# Prerequisites

1. Create azure subscription
2. Create azure devops organization
3. Read information about github flow branching strategy
4. terraform should be installed
5. Terraform knowledge is also required to do the stuff
6. Az cli should be installed

# Homework

## Part 1 – Configure application

1. Create a service connection in a Azure DevOps project to your subscription - <https://learn.microsoft.com/en-us/azure/devops/pipelines/library/service-endpoints?view=azure-devops&tabs=yaml>
2. Find a .net pet project for the experiments
3. Build your app locally .net project via dotnet tool. dotnet restore/build/run
4. Create an Azure DevOps repo - <https://learn.microsoft.com/en-us/azure/devops/repos/git/create-new-repo?view=azure-devops> You can use import repository to import from existing source control version like github
5. Create a branching policy for you application. Added yourself as a reviewer - <https://learn.microsoft.com/en-us/azure/devops/repos/git/branch-policies?view=azure-devops&tabs=browser> As branching strategy use a github flow (It will be applied by default when you strict commit to your main branch)

## Part 2 – Configure a pipeline to deploy infrastructure

Below is describing on how to do it via terraform. If you want to use terraform you need to create service connection in manual way. Otherwise you won’t be able to deploy your iac – Navigate to the last section

### Terraform storage account

1. Create a separate resource group and deploy azure storage account - <https://learn.microsoft.com/en-us/azure/storage/common/storage-account-create?tabs=azure-portal>
2. Create a container with the name “tfstate” and remember the name. use portal settings Graphical user interface, application

   Description automatically generated Graphical user interface, text, application

   Description automatically generated

In this storage account you will be store your tf state file

### Terraform preparation

1. Create another repo to store devops code
2. Create a folder terraform
3. Add app service implementation - <https://learn.microsoft.com/en-us/azure/app-service/provision-resource-terraform>
4. Integrate application insights with app service
5. Updated backend “azurerm” according to the guide - <https://learn.microsoft.com/en-us/azure/developer/terraform/store-state-in-azure-storage?tabs=azure-cli> Graphical user interface, application, Word

   Description automatically generated
6. Run az login or Connect-AzAccount to connect the azure subscription from your local
7. Run terraform apply to deploy infrastructure

**Important note: Use only freshest version of tf module like** <https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/resources/windows_web_app>

**Important note: Don’t forget to destroy your application once completed**

### Create a terraform pipeline

1. Create a yaml pipeline with the following steps: terraform install, terraform init, terraform plan/apply. Plan is an optional one
2. Inside yaml pipeline add trigger to main branch. The scenario – when main is updated, pipeline should run automatically - <https://learn.microsoft.com/en-us/azure/devops/pipelines/yaml-schema/trigger?view=azure-pipelines>
3. Added 3 steps – terraform install, terraform init, terraform plan/apply. Plan is an optional one. You may add it as 4th step

## Part 3 – Create a deploy pipeline to app service

1. Add yml pipeline to the application folder
2. Your pipeline structure should contain 2 stages. 1st – build, create zip archieve, and publish an artifact. 2nd – download an artifact and deploy it to azure app service
3. To deploy .zip to app service use azure app service deployment task

# Service connection – manual way

<https://4bes.nl/2019/07/11/step-by-step-manually-create-an-azure-devops-service-connection-to-azure/>

Don’t forget to grant access on the subscription level for your enterprise application (service principal)

# Useful readings

1. How to share variables A picture containing graphical user interface

   Description automatically generated
2. Templates example for variables - <https://learn.microsoft.com/en-us/samples/azure-samples/azure-pipelines-variable-templates/azure-pipelines-variable-templates/>
3. Good example how to do a pipeline to build .net app and deplot tf iac - <https://azuredevopslabs.com/labs/vstsextend/terraform/> Only via UI. Hence don’t forget about view yaml button in UI
4. Example of the Angular application from lecture 2 - <https://epam-my.sharepoint.com/:u:/p/yevhen_husiev/EWXdflfwT7pBijqGNXZnvRgBRdpB_EXlN0cJy8_SFA6_eA?e=Fc3LQW> password – AQ!sw2DE£fr4