HIGH LEVEL FLOW FOR MLOPS ON AZURE CLOUD

Resource Creation:
- All resources that are used for MLOps in Azure can be created using ARM templates in IaC Pipelines

CI Pipeline:
- Create repository on the Azure project to host the source code. Enable pipelines to perform CI/CD operations
- Create CI pipelines to perform code quality tests and unit testing on every commit
- Create tasks in CI pipeline to build, publish and trigger ML Pipeline to Azure ML and CI pipeline would wait until the tasks have finished their job

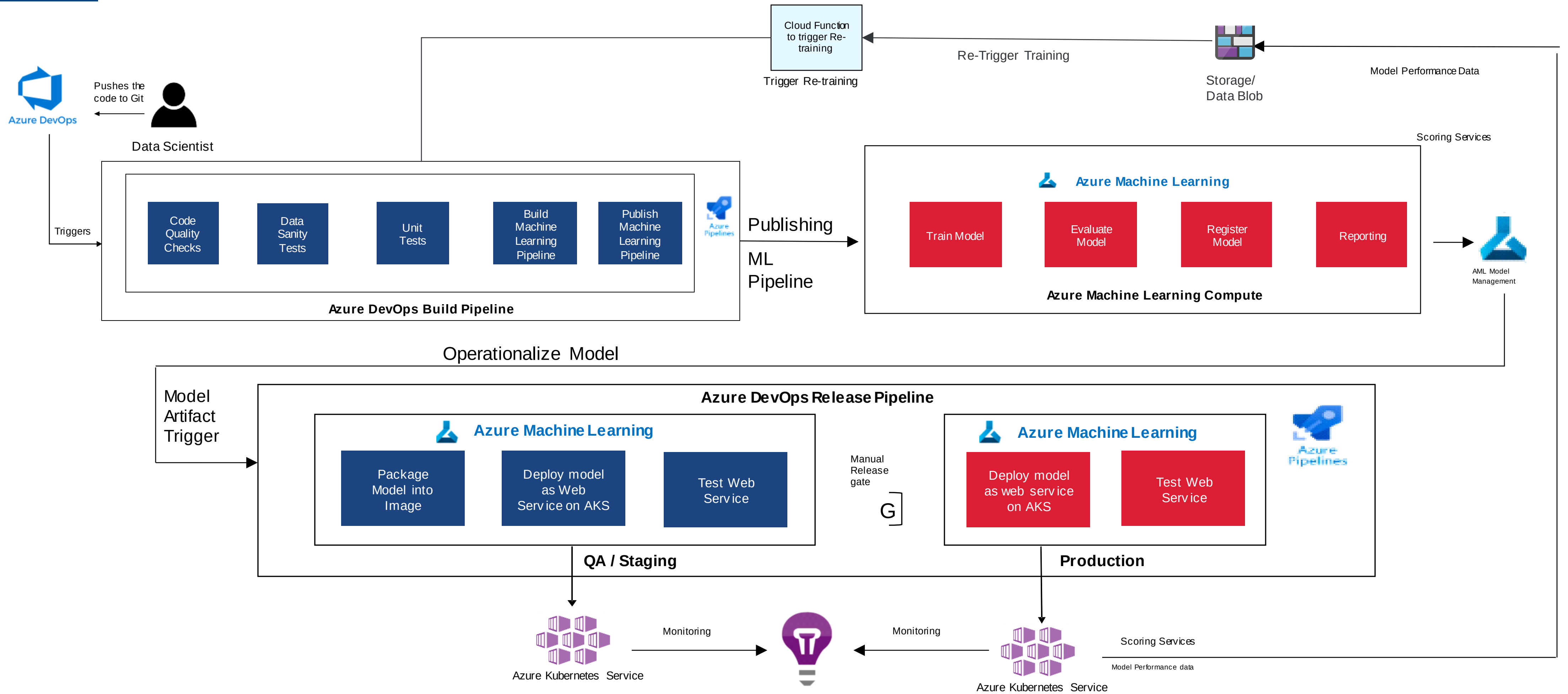
ML Pipeline:
- After the ML pipeline is published and triggered on Azure ML workspace a trained model is given as the output
- Perform model evaluation with existing model in production and register if evaluation succeeds
- As soon as a model is registered the CD Pipeline would be triggered automatically

CD Pipeline:
- The CD pipeline would be configured to pick the latest model from the model registry, package it as a service. Model registry is a version control system for the built models. Any selected previous version of the model can be deployed if desired
- Deploy the packaged model as Webservice onto desired environment (AKS cluster) which provides the REST API Endpoint for consumption of the model

Monitoring:
- Azure App Insights and Azure Monitor Services help us generate and monitor custom logs along with viewing live data in analytics of the deployed cluster and service.
- Azure ML workspace has all the historical runs which provides the complete meta information on items like the data set that has been used for that run, what it's accuracy is, and if the model was deployed, what the features & significance were, etc. Model evolution is not just confined to code change versions in the version control system

Re-Training:
- Azure Functions are created to trigger retraining of pipelines based on various conditions such as data drift or scheduled time intervals

END-TO-END MLOPS ARCHITECTURE USING AZURE ASSETS



ARCHITECTURE FLOW FOR MLOPS

End-to-End MLOps pipeline on Azure can be architected using ML Build and ML Release pipelines

Step 1:

Data Scientist writes/updates the code and push it to Azure repo. This triggers the Azure DevOps build pipeline (continuous integration) and performs code quality check, data sanity tests and unit tests on the new code and publishes the test results.

Step 2:
Azure DevOps builds the pipeline and publishes ML Pipeline to Azure ML Workspace

Step 3:

ML Pipeline typically performs the below tasks

- **Feature Engineering** task prepares the features required for the model
- **Train Model** task executes model training script on Azure ML Compute.
- **Evaluate Model** task evaluates the performance of newly trained model with the model in production. If the new model performs better than the production model, the following steps are executed. If not, they will be skipped.
- **Register Model** task takes the improved model and registers it with the Azure ML Model registry with version control enabled.

Step 4:

Release pipeline triggers when the new model registered to the Model registry and deploys to QA/Staging AKS and following to Production AKS cluster post the approval from the Approvers as Manual approvals and gates are configured.

Step 5:

The deployed services are available as REST API Endpoints