Create Serverless Applications

**Choose the best Azure service to automate your business processes**

In Azure, Workflows are business processes modeled. These workflows can be integrated using multiple systems such as Logic Apps, Microsoft Power Automated, WebJobs and Azure Functions.

All these technologies accept inputs, can run actions to modify data or execute one or more actions, include conditions to decide when an action should take effect and outputs.

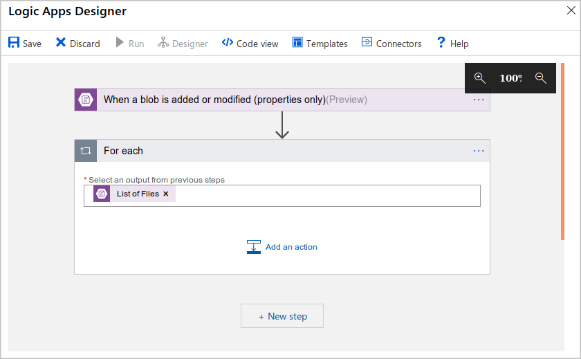
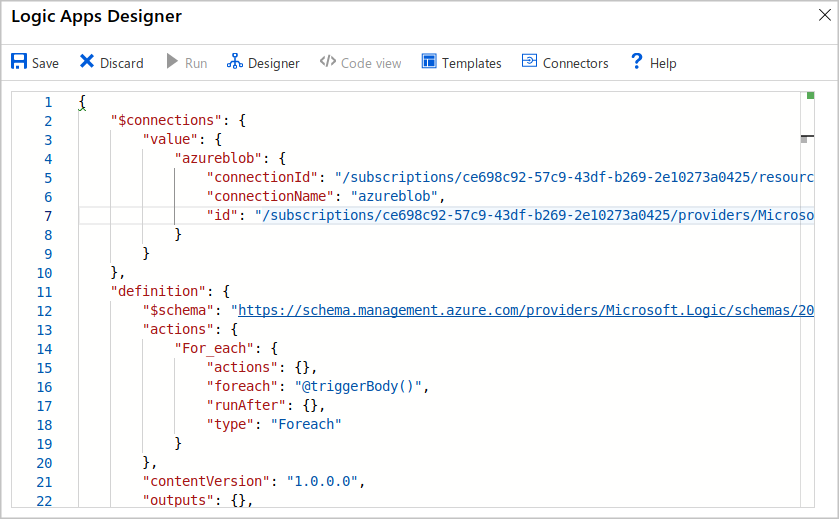
Design-first technologies comparison: Microsoft Power Automated vs Logic Apps

These technologies allow to perform a similar approach to designing a workflow as business analysts discuss and plan a business process.

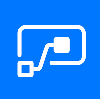
|  | **Microsoft Power Automate** | **Logic Apps** |
| --- | --- | --- |
| **Intended users** | Office workers and business analysts | Developers and IT pros |
| **Intended scenarios** | Self-service workflow creation | Advanced integration projects |
| **Design tools** | GUI only. Browser and mobile app | Browser and Visual Studio designer. Code editing is possible |
| **Application Lifecycle Management** | Power Automate includes testing and production environments | Logic Apps source code can be included in Azure DevOps and source code management systems |

Logic Apps

It’s a service use to automate, orchestrate, and integrate components. It allows to draw out complex workflows to solve complex business processes. Azure provides a user interface; as well as, a source code view using JSON notation. Logic Apps are fit for IT professionals, Developers or DevOps.

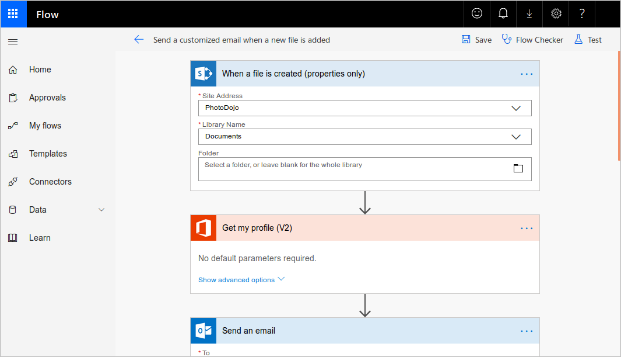
Logic Apps include over 200 connectors and hundreds of pre-built connectors. Each connector is a component which provides an interface to an external service. You can even create your own connector if your system exposes a REST API.

Microsoft Power Automate

It allows the user to create workflows without development or IT experience. Each workflow can integrate and orchestrate different components using the Azure website or Microsoft Power Automated mobile app. Under the hood, Microsoft Power Automated is built on Logic Apps; therefore, it supports the same connectors and actions. Microsoft Power Automated is more fit for non-technical staff.

There are four flows that can be created:

* Automated: The workflow starts with a trigger triggered by an event.
* Button: A button flow can be used to run a repetitive task from your mobile device.
* Scheduled: A flow can be executed on a scheduled basis such as once or more per week.
* Business Process: A flow that models a business process.



Code-first technologies comparison: Azure WebJobs vs Azure Functions

These technologies allow your developers to write code to orchestrate and integrate different business applications into a single workflow.

|  | **Azure WebJobs** | **Azure Functions** |
| --- | --- | --- |
| Supported languages | C# if you are using the WebJobs SDK | C#, Java, JavaScript, PowerShell, etc. |
| Automatic scaling | No | Yes |
| Development and testing in a browser | No | Yes |
| Pay-per-use pricing | No | Yes |
| Integration with Logic Apps | No | Yes |
| Package managers | NuGet if you are using the WebJobs SDK | Nuget and NPM |
| Can be part of an App Service application | Yes | No |
| Provides close control of JobHost | Yes | No |

WebJobs and the WebJobs SDK

WebJobs are part of the Azure App Service and allows to run a program or script automatically. The Azure App Service is a cloud-based hosting service for web applications, mobile back-ends, and RESTful APIs. There are two kinds of WebJobs. WebJobs that run in a continuous loop and WebJobs that are triggered manually or by a schedule.

WebJobs allows for the use of different languages and script languages such as Java, PHP, Python, Node.js and Shell Script (Windows, PowerShell and Bash). Also, it allows the use of the .NET Framework/.NET Core Framework by using C# or VB.NET languages.

WebJobs SDK provides a range of classes, such as JobHostConfiguration and HostBuilder, which reduce the amount of code required to interact with the Azure App Service. WebJobs SDK only supports C# and the NuGet package Manager.

There could be extra reasons to use WebJobs such as wanting to have the code be part of the existing App Service application or wanting to have close control of the object that listens for events triggering the code, for example, the JobHost class.

Azure Functions

Azure Functions allows the developer to write small pieces of code without having to setup an infrastructure to run them. There is a range of supported languages such as C#, Java, JavaScript, PowerShell, Python and many more. In addition, you only pay for the time when the code runs with the consumption plan option. Azure will automatically scale these functions in response of the demand.

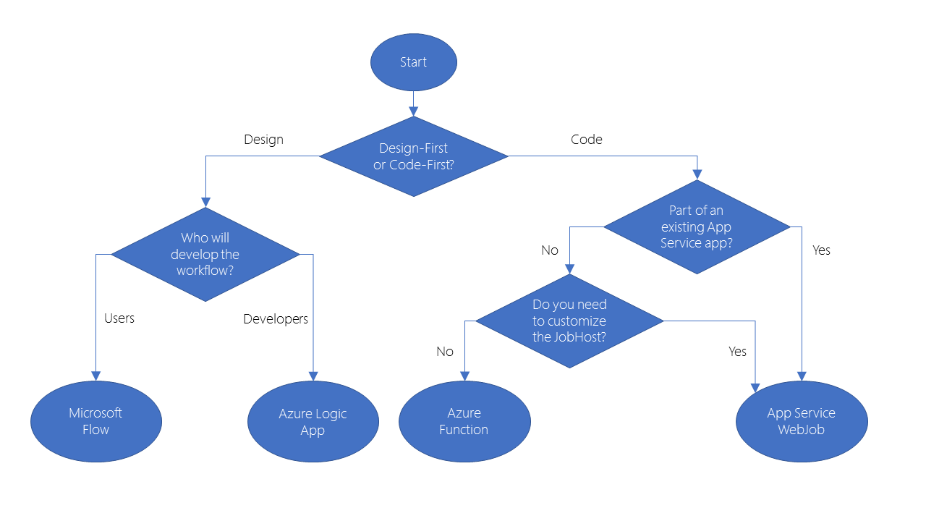
Development can be done on the portal or using a code management such as GitHub or Azure DevOps Services.

The process of creating an Azure Function is by choosing a template from a list of templates, each with a specific trigger. Since there are many different services (both in Azure and from third parties), there are many triggers available. Four common triggers are:

* HTTPTrigger: The code is executed when a request is sent to the function through the HTTP protocol.
* TimerTrigger: The code is executed according to a schedule.
* BlobTrigger: The code is executed when a new blob is added to an Azure Storage Account
* CosmosDBTigger: The code is executed when new or updated documents are created in this NoSQL database.

How to Choose a Service

Creating a workflow will depend if a developer or a user will build it. There are times when you will mix technoloagies such as to give users control over a small portion of the workflow and allow developers to control the rest. For example, you can implement a section in MS Power Automate, then call a flow from a Logic App, WebJob or Function App.



When Choosing Design-First Technology to Automate a Business Process

In this scenario, let us assume we want to automate the booking process for a bike rental business. We want to integrate a bike tracking technology on an existing bike business.

This bike business is conformed of five bike rental shops, each with a list of bikes for rent and their own database to record the bikes, features, and if they are already rented or inside the shop.

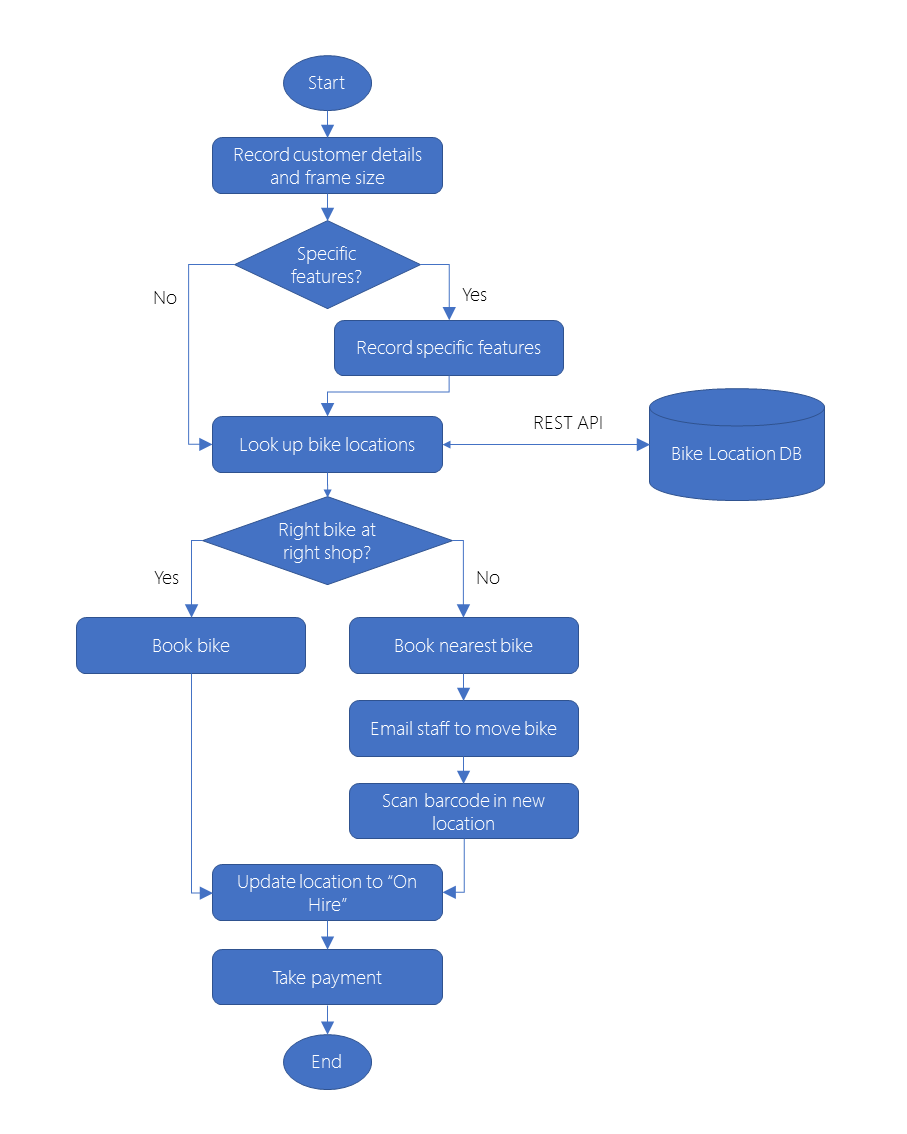
Each bike can only be rented from its home shop. If a bike from a different shop is returned, then the staff will move it back to the shop where it is listed on the database.

We want to change this process so each bike can be rented from any shop while ensuring that the staff can find out where each bike is.

Please notices that on a different state, one of the bike rentals are using a third-party system to track the bike locations. When a bike returns to that store, a unique barcode, on the crossbar, is scanned. The tracking system’s database will update the record with the name of the shop that scanned the barcode. When bikes are rented by customers, the location is changed to “On Hire” and the customer name is recorded in a separate column. This bike location database has a REST API for which other systems can call.

The customer can ask for a bike with specific frame size and features, such as having an electric motor or all-terrain suspension. If there is not a bike with the specifications, the shop can find out where suck bike is and get it or send the customer to the right shop.

The following is a workflow of the bike reservation and rental process:



The flow works as follow:

1. A customer requests a bike through the website, phone or in person.
2. The staff save the customer’s details and the bike’s frame size.
3. Does the customer specify features? If so, then what are those features?
4. What is the location of all bikes with that frame size and those features? The information should be obtained from the bike location database and should be kept updated by the barcode scanning system.
5. Is there a right bike with the frame size and features in the store? If so, then book the bike.
   1. If not, then locate the nearest bike and reserve it.
   2. Notify the staff via email to move the bike to the customer.
   3. Scan the barcode in the new location
6. Provide the bike to the customer and update the location to “On Hire”.
7. Obtain payment from the customer.

We are omitting cases such as no bike with desired frame size or features is available for rent to simplify the process of the example.

The next step is to choose the technology. All the technologies we have available has the capability to build a workflow for this business process and each of them can integrate with any REST API; therefore, we need to decide is this will be a design-first or code-first project.

We are aware that the Managing Director and the staff would like to understand the workflow, at a higher level, instead of coding and implementing. Plus, the manager doesn’t like to have separated documents describing the process, in case they become obsolete due a change in the process.

If we decide to take this approach, then we want to visualize the workflow for an easy-to-understand design surface. Plus, the diagram will prevent to have separate document by providing a clear picture of the process implemented.

Now, that we decided to take the design-first approach, we need to pick the technology: Microsoft Power Automated or Azure Logic Apps?

Nothing in this scenario indicates that the staff will be able to modify the business process and connect to the bike location database via REST API. We will need to create a custom connector which means that it’s a developer task to do so.

It makes sense that the development of the custom connector and the workflow should be made by the same person or team; therefore, the best choice is to use Azure Logic Apps.

As you witness, we narrowed down the technology to use given the solution needed based on the business process and the audience.

When Choosing Code-First Technology to Automate a Business Process

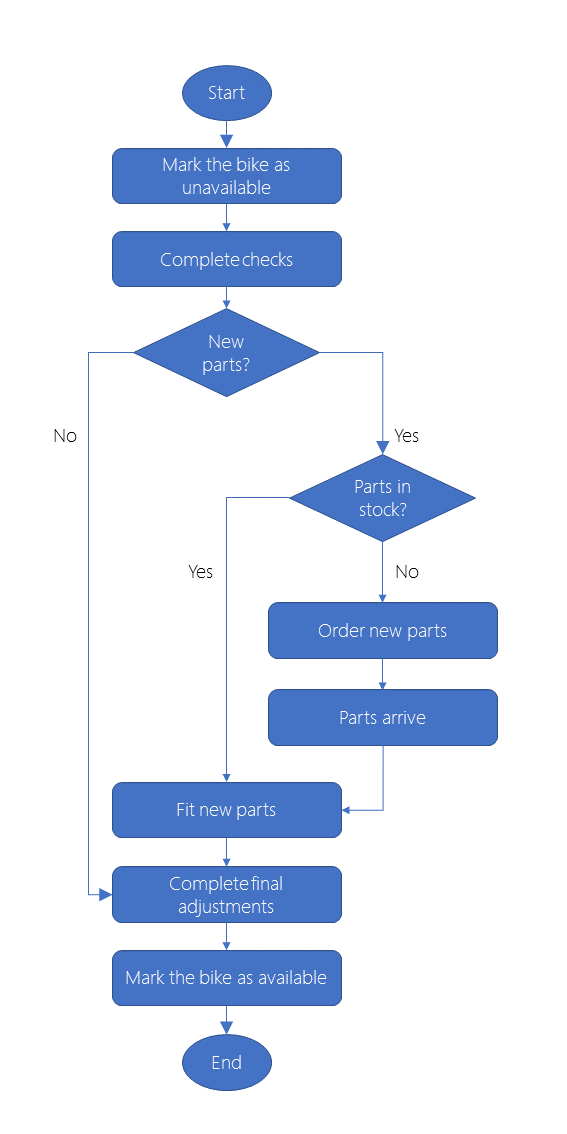
We got some bike technicians who are using a spreadsheet to keep record of their actions when repairing and maintaining each bike as their arrive. A problem to solve is when spare parts are needed since there is no way to know when a bike is waiting for repairs or parts. This created already some problems when bikes were rented with worn brake pads, flat tires and other faults which ended damaging the brand of our stores as a high-quality bike rental company.

We need to build a system that allows us to control the maintenance and repair process plus allow our technicians to find answers to the following questions:

* What jobs have been completed on a bike?
* What jobs remain to be completed before the bike can be rented out again?
* Which bikes are currently available to rent?
* Which bikes are currently unavailable to rent?
* For each bike that is unavailable?
  + What is the reason that can’t be rested out?
  + Are they waiting for any parts and what are those parts?
  + What is the ETA of the bike be ready for renting again?

We would like to integrate this system to our existent bike booking and rental process, from the last unit, in that way the staff can search for available bikes to rent. Currently, the staff can only find bikes that are currently available for rent only and the Manager wish the developers to build the solution.

The following is the workflow for the maintenance of the bikes:



The process goes as follow:

1. When a customer returns a bike, to any of the locations, the bike maintenance will begin.
2. The technician will mark the bike as unavailable.
3. The technician will complete a list of check including the state of the tires, brakes, drive chain and lights.
4. Are new parts required?
   1. If so, the technician fits the new parts
   2. If so but we there are not parts in stock, then the technician orders new parts
5. The technician then completes the final changes.
6. The bike is marked as available to rent again.

Our next step is to choose the technology. Since this project requires too many low-level details, such as accessing an inventory system and place orders with a third-party parts company, it will require a developer to implement; therefore, we can eliminate the idea of using a design-first approach. We can create a solution by using a custom connector to integrate to other workflows created using design-first tools such as Logic Apps and Power Automate. The developer will have the flexibility to solve this using the code-first approach.

For the code-first approach, we have two technologies at our disposition, Azure Functions and Azure App Service WebJobs. Part of the process of selection goes in the cost and integration factors.

When talking about cost, with WebJobs, we can pay for the entire VM or App Service Plan to host the job. Azure Functions can run using a consumption plan; therefore, we only pay when the function runs. After all, only when the bike is returned that the process kicks in; therefore, we might be able to save money that way.

When talking about integrations, we wish to integrate the maintenance workflow with the Logic App that we built for the bike booking and rental process shown previously. Even do, we could call the WebJob from a Logic App, the integration between Logic Apps and Functions is closer. This means that it can be easier to control our calls to a function from the Logic Apps designer.

Knowledge Check

We got two scenarios to test our knowledge.

Scenario 1: TV Adverts

Our company does TV adverts and we wish to formalize two business processes:

* The advert review process: Throughout an editorial process, we wish to ensure that the advert meets the standards of taste, decency, grammar, style, and legal requirements in the jurisdiction where it will be broadcast.
* The feedback collection process: The advert will also being put throughout this process in which the customers, the director, and members of the board of directors can provide feedback.

The advert review process should be managed by the members of the creative team since they will need to perform chances on regular basis. Plus, they would prefer not to have wait for a developer to perform changes.

The feedback collection process will call an on-premises SharePoint Server since this server is not as reliable as a cloud-based server is; therefore, developers want to carefully control the way the workflow retries this connection if there happens to be a failure.

Scenario 2: Camera Company Merge

Our company makes digital cameras and just recently has being acquired by a smaller company that makes lenses. We wish to ensure that we all use the same procedures throughout the company. These processes are:

* Lens quality control: We have a good reputation for lens reliability due our quality control procedure. We wish to ensure this process goes across the merged company by integrate it with our parts ordering system, which includes a REST API for convenience.
* Ordering and dispatch: Currently, the company acquired doesn’t have a formal order and dispatch procedure. We wish to ensure that the employees use our original business procedure for such tasks. The ordering system has a user interface which is built as an Azure App service web app; however, we wish to manage the order and dispatch workflow as a separate project.

We have hired a small team of developers to do the work and we advise to use a design-first approach.

* In the television advert company, which technology would you use for the advert review process?
  + [Incorrect] Azure Logic Apps: Azure Logic Apps don't permit the creative team, who are not developers, to manage the process.
  + [Correct] Microsoft Power Automate: This allows the creative team, who are not developers, to manage the flow.
* In the television advert company, which technology would you use for the feedback collection process?
  + [Incorrect] Microsoft Power Automate: It doesn’t allow developers to control retry polices as WebJobs do.
  + [Correct] Azure App Service WebJobs: this is the only technology that permits developers to control retry policies.
* In the merged camera company, which technology would you use for the lens quality control procedure?
  + [Incorrect] Azure Functions: It doesn’t provide a design-first approach intended for developers. Logic Apps do.
  + [Correct] Azure Logic Apps: This is the only technology that provides a design-first approach intended for developers.
* In the merged camera company, which technology would you use for the ordering and dispatch procedure?
  + [Incorrect] Azure App Service WebJobs: We want to manage the procedure code separately. The only technology that provides a design-first approach is Azure Logic App
  + [Correct] Azure Logic Apps: This is the only technology that provides a design-first approach intended for developers.

Summary

Some details that are important to check is if the process involves different steps, loops or conditional branches. Next, if the process will be long-running and complete over days or weeks as the staff become available or other delays. Then, we need to check if the process involves different systems such as web services, email servers, database servers and other components. Also, we need to pay attention if we need to integrate our processes to a custom or third-party system, for which might require a custom connector. Finally, we want to have in consideration if non-developers are going to need to modify and update the workflow without developer’s assistance.

**Create Serverless Logic with Azure Functions**

Without having to create and manage an infrastructure, Azure functions allows developers to host business logic.

Let us assume we are an escalator company and we are applying IoT technology to monitor our products. We wish to oversee the processing of temperature sensor data from the drive gears of our products. We monitor the temperature data and included a data flag to indicate when the gears are too hot. By downstream this data, we can determine the maintenance needed.

Our company receives sensor data from different locations and different escalator models. The data may arrive in different formats, including batch files uploads, from schedule database pulls, messages on queue, and even coming data from an event hub.

Normally, we would need to plan and install the infrastructure, coordinate with IT to do the management, and much more work. With serverless computer, you allow the could provider to manage the provisioning and maintenance of the infrastructure while we can focus building the app logic.

Azure Functions will enable us to run pieces of code or functions, in the cloud, in the programming language of our choosing.

Is Serverless Computing Right for The Business Needs\

Serverless compute are functions as a service (FaaS), or microservices hosted on the cloud platform. Without having to provision or scale infrastructure, we can run our business logic as functions. Our app, depending of the load, will automatedly be scaled out or down. There are two approaches that are commonly used: Azure Logic Apps and Azure Functions.

Azure Functions

Azure Functions, a serverless application platform, allows developers to host business logics that can be executed without having to worry to setup an infrastructure. Apart of intrinsic scalability, we can write our functions in different languages including JavaScript, Python, PowerShell, C# and F#. It also supports packages managers like NuGet and NPM, allowing us to use popular libraries to complete our business logic.

Some other benefits, apart of those listed above, is that we can avoid over-allocation of infrastructure. We don’t need to provision and configure VM servers in order to handle peak loads or waste resources when there is no a peak because we gain automatic scaling and get billed only when the function is processing work.

Stateless functions, such as these, allow instances to be created or destroyed on demand. If by any chance a state is required, then we can associate a storage service to be used by the functions.

All Azure Functions are driven by events aka. triggers. For example, when we receive an HTTP request (HTTP trigger), we can have the function take some input and generate an output. There is no need to do coding for the triggers since there are many available.

There are some drawbacks of using such solution. Azure Functions have a default timeout of 5 minutes. We can set a timeout value of maximum 10 minutes, but it requires to host it on a VM. If the function is trigger via HTTP, there is a timeout restriction of 2.5 minutes for a response be generated. Finally, if we wish to orchestrate the execution of multiples functions without any timeout, there is an alternative called Durable Functions.

Depending of the execution frequency, times you think multiple clients may call the functions, it might be cheaper to hos your service on a VM.

Be aware that one function app instance can be created every 10 seconds, for up to a total of 200 instances. Each instance can provide service to multiple concurrent executions without limit of how much traffic a single instance can handle.

Depending of the trigger, there are different scaling requirements; therefore, we are required to research which trigger to use and its limits.

When we create a function app, we define it to a logical group and structure as a compute resource in Azure. For our escalator project, we could create a function app to host the escalator drive gear temperature service for example.

Either way, in order to create a function app, we must choose a service plan and select a compatible storage account.

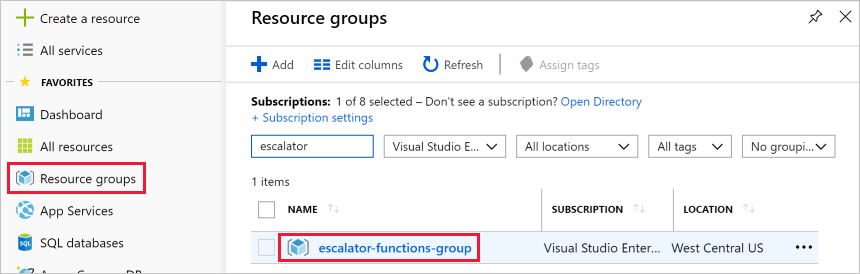
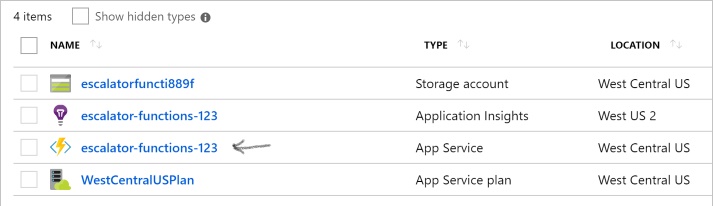
There are two types of service plans at our disposition. When using the Azure serverless application platform, the first plan is the Consumption service plan. This plan provides automatic scaling while billing us when our function is running. When selecting this plan, we can configure the function app’s timeout period which is 5 minutes by default, and can we be extend up to 10 minutes as explained before. When using a VM, the second plan is Azure App service plan. This is not a serverless plan. When using this plan, we are responsible to manage the app resource in where the function runs on. This option is better when functions are used continuously or when more processing power or execution time, than those provided by the Consumption plan, is required.

Functions apps require us to have a storage account. We can create a new storage or select an existing storage. This storage is used for internal operations such as logging executions plus managing triggers. If the Consumption service plan is used, the storage is used to store the function code and the configuration file.

To create a function app, we must first sign into the Azure portal.

1. From the portal menu, we must select Create a resource:   
   
2. When creating a resource, in the Azure Marketplace, we wish to select Compute and then Function App:  
   
3. Our function requires a unique name, for example, escalator-functions-xxxx
   1. This name should be globally unique.
   2. Valid characters are a to z, 0 to 9 and dashes (-)
4. Select a subscription such as Concierge Subscription.
5. Select an existent resource group.
6. Select the operative system. In our case, we will choose Windows.
7. Select the Hosting plan. In our case, we select Consumption Plan which is serverless hosting.
8. For the Runtime Stack, we will select Node.js which uses JavaScript as its script language.
9. If you don’t have a storage account, then create a new one. It must be unique.
10. Select the region closest to you.
11. Make sure Azure Application Insights in On
12. Then press Create. The deployment will take some time to be ready and we will receive a notification when completed.

To verify our azure function app is ready for us to use:

1. From the Azure portal left-hand menu, select Resource group:  
   
2. We should see a resource list: 
3. If we click on our new function, we can see that see that it has a public URL assigned to it for which we can call with our browser. By doing that, we will receive a default web page indicating that the Function App is running.