Service Fabric Hackathon

October 4th-5th in Bengaluru, 8th – 9th in Mumbai

# Overview

# Azure Service Fabric is a distributed systems platform that makes it easy to package, deploy, and manage scalable and reliable microservices and containers. Service Fabric also addresses the significant challenges in developing and managing cloud-native applications.

# Service Fabric Hackathon is focused on ‘Lift and Shift’ developer scenario to move a legacy application by containerizing it into Azure by using Service fabric Containers platform. The challenges are geared towards teaching you the basic/real-world features of Service fabric. The initial challenges start easy and help a new person understand concepts of containers and Service fabric and later challenges lets developer build expertise around powerful features of Service fabric.

# Event Structure

This is a team-based event. Each team consisting of 3 team members will storm through challenges. The challenges are defined but the solutions are not. You will have to explore Service fabric docs and internet to define and implement solutions.

We have planned the following structure for you

## Day1

Talk from Speakers

Setup of your laptop environment if you haven’t finished pre requisites

Ready Set Go! Start your challenges

LUNCH

Chale Chalo! Keep storming through your challenges

Speaker present on topics covered in first few challenges

Q & A - Ask Anything except food 😊

Disable Azure cluster resources before heading home

## Day 2

Pick a new challenge and breeze through it

If not breezing, sweat through it!

LUNCH

Free Hand- Create new website/services and deploy to Service fabric

Presentation on technical concepts covered

Q&A

Disable Azure cluster resources before heading home

Closing ceremony – With awards and complements

# Feedback

Feedback is super important to us. We need to know what’s working well and what isn’t. We will run a short survey at the end of the event to get your thoughts. However, you can grab a proctor at any time and ask them to pass on feedback directly to us.

If you have feedback about documentation, please comment, raise an issue or open a pull-request against the offending document. Our open documentation is all backed by GitHub so it should be super easy to do!

If you have issues with Service Fabric, its tooling, or just a great idea on how to improve the product, please submit in the survey or create an issue directly on the Service Fabric issue page or submit a user voice post.

If all else fails, stand on a chair and start shouting! I’m sure someone will take notice soon enough!

# Setup & Prerequisites

To ensure you have the best experience whilst completing the challenges, we have put together a list of tools you’ll need to install to get setup.

We will provide an Azure subscription with credit, but please ensure you bring your own laptop and have followed one of the "Set up your development environment" guides below:

## Required

* Service Fabric SDK, runtime and tools
  + [Windows](https://aka.ms/DevWindows)
  + [Linux](https://aka.ms/DevLinux)
  + [MacOS](https://aka.ms/DevMacOS)
* [PowerShell](https://docs.microsoft.com/en-us/powershell/) (Should be installed by default on Windows OS)
* Docker
  + [Windows](https://store.docker.com/editions/community/docker-ce-desktop-windows?tab=description)
  + [Mac](https://docs.docker.com/docker-for-mac/)
  + [Ubuntu](https://docs.docker.com/install/linux/docker-ce/ubuntu/)
* [Visual Studio 2017 15.7+](https://visualstudio.microsoft.com/downloads/)
* [Git](https://git-scm.com/downloads)

## Optional

* [.NET Core](https://www.microsoft.com/net/download/dotnet-core/2.1)
* [Azure CLI](https://docs.microsoft.com/en-us/cli/azure/install-azure-cli?view=azure-cli-latest)
* [Service Fabric CLI](https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-cli)
* [Service Fabric Mesh Tooling](https://docs.microsoft.com/en-us/azure/service-fabric-mesh/service-fabric-mesh-howto-setup-developer-environment-sdk)

# Azure Subscription

Azure Subscription – Azure pass code will be passed on to you to give you access to Azure resources for a month from activation. The subscription will give access to all Azure 1st party services which is what is needed for your challenges.

Redeem pass code by clicking this link: <https://www.microsoftazurepass.com/> and following process listed here <https://www.microsoftazurepass.com/Home/HowTo>.

# Up and Running

Make sure you have finished and installed all the Pre-requisites. Make sure the following is running on your system

1. Docker is up and running – You will see docker icon in your task bar. Click on the icon and ensure the docker is up and running.
2. Service Fabric local cluster is running - Service Fabric Local Cluster Manager icon should appear in the notification/ system tray. If not, search for “Service Fabric Local Cluster Manager” in the Windows search and launch it. Right click on the cluster icon and select “Setup Local Cluster -> 1 Node”. Wait for the notification to appear as “Local Cluster setup completed successfully”. It should take couple of minutes.
3. Right click on the Service Fabric cluster icon in tray and select “Manage Local Cluster”. This should bring up the Service Fabric Explorer in a browser. You shouldn’t see any errors on the screen and be able to browser through the nodes, system services etc. Take few minutes to go through the document below to understand the functionality of Service Fabric Explorer. This is a key app that you will be using to visualize and monitor your applications on Service Fabric cluster.

<https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-visualizing-your-cluster>

1. Open Visual Studio and verify the version is greater than 15.7.

# Challenges

Welcome to Service Fabric Challenge! The challenges are intended to be a challenge! They should help you understand the problem, the potential solutions, and give you experience trying to implement. Useful resources will be referenced to help you work through each challenge but step by step instructions will not be provided. To ensure you progress at a good pace try to make sure that your workload is divided between team members. This may mean anticipating work that might be required in a later challenge

## Lift and Shift existing .net Application as containers using service fabric

Among many scenarios where Azure Service Fabric shines, one of the latest has been the lift and shift legacy applications into Azure with minimal code changes on Service Fabric. With Windows Service 2008 and Windows Server 2005 getting out of support, there exists thousands of .net applications in want of supported environment. Many organizations are taking advantage of this situation and looking for ways to move their legacy business-critical applications into cloud with Service Fabric and why not? Azure Service Fabric gives your legacy application all the tested feature of scale, availability, reliability, and resilience in one shot with minimal code change.

The legacy application can be moved into Service Fabric by containerizing java legacy with Linux containers and .Net with Windows containers. As of now, Service Fabric is the only product that support moving .Net legacy apps with Windows containers into cloud. Through this challenge, we would like to showcase this momentum to all the partners, ISV, and customer participants in the hackathon.

Use following application in the repository for challenges 1.1 to 1.5- <https://github.com/kagowda/India-Hackathon/tree/master/eshopwebformApplication/eShopLegacyWebFormsSolution>

### Containerize legacy .Net application using Service Fabric as container orchestrator in Visual Studio

Build the solution and run the application to see an eShop web page in the browser.

In this challenge, containerize the typical Asp.net IIS application using Service Fabric. We have designed the challenge to not give any specific resources around this and find your way in Service Fabric docs. How’s that!!

##### Resources:

You can use documentation at <https://docs.microsoft.com/en-us/azure/service-fabric/>

### Build and publish to local cluster and run application

Publish the containers to local service fabric clusters and run the application. Take a point of the port you have exposed for service host in your manifest files. Use this port to access your application on local cluster. Ha! Very detailed instruction, isn’t it?

##### Resources:

<https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-host-app-in-a-container>

<https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-service-manifest-resources>

<https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-get-started-containers>

### Push Container to Azure Container Registry (ACR)

Simple challenge to create new Azure container registry and then push the above container image into registry. The previous challenges have created containers on your local machine. Find the container and push the container image to newly created ACR. Use your Azure pass subscription to create this resource.

##### Resources:

<https://docs.microsoft.com/en-us/azure/container-registry/container-registry-get-started-powershell>

<https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-get-started-containers#push-the-image-to-the-container-registry>

Docker commands- https://docs.docker.com/engine/reference/commandline/docker/

### Create Azure Service fabric cluster and access Service fabric explorer

Now let’s walk these containers to azure. First, create Service fabric cluster using Azure resource manager and deploy the containerized services to azure service fabric cluster. Successful completion will leave you with a healthy SF cluster and container instances.

Do not move out of the challenge unless you can run your eshop application in a browser.

##### Resources:

<https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-cluster-creation-via-arm>

### Upgrade the application and parametrize environment variables

In this challenge, you will enable parameterization of application configuration for different environments. We will also roll-out an upgrade to the Service Fabric application.

If you open up SiteMaster.cs, we see that it reads environment variable “eShopTitle” to set page title. Let’s put environment variable in service manifest and parameterize this at application level and specify different values for local cluster and azure cloud.

Make some code changes and upgrade the application to cluster again. Watch the upgrade process through Upgrade domains with zero downtime for your application.

##### Resources:

<https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-how-to-parameterize-configuration-files>

<https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-how-to-specify-environment-variables>

<https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-tutorial-deploy-container-app-with-cicd-vsts#update-the-application>

<https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-get-started-containers>

### Add Inter service communication with DNS

From here on – Use the following application of eshopAppWithWepApi [https://github.com/kagowda/India-Hackathon/tree/master/eshopAppWithWepApi](https://github.com/kagowda/India-Hackathon/tree/master/eshopAppWithWepApi%20) .

This application contains a webapi service along with Asp.net eshopwebform application which you have already seen in above challenges. We have added a service to showcase the inter communication among services.

The solution is already containerized for Service fabric orchestrator. Delete existing application in your Azure cluster by using Service fabric explorer.

Enable DNS for your cluster in Azure and update eshopwebform application to use dns url of webapi to access the service. Publish the application to Azure SF cluster. Access your application and make sure everything is running the same.

Now enable reverse proxy for your cluster and see the changes in accessing your service inside containers.

##### Resources:

[https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-dnsservice](https://apac01.safelinks.protection.outlook.com/?url=https%3A%2F%2Fdocs.microsoft.com%2Fen-us%2Fazure%2Fservice-fabric%2Fservice-fabric-dnsservice&data=02%7C01%7C%7C52363e50f93b438e05d708d6223e3a04%7C72f988bf86f141af91ab2d7cd011db47%7C1%7C0%7C636734047976763279&sdata=Uud6mWhb%2F2L8Du1E65dsUTDXlFIiF8nR7QXTR0eZkS4%3D&reserved=0)

https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-reverseproxy-setup

[https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-reverseproxy](https://apac01.safelinks.protection.outlook.com/?url=https%3A%2F%2Fdocs.microsoft.com%2Fen-us%2Fazure%2Fservice-fabric%2Fservice-fabric-reverseproxy&data=02%7C01%7C%7C52363e50f93b438e05d708d6223e3a04%7C72f988bf86f141af91ab2d7cd011db47%7C1%7C0%7C636734047976763279&sdata=KCp7Wrq3ZSTMUX6Wp4Skr5joeYr7fMcNRdpWB8gVxqQ%3D&reserved=0)

[https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-connect-and-communicate-with-services](https://apac01.safelinks.protection.outlook.com/?url=https%3A%2F%2Fdocs.microsoft.com%2Fen-us%2Fazure%2Fservice-fabric%2Fservice-fabric-connect-and-communicate-with-services&data=02%7C01%7C%7C52363e50f93b438e05d708d6223e3a04%7C72f988bf86f141af91ab2d7cd011db47%7C1%7C0%7C636734047976773283&sdata=M4Hn7RFuCEUCDk8AIKvRsi1R%2Bj8n0t7Sz7xUV%2BOToDQ%3D&reserved=0)

### Add monitoring, diagnostics for your containers

Running and operating a distributed system is hard. Without good instrumentation and monitoring it is impossible. To ensure you have a good understanding of how the application is performing and where potential issues may occur, you should add comprehensive monitoring to the solution.

This challenge requires you to add extensive monitoring that covers the following 3 areas:

1. Application Monitoring 2. Platform/Cluster Monitoring 3. Infrastructure/Performance Monitoring

#### Application Monitoring:

Application monitoring is essential to understanding what the application is doing and if anything has gone wrong. When things do go wrong, the telemetry sent by the application is often the first place to look! Currently the application writes logs using each services own EventSource. This harnesses the power of ETW on Windows. ETW is rich, extensible and fast. However, without something to forward those logs off the node into a central repository, they’ll be very hard to access and comprehend. This challenge requires that you add application level monitoring that allows you to see what the application is doing, how it is performing and what errors are occurring.

##### Resources:

<https://docs.microsoft.com/en-us/azure/service-fabric/servicefabric-diagnostics-overview#application-monitoring>

https://docs.microsoft.com/en-us/azure/service-fabric/servicefabric-diagnostics-event-analysis-appinsights

https://github.com/Microsoft/ApplicationInsightsaspnetcore/wiki/Getting-Started#add-the-instrumentation-key-toappsettingsjson

https://docs.microsoft.com/en-us/azure/service-fabric/servicefabric-diagnostics-event-aggregation-eventflow

https://github.com/Microsoft/ApplicationInsights-ServiceFabric

<https://blogs.msdn.microsoft.com/softwaresimian/2017/06/16/azureservice-fabric-application-insights-working-sample/>

Add monitoring for your containers on by plugging into Log Analytics in Azure cluster.

<https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-tutorial-monitoring-wincontainers>

#### Cluster monitoring

When using a distributed platform such as Service Fabric, it is important to ensure you have deep insight into how it is performing and whether you need to take any action to mitigate issues. The Service Fabric agents write logs out to each node, you need to collect all of these into a central location for triage, alerting and analysis.

Some recommended approaches are listed below.

##### Resources:

https://docs.microsoft.com/en-us/azure/service-fabric/servicefabric-diagnostics-overview#platform-cluster-monitoring

Log Analytics (OMS)

https://docs.microsoft.com/en-us/azure/service-fabric/servicefabric-diagnostics-oms-setup

<https://docs.microsoft.com/en-us/azure/service-fabric/servicefabric-diagnostics-event-analysis-oms>

#### Infrastructure Monitoring

Service Fabric runs on a set of nodes (virtual or physical machines). In Azure deployed clusters this infrastructure layer is provided by Virtual Machine Scale Sets. We want to know how these machines are performing. This information is critical to understanding whether or not we have enough capacity in the cluster to handle our workload. Extracting key metrics such as Performance Counters and OS logs will help us build a holistic picture of the state of our cluster.

This challenge requires you to enable Performance Counters and set an alert for high memory and CPU usage across the cluster.

##### Resources

Some recommended approaches are listed below.

https://docs.microsoft.com/en-us/azure/service-fabric/servicefabric-diagnostics-overview#performance-monitoring

Log Analytics

https://docs.microsoft.com/en-us/azure/service-fabric/servicefabric-diagnostics-oms-agent • https://docs.microsoft.com/en-us/azure/service-fabric/servicefabric-diagnostics-perf-wad

### Rapid Scale the containers by increasing the instance count of your containers -??

Azure Service Fabric makes it easy to build scalable applications by managing the services, partitions, and replicas on the nodes of a cluster. Running many workloads on the same hardware enables maximum resource utilization, but also provides flexibility in terms of how you choose to scale your workloads.

In this challenge, you will look at understanding manual scaling, which can also be done programmatically. But programmatical scaling is not included here, you can always explore more.

#### Service/Container Instance scaling-

Look at your container instances in your cluster. You should see that containers are installed on all of the 5 VM as instance count for containers in your manifest is set to -1.

Reduce the instance count to 1 and republish to observer the container instance changes in SF explorer. You will see only one container instance on only one node. Now increase the instance count = (number of nodes -1), ie 4 and republish. See how quick your container instances scale and start listening for new requests.

Now disable one of the nodes that hosts a container instance from the cluster. Use SF explorer to do this, right click on the node and say disable. Observe how the container instance on this node will be moved to another working node super quick.

#### Scaling with adding/removing nodes from the cluster

Begin by setting instance count to -1 for your apps/webapi and republish to Azure. Wait for your publish to finish and see your container instance on all nodes. Now manually add new VM to your cluster and observe the changes. When a new VM is successfully attached to your cluster, SF automatically create new container instance and deploy on the new VM.

##### Resources

<https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-conceptsscalability>

Overview- https://docs.microsoft.com/en-us/azure/service-fabric/service-fabriccluster-scaling Commands to Scale - https://docs.microsoft.com/en-us/azure/servicefabric/service-fabric-tutorial-scale-cluster

Scale programmatically - https://docs.microsoft.com/en-us/azure/servicefabric/service-fabric-cluster-programmatic-scaling

Auto Scaling - https://docs.microsoft.com/en-us/azure/service-fabric/servicefabric-cluster-resource-manager-autoscaling

Manual Scale out - https://docs.microsoft.com/en-us/azure/service-fabric/servicefabric-cluster-scale-up-down

### Add resource constraints for your containers

When you're running multiple services on the same node or cluster, it's possible that one service might consume more resources, starving other services in the process. This problem is referred to as the "noisy neighbor" problem. Azure Service Fabric enables the developer to specify reservations and limits per service to guarantee resources and limit resource usage.

##### Resources:

https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-resource-governance

# Choose from following exercise:

## Service fabric mesh

Azure Service Fabric Mesh is a fully managed service that enables developers to deploy microservices applications without managing virtual machines, storage, or networking. Applications hosted on Service Fabric Mesh run and scale without you worrying about the infrastructure powering it. Service Fabric Mesh consists of clusters of thousands of machines. All cluster operations are hidden from the developer. Simply upload your code and specify resources you need, availability requirements, and resource limits.

SF Mesh is in public preview as of now and is planned to go General availability next year.

### Setup your development experience

You have to setup your development box before you proceed with SF mesh. Please follow the following links to get the dev setup.

<https://docs.microsoft.com/en-us/azure/service-fabric-mesh/service-fabric-mesh-howto-setup-developer-environment-sdk>

<https://docs.microsoft.com/en-us/azure/service-fabric-mesh/service-fabric-mesh-howto-setup-cli>

Open Windows powershell and type “az-mesh”. If the command runs successfully then your local box is setup.

### Create a new .net application and deploy on Mesh

You'll learn how to use Visual Studio to create an Azure Service Fabric Mesh app that has an ASP.NET web front-end and an ASP.NET Core Web API back-end service. Then you'll debug the app on your local development cluster. You'll publish the app to Azure and then make config, code changes, and upgrade the app. Finally, you'll clean up unused Azure resources so that you are not charged for what you aren't using.

The resource link below has the direction to source code that you can use to perform the exercise.

##### Resources:

https://docs.microsoft.com/en-us/azure/service-fabric-mesh/service-fabric-mesh-tutorial-create-dotnetcore

## CI/CD Pipeline

### Create CI/CD pipeline to Azure SF cluster

##### In order for you to safely make iterative changes to the application code, it is important that you move from manually deploying updates to an automated system. This challenge requires you to create an automated build, test and deployment pipeline. Once you’ve set it up, all modifications to the application should be pushed through the pipeline and into your cluster.

##### Your CI should build and test any code changes committed or merged on to your master branch. The build should then publish the build artifacts to a staging area that can be accessed by a release management tool in CD.

##### Your CD should pick up these build artifacts and execute a release into your cluster leveraging Service Fabric’s rolling update mechanism.

##### You can apply additional items such as gated check ins, notifications and policies if you wish.

##### Resources:

<https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-tutorial-deploy-container-app-with-cicd-vsts#commit-and-push-changes-trigger-a-release>

<https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-tutorialdeploy-app-with-cicd-vsts>

https://docs.microsoft.com/en-us/vsts/accounts/create-accountmsa-or-work-student?view=vsts

https://about.gitlab.com/

## Auto scale existing application

<<Naveen>> to add the description, links and solution file.

## Reliable Collection

<<Naveen>> Will add this.

## Create new web application/Microservices and deploy to Service fabrics

This challenge is your opportunity to get hands on building a new Service Fabric service. You can write this application using whatever framework, language or tooling you wish to. If you choose to use .NET, you can write a new .net core application and deploy as guest executables or plugin to the Service Fabric Reliable Stateless service programming model. This is an open challenge, go wild!

##### Resources

https://docs.microsoft.com/en-us/azure/service-fabric/servicefabric-tutorial-create-dotnet-app https://docs.microsoft.com/en-us/azure/service-fabric/quickstartguest-app https://azure.microsoft.com/en-gb/resources/samples/servicefabric-dotnet-web-reference-app/ https://docs.microsoft.com/en-us/azure/service-fabric/servicefabric-deploy-existing-app