



Taking the Fabric Development Experience to the next level



DUTCH FABRIC USER GROUP

STAY CONNECTED!



/company/DutchFabricUserGroup



DutchFabricUserGroup.com

Or follow us on the Fabric Community Website →





OUTCH FABRIC USER GROUP

Our next meetup!

Tuesday January 21st, 2025

Speakers and location to be announced!

Setting the scene



CI / CD

Continuous Integration

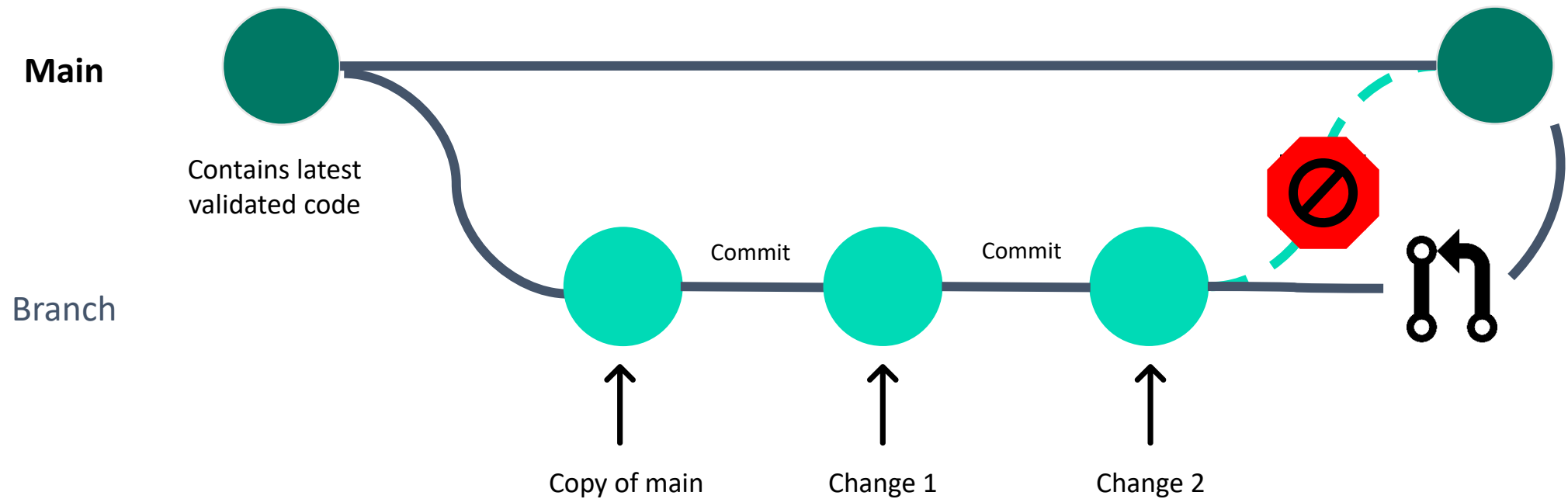
- Small but frequent changes
- Working with multiple people on various aspects of the solution
- Merging back into main branch

Continuous Deployment

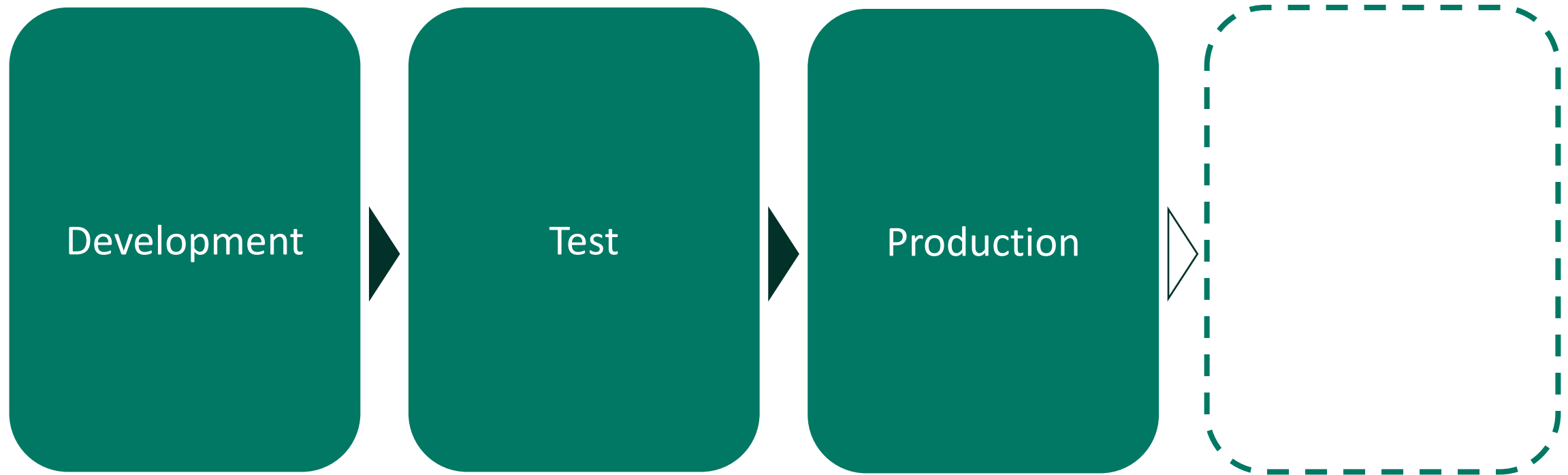
- Working in short cycles (often Agile)
- High frequency updates and releases
- Repeatable deployment process

This all, to improve the developer experience.

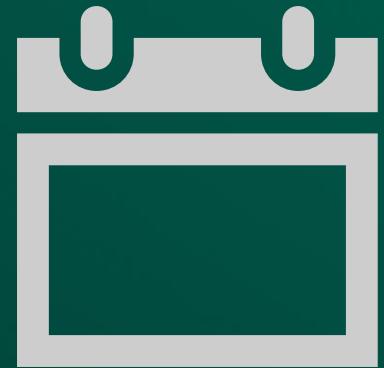
Git concept



Staged approach

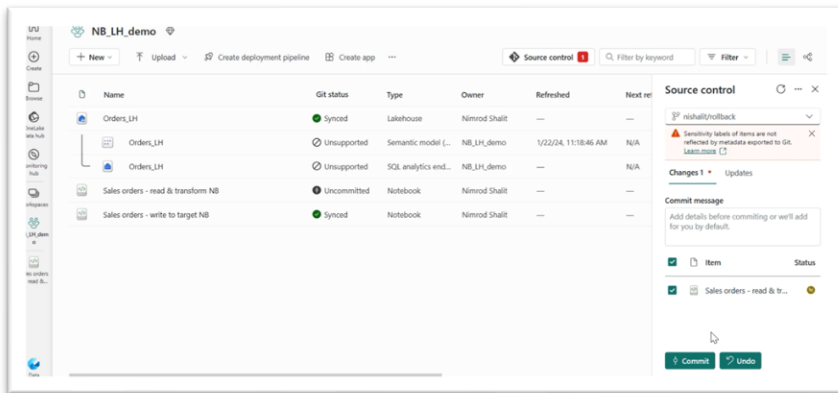


What do we have today?



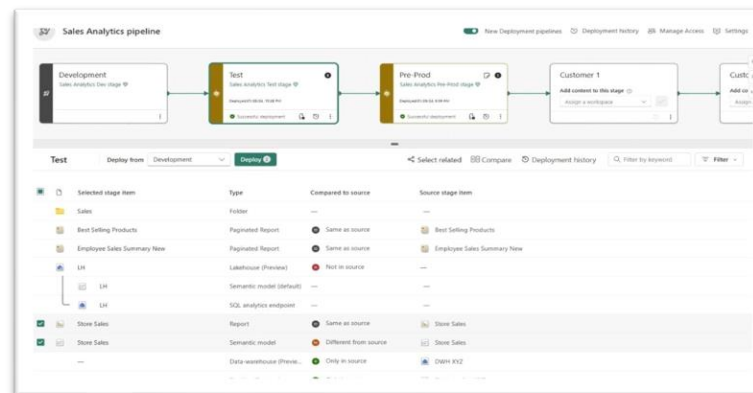
Fabric CI/CD platform

Built-in git integration



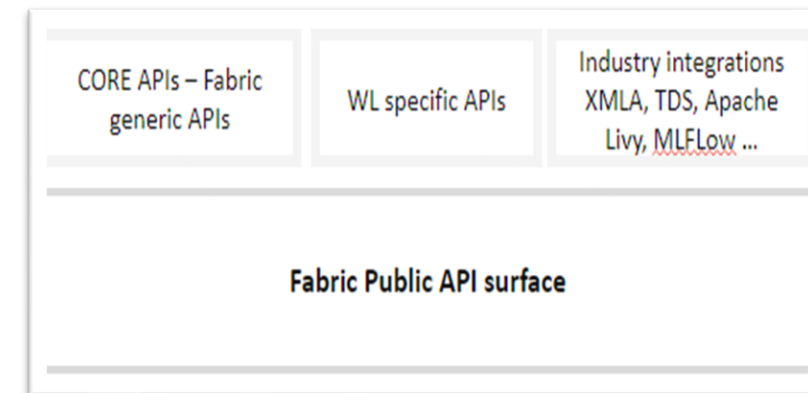
General Available

Deployment pipelines



General Available

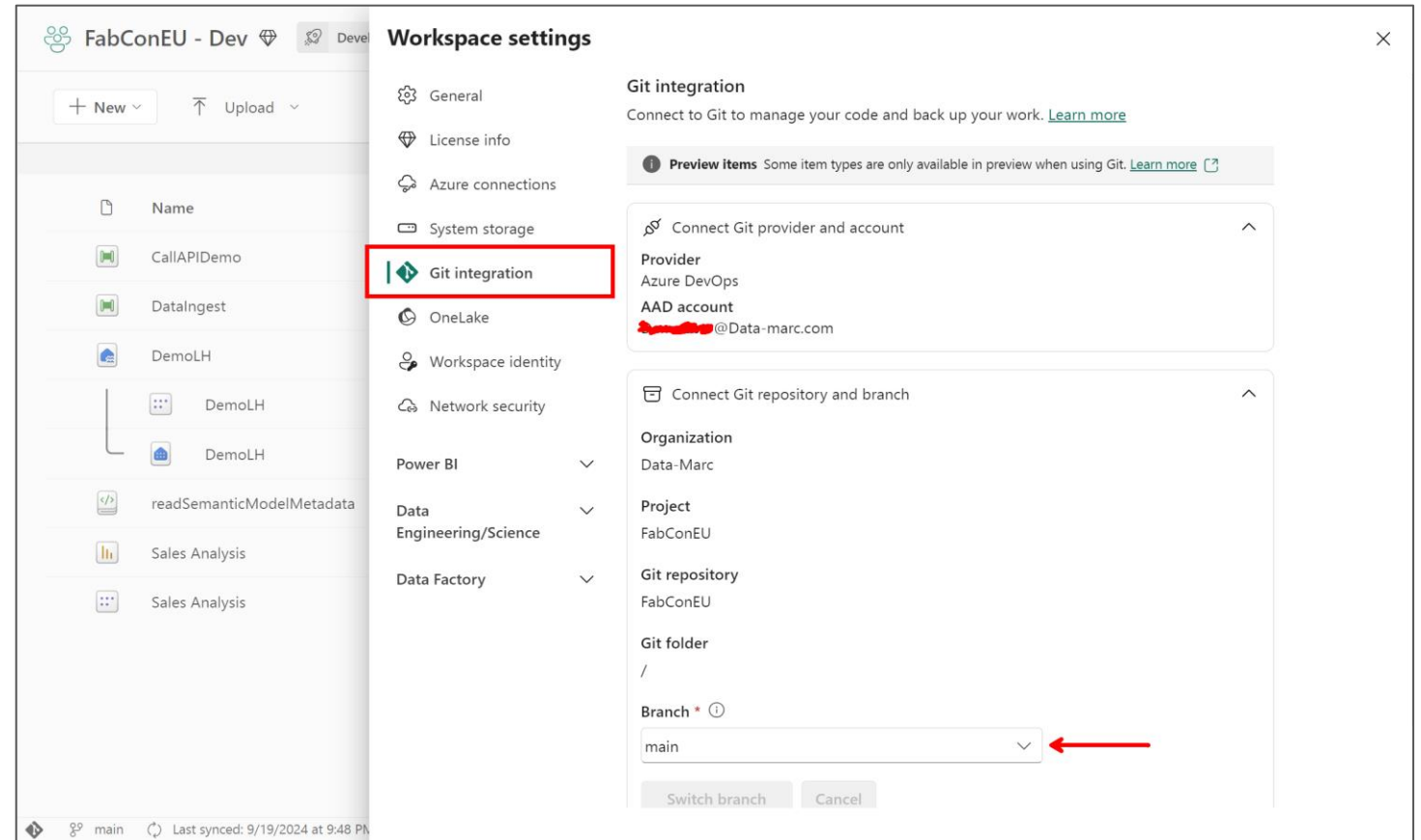
Fabric REST APIs



Public preview

Git integration

- Sync a Workspace to a Git branch
- Git providers
 - Azure DevOps
 - GitHub **New**
 - GitHub Enterprise **New**
- Fabric git APIs – REST APIs & PowerShell samples.
- Manage branches
 - Switch branch
 - Checkout new branch
 - Branch out to new workspace



DEMO TIME!



Deployment Pipelines

- Deploy items across Workspaces
- Apply rules on configuration
- Majority of Fabric items supported (*and more to come*)
- Compare changes on code-level
- Create a pipeline of 2-10 stages
 - Pipeline designer at creation
 - Ability to add custom stage names

The screenshot displays the Microsoft Fabric Deployment Pipelines interface. The top section shows a visual pipeline with stages: Development (FabConEU - Dev), Test, Custom Stage Name 2, and Custom Stage Name 3. Each stage has an 'Add content to this stage' button and an 'Assign a workspace' dropdown. The bottom section is a table titled 'Development' showing a list of items with columns for 'Selected stage item', 'Type', 'Compared to source', and 'Source stage item'.

Selected stage item	Type	Compared to source	Source stage item
CallAPIDemo	Data pipeline (Preview)	—	—
DataIngest	Data pipeline (Preview)	—	—
DemoLH	Lakehouse (Preview)	—	—
DemoLH	Semantic model (default)	—	—
DemoLH	SQL analytics endpoint	—	—
readSemanticModelMetadata	Notebook (Preview)	—	—
Sales Analysis	Report	—	—
Sales Analysis	Semantic model	—	—

Fabric User APIs

- Automated operations on behalf of Fabric users.
- Supporting CRUD operations
- Example usage scenarios:
 - Item management (*see table*)
 - Item definition
 - Workspace management
 - Workspace access management
 - Execute item jobs

Most up to date status, see:

[Item management - Microsoft Fabric REST APIs](#) | [Microsoft Learn](#)

Item type	Create (without definition)	Get	Update	Delete	List
Dashboard	✗	✗	✗	✗	✓
DataPipeline	✗	✓	✓	✓	✓
Datamart	✗	✗	✗	✗	✓
Eventhouse	✓	✓	✓	✓	✓
Eventstream	✓	✓	✓	✓	✓
KQLDatabase	✗	✓	✓	✓	✓
KQLQueryset	✓	✓	✓	✓	✓
Lakehouse	✓	✓	✓	✓	✓
MLExperiment	✓	✓	✓	✓	✓
MLModel	✓	✓	✓	✓	✓
MirroredWarehouse	✗	✗	✗	✗	✓
Notebook	✓	✓	✓	✓	✓
PaginatedReport	✗	✗	✗	✗	✓
Report	✗	✓	✗	✓	✓
SemanticModel	✗	✓	✗	✓	✓
SparkJobDefinition	✓	✓	✓	✓	✓
SQLEndpoint	✗	✗	✗	✗	✓
Warehouse	✓	✓	✓	✓	✓

New announcements



New UI for Deployment Pipelines

- Switch to enable/disable UI
- Easier navigate
- More focused (per stage)
- Smoother flow
- Folder structure
- Identify unsupported items

You decide what you like best

The screenshot displays the 'Sales Analytics pipeline' interface. The top navigation bar includes 'New Deployment pipelines', 'Deployment history', 'Manage Access', and 'Settings'. The pipeline stages are 'Development' (Sales Analytics Dev stage), 'Test' (Sales Analytics Test stage), 'Pre-Prod' (Sales Analytics Pre-Prod stage), and 'Customer 1'. The 'Test' stage is selected, showing a 'Deploy from' dropdown set to 'Development' and a 'Deploy' button. Below the stages, a table compares 'Selected stage item' with 'Source stage item'.

Selected stage item	Type	Compared to source	Source stage item
Sales	Folder	—	—
Best Selling Products	Paginated Report	Same as source	Best Selling Products
Employee Sales Summary New	Paginated Report	Same as source	Employee Sales Summary New
LH	Lakehouse (Preview)	Not in source	—
LH	Semantic model (default)	—	—
LH	SQL analytics endpoint	—	—
Store Sales	Report	Same as source	Store Sales
Store Sales	Semantic model	Different from source	Store Sales
—	Data-warehouse (Preview)	Only in source	DWH XYZ
—	Pipeline (Preview)	Only in source	Data pipeline XYZ

GitHub integration

- Second git provider next to Azure DevOps
 - GitHub + GitHub Enterprise
- Tenant admin explicitly must activate the feature
- Potential multi-geo restrictions not enforced
- API support coming soon

The screenshot displays the 'Workspace settings' interface. On the left is a sidebar with a search bar and a list of settings: General, License info, Azure connections, System storage, Git integration (highlighted with a green bar), and OneLake. The main content area is titled 'Git integration (Preview)' and includes a link to 'Learn more'. Below this, it says 'Connect Git provider and account'. Under the 'Git provider' section, there are two buttons: 'Azure DevOps' and 'GitHub'. The 'GitHub' button is highlighted with a black border. A callout box in the top right corner provides additional details about the 'Git integration' feature, stating it is enabled for the entire organization and that users can select GitHub as their Git provider.

Git integration

- △ Users can sync workspace items with **GitHub** repositories
Enabled for the entire organization

Users can select **GitHub** as their Git provider and sync items in their workspaces with **GitHub** repositories.

☒ Enabled

Workspace settings

Search

- General
- License info
- Azure connections
- System storage
- Git integration**
- OneLake

Git integration (Preview)

Connect to Git to manage your code and back up your work. [Learn more](#)

Connect Git provider and account

Git provider

Azure DevOps

GitHub

Generally available

Fabric Git integration



Supported
items



Data pipeline



Lakehouse



Warehouse



Reflex

New



Report



Paginated Report



Semantic Model



Real-Time Dashboard

New



Notebook



Spark Job Definition



Spark Environment



Queryset

New



Dataflow Gen2



Org App



Eventhouse



Metrics Set



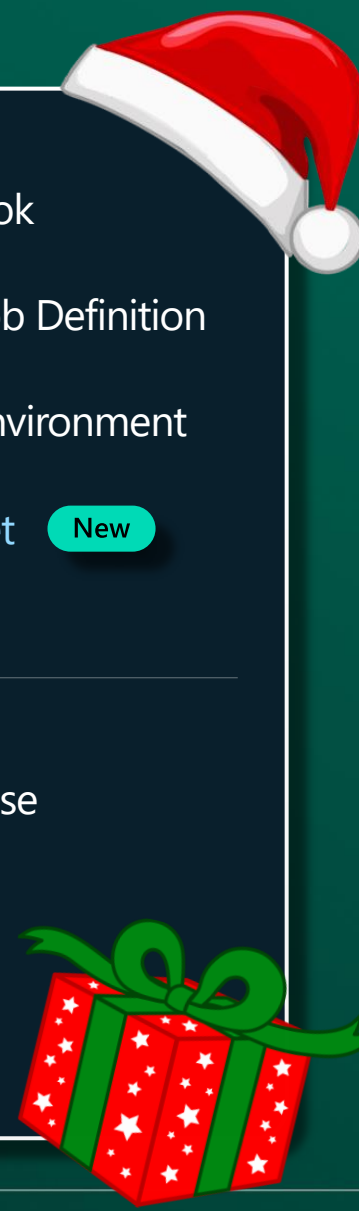
Eventstream



Mirrored Database



GraphQL API

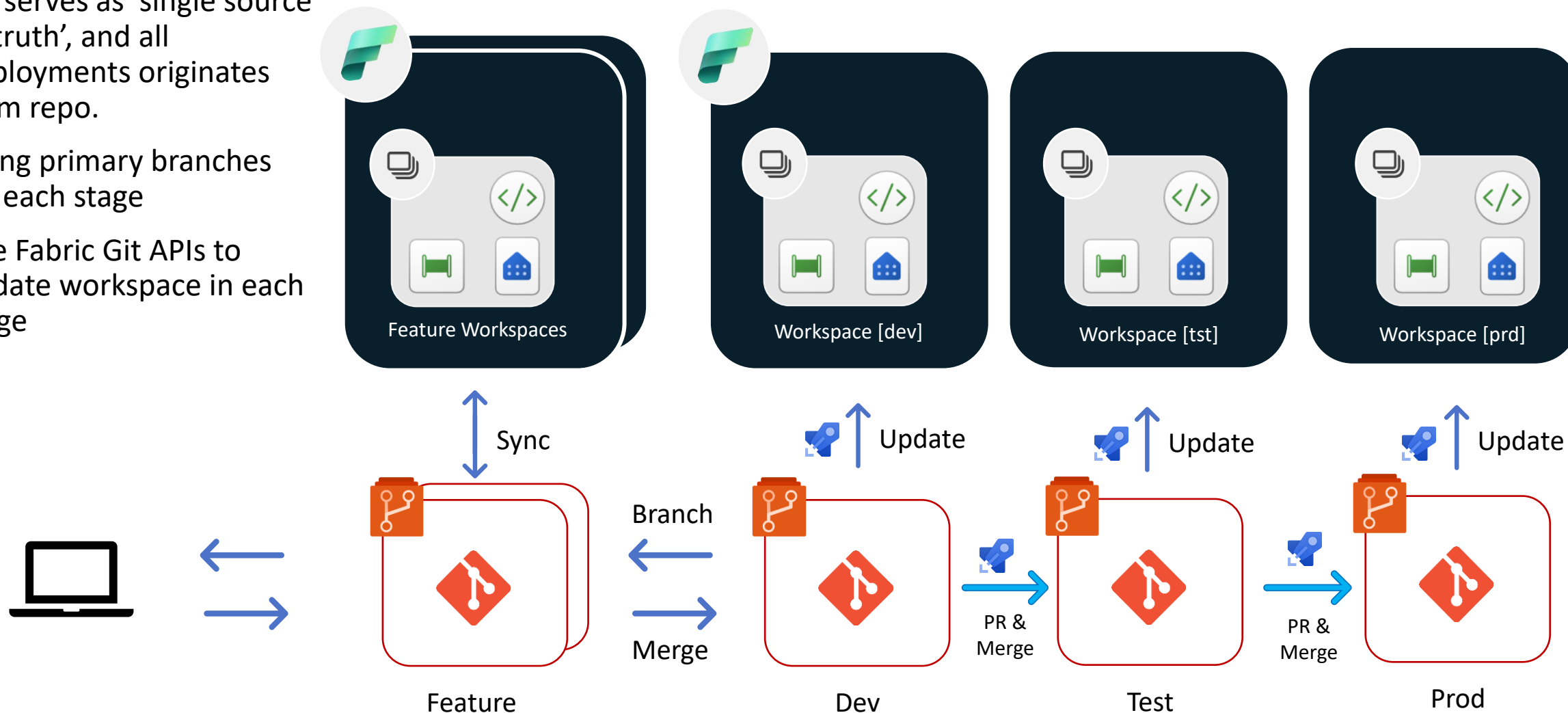


Deployment strategies



Scenario 1 – Git based deployments

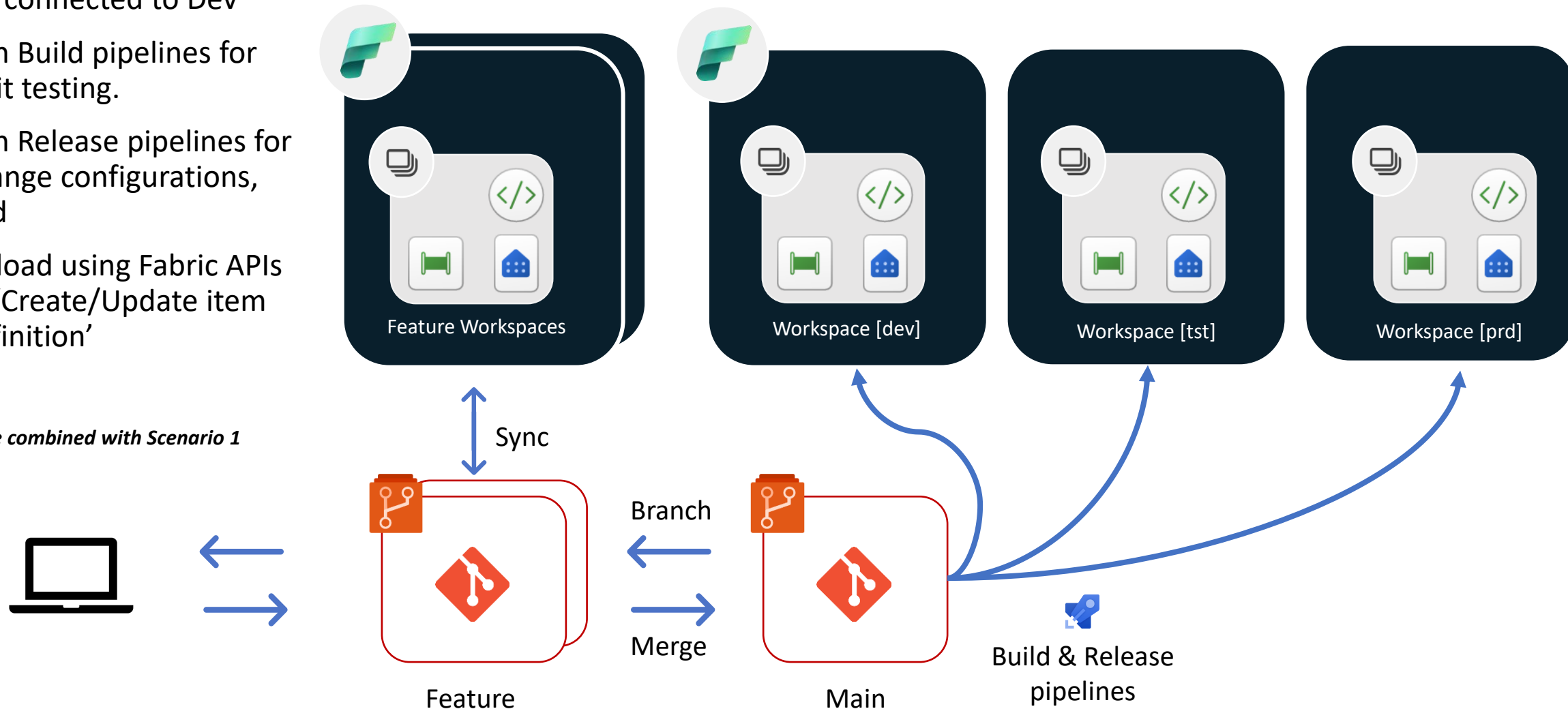
- Git serves as 'single source of truth', and all deployments originates from repo.
- Using primary branches for each stage
- Use Fabric Git APIs to update workspace in each stage



Scenario 2* – Git & Build environments

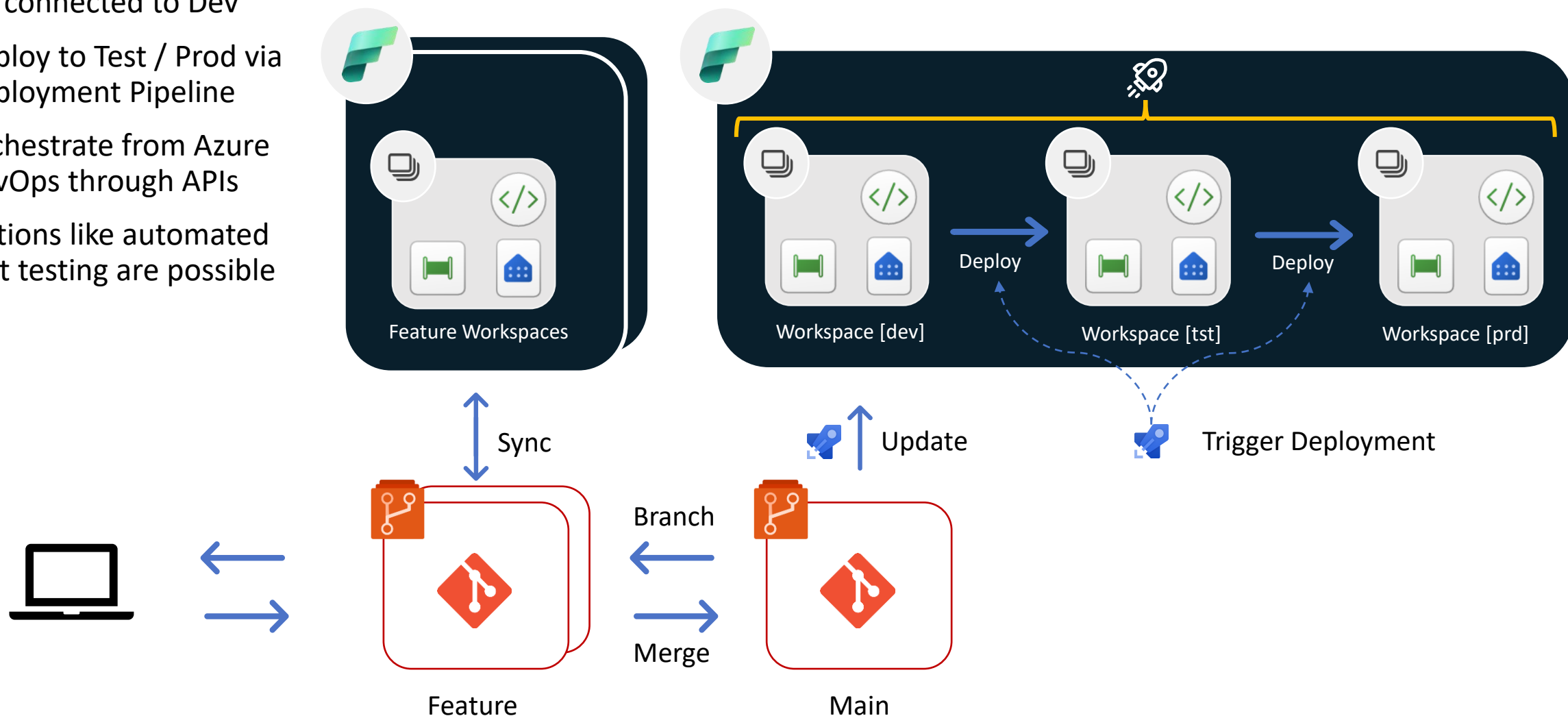
- Git connected to Dev
- Run Build pipelines for Unit testing.
- Run Release pipelines for change configurations, and
- Upload using Fabric APIs to 'Create/Update item definition'

* Can be combined with Scenario 1



Scenario 3 – Git & Deployment pipelines

- Git connected to Dev
- Deploy to Test / Prod via Deployment Pipeline
- Orchestrate from Azure DevOps through APIs
- Options like automated unit testing are possible



Opinionated view!

Entry point	Scenario	Code heaviness
Prefer an user interface?	Start with deployment pipelines interface, grow into scenario 3 over time	Low
Comfortable working with git/code?	Your go-to scenario will be scenario 1 .	Medium
Require a lot of customization?	Using the APIs in scenario 2 allows you to customize everything to your wish	High

In greenfield scenario – opt for scenario 1!

DEMO TIME!



Round-up & Questions?

