A screenshot of a blue screen

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A diagram of a software

AI-generated content may be incorrect.

1. Clone the **backend Project in your local system through either VS code or Visual studio**

**Backend Clone URL**: <https://akspks880187@dev.azure.com/akspks880187/QuickCart/_git/QuickCart>

* Go to file in Visual Studio 2022 -> Open -> folder -> QuickKart(backend)

2. Clone the frontend **Project in your local system through either VS code or Visual studio**

**Frontend Clone URL:** <https://akspks880187@dev.azure.com/akspks880187/QuickCart/_git/Quick-Cart-FrontEnd>

1. Create Azure Storage Account (To store images for frontend application)

* Create storage account.
* Go to Container -> create one container (you can give any name). -> Change the access level to anonymous so that it can accessible –(Storage account -> overview -> right window properties -> Blob anonymous access -> Allow Blob anonymous access -> Enabled -> save.
* Go to Container -> Select container -> Change access level -> Select container(third option).

> go inside and upload the images.

Created Storage Account

A screenshot of a computer

AI-generated content may be incorrect.

Created a Container with Blob Storage access.

A screenshot of a computer

AI-generated content may be incorrect.

Uploaded images in container

A screenshot of a computer

AI-generated content may be incorrect.

 Backend

Configuring Server connection string

A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

"DBConnectionString": "Data Source=sqldatabsebatch36.database.windows.net;Initial Catalog=sqldatabsebatch36;user id=batch36; password=batch@36123"

Using dotnet run

A screenshot of a computer program

AI-generated content may be incorrect.

Launching the web

A screen shot of a computer

AI-generated content may be incorrect.

Launching Front End

Using ng serve

A screenshot of a computer

AI-generated content may be incorrect.

**TASK-2:  IMPLEMENTING CONTINUOUS INTEGRATION FOR BACKEND**

1. Learn to build CI Pipeline in Azure Devops using YAML
2. Create a **new Agent Pool** and **Self-Hosted Agent** in Azure Devops in VM
3. PAT Bh4B3OeDl8COaMAm4w92XxtEJvm68vbF00eZ667TC8YdQC0pHg1EJQQJ99BFACAAAAA8jZvlAAASAZDO2HBK A screenshot of a computer

   AI-generated content may be incorrect.A screenshot of a computer

   AI-generated content may be incorrect.A screenshot of a computer

   AI-generated content may be incorrect.A screenshot of a computer

   AI-generated content may be incorrect.

CI using Self Hosted Agent

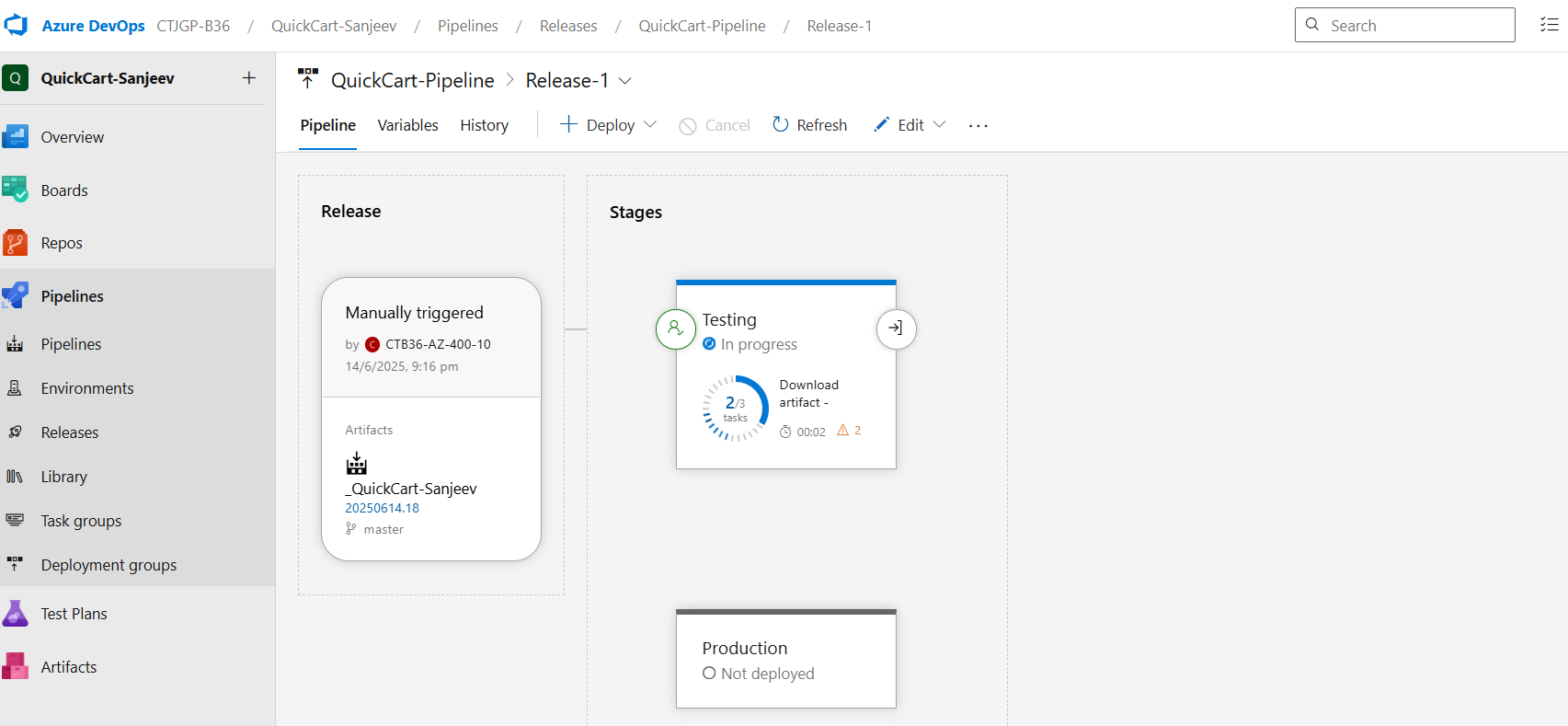
A screenshot of a computer

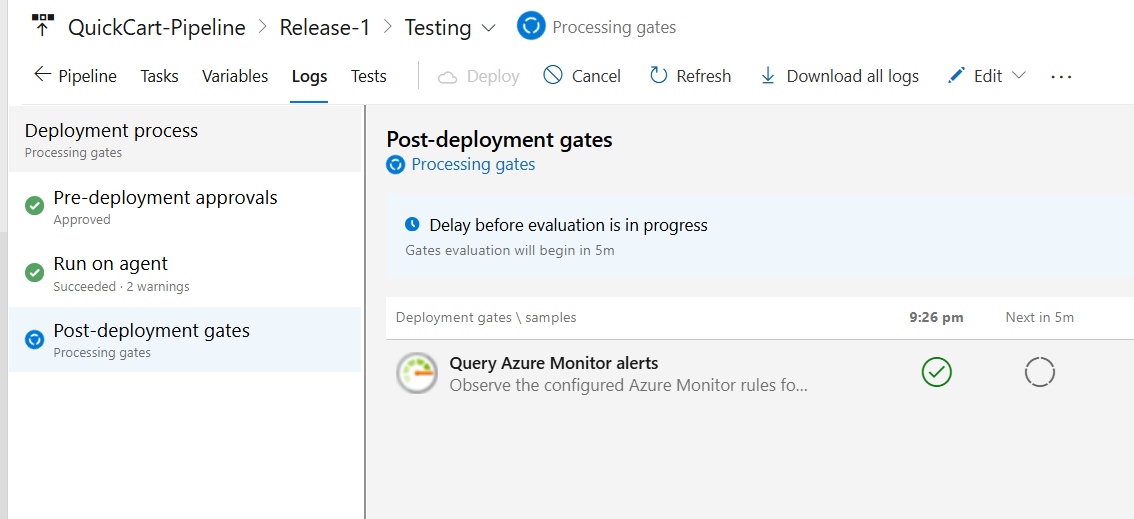
AI-generated content may be incorrect.

Created CI and CD PipelinesA screenshot of a computer

AI-generated content may be incorrect.

Created Release Gates



Monitoring Alert of Gates

Gates RunningA screenshot of a computer

AI-generated content may be incorrect.

SuccessA screenshot of a computer

AI-generated content may be incorrect.

Alerts CreatedA screenshot of a computer

AI-generated content may be incorrect.

FRONTEND using Devops

Imported Repository and made CI pipeline

A screenshot of a computer

AI-generated content may be incorrect.

Created a Static Web App

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Output of static web app

A screenshot of a computer

AI-generated content may be incorrect.

"DBConnectionString": "Data Source=sqldatabsebatch36.database.windows.net;Initial Catalog=sqldatabsebatch36;user id=batch36; password=batch@36123"

**STORING DB CREDS IN KEY VAULT AND LINNK WITH PROJECT**

1. Create a new Keyvault  
  Fill in details in Basic Tab  
  In Access Configurations --> Select Vault Access Policy --> Select Your Account --> Create Keyvault account

2. Add secrets in Keyvault. --> Add as Name DBConnectionString --> Add URL as vaule --> Create  
Name: "DBConnectionString"  
URL: "DBConnectionString": "Data Source=quicksqlserver.database.windows.net;Initial Catalog=quickdb ;user id=sqladmin; password=Quicksql@12"

3. Now to integrate this with Azure pipeline, go to Azure DevOps --> Azure Pipeline --> Library

4. Create a New Variable Group

5. Give name --> Enable "Link Secrets from Azure KeyVault

6. Authenticate the subscription and keyvault here

7. If not connecting --> Go to Keyvault --> Go to Access Policy --> Select all the permissions

8. Once variable group is created, reference it in release pipeline.

9. Edit release pipeline --> Go to variables --> Go to Variable Groups --> Click on "Link Variable Group" --> Select Scope --> Link

10. Goto Task in the same release pipeline --> Add "Azure App Service Settings" --> This task should be first task in that stage.

11. Fill in the necessery details --> Scroll down --> In connection string section --> Add keyvault details  
[  
   {  
    "name": "kvdemo1",  
    "value": "$(DBConnectionString)"  
   }  
]

12. Keyvault name is Variable Group name. Value to be referrences with $ symbol. --> Save and RUn

FUNTIONAPP

<https://quickcart-function.azurewebsites.net/dB4RvSf0Gka8CkWl8kl64oNzjQQQKLlNZk4cVK28mNyTAzFuS0Do_A==>

1. Clone Login Functionality repo from the given link  
<https://akspks880187@dev.azure.com/akspks880187/QuickCart/_git/LoginService.git>

2. Open LoginFunctionApp.sln in Visual Studio 2022

3. Open Solution Explorer

4. Go to portal.azure.come and create a Azure Function in the same resource group as your QuickKart application  
  Runtime stack   --> .Net  
  Version   --> 8 (LTS)  
  Rest all Default Options  
  Review & Create

5. Go to Visual Studio  2022  --> Right click on the project from Solution Explorer  
Click on Publish

6. First time if you are doing, then create a + New profile

7. Select Target --> Azure

8. Select Specific Target --> Azure Function App (Windows)

9. Check the account with which the account is set up.   
  Email ID Should match with your DevOps portal account  
  If not, re-enter your credentials

10. Once credentials are entered correctly you will see your subscription and also the Resource Group.  
  Select resource group and select your Azure Function App.  
  Deployment Type --> Publish

  Profile will be created now.

11. Download and install Azure Functions Runtime  
<https://learn.microsoft.com/en-us/azure/azure-functions/functions-run-local?tabs=windows%2Cisolated-process%2Cnode-v4%2Cpython-v2%2Chttp-trigger%2Ccontainer-apps&pivots=programming-language-csharp>

12. Click on Publish

13. Go to Azure Portal with Function App. Scroll down and see LoginFunction getting deployed.

14. Click on LoginFunction --> Click on Get Function URL --> Copy the Link

15. Make changes in the home-page.service.ts in Azure Repos

16. In the validateUser function, add your Azure Function link (Copied from portal)

17. Enable CORS policy for Azure Functions  
  Copy static web app URL.  
  Go inside the LoginFunction App  
  Search for CORS  
  Add your static web app link  
  Add \* in next text box  
  Save

18. Will take upto 10 min to reflect this functionality.

19. See to it that pipeline is running. Once the pipeline finishes, then login.

20. You should be able to login now.

Created Az function

A screenshot of a computer

AI-generated content may be incorrect.

Publishing from VS 2022

A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.