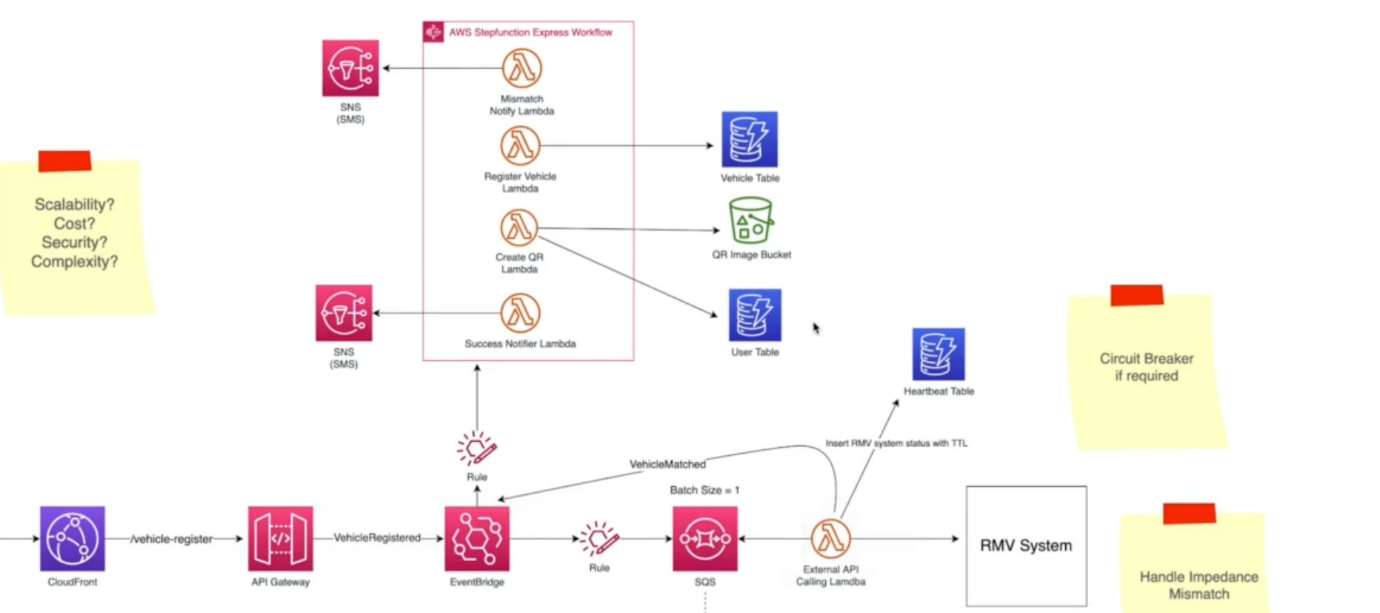
**13. Achieving Idempotency in Serverless Architectures**



* I want to discuss with you is related to idempotency. So, if your systems are idempotent, that means if the same request got executed a couple of times, you will get the same result.

--- **step function express workflow**

* Now have a look at this in the step function express workflow. We have a step where a lambda function. Try to register the vehicle details in a vehicle table. Now, what if this request occurs a couple of times. you can call your system a idempotent if only one record will be created in the Dynamo DB table.
* So, the next time is trying to create the same vehicle detail in the vehicle table. It will first check whether this information is available. If not, only it will create, otherwise it will return an error.
* Now we can achieve idempotency. If we are working with Dynamo DB by using conditional attributes like **attribute\_not\_exist**. So, when Lambda functions call out to this Dynamo DB through the Dynamo DB SDK, we can simply pass that logical condition add only if this record not existing in the table already. So, inserting maybe not a big issue, but what about an update?
* Now, this may be not a good example, but imagine this an e-commerce web application. So, this function. So, let us say this function is trying to charge the customer. So due to some reason, if the change happened multiple times, that would be a real trouble. Customer will start complaining.

--- **lambda power tools**

* So, we had to make sure only once the change is happening. So, handling it idempotency is a challenging task. So that is why we have tools like Lambda Power tools. Now this Lambda Power tools, it contains multiple utilities and one of the utilities is it in idempotency and Lambda Power Tools is available for different languages for example, Python, Java, TypeScript and so on.
* So, if you use Lambda Power tools, you can use these utilities. So here, in order to insure idempotency, we need to create something called idempotency Key and there is nothing but some unique identifier for this request. Then it will do something like what we did with Circuit Breaker. So, the idempotency key will be updated in a table. Now this will happen after the first-time update and when it tries to update for the second time, it will check that it idempotency table.
* It is going to look up with the idempotency key if some value is already existing. If so, then it's not going to do the update. It will return. So, it is important when you design your architecture with different managed services here, in this case, express step function workflows to check what are the sales or what are the guarantees of these services.
* So, it is always good that you have this idempotency checks before you use this service or else you can also find different services that will do only once delivery and the service itself will make sure, there will not be any duplicates. Now these are some of the considerations you need to think about.