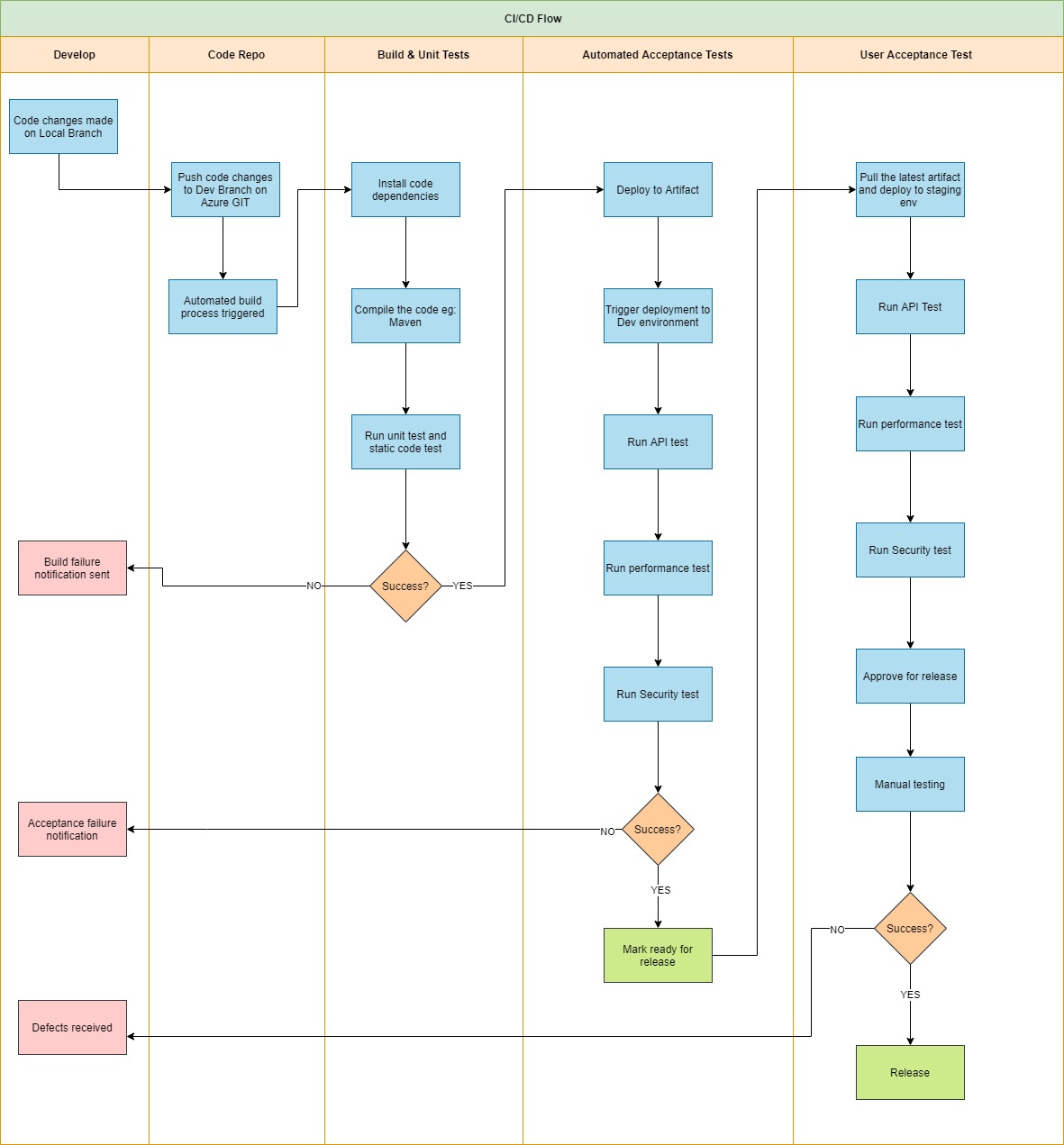
Scenario 1:

CI/CD for DevOps deployment in FastCarz should have something like below flow for their code changes to be merged from lower to higher environment:



Flow Walkthrough:

* Developer make code changes in their local repo and trigger a Pull request to merge code to Dev Branch. (Developer can build the code using any build tool like: Maven, ant etc)
* Once the PR is approved the automated build process gets triggered.
* New version gets build. Unit and Code coverage gets triggered. On failure notification gets generated and the build stops.
* Post successful testing, Artifacts gets generated.
* An automatic deployment gets triggered for Dev (here we can enable approval process in Release pipeline for higher environment.)
* Post successful deployment, automated performance and security test could be triggered. In case of any failure notification to be generated.
* Upon successful test, the deployment is ready to go for QA and PROD.
* Approvals can be added as needed.

Scenario 2:

Since I haven’t worked with terraform yet, so I’ll use ARM templates for this problem.

1. For this example, we can prefer to create an Azure Pipeline artifact. Another option can be Github Artifact
2. We can use the Azure DevOps Repo (recommended), bitbucket(can be integrated with DevOps).
3. Before we start, we should have Resource Group level (1 for each environment) and 1 Subscription level Service connections for Azure and DevOps to connect and deploy resources.
4. For ARM templates have a template.json and parameter.json file, the focus should be to keep one parameter.json for 1 environment. And we should parameterize as much variables possible.

* Once the ARM templates for resources are ready, we can use Azure Repo to store them.
* Start creating pipeline, click on new pipeline->select the repo->select the path where the ARM template is being kept.
* Select “Starter pipeline”
* Search for ARM template deployment in “tasks” and fill in the required details.
* Variables values could be kept here that needs to be passed though this pipeline.

1. In order to access the keyvault from Azure Devops first we need to set up a service principal that will have access to Azure pipelines.
2. Go to the Keyvault in Azure portal, select the vault and select “access policies”
3. Now “Add access policy” . Add “Get” and “List” permissions.
4. Add the service principal . and seclet ok
5. In DevOps search for Azure key vault task, add in the required details like: Subsscription, Keyvault name etc.