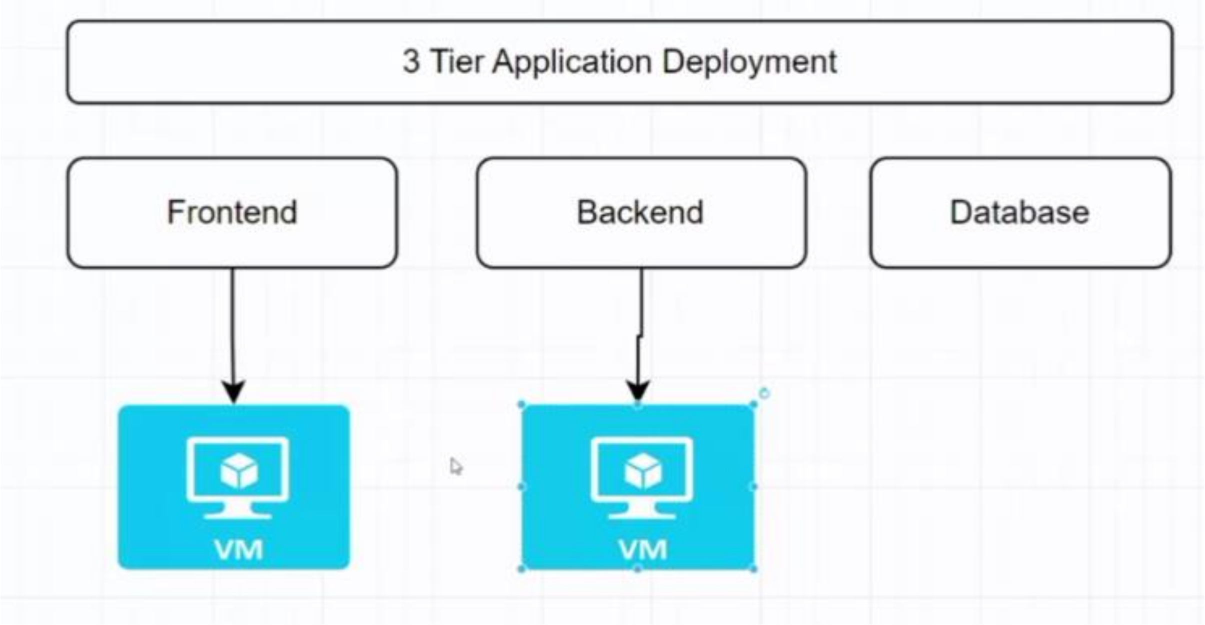
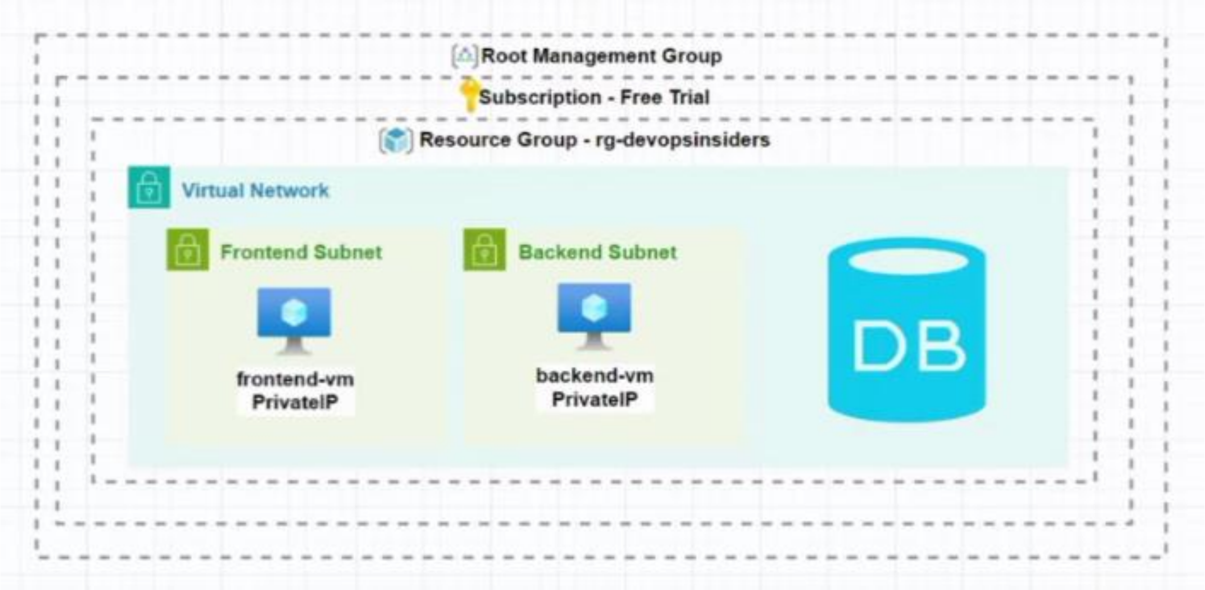
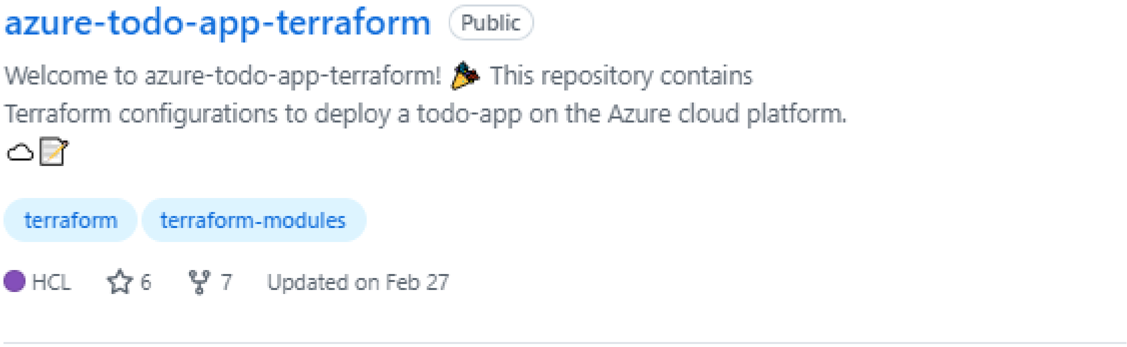
3 AUGUST 2024





1) Using "azure-todo-app-terraform" from website



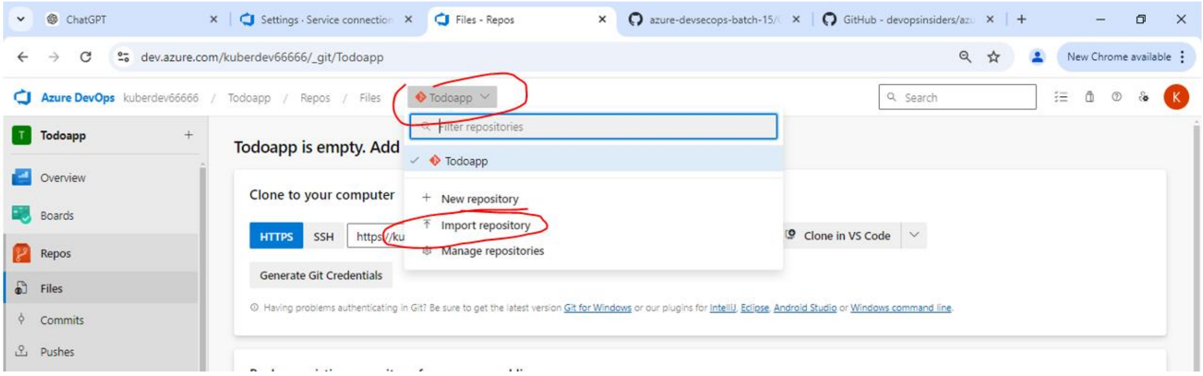
azure-todo-app-terraform Public

Welcome to azure-todo-app-terraform! 🚀 This repository contains Terraform configurations to deploy a todo-app on the Azure cloud platform. ☁️📝

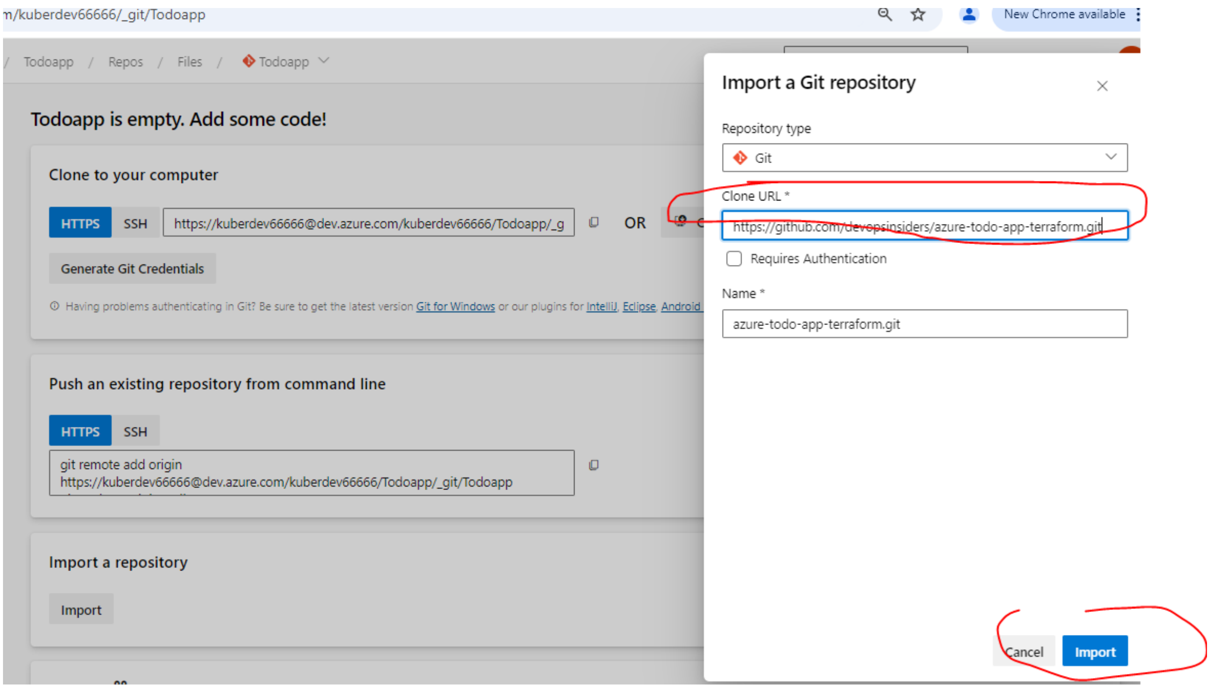
terraform terraform-modules

● HCL ☆ 6 7 Updated on Feb 27

2) Make new project and do import repo as shown below



Having problems authenticating in Git? Be sure to get the latest version Git for Windows or our plugins for IntelliJ, Eclipse, Android Studio or Windows command line.



https://kuberdev66666@dev.azure.com/kuberdev66666/Todoapp/_g

Having problems authenticating in Git? Be sure to get the latest version Git for Windows or our plugins for IntelliJ, Eclipse, Android

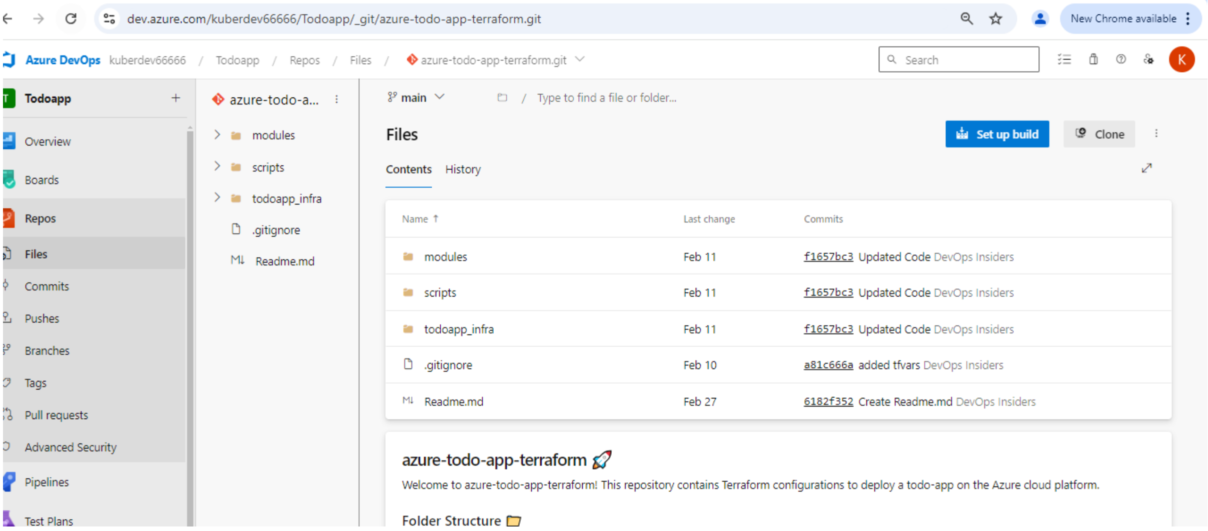
```
git remote add origin
https://kuberdev66666@dev.azure.com/kuberdev66666/Todoapp/_git/Todoapp
```

Repository type: Git

Clone URL *: https://github.com/devopsinsiders/azure-todo-app-terraform.git

☐ Requires Authentication

Name *: azure-todo-app-terraform.git



Name ↑	Last change	Commits
modules	Feb 11	f1657bc3 Updated Code DevOps Insiders
scripts	Feb 11	f1657bc3 Updated Code DevOps Insiders
todoapp_infra	Feb 11	f1657bc3 Updated Code DevOps Insiders
.gitignore	Feb 10	a81c666a added tfvars DevOps Insiders
Readme.md	Feb 27	6182f352 Create Readme.md DevOps Insiders

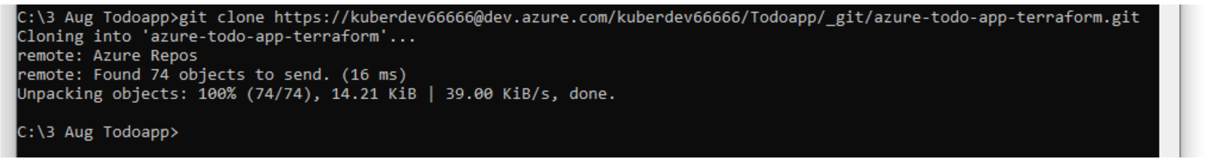
azure-todo-app-terraform 🚀

Welcome to azure-todo-app-terraform! This repository contains Terraform configurations to deploy a todo-app on the Azure cloud platform.

Folder Structure 📁

3) Now we will make pipeline to deploy this imported code

4) After making pipeline git clone the repo in our local and open with vs code



```
C:\3 Aug Todoapp>git clone https://kuberdev66666@dev.azure.com/kuberdev66666/Todoapp/_git/azure-todo-app-terraform.git
Cloning into 'azure-todo-app-terraform'...
remote: Azure Repos
remote: Found 74 objects to send. (16 ms)
Unpacking objects: 100% (74/74), 14.21 KiB | 39.00 KiB/s, done.

C:\3 Aug Todoapp>
```

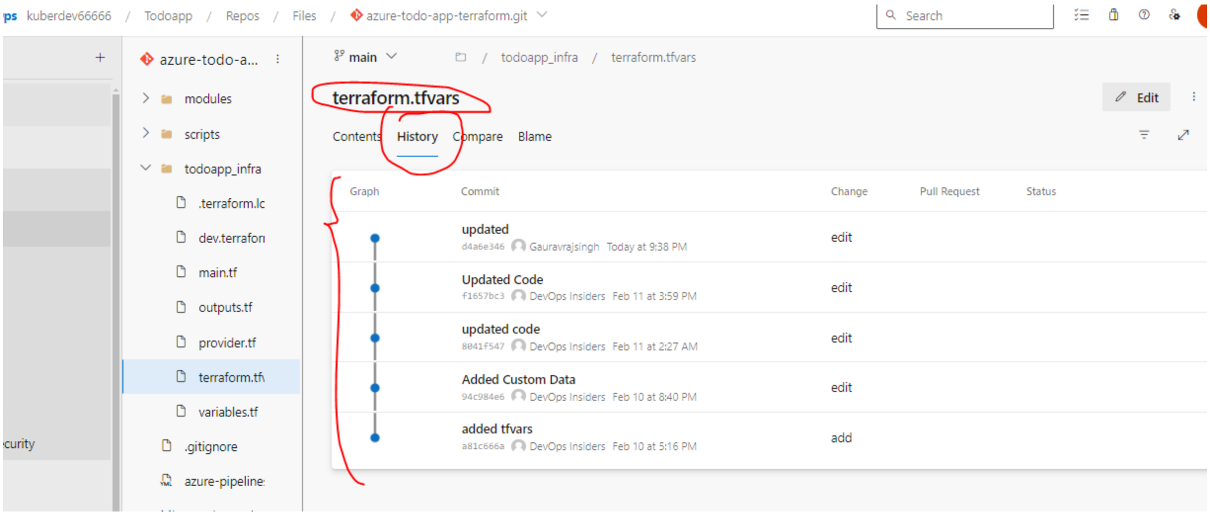
5) Now in vs code made changes and open terminal and perform below steps

i) git add .

ii) git commit -m “updated tfvars”

iii) git push

6) Now refresh dev.azure.com, which will show updated code in it



terraform.tfvars

Graph	Commit	Change	Pull Request	Status
●	updated d4a6e346 Gauravrajsingh Today at 9:38 PM	edit		
●	Updated Code f1657bc3 DevOps Insiders Feb 11 at 3:59 PM	edit		
●	updated code 8041f547 DevOps Insiders Feb 11 at 2:27 AM	edit		
●	Added Custom Data 94c984e6 DevOps Insiders Feb 10 at 8:40 PM	edit		
●	added tfvars a81c666a DevOps Insiders Feb 10 at 5:16 PM	add		

7)

++

8) Functions used in code

i) lookup

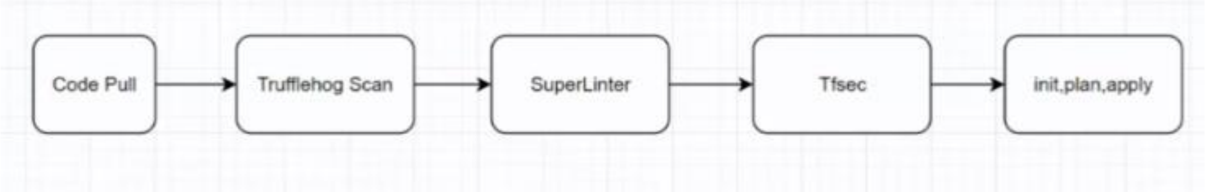
ii) ceil

iii) transpose

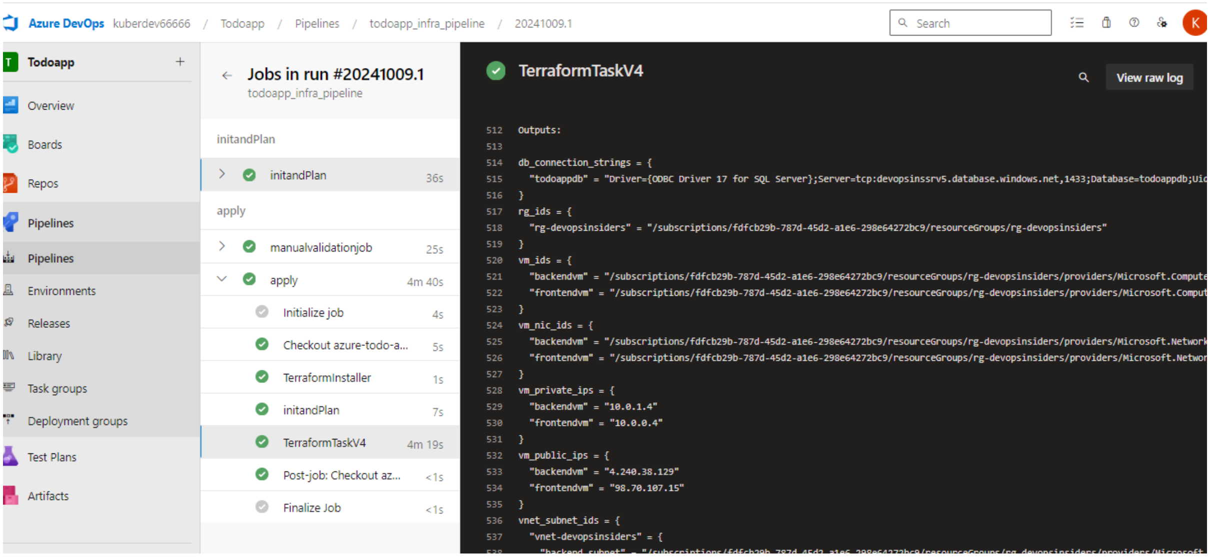
iv)

++

8a) So the flow is as below



9) Now our pipeline is successful means our infra is successfully created to deploy application on it.



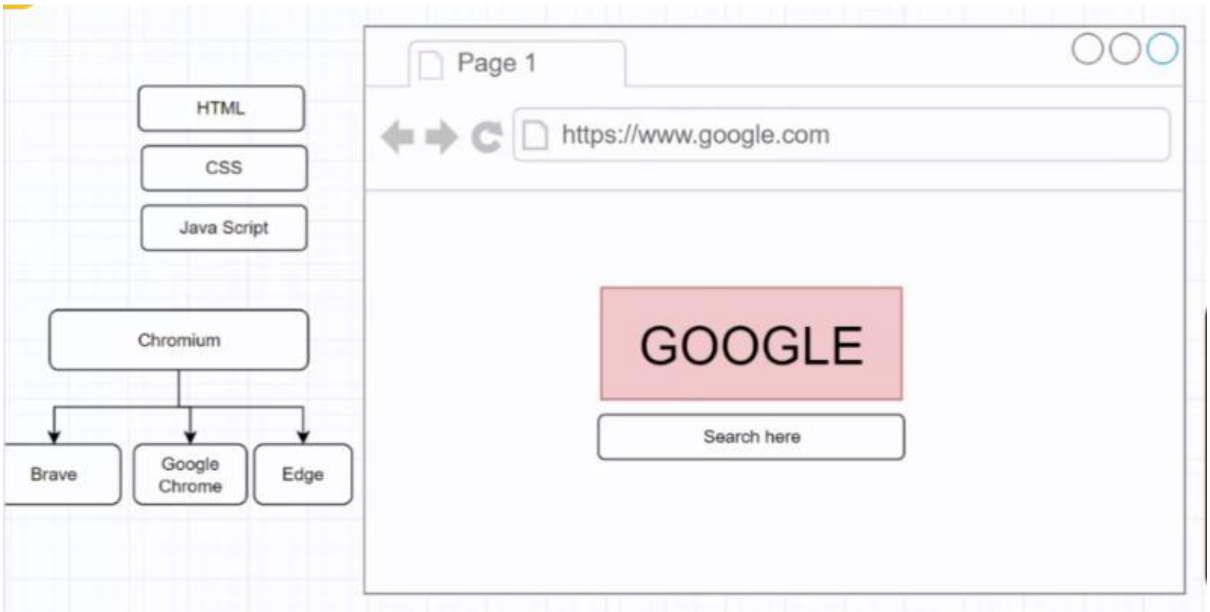
Jobs in run #20241009.1
todoapp_infra_pipeline

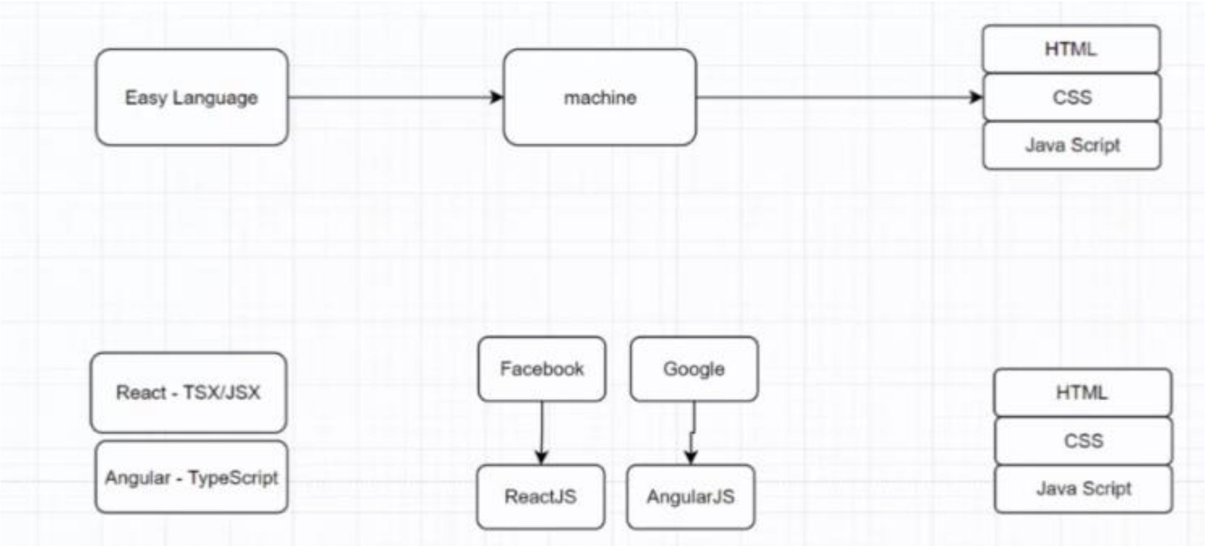
initandPlan
- initandPlan 36s

apply
- manualvalidationjob 25s
- apply 4m 40s
  - Initialize job 4s
  - Checkout azure-todo-a... 5s
  - TerraformInstaller 1s
  - initandPlan 7s
  - TerraformTaskV4 4m 19s
  - Post-job: Checkout az... <1s
  - Finalize Job <1s

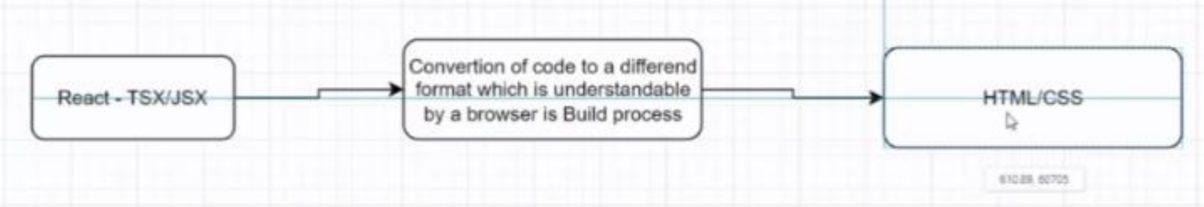
TerraformTaskV4

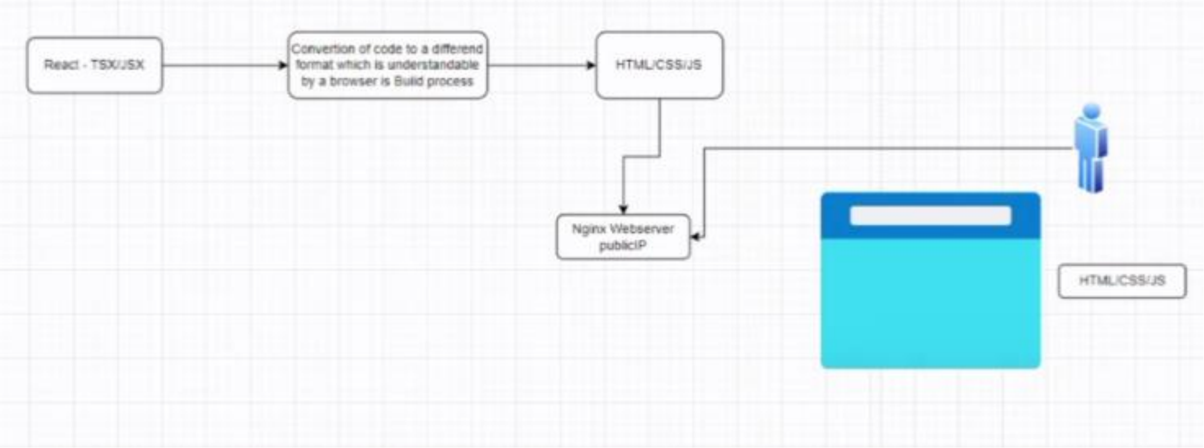
```
Outputs:

db_connection_strings = {
  "todoappdb" = "Driver={ODBC Driver 17 for SQL Server};Server=tcp:devopsinssrv5.database.windows.net,1433;Database=todoappdb;Uid
}
rg_ids = {
  "rg-devopsinsiders" = "/subscriptions/fdfcb29b-787d-45d2-a1e6-298e64272bc9/resourceGroups/rg-devopsinsiders"
}
vm_ids = {
  "backendvm" = "/subscriptions/fdfcb29b-787d-45d2-a1e6-298e64272bc9/resourceGroups/rg-devopsinsiders/providers/Microsoft.Compute
  "frontendvm" = "/subscriptions/fdfcb29b-787d-45d2-a1e6-298e64272bc9/resourceGroups/rg-devopsinsiders/providers/Microsoft.Comput
}
vm_nic_ids = {
  "backendvm" = "/subscriptions/fdfcb29b-787d-45d2-a1e6-298e64272bc9/resourceGroups/rg-devopsinsiders/providers/Microsoft.Network
  "frontendvm" = "/subscriptions/fdfcb29b-787d-45d2-a1e6-298e64272bc9/resourceGroups/rg-devopsinsiders/providers/Microsoft.Networ
}
vm_private_ips = {
  "backendvm" = "10.0.1.4"
  "frontendvm" = "10.0.0.4"
}
vm_public_ips = {
  "backendvm" = "4.240.38.129"
  "frontendvm" = "98.70.107.15"
}
vnet_subnet_ids = {
  "vnet-devopsinsiders" = {
    "backend_subnet" = "/subscriptions/fdfcb29b-787d-45d2-a1e6-298e64272bc9/resourceGroups/rg-devopsinsiders/providers/Microsoft
```



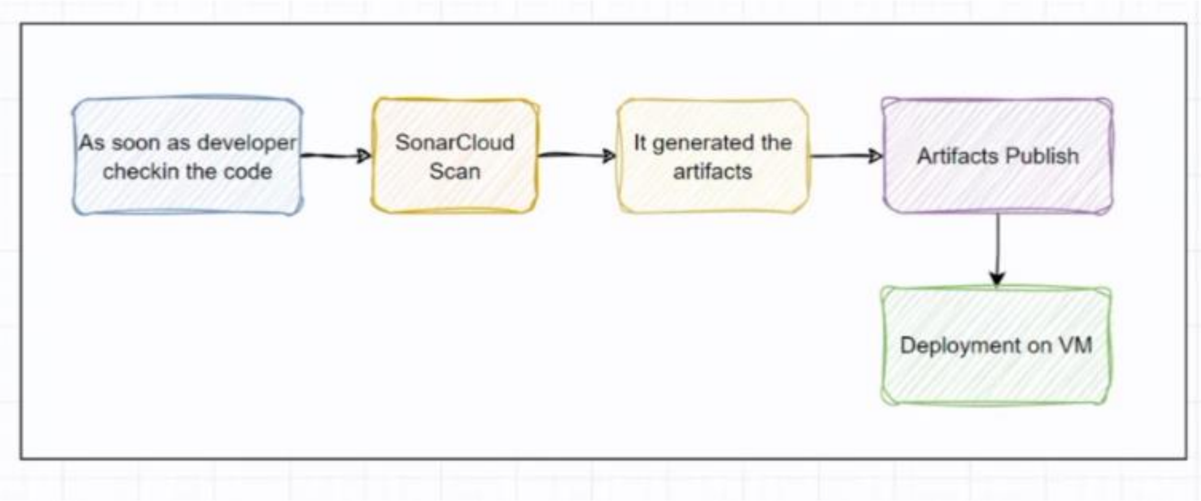


10) **BUILD PROCESS =** Conversation of code to a different format which is understandable by a browser is BUILD PROCESS.


React - TSX/JSX
Convertion of code to a differend format which is understandable by a browser is Build process
HTML/CSS


React - TSX/JSX
Convertion of code to a differend format which is understandable by a browser is Build process
HTML/CSS/JS
Nginx Webserver publicIP
HTML/CSS/JS

11) Now for deployment


As soon as developer checkin the code
SonarCloud Scan
It generated the artifacts
Artifacts Publish
Deployment on VM

++

1) After this monolithic architecture end to end manual deployment will be completed

i) Landing zone with terraform pipeline

ii) nginx data script se khatam

iii) Pipeline deployment completed with this setup i.e. fronted and backend

iv) code quality completed by sonar cube

v) Unit test case alag team

vi) Vulnerability scan by checkmarks tool and checkov tool

vii) infrastructure vulnerability completed by tfsec

viii) Trufflehog se secret management completed

ix) Linting process completed by super linter

x)