Microsoft Fabric — One■Repo CI/CD Strategy (Expanded)

1) Purpose

Standardize CI/CD using a single Azure DevOps repo with three top■level folders (Bronze/Silver/Gold) and three pipelines (one per layer). Promote changes Dev → UAT → Prod with approvals and environment■specific bindings.

2) Repo Layout

```
lls-fabric/
  bronze/
  silver/
  gold/
Branches: dev, uat, prod
```

3) Workspace ↔ Branch/Folder Map

Workspace	Repo	Branch	Folder
DEV_Bronze_WS	lls-fabric	dev	/bronze
DEV_Silver_WS	lls-fabric	dev	/silver
DEV_Gold_WS	lls-fabric	dev	/gold
UAT_Bronze_WS	lls-fabric	uat	/bronze
UAT_Silver_WS	lls-fabric	uat	/silver
UAT_Gold_WS	lls-fabric	uat	/gold
PROD_Bronze_WS	lls-fabric	prod	/bronze
PROD_Silver_WS	lls-fabric	prod	/silver
PROD_Gold_WS	lls-fabric	prod	/gold

4) Branching, PRs & Ownership

• Use feature branches per layer (e.g., feat/bronze-123). Only modify that layer's folder. • PRs are branch→branch (no folder picker). Folder■scoped changes make the PR layer■specific naturally. • Branch policies on uat/prod enforce required reviewers by path. • Optional CODEOWNERS ensures right reviewers are auto■requested.

5) Pipelines (3 total — one per layer)

```
trigger:
  branches: [ dev, uat, prod ]
  paths: { include: [ 'bronze/**' ] }

pr:
  branches: [ uat, prod ]
  paths: { include: [ 'bronze/**' ] }
stages:
```

```
- stage: Dev
  jobs:
  - job: DeployBronzeDev
    steps:
    - bash: python deploy_fabric.py --layer bronze --workspace $(DEV_BRONZE_WS_ID) --branch dev --pa
```

6) Real■World Example — Enterprise Parallel Work

```
Alice (Bronze):
  feat/bronze-csv → PR→dev → Bronze Dev stage → PR dev→uat → Bronze UAT stage → PR uat→prod → B
Bob (Silver):
  feat/silver-clean → PR→dev → Silver Dev stage → PR dev→uat → Silver UAT stage → PR uat→prod →
Carol (Gold):
  feat/gold-dashboard → PR→dev → Gold Dev stage → PR dev→uat → Gold UAT stage → PR uat→prod → G
```