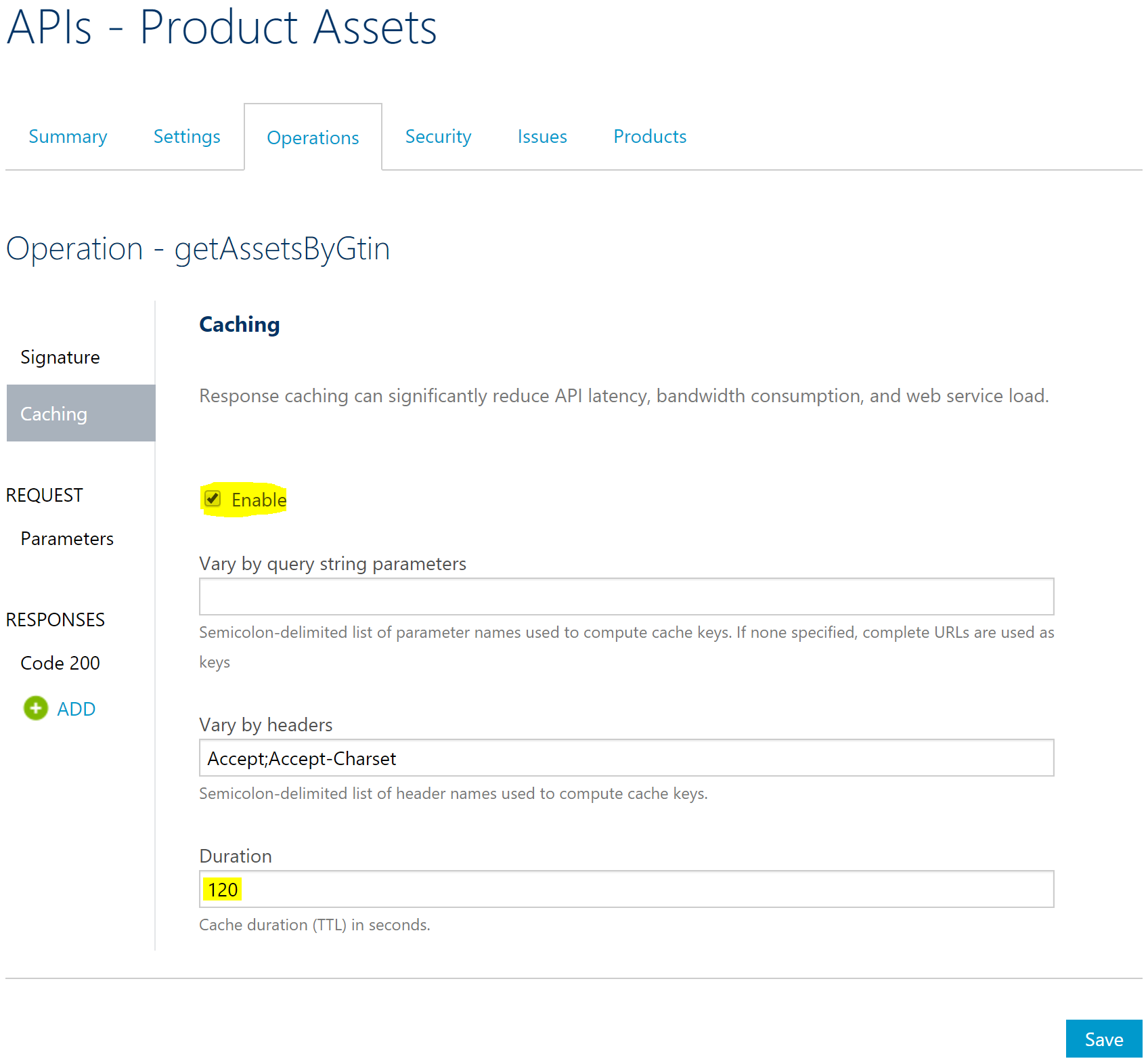
# Configure Service Response Caching

1. Go to Publisher Portal 🡪 APIs 🡪 Product Assets 🡪 Operations 🡪 getAssetsByGtin 🡪 Caching
2. Enable response caching for 2 minutes as shown below.



1. Go to the policy editor of the getAssetsbyGtin operation and note that the above configuration has added caching policies to both the outbound (to cache response) and inbound (to fetch from cache) sections of the service, as shown below.



1. Access the Product Assets service (<https://pgecommerce.azure-api.net/dam/assets/47400656109>) and the aggregated Products service (<https://pgecommerce.azure-api.net/products/47400656109>). Look at the value of **assetsLastUpdatedAt** attribute.
2. Continue accessing the APIs within a span of 2 minutes and notice that the **assetsLastUpdatedAt** values does not change, indicating that the service response is being cached. After 2 minutes, the value gets updated indicating that cache has been updated with new service call result.

# CACHED RESULT WHEN SERVICE IS DOWN

1. In the Azure Portal, go ahead and stop Product Assets API App.
2. Access the Product Assets service (<https://pgecommerce.azure-api.net/dam/assets/47400656109>) and the aggregated Products service (<https://pgecommerce.azure-api.net/products/47400656109>). Note that the service and the corresponding HTML pages continue to function without any disruption for the cache duration configured.
3. **Ensure to restart the Product Assets API app for the remaining hands-on exercises.**

# References

* Basic response caching – <https://docs.microsoft.com/en-us/azure/api-management/api-management-howto-cache>
* Custom caching / fragment caching – <https://docs.microsoft.com/en-us/azure/api-management/api-management-sample-cache-by-key>
* Caching policies – <https://docs.microsoft.com/en-us/azure/api-management/api-management-caching-policies>