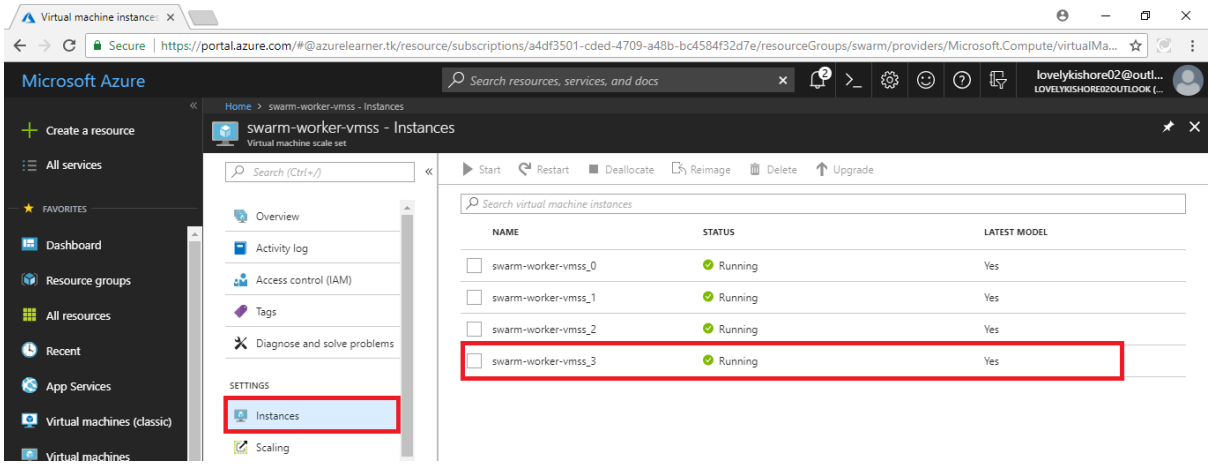
Go to the **Worker-VMSS** in the resources that you created initially.



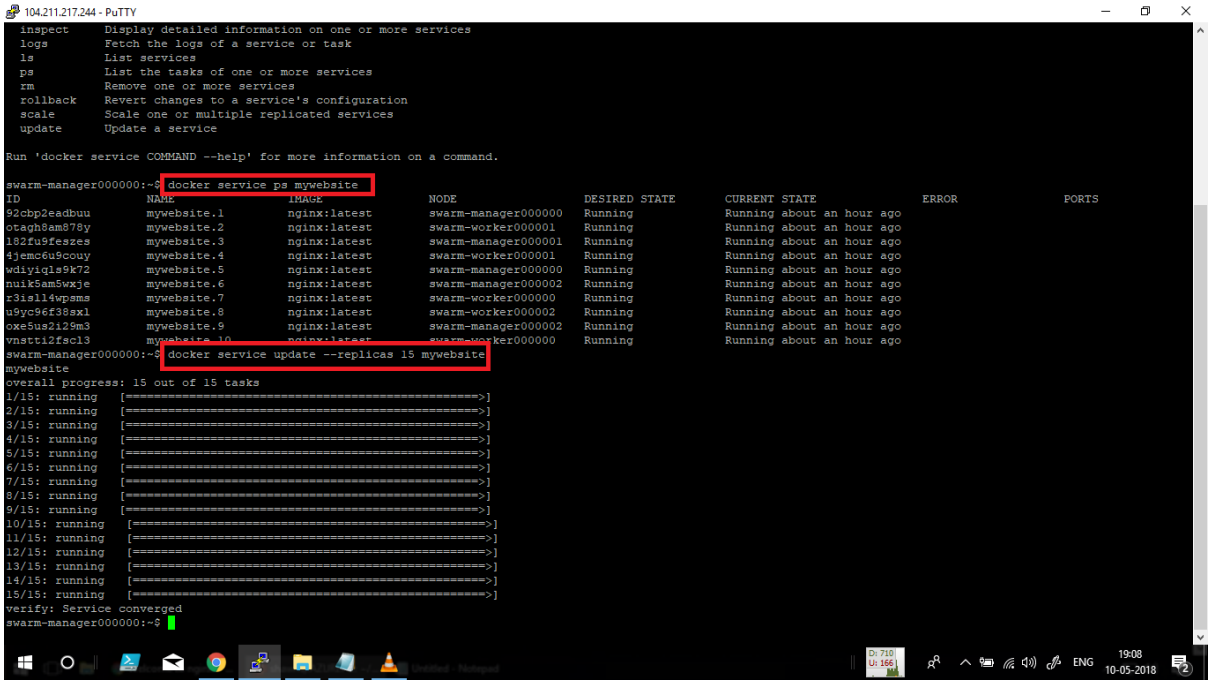
In the left side menu, click on **Scaling** and change the instance count to your requirement and click on save button.



Click on the instances menu in the left side. Wait for the additional instances to spin up. In a while, you will be able to find the newly scaled instances.



At last run the **docker service ps mywebsite** command to get the state of all your containers. Again run the command **docker service update --replicas 15 mywebsite .** This will be spinning up another 5 instances for your **NGINX** server.



At last run the command **docker service ps mywebsite** to get details about every process and the container instance in which it is running.