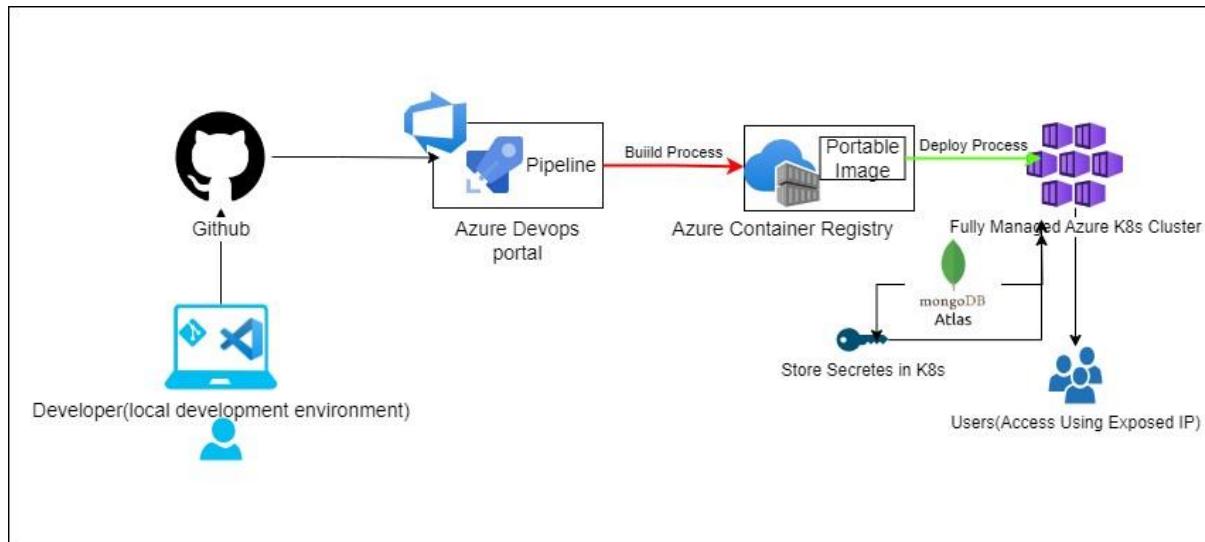


Aim: Kubernetes Deployment.

Architecture:



Steps:

1. Create a Kubernetes cluster:

The screenshot shows the 'Create Kubernetes cluster' wizard on the Azure portal. The current step is 'Basics'. The 'Project details' section shows a selected subscription 'Azure for Students' and a resource group '(New) K8sProject'. The 'Cluster details' section shows a selected cluster preset configuration 'Dev/Test'. At the bottom, there are navigation buttons: '< Previous', 'Next : Node pools >', and 'Review + create'.

The screenshot shows the 'Create Kubernetes cluster' wizard on the Azure portal. The current step is 'Cluster details'. The 'Kubernetes cluster name' is set to 'K8sProjectCluster'. Other settings include Region '(Asia Pacific) Central India', Availability zones 'None', AKS pricing tier 'Free', and Kubernetes version '1.27.7 (default)'. Under 'Authentication and Authorization', it is set to 'Local accounts with Kubernetes RBAC'. A note at the bottom says: 'Once the cluster is deployed, use the Kubernetes CLI to manage RBAC configurations.' Navigation buttons are at the bottom: '< Previous', 'Next : Node pools >', and 'Review + create'.

portal.azure.com/#create/Microsoft.AKS

E-CLASS Announcements Getting Started with... DataCamp's Fast Tra... Best Free Online Co... HTML, CSS, and Jav... ESSO-INCOIS-India... Coderbyte | The #1... All Bookmarks

Microsoft Azure Search resources, services, and docs (G+ /)

diveshkkolhe@gmail.com DEFAULT DIRECTORY DIVESHEK...

Home > Kubernetes services > Create Kubernetes cluster ...

Add node pool Delete

Name	Mode	Node size	OS SKU	Node count
agentpool	System	Standard_DS2_v2 (change)	AzureLinux	1 - 2

Enable virtual nodes

Virtual nodes allow burstable scaling backed by serverless Azure Container Instances. [Learn more about virtual nodes](#)

Enable virtual nodes

Node pool OS disk encryption

By default, all disks in AKS are encrypted at rest with Microsoft-managed keys. For additional control over encryption, you can supply your own keys using a disk encryption set backed by an Azure Key Vault. The disk encryption set will be used to encrypt the OS disks for all node pools in the cluster. [Learn more](#)

Encryption type

< Previous Next : Networking > Review + create Give feedback

Type here to search

Windows Start button Taskbar 05:04 PM 04-12-2023 ENG

portal.azure.com/#create/Microsoft.AKS

E-CLASS Announcements Getting Started with... DataCamp's Fast Tra... Best Free Online Co... HTML, CSS, and Jav... ESSO-INCOIS-India... Coderbyte | The #1... All Bookmarks

Microsoft Azure Search resources, services, and docs (G+ /)

diveshkkolhe@gmail.com DEFAULT DIRECTORY DIVESHEK...

Home > Kubernetes services > Create Kubernetes cluster ...

Azure CNI networking plugin is required for virtual nodes.

Bring your own virtual network

Virtual network * Create new

Cluster subnet * (new) default (10.224.0.0/16)

Virtual nodes subnet * (new) virtual-node-aci (10.239.0.0/16)

Kubernetes service address range * 10.0.0.0/16

Kubernetes DNS service IP address * 10.0.0.10

DNS name prefix * K8sProjectCluster-dns

Network policy None
Allow all ingress and egress traffic to the pods
 Calico
Open-source networking solution. Best for large-scale deployments with strict security requirements
 Azure

< Previous Next : Integrations > Review + create Give feedback

Type here to search

Windows Start button Taskbar 05:04 PM 04-12-2023 ENG

The screenshot shows the Azure portal interface for creating a Kubernetes cluster. At the top, there's a navigation bar with links like 'Home', 'Kubernetes services', and 'Create Kubernetes cluster'. Below the navigation is a form for configuring the cluster. It includes sections for 'Alerts' (with a checked checkbox for 'Enable recommended alert rules'), 'Alert rules' (with a section for 'Alert me if' and 'Notify me by' email), and 'Azure Policy' (with 'Enabled' selected). At the bottom of the form are buttons for '[Review + create](#)' and '[Give feedback](#)'. The taskbar at the bottom shows various pinned icons.

Create with this basic and simple configuration.

The screenshot shows the Azure portal overview page for a Kubernetes cluster named 'microsoft.aks-20231204170133'. The main heading says 'Your deployment is complete'. Below it, there's a table of deployment details:

Resource	Type	Status	Operation details
ClusterSubnetRoleAssignmentDeploy	Microsoft.Resources/deployments	OK	Operation details
K8sProjectCluster	Microsoft.ContainerService/managedClusters	OK	Operation details
AciSubnetRoleAssignmentDeploy	Microsoft.Resources/deployments	OK	Operation details
InsightsMetricAlertsDepl-111f368	Microsoft.Resources/deployments	OK	Operation details
K8sProjectCluster	Microsoft.ContainerService/managedClusters	OK	Operation details
VnetDeployment-111f368-4597-	Microsoft.Resources/deployments	OK	Operation details
InsightsActionGroupDepl-111f368	Microsoft.Resources/deployments	OK	Operation details

On the right side, there are promotional cards for 'Cost Management', 'Container Insights', and 'Free Microsoft tutorials'. The taskbar at the bottom shows pinned icons.

2. Connect to Cluster Using Cloud Shell.

The screenshot shows the Azure portal interface with the URL `portal.azure.com/#@diveshkkolhe@gmail.onmicrosoft.com/resource/subscriptions/b858338e-efa7-486d-aebd-c1573aab0e40/resourcegroups/K8sProject/providers/Microsoft.ContainerService/managedClusters/K8sProjectCluster`. On the left, there's a sidebar with 'K8sProjectCluster' selected under 'Kubernetes service'. The main area shows the cluster's overview with details like Resource group: K8sProject, Status: Succeeded (Running), Location: Central India, Subscription: Azure for Students, and a single node pool named 'Standard_DS2_v2'. On the right, a 'Connect to K8sProjectCluster' dialog is open. It has tabs for 'Cloud shell', 'Azure CLI', and 'Run command'. Under 'Cloud shell', it provides a command line to run: `az account set --subscription b858338e-efa7-486d-aebd-c1573aab0e40`. Below that, it says 'Download cluster credentials' with the command: `az aks get-credentials --resource-group K8sProject --name K8sProjectCluster`. A 'Sample commands' section shows examples like 'az deployment list' and 'kubectl get deployments --all-namespaces'. At the bottom right of the dialog is a 'Close' button.

This screenshot is similar to the one above, showing the 'Connect to K8sProjectCluster' dialog. However, a modal window is overlaid on the main content. The modal title is 'You have no storage mounted'. It contains text explaining that Azure Cloud Shell requires an Azure file share to persist files, with a link to 'Learn more' and a note about a small monthly cost. It also states that Azure Cloud Shell will register the subscription with MicrosoftCloudShell resource provider. The modal has a dropdown for 'Subscription' currently set to 'Azure for Students', a 'Show advanced settings' link, and two buttons at the bottom: 'Create storage' and 'Close'. The background of the main window is darkened.

Establish the connection.

The screenshot shows the Microsoft Azure portal interface. In the center, a modal dialog titled "Connect to K8sProjectCluster" is open. It contains three sections: "Set the cluster subscription" with the command "az account set --subscription b858338e-efa7-486d-aebd-c1573aab0e40", "Download cluster credentials" with the command "az aks get-credentials --resource-group K8sProject --name K8sProjectCluster", and "Sample commands". Below the dialog is a "Bash" terminal window. The terminal output shows the user running "az account set" and "az aks get-credentials" commands, which both succeed. The user then runs "kubectl get services", which lists a single service named "kubernetes" with a Cluster-IP of 10.0.0.1 and an External-IP of <none>. The terminal window has a dark theme and is part of a Windows desktop environment.

```
Requesting a Cloud Shell. Succeeded.  
Connecting terminal...  
  
divesh [ ~ ]$ az account set --subscription b858338e-efa7-486d-aebd-c1573aab0e40  
divesh [ ~ ]$ az aks get-credentials --resource-group K8sProject --name K8sProjectCluster  
Merged "K8sProjectCluster" as current context in /home/divesh/.kube/config  
divesh [ ~ ]$  
  
divesh [ ~ ]$ kubectl get services  
NAME      TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE  
kubernetes   ClusterIP  10.0.0.1    <none>        443/TCP   9m37s  
divesh [ ~ ]$
```

This screenshot is nearly identical to the one above, showing the same "Connect to K8sProjectCluster" dialog and the "Bash" terminal window. The terminal output shows the user running "kubectl get services" again, with the same result. The desktop taskbar at the bottom includes icons for File Explorer, Task View, Edge browser, and other standard Windows applications.

```
divesh [ ~ ]$ kubectl get services  
NAME      TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE  
kubernetes   ClusterIP  10.0.0.1    <none>        443/TCP   9m37s  
divesh [ ~ ]$
```

I have already created a docker container of my code and stored it on docker hub.

The screenshot shows the Docker Hub interface for the repository `skstudies/blog-website`. The repository has one tag, `v1.0.0`, which was pushed 2 hours ago. The Docker commands section shows the command to push a new tag: `docker push skstudies/blog-website:tagname`. The interface includes tabs for General, Tags, Builds, Collaborators, Webhooks, and Settings. A sidebar on the right provides information about automated builds.

Create a deployment.yaml file

The screenshot shows the Azure Cloud Shell interface. A terminal window displays the creation of a `deployment.yaml` file:

```
1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
4   name: blog-app-deployment
5 spec:
6   replicas: 3
7   selector:
8     matchLabels:
9       app: blog-app
10  template:
11    metadata:
12      labels:
13        app: blog-app
14    spec:
15      containers:
16        - name: blog-app
17          image: skstudies/blog-website:v1.0.0
18          ports:
19            - containerPort: 5000
20          env:
21            - name: MONGODB_URL
```

After saving the file, the terminal shows the deployment process:

```
Requesting a Cloud Shell.Succeeded.
Connecting terminal...
divesh [ ~ $] code blog-app-deployment.yaml
divesh [ ~ $] kubectl get pods
No resources found in default namespace.
divesh [ ~ $] kubectl get services
```

The terminal also lists available services:

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
------	------	------------	-------------	---------	-----

Create a yaml file for service;

The screenshot shows a Microsoft Azure Bash terminal window. At the top, there's a browser-like header with tabs and links. Below it is the terminal interface with a dark background and white text. The user has created a file named 'blog-app-service.yaml' containing the following YAML configuration:

```
1 apiVersion: v1
2 kind: Service
3 metadata:
4   name: blog-app-service
5 spec:
6   selector:
7     app: blog-app
8   ports:
9     - protocol: TCP
10    port: 80
11    targetPort: 5000
12   type: LoadBalancer
```

After saving the file, the user runs the command `kubectl get services` to check the status of the service. The output shows a single ClusterIP service named 'ClusterIP' with port 443/TCP and an age of 32m.

```
divesh [ ~ ]$ kubectl get services
NAME      TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes  ClusterIP  10.0.0.1   <none>        443/TCP   32m
```

The terminal also shows the user navigating between files ('code blog-app-service.yaml') and running the command again ('code blog-app-service.yaml').

Apply the two created files.

This screenshot shows the continuation of the Azure Bash session. The user has applied the 'blog-app-deployment.yaml' file using the command `kubectl apply -f blog-app-deployment.yaml`. This creates a Deployment named 'blog-app-deployment' with three pods: '796b7f9d7d-55tzq', '796b7f9d7d-v24wh', and '796b7f9d7d-vlqrw'. All pods are currently in a 'ContainerCreating' state.

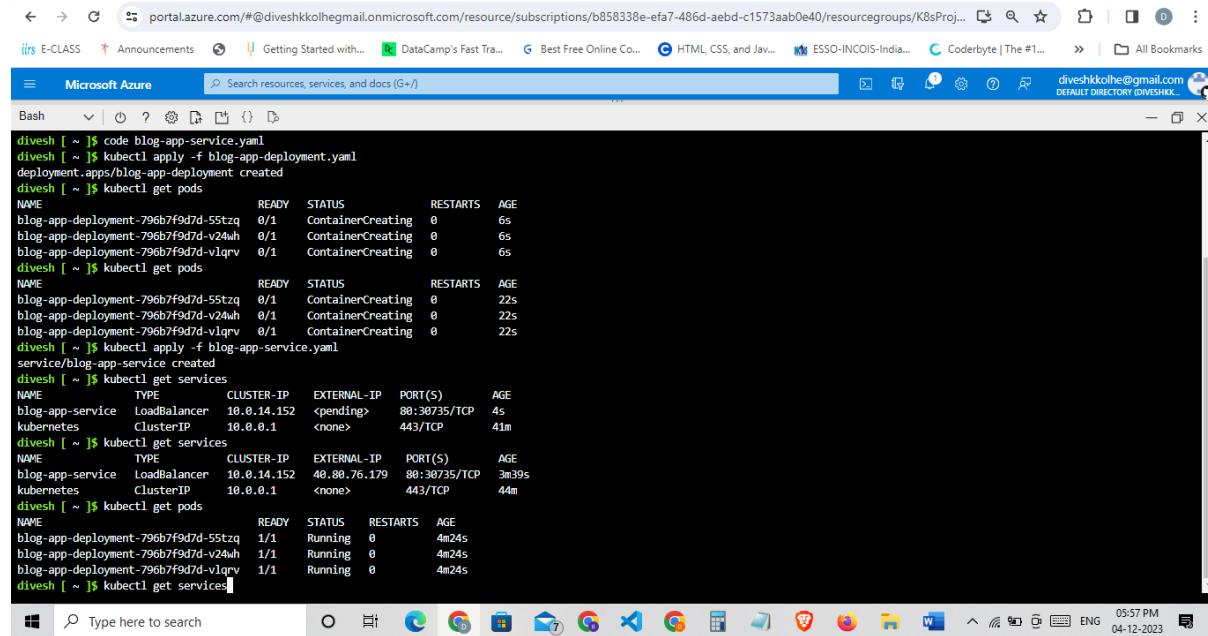
```
divesh [ ~ ]$ kubectl apply -f blog-app-deployment.yaml
deployment.apps/blog-app-deployment created
```

Next, the user applies the 'blog-app-service.yaml' file using the command `kubectl apply -f blog-app-service.yaml`. This creates a Service named 'blog-app-service' of type 'LoadBalancer' with an external IP of '10.0.14.152' and port 80 mapped to the internal port 30735/TCP.

```
divesh [ ~ ]$ kubectl apply -f blog-app-service.yaml
service/blog-app-service created
```

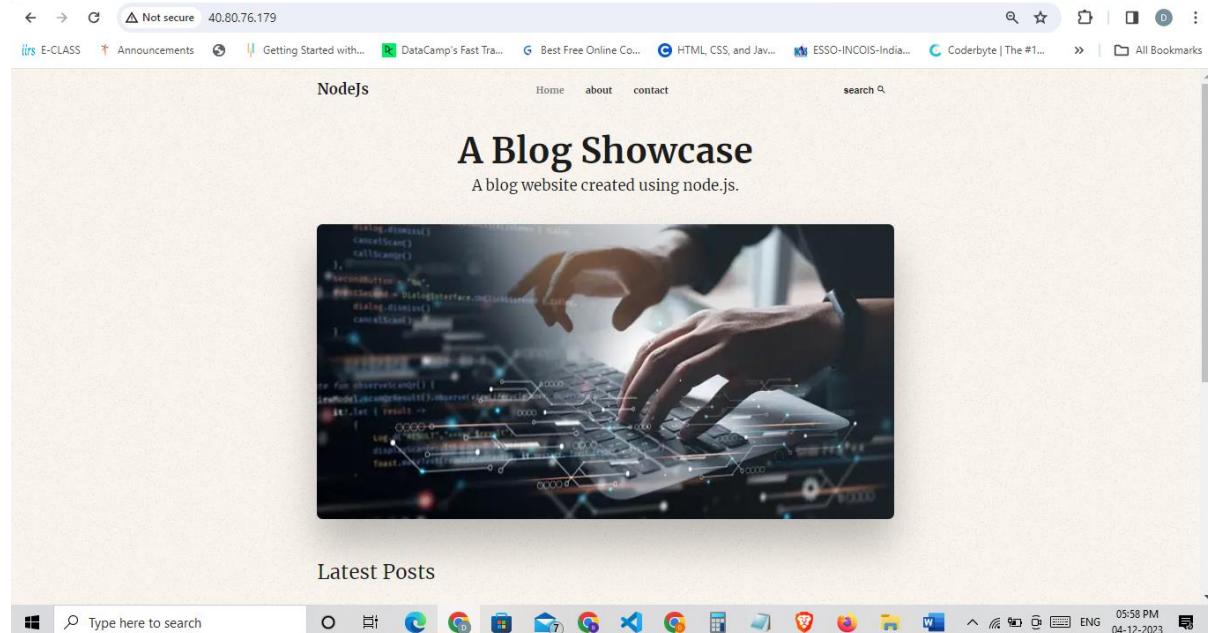
The terminal also shows the user navigating between files ('code blog-app-service.yaml') and running the command again ('code blog-app-service.yaml').

Wait for pods to run.



```
divesh [ ~ ]$ code blog-app-service.yaml
divesh [ ~ ]$ kubectl apply -f blog-app-deployment.yaml
deployment.apps/blog-app-deployment created
divesh [ ~ ]$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
blog-app-deployment-796b7f9d7d-55tzq  0/1     ContainerCreating  0          6s
blog-app-deployment-796b7f9d7d-v24wh  0/1     ContainerCreating  0          6s
blog-app-deployment-796b7f9d7d-vlqrw  0/1     ContainerCreating  0          6s
divesh [ ~ ]$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
blog-app-deployment-796b7f9d7d-55tzq  0/1     ContainerCreating  0          22s
blog-app-deployment-796b7f9d7d-v24wh  0/1     ContainerCreating  0          22s
blog-app-deployment-796b7f9d7d-vlqrw  0/1     ContainerCreating  0          22s
divesh [ ~ ]$ kubectl apply -f blog-app-service.yaml
service/blog-app-service created
divesh [ ~ ]$ kubectl get services
NAME            TYPE           CLUSTER-IP      EXTERNAL-IP      PORT(S)        AGE
blog-app-service   LoadBalancer   10.0.14.152   <pending>       80:30735/TCP   4s
kubernetes       ClusterIP      10.0.0.1       <none>          443/TCP       41m
divesh [ ~ ]$ kubectl get services
NAME            TYPE           CLUSTER-IP      EXTERNAL-IP      PORT(S)        AGE
blog-app-service   LoadBalancer   10.0.14.152   40.80.76.179   80:30735/TCP   3m39s
kubernetes       ClusterIP      10.0.0.1       <none>          443/TCP       44m
divesh [ ~ ]$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
blog-app-deployment-796b7f9d7d-55tzq  1/1     Running   0          4m24s
blog-app-deployment-796b7f9d7d-v24wh  1/1     Running   0          4m24s
blog-app-deployment-796b7f9d7d-vlqrw  1/1     Running   0          4m24s
divesh [ ~ ]$ kubectl get services
```

Search the external ip



```
1 az account set --subscription b858338e-efa7-486d-aebd-c1573aab0e40
2 az aks get-credentials --resource-group K8sProject --name K8sProjectCluster
3 kubectl get pods
4 clear
5 kubectl get services
6 ls
7 apt update
8 clear
9 code blog-app.yaml
10 az account set --subscription b858338e-efa7-486d-aebd-c1573aab0e40
11 az aks get-credentials --resource-group K8sProject --name K8sProjectCluster
12 clear
13 code blog-app-deployment.yaml
14 kubectl get pods
15 kubectl get services
16 code blog-app-service.yaml
17 kubectl apply -f blog-app-deployment.yaml
18 kubectl get pods
19 kubectl apply -f blog-app-service.yaml
20 kubectl get services
21 kubectl get pods
22 code blog-app-service.yaml
23 code blog-app.yaml
24 code blog-app-deployment.yaml
25 cat blog-app-deployment.yaml
26 history
```

This Was The Manual Deployment.

Aim: Lets Try The CICD Deployment.

1. Create ACR

The screenshot shows the 'Create container registry' wizard in the Azure portal. The 'Project details' section is filled with the following information:

- Subscription: Azure for Students
- Resource group: (New) todoCI/CDk8s

The 'Instance details' section includes:

- Registry name: todoacrk8s
- Location: Central India
- Use availability zones: (checkbox unchecked)
- Pricing plan: Basic

At the bottom, there are 'Review + create' and 'Next: Networking >' buttons.

Enable Access

The screenshot shows the 'Access keys' page for the 'todoacrk8s' container registry. The left sidebar shows navigation options like Overview, Activity log, and Settings. Under Settings, 'Access keys' is selected. The main pane displays two sets of access keys:

Name	Password	Action
password	8gQ78XMFQjE7oPG9LUb5fquA+CZZGJxWVlGvH/4+A...	Regenerate
password2	taqy8acBjs/96aCF9osJr4Xfm2zYGEwNlojNjCG/l+ACRDOj...	Regenerate

2. Create K8s Cluster:

The screenshot shows the 'Create Kubernetes cluster' wizard on the Azure portal. The 'Basics' tab is selected. The 'Resource group' dropdown is set to 'ToDoCIDk8s'. The 'Cluster preset configuration' dropdown is set to 'Dev/Test'. The 'Kubernetes cluster name' input field contains 'toDocidk8scluster'. The 'Region' dropdown is set to '(Asia Pacific) Central India'. The 'Availability zones' dropdown is set to 'None'. The 'AKS pricing tier' dropdown is set to 'Free'. At the bottom, there are 'Previous', 'Next', and 'Review + create' buttons.

Create Kubernetes cluster

Basics Node pools Networking Integrations Monitoring Advanced Tags Review + create

Resource group * Create new

Cluster details

Cluster preset configuration *

To quickly customize your Kubernetes cluster, choose one of the preset configurations above. You can modify these configurations at any time.
[Compare presets](#)

Kubernetes cluster name *

Region *

Availability zones

AKS pricing tier

Previous Next Review + create Give feedback

The screenshot shows the 'Create Kubernetes cluster' wizard on the Azure portal. The 'Authentication and Authorization' section is expanded. It shows 'Local accounts with Kubernetes RBAC' selected. A note says: 'Once the cluster is deployed, use the Kubernetes CLI to manage RBAC configurations.' Below this is a 'Learn more' link. At the bottom, there are 'Previous', 'Next', and 'Review + create' buttons.

Create Kubernetes cluster

Basics Node pools Networking Integrations Monitoring Advanced Tags Review + create

Kubernetes cluster name *

Region *

Availability zones

AKS pricing tier

Kubernetes version *

Automatic upgrade

Choose between local accounts or Azure AD for authentication and Azure RBAC or Kubernetes RBAC for your authorization needs.

Authentication and Authorization

Once the cluster is deployed, use the Kubernetes CLI to manage RBAC configurations. [Learn more ↗](#)

Previous Next Review + create Give feedback

Screenshot of the Azure portal showing the 'Create Kubernetes cluster' wizard. The 'Node pools' tab is selected. A table lists one node pool named 'agentpool' with a mode of 'System', OS SKU of 'Standard_DS2_v2', and a node count of '1 - 5'. The 'Enable virtual nodes' section has a checked checkbox. The 'Node pool OS disk encryption' section is collapsed. At the bottom, there are 'Previous', 'Next', and 'Review + create' buttons.

Screenshot of the Azure portal showing the 'Create Kubernetes cluster' wizard. The 'Networking' tab is selected. Fields include: 'Virtual network' set to '(New) toDoCIk8s-vnet', 'Cluster subnet' set to '(New) default (10.224.0.0/16)', 'Virtual nodes subnet' set to '(New) virtual-node-aci (10.239.0.0/16)', 'Kubernetes service address range' set to '10.0.0.0/16', 'Kubernetes DNS service IP address' set to '10.0.0.10', and 'DNS name prefix' set to 'toDocidk8scluster-dns'. Under 'Network policy', 'None' is selected. At the bottom, there are 'Previous', 'Next', and 'Review + create' buttons.

portal.azure.com/#create/Microsoft.AKS

Microsoft Azure

Create Kubernetes cluster

Basics Node pools Networking Integrations Monitoring Advanced Tags Review + create

Microsoft Defender for Cloud
Microsoft Defender for Cloud provides unified security management and advanced threat protection across hybrid cloud workloads. [Learn more ↗](#)

Your subscription is protected by Microsoft Defender for Cloud basic plan.

Azure Container Registry
Connect your cluster to an Azure Container Registry to enable seamless deployments from a private image registry. [Learn more ↗](#)

Container registry: todoacrk8s [Create new](#)

Azure Policy
Apply at-scale enforcements and safeguards for AKS clusters in a centralized, consistent manner through Azure Policy. [Learn more ↗](#)

Azure Policy: Enabled Disabled
Azure policy is recommended for dev/test configuration.

Previous Next Review + create Give feedback

portal.azure.com/#create/Microsoft.AKS

Microsoft Azure

Create Kubernetes cluster

Basics Node pools Networking Integrations Monitoring Advanced Tags Review + create

Grafana
Selecting a fully managed instance of Grafana to visualize your managed Prometheus data stored in your Azure Monitor workspace. [Learn more about pricing ↗](#)

Enable Grafana

Managed Prometheus
Managed Prometheus provides a highly available, scalable, and secure metrics platform to monitor your containerized workloads. [Learn more ↗](#)

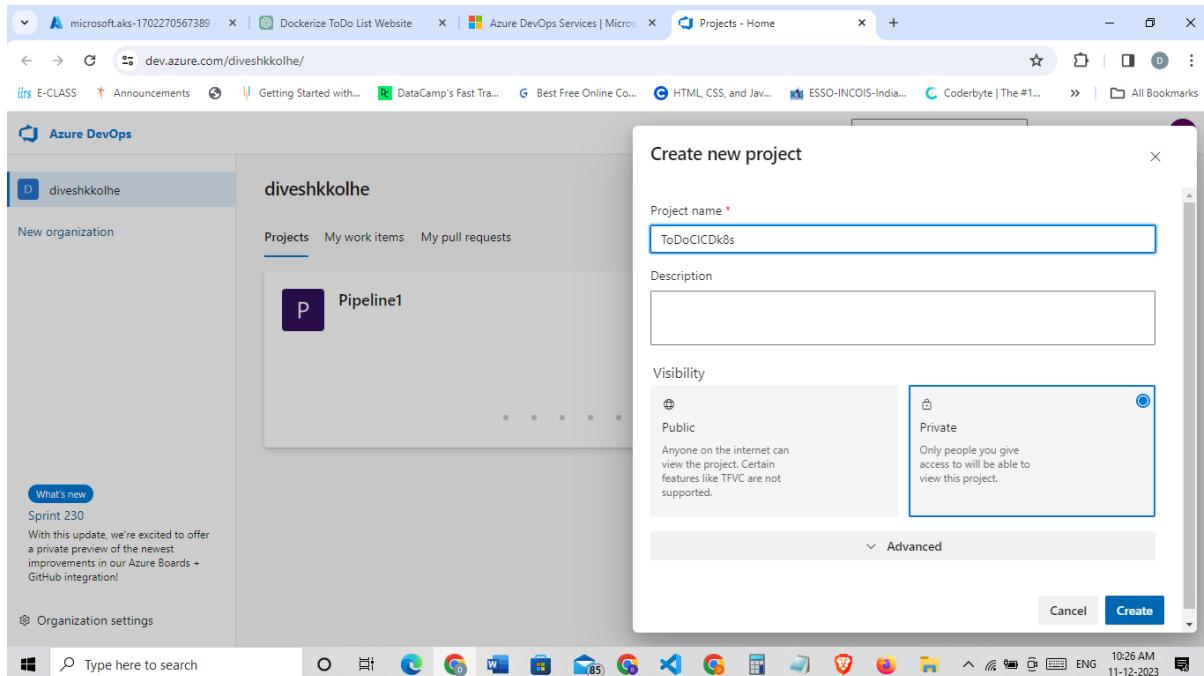
Enable Prometheus metrics

Alerts
To Monitor events on your cluster, enable alerts recommended for your monitoring configuration above (you can select and customize the rules) [Learn more about recommended alert rules ↗](#)

Enable recommended alert rules

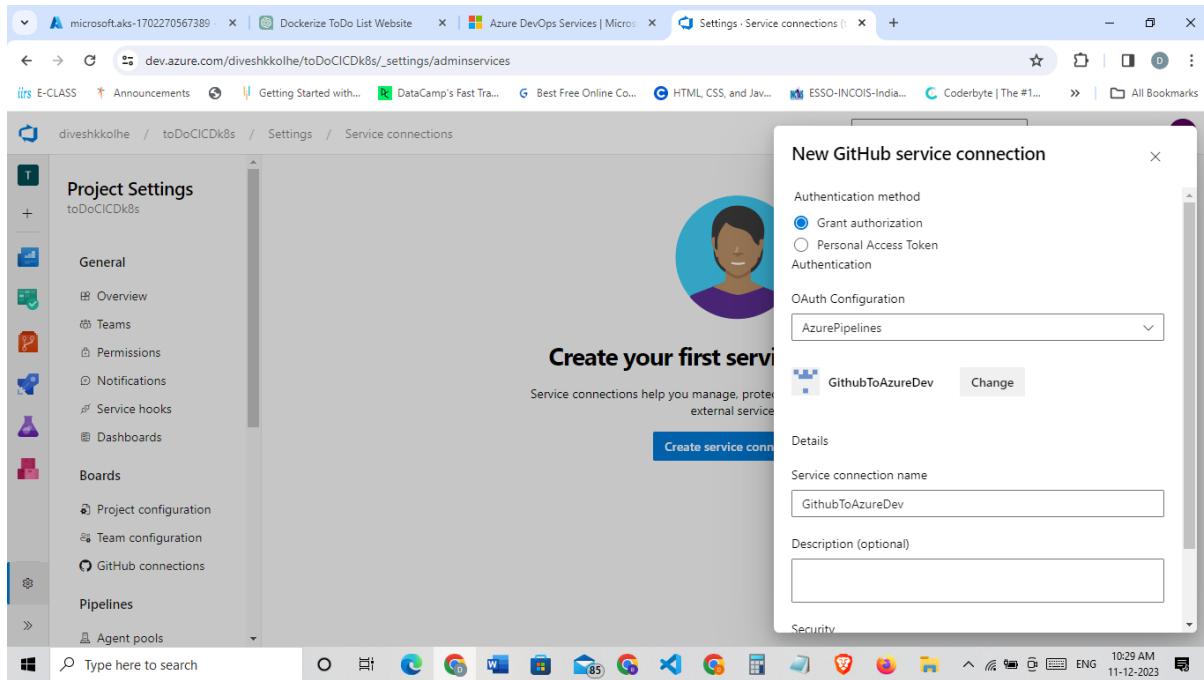
Previous Next Review + create Give feedback

3. Create the pipeline on Devops portal.



Create service connections

Github:-



The screenshot shows the 'Service connections' page in the Azure DevOps portal. A modal window titled 'New GitHub service connection' is open. In the OAuth Configuration dropdown, 'AzurePipelines' is selected. The service connection name is 'GithubToAzureDev'. There is an optional description field and a checked checkbox for granting access permission to all pipelines. The 'Save' button is visible at the bottom right of the modal.

AzurePortal:-

The screenshot shows the 'Service connections' page in the Azure Portal. A modal window titled 'New Azure service connection' is open, specifically for 'Azure Resource Manager'. Under 'Authentication method', the 'Service principal (automatic)' option is selected. Other options like 'Workload Identity federation (automatic)', 'Workload Identity federation (manual)', 'Service principal (manual)', 'Managed identity', and 'Publish Profile' are also listed. A link 'Need help choosing a connection type?' is present at the bottom of the modal, along with 'Back' and 'Next' buttons.

The screenshot shows the 'Service connections' configuration page in the Azure DevOps interface. The left sidebar lists project settings like General, Boards, Pipelines, and Agent pools. The main area displays a list of existing service connections, including 'github.com_SKstudiesTOAzureDevops'. A modal window titled 'New Azure service connection' is open, showing options for authentication methods: Workload Identity federation (automatic) (Recommended), Workload Identity federation (manual), Service principal (automatic) (selected), Service principal (manual), Managed identity, and Publish Profile. Below the modal is a link 'Need help choosing a connection type?' and buttons for 'Back' and 'Next'.

The screenshot shows the 'Service connections' configuration page in the Azure DevOps interface for a project named 'toDoCI_CDk8s'. The left sidebar lists project settings. The main area displays a list of existing service connections, including 'GithubToAzureDev'. A modal window titled 'New Azure service connection' is open, showing configuration details: Scope level (Subscription selected), Subscription (Azure for Students), Resource group (toDoCI_CDk8s), Service connection name (AzureToDwvops), and a Description field (empty). The Security section includes a link 'Grant access permission to all pipelines'. The status bar at the bottom indicates the date and time as 11-12-2023 10:31 AM.

Create Pipeline

The screenshot shows the Azure DevOps Pipelines interface. On the left, a sidebar menu for the project 'ToDoCIk8s' is visible, with 'Pipelines' selected. The main area features a cartoon illustration of a robot and a person working on a laptop. Below the illustration, the text 'Create your first Pipeline' is displayed, followed by the sub-instruction 'Automate your build and release processes using our wizard, and go from code to cloud-hosted within minutes.' A prominent blue 'Create Pipeline' button is centered at the bottom of this section.

This screenshot shows the 'New pipeline - Pipelines' wizard in progress. The top navigation bar indicates the current step is 'Connect'. The main content area is titled 'Where is your code?' and lists four options for connecting code repositories:

- Azure Repos Git (YAML) - Free private Git repositories, pull requests, and code search
- Bitbucket Cloud (YAML) - Hosted by Atlassian
- GitHub (YAML) - Home to the world's largest community of developers
- GitHub Enterprise Server (YAML) - The self-hosted version of GitHub Enterprise

The sidebar on the left remains the same as the previous screenshot, showing the 'ToDoCIk8s' project structure.

Screenshot of the Azure DevOps Pipelines "Select a repository" step.

The pipeline configuration steps are: Connect, Select, Configure, Review. The "Select" step is active.

The "Select a repository" interface shows a list of repositories under "My repositories":

- SKstudies/to-do-simple (38m ago)
- SKstudies/TestTask
- SKstudies/testblog (Nov 1)
- SKstudies/Blogs (Oct 31)
- SKstudies/Tetris (private, Oct 25)

Bottom navigation bar includes: Home, Boards, Repos, Pipelines, Environments, Releases, Library, Task groups, Deployment groups, Project settings, and a search bar.

Screenshot of the Azure DevOps Pipelines "Configure your pipeline" step.

The pipeline configuration steps are: Connect, Select, Configure, Review. The "Configure" step is active.

A warning message is displayed: "⚠️ You selected a public repository, but this is not a public project. Go to [project settings](#) to change the visibility of the project. [Learn more](#)"

The "Configure your pipeline" interface shows a list of available tasks:

- Docker (Build a Docker image)
- Docker (Build and push an image to Azure Container Registry)
- Deploy to Azure Kubernetes Service (Build and push image to Azure Container Registry; Deploy to Azure Kubernetes Service)
- Node.js (Build a general Node.js project with npm)
- Node.js Express Web App to Linux on Azure (Build a Node.js Express app and deploy it to Azure as a Linux web app)
- Node.js with Vue (Build a Node.js project that uses Vue)

Bottom navigation bar includes: Home, Boards, Repos, Pipelines, Environments, Releases, Library, Task groups, Deployment groups, Project settings, and a search bar.

The screenshot shows the Azure DevOps Pipelines interface. On the left, the sidebar has 'Pipelines' selected. The main area shows a 'Configure' step for a new pipeline named 'New pipeline'. A modal window titled 'Deploy to Azure Kubernetes Service' is open, prompting for cluster selection ('toDocidk8scluster'), namespace ('default'), container registry ('todoacrk8s'), image name ('skstudiestodosimple'), and service port ('3000'). Below the modal, a list of pipeline steps is visible, including Docker, Deploy to Azure Kubernetes Service, Node.js, Node.js Express Web App to Linux on Azure, and Node.js with Vue.

Connect cloud shell and deploy secrectes.yml

The screenshot shows the Microsoft Azure portal. On the left, the 'Kubernetes services' blade is open, showing the 'toDocidk8scluster' service with details like resource group, status, location, subscription, and tags. A 'Connect' button is highlighted. To the right, a 'Connect to toDocidk8scluster' panel is displayed, containing commands to set the cluster subscription ('az account set --subscription b858338e-efa7-486d-aebd-c1573aab0e40') and download cluster credentials ('az aks get-credentials --resource-group toDocIDk8s --name toDocidk8scluster'). Below this is a 'Cloud Shell' terminal window titled 'Bash'. The terminal shows the user has successfully connected to the cluster and is now in the Azure Cloud Shell environment, ready to use the Azure CLI.

Connect to toDocidk8scluster

Set the cluster subscription
az account set --subscription b858338e-efa7-486d-aebd-c1573aab0e40

Download cluster credentials
az aks get-credentials --resource-group toDocidk8s --name toDocidk8scluster

Sample commands

```
2 kind: Secret
3 metadata:
4   name: to-do-app-secrets
5   type: Opaque
6 data:
7   MONGODB_URI: bm9uZ29kYjzcnY6Ly9icmF22XVu25vd24xIjM6H0J5VkhNUjFOSszME10UEBjbhVzdGVyMC4ZnBocGNLm1vbmdvZGJubmV0L3RvZG9saXN0REI=
8
9
```

divesh [~ \$] code secrets.yaml

Connect to toDocidk8scluster

Set the cluster subscription
az account set --subscription b858338e-efa7-486d-aebd-c1573aab0e40

Download cluster credentials
az aks get-credentials --resource-group toDocidk8s --name toDocidk8scluster

Sample commands

```
divesh [ ~ $] az account set --subscription b858338e-efa7-486d-aebd-c1573aab0e40
divesh [ ~ $] az aks get-credentials --resource-group toDocidk8s --name toDocidk8scluster
Merged "toDocidk8scluster" as current context in /home/divesh/.kube/config
divesh [ ~ $] code secrets.yaml
divesh [ ~ $] kubectl apply -f secrets.yaml
secret/to-do-app-secrets created
divesh [ ~ $] kubectl get secrets mysecrets
Error from server (NotFound): secrets "mysecrets" not found
divesh [ ~ $] kubectl get secrets
NAME          TYPE        DATA  AGE
azdev-sa-227413-secret  kubernetes.io/service-account-token  3      11m
to-do-app-secrets    Opaque       1      12s
```

Save and run the pipeline

The screenshot shows the Azure DevOps Pipelines interface for a project named 'ToDoCIk8s'. The left sidebar is open, showing options like Overview, Boards, Repos, Pipelines, Environments, Releases, Library, Task groups, Deployment groups, and Project settings. The main area is titled 'Review your pipeline YAML' and displays the following YAML code:

```
50
51   jobs:
52     - deployment: Deploy
53       displayName: Deploy
54       pool:
55         vmImage: $(vmImageName)
56         environment: 'SKstudiestodosimple.default'
57       strategy:
58         runOnce:
59           deploy:
60             steps:
61               - task: KubernetesManifest@0
62                 displayName: Create imagePullSecret
63                 inputs:
64                   action: createSecret
```

Below the code, there are tabs for 'Variables' and 'Save and run'. A status bar at the bottom right indicates the time as 10:49 AM and the date as 11-12-2023.

The screenshot shows the same Azure DevOps Pipelines interface as the previous one, but with a modal dialog box titled 'Save and run' overlaid. The dialog contains the following message: 'Saving will commit azure-pipelines.yml to the repository.' Below this is a 'Commit message' field containing 'Set up CI with Azure Pipelines'. There is also an 'Optional extended description' field with a placeholder 'Add an optional description...'. At the bottom of the dialog, there are two radio buttons: 'Commit directly to the main branch' (which is selected) and 'Create a new branch for this commit'. A 'Save and run' button is located at the bottom right of the dialog. The status bar at the bottom right shows 10:50 AM and 11-12-2023.

Search the ip

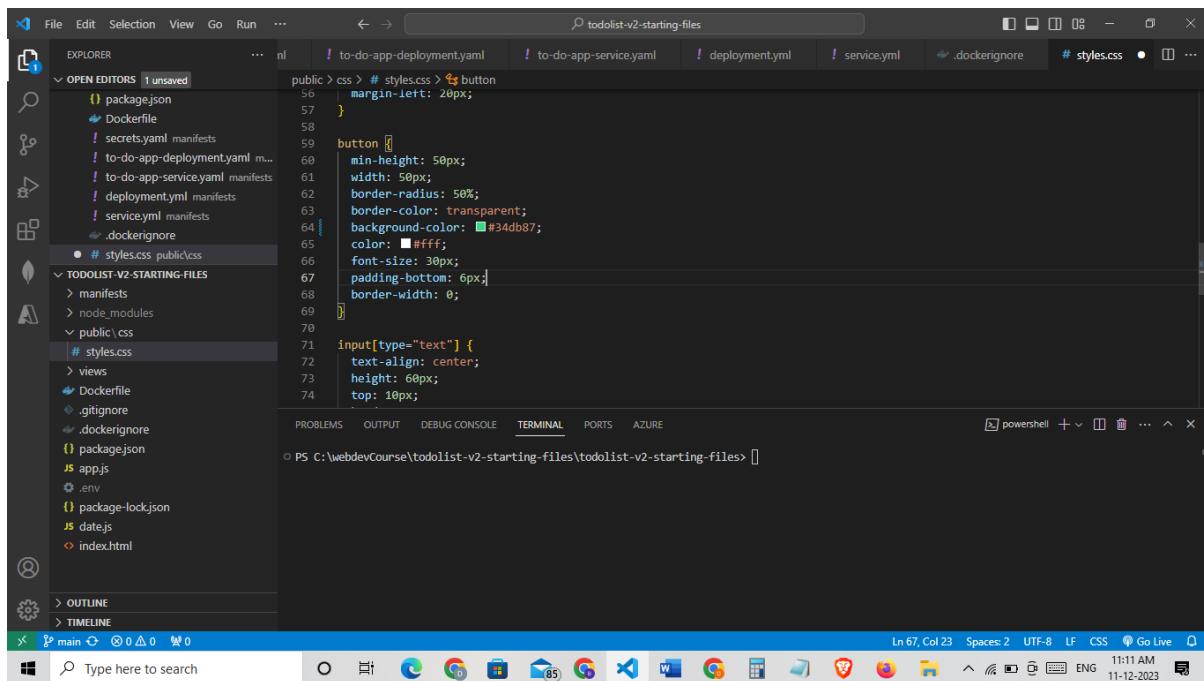
The screenshot shows the Azure DevOps Pipelines interface. On the left, there's a sidebar with options like Overview, Boards, Repos, Pipelines, Environments, Releases, Library, Task groups, Deployment groups, and Project settings. The Pipelines option is selected. In the main area, it says "Jobs in run #20231211.3 SKstudies.to-do-simple". Below that, it lists the stages: Build stage (Build, 35s) and Deploy stage (Initialize job, <1s; Download Artifact, 4s; Create imagePullSecret, 2s; Deploy to Kubernetes cluster, 6s; Finalize Job, <1s). The "Deploy to Kubernetes cluster" step is expanded, showing a YAML configuration snippet. A specific line in the log highlights the IP address: "ip": "20.219.241.34".

The screenshot shows a web browser window with the URL "Not secure 20.219.241.34/home". The page title is "Home". It displays a list of items in a table format:

<input type="checkbox"/>	hello
<input type="checkbox"/>	hello2
<input type="checkbox"/>	hello3
<input type="checkbox"/>	home

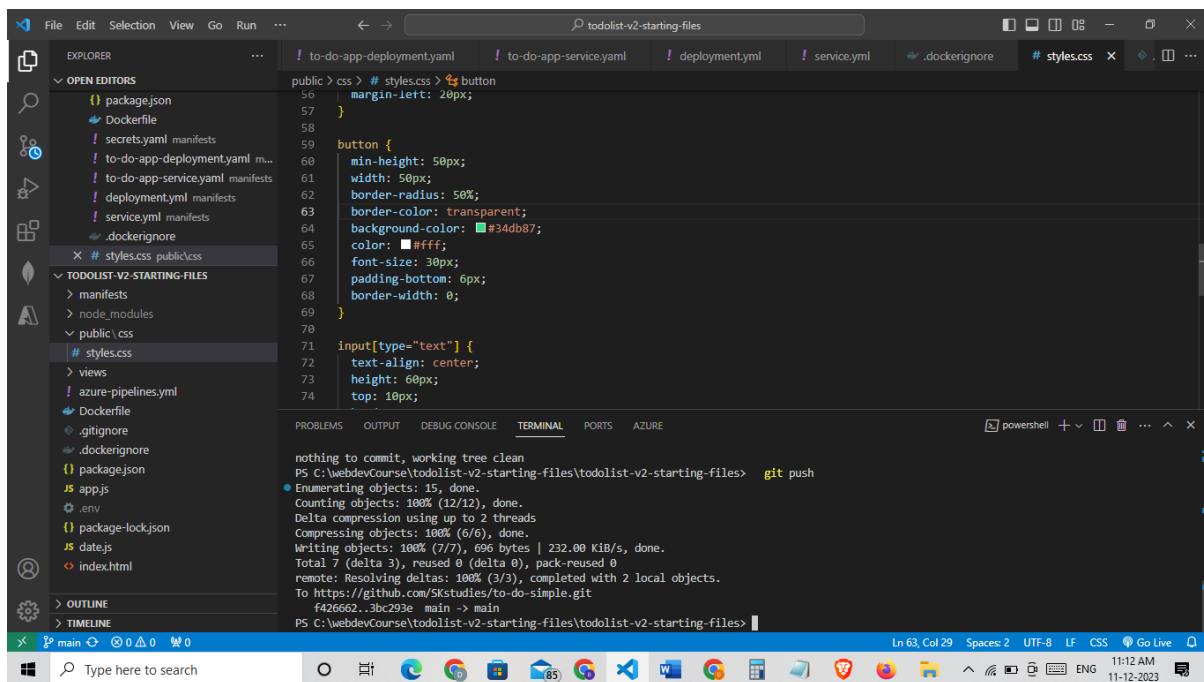
At the bottom of the list is a "New Item" button with a plus sign. The footer of the page says "Divesh • Built with NodeJs and MongoDB".

verify the cicd



```
public > # styles.css > button
56 margin-left: 20px;
57 }
58
59 button {
60 min-height: 50px;
61 width: 50px;
62 border-radius: 50%;
63 border-color: transparent;
64 background-color: #34db87;
65 color: #fff;
66 font-size: 30px;
67 padding-bottom: 6px;
68 border-width: 0;
69 }
70
71 input[type="text"] {
72 text-align: center;
73 height: 60px;
74 top: 10px;
```

Changed the colour



```
nothing to commit, working tree clean
PS C:\webdevCourse\todolist-v2-starting-files\todolist-v2-starting-files> git push
● Enumerating objects: 15, done.
Counting objects: 100% (12/12), done.
Delta compression using up to 2 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (7/7), 696 bytes | 232.00 Kib/s, done.
Total 7 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), completed with 2 local objects.
To https://github.com/SKstudies/to-do-simple.git
 f426662..3bc293e main -> main
PS C:\webdevCourse\todolist-v2-starting-files\todolist-v2-starting-files>
```

Pipeline automatically triggered

The screenshot shows the Azure DevOps Pipelines interface. On the left, a sidebar lists project navigation options like Overview, Boards, Repos, Pipelines, Environments, Releases, Library, Task groups, Deployment groups, and Project settings. The 'Pipelines' section is currently selected. In the main area, the title 'SKstudies.to-do-simple' is displayed above a table of recent pipeline runs. The table has columns for 'Description', 'Stages', and two timestamp columns. The first run, '#20231211.4', is shown as completed with a green checkmark, while the others are still in progress (indicated by a circle with a dot). A 'Run pipeline' button is located at the top right of the main area.

Wait for build to finish

This screenshot displays the details of a specific pipeline run, '#20231211.4'. The interface is similar to the previous one, with the 'Pipelines' section selected in the sidebar. The main area shows the run's ID and a detailed view of the 'Jobs in run #20231211.4'. Each job is listed with its name, status (green checkmark), duration, and a small icon. Below this, a 'Deploy' stage is expanded, showing its individual steps and their statuses. To the right of the job list, there is a large panel titled 'Deploy' containing the raw log output. The log shows the process of acquiring an agent from the cloud, starting the job, and performing various tasks like building and pushing artifacts. The log ends with the deployment to Kubernetes. The bottom of the screen shows the Windows taskbar with various pinned icons.

Button colour changed automatically

