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 →

DUTCH FABRIC USER GROUP

Our next meetup!

Tuesday January 21st, 2025

Speakers and location to be announced!

Setting the scene

CI / CD

Continuous Integration

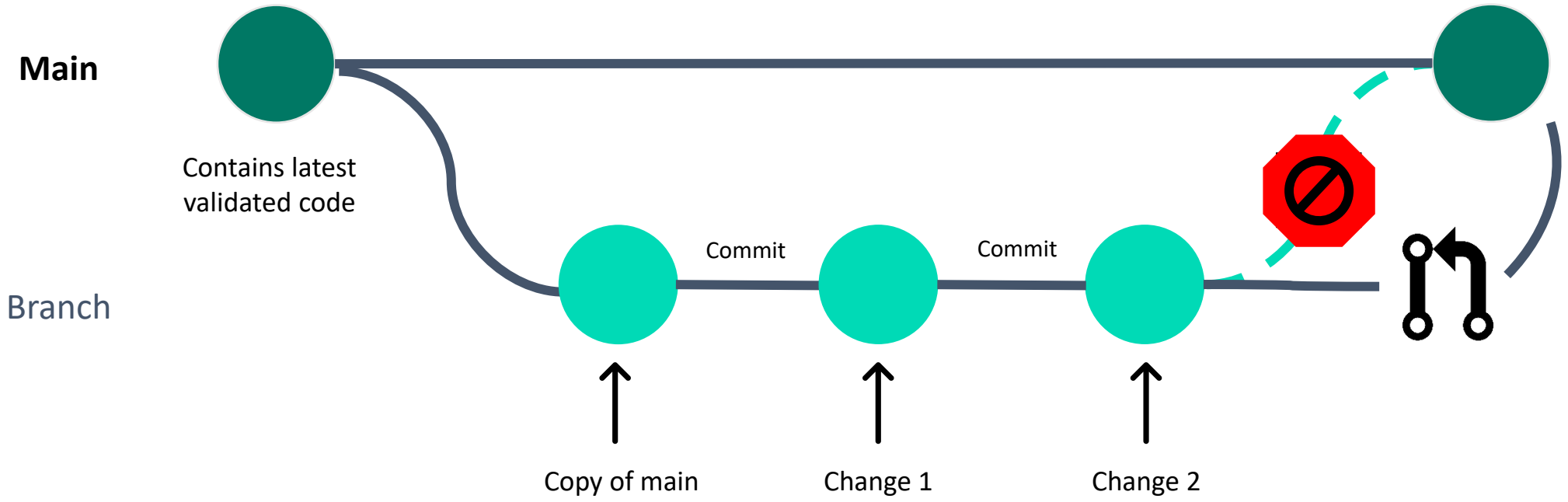
- Small but frequent changes
- Working with multiple people on various aspects of the solution
- Merging back into main branch

Continuous Deployment

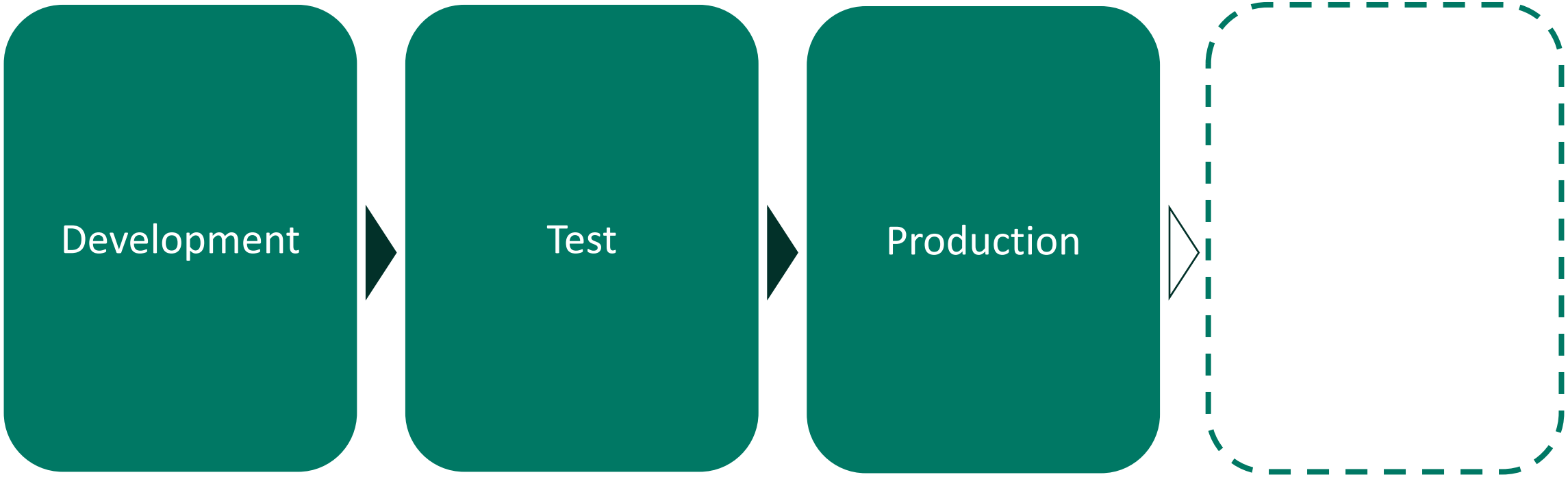
- Working in short cycles (often Agile)
- High frequency updates and releases
- Repeatable deployment process

This all, to improve the developer experience.

Git concept


Main
Contains latest validated code
Branch
Commit
Commit
Copy of main
Change 1
Change 2

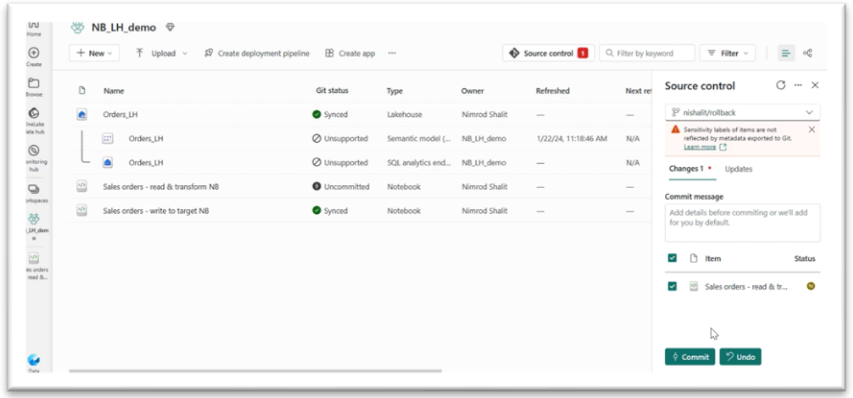
Staged approach



What do we have today?

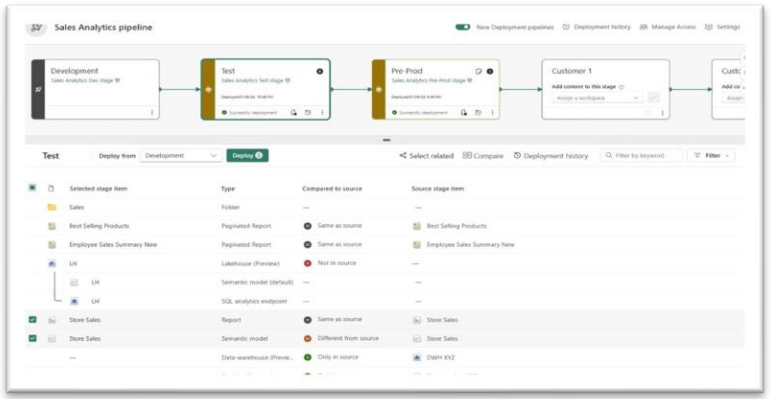
Fabric CI/CD platform

Built-in git integration



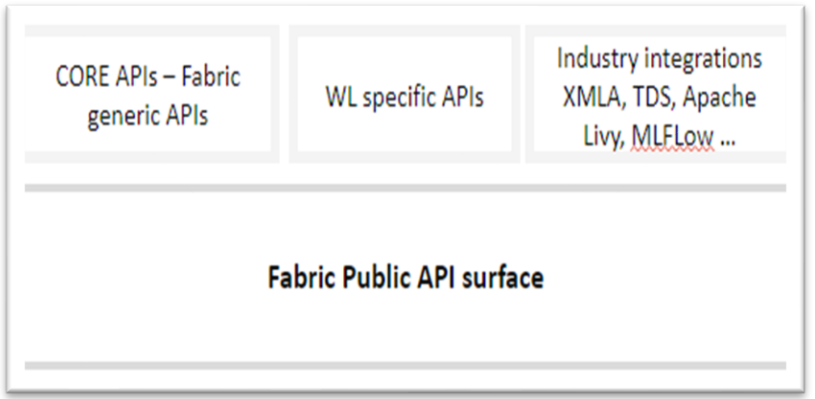
General Available

Deployment pipelines



General Available

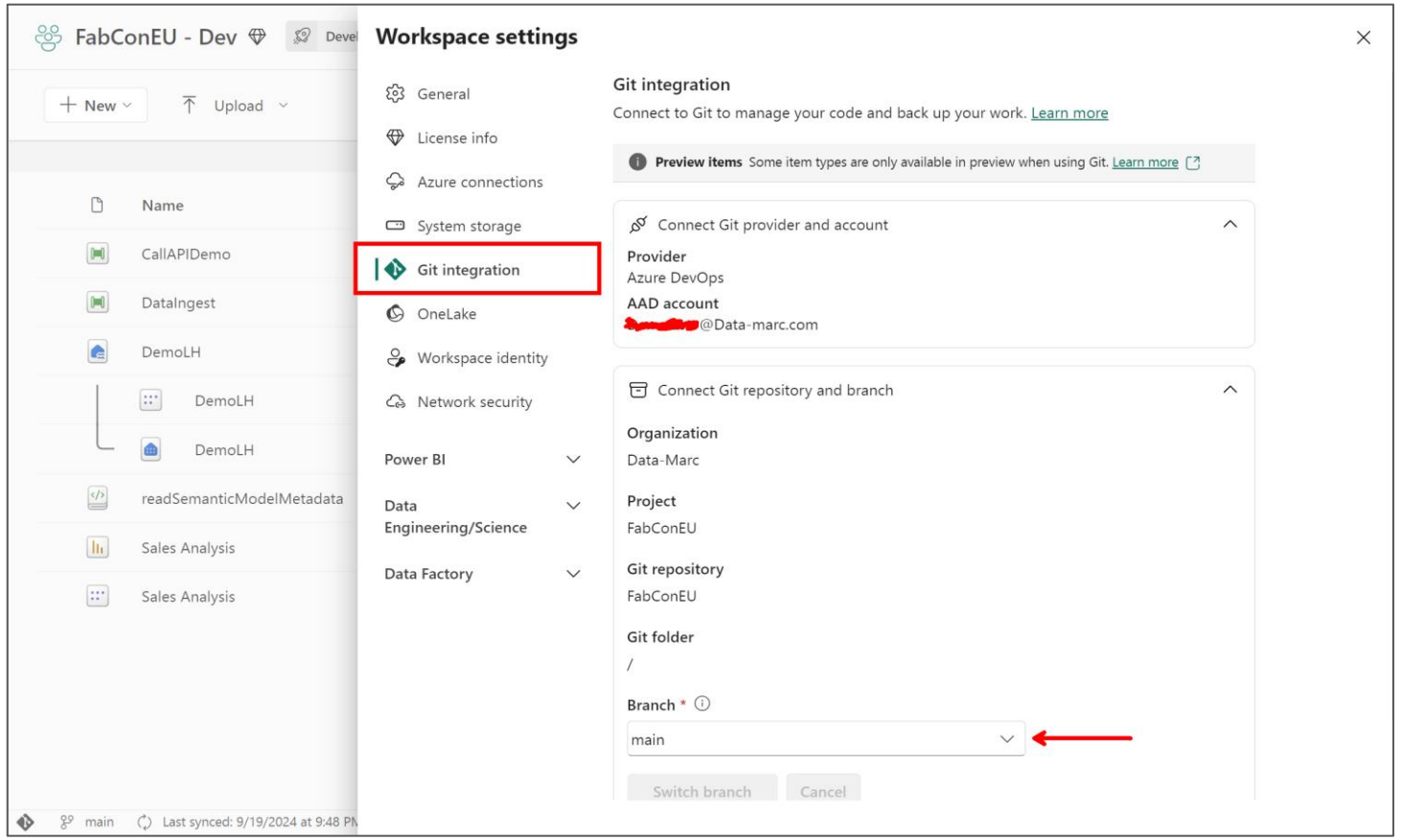
Fabric REST APIs



Public preview

Git integration

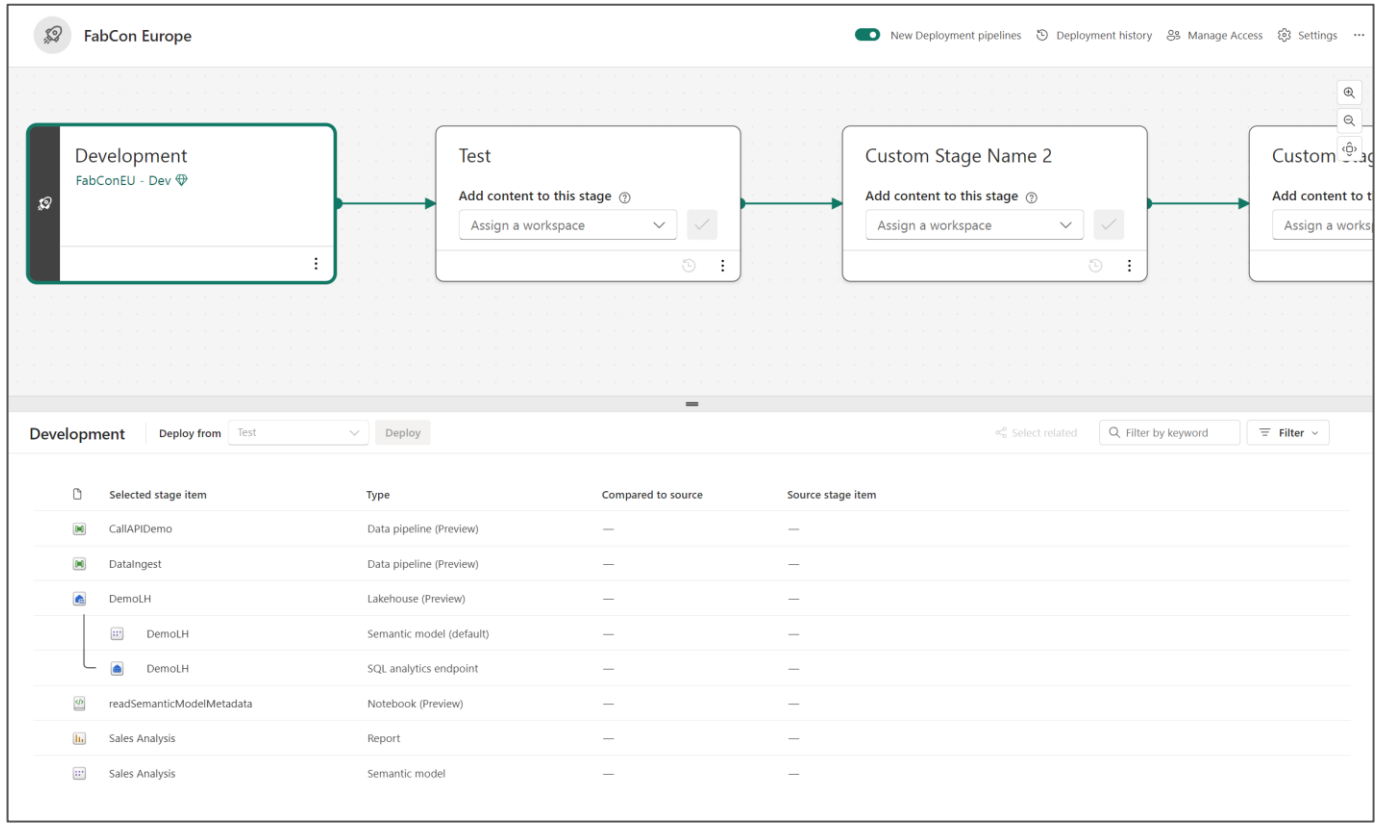
- Sync a Workspace to a Git branch
- Git providers
  - Azure DevOps
  - GitHub New
  - GitHub Enterprise New
- Fabric git APIs – REST APIs & PowerShell samples.
- Manage branches
  - Switch branch
  - Checkout new branch
  - Branch out to new workspace



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



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]

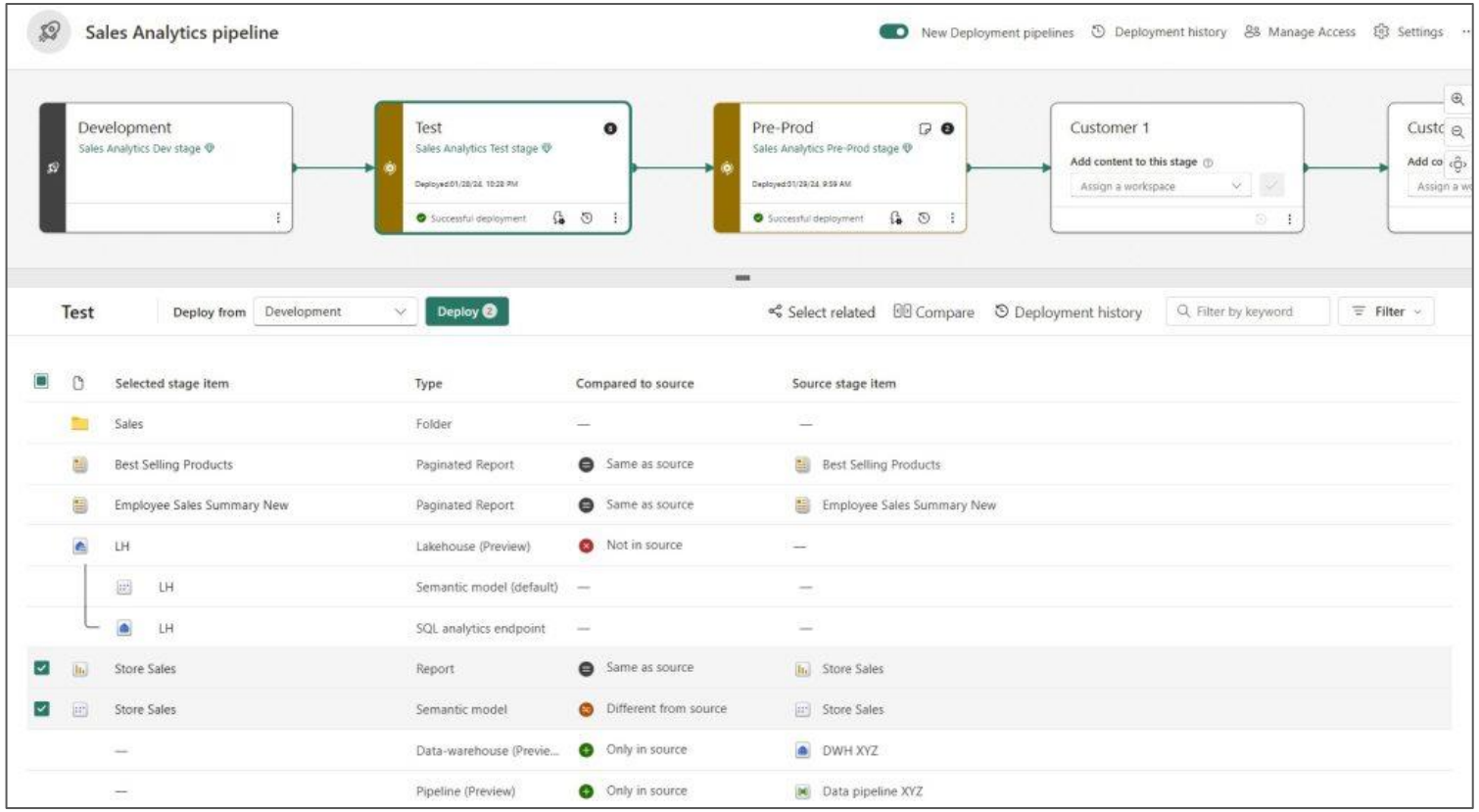
Item type	Create (without definition)	Get	Update	Delete	List
Dashboard	✗	✗	✗	✗	✓
DataPipeline	✗	✓	✓	✓	✓
Datamart	✗	✗	✗	✗	✓
Eventhouse	✓	✓	✓	✓	✓
Eventstream	✓	✓	✓	✓	✓
KQLDatabase	✗	✓	✓	✓	✓
KQLQueryset	✓	✓	✓	✓	✓
Lakehouse	✓	✓	✓	✓	✓
MLExperiment	✓	✓	✓	✓	✓
MLModel	✓	✓	✓	✓	✓
MirroredWarehouse	✗	✗	✗	✗	✓
Notebook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

New announcements

New UI for Deployment Pipelines

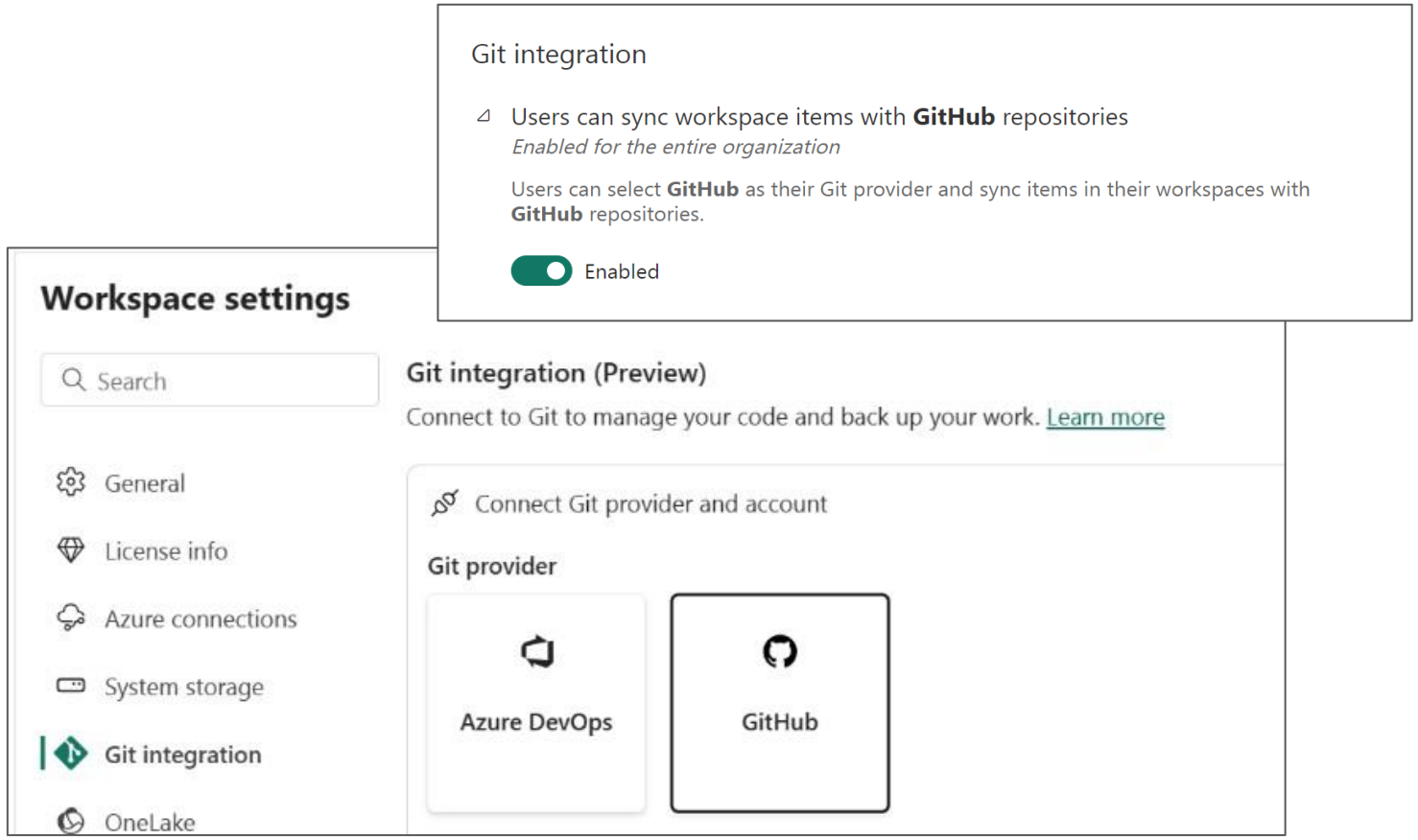
- Switch to enable/disable UI
- Easier navigate
- More focused (per stage)
- Smoother flow
- Folder structure
- Identify unsupported items

You decide what you like best



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



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

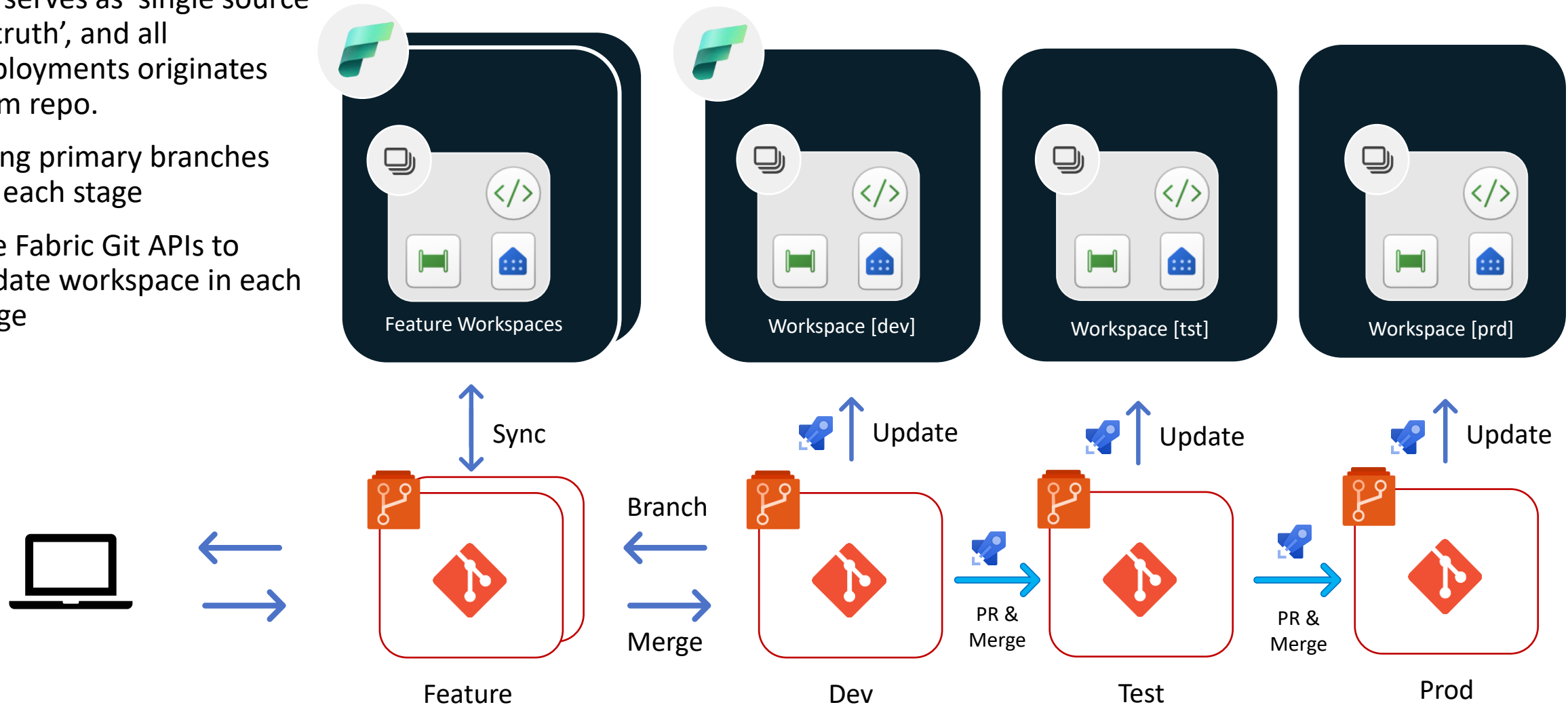
By end of the year

- Dataflow Gen2
- Org App
- Eventhouse
- Metrics Set
- Eventstream
- Mirrored Database
- GraphQL API

Deployment strategies

Scenario 1 – Git based deployments

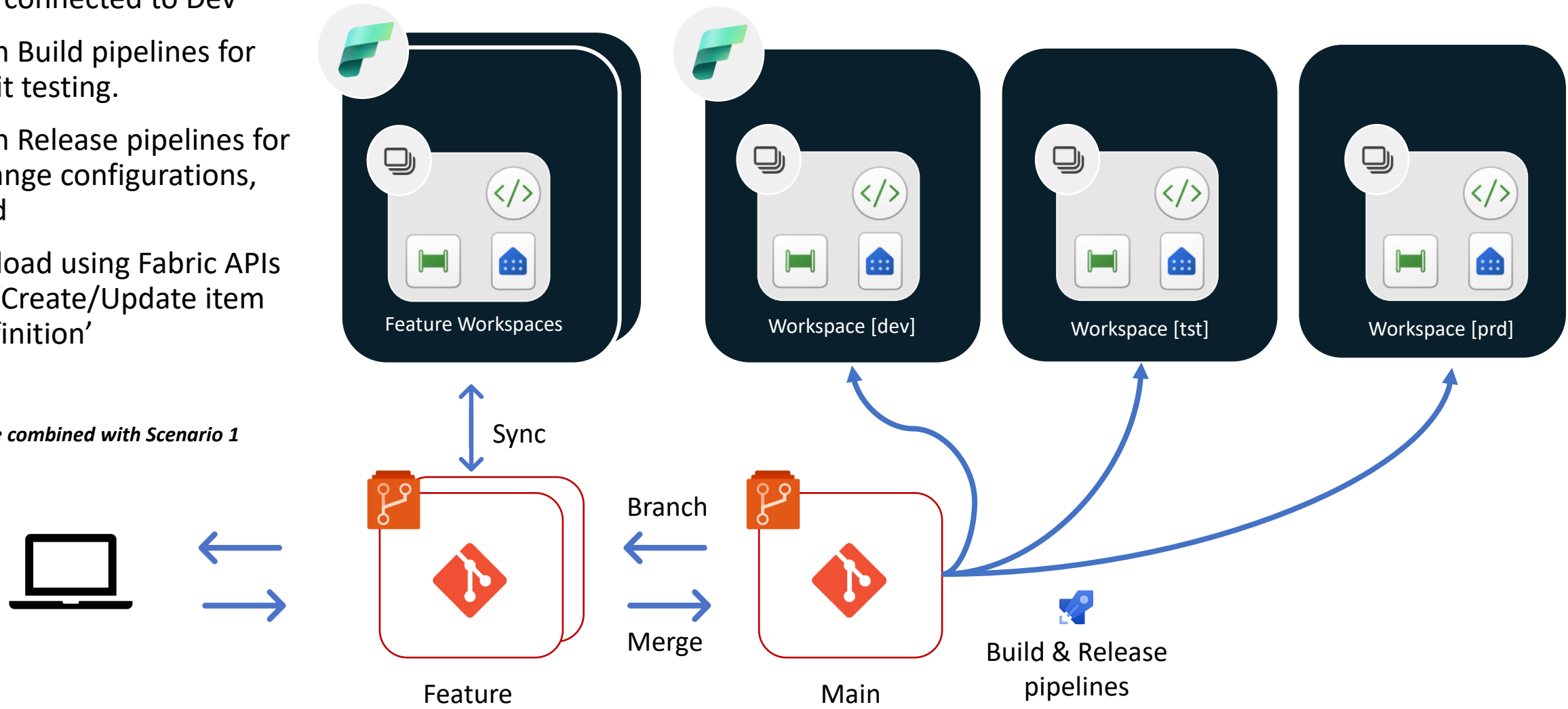
- Git serves as ‘single source of truth’, and all deployments originates from repo.
- Using primary branches for each stage
- Use Fabric Git APIs to update workspace in each stage



Scenario 2* – Git & Build environments

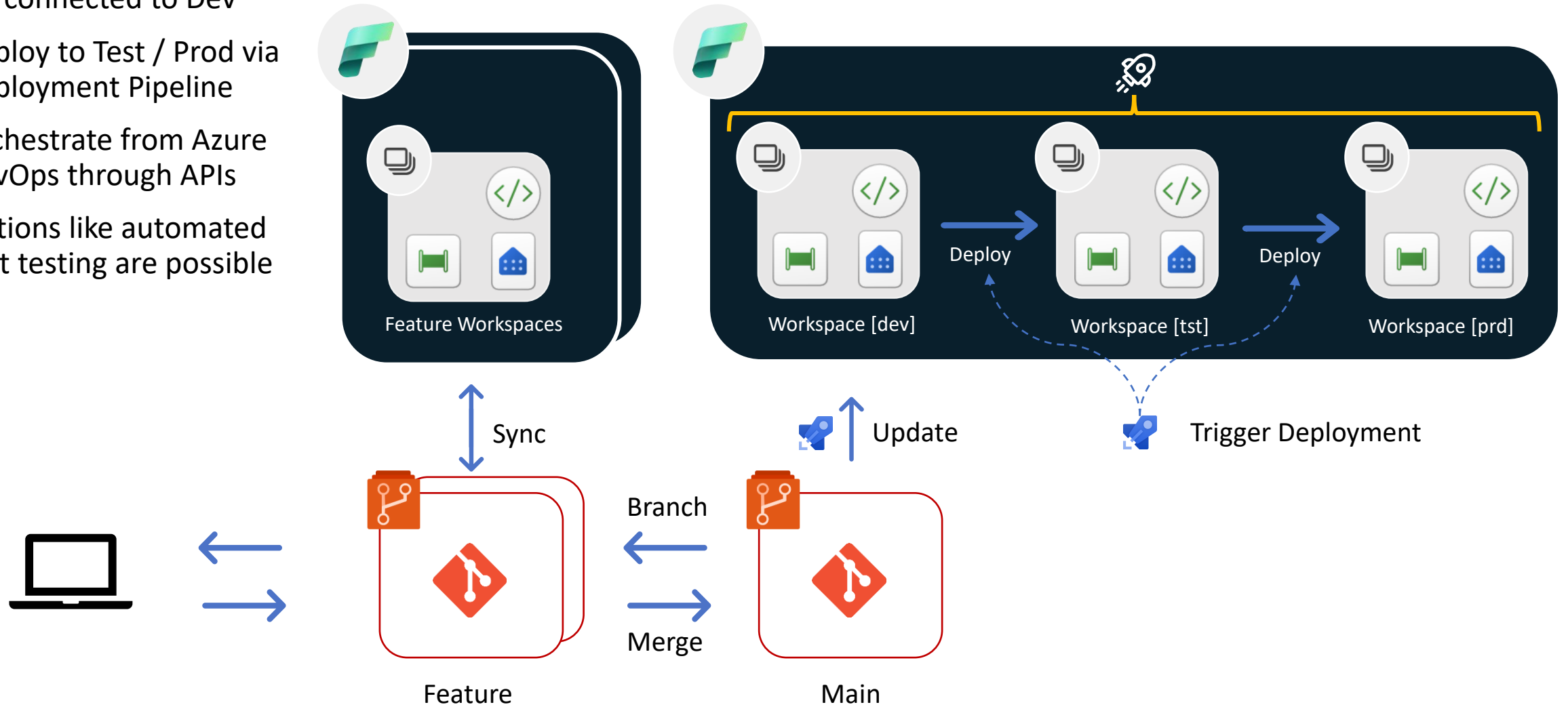
- Git connected to Dev
- Run Build pipelines for Unit testing.
- Run Release pipelines for change configurations, and
- Upload using Fabric APIs to ‘Create/Update item definition’

**** 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?