Create your first serverless function & workflow

Background

In this challenge you will start creating your first serverless functions and workflows, which are key skills for the upcoming challenges which make up this OpenHack.

Challenge

Your first goal is for each team member to create and debug an HTTP triggered Azure Function locally on your own development machine. This function has the following requirements:

* **Verb**: GET
* **Input parameters**: A productId value is passed to it as a query parameter or an HTTP route
* **Result**: “The product name for your product id {the id passed in to the function} is Starfruit Explosion”

The second goal is for each team member to create their own Azure Resource Group in your team’s Azure Subscription, and then deploy the function from the first goal to it. **Make sure to use different, unique Function App names!** Then test the function after deployed.

Your final goal for this challenge is for each team member to create an HTTP triggered Logic App in their Resource Group. The Logic App has the following trigger requirements:

* **Verb**: POST
* **Input payload example**:
* {
* "productId": "75542e38-563f-436f-adeb-f426f1dabb5c"

}

Next, you should configure the Logic App to call the Azure Function created earlier, passing the productId into it, then reply successfully with the response text content of **“{response from function} and the description is This starfruit ice cream is out of this world!”**.

So, for the example inputs above the Logic App would reply with **“The product name for your product id 75542e38-563f-436f-adeb-f426f1dabb5c is Starfruit Explosion and the description is This starfruit ice cream is out of this world!”**.

**Hint**: To pass query parameters from Logic Apps to a function in the Azure Functions action, you can write the name and values as JSON in the Queries field. Example:

{

"queryParamName": "queryparamValue"

}

Success Criteria

*Given the complexity and variety of potential solutions, the goals for all challenge are verified by your team’s coach.*

* Everyone on your team must demonstrate that you created an HTTP triggered Azure Function locally that accepts a GET verb. Show to your coach that you can call that function and receive a successful response with the right text provided and the value of productId properly filled in.
* Everyone on your team must demonstrate that you deployed that HTTP triggered Azure Function to Azure. Show to your coach that you can call that deployed function and receive a successful response.
* Everyone on your team must demonstrate that you can call the Logic App via HTTP POST which then calls the Function, and that it replies successfully with the correct extended text.
* Infrastructure Bonus: If your team has a member working on IaC, demonstrate the ability to deploy an Azure Function and Logic App services with code, instead of performing the tasks related to the other success critera.

References

* [An introduction to Azure Functions](https://docs.microsoft.com/azure/azure-functions/functions-overview)
* [Supported languages in Azure Functions](https://docs.microsoft.com/azure/azure-functions/supported-languages)
* [Write Azure functions in VSCode](https://docs.microsoft.com/azure/azure-functions/functions-develop-vs-code?tabs=csharp)
* [Test your project locally](https://docs.microsoft.com/azure/developer/javascript/tutorial-vscode-serverless-node-03)
* [Strategies for testing your code in Azure Functions](https://docs.microsoft.com/azure/azure-functions/functions-test-a-function" \t "_blank)
* [Code and test Azure Functions locally](https://docs.microsoft.com/azure/azure-functions/functions-develop-local" \t "_blank)
* [Create a function on Linux using a custom container](https://docs.microsoft.com/azure/azure-functions/functions-create-function-linux-custom-image" \t "_blank)
* [Logic Apps Overview](https://docs.microsoft.com/azure/logic-apps/logic-apps-overview" \t "_blank)
* [Create a Logic App](https://docs.microsoft.com/azure/logic-apps/quickstart-create-first-logic-app-workflow)
* [Logic Apps Call, trigger, or nest workflows with HTTP endpoints in Logic Apps](https://docs.microsoft.com/azure/logic-apps/logic-apps-http-endpoint" \t "_blank)
* [First Azure Function Visual Studio](https://docs.microsoft.com/azure/azure-functions/functions-create-your-first-function-visual-studio" \t "_blank)
* [Azure Functions Python developer guide](https://docs.microsoft.com/azure/azure-functions/functions-reference-python)
* [Azure Functions create first Python Function](https://docs.microsoft.com/azure/azure-functions/functions-create-first-function-python) for latest version/environment configuration information.

Bonus - Infrastructure as Code

It’s important for all teams to be able to deploy and experiment with new technologies to help determine how they can be used, but at some point they must be operationalized. While the application developers are learning to work with Azure Functions and Logic Apps, develop code that can deploy those services in a repeatable manner using a consistant naming pattern. Consider common domain specific languages for infrastructure deployments and work with your team so they can be prepared to use these components for challenges going forward.

* [What is Infrastructure as Code?](https://docs.microsoft.com/en-us/devops/deliver/what-is-infrastructure-as-code" \t "_blank)
* [Cloud-Native IaC](https://docs.microsoft.com/en-us/dotnet/architecture/cloud-native/infrastructure-as-code)
* [What is Azure Resource Manager](https://docs.microsoft.com/en-us/azure/azure-resource-manager/management/overview)
* [Azure Resource Manager Templates](https://azure.microsoft.com/en-in/services/arm-templates/)
* [Terraform AzureRM](https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs)