Scenario

DataTek, a customer analytics company, has developed a working churn prediction solution (the local codebase you were given). The leadership team now wants to **operationalize this model at scale** so business teams can consume predictions reliably through an API. They also require the solution to be production-grade with versioning, automated retraining, CI/CD, and monitoring, all aligned to **Azure Machine Learning best practices**.

Your task is to transform the existing repo into a full **Azure MLOps pipeline**.

Code is provided (data_utils.py, train.py, predict.py)

Core Requirements

1. **Model Training Pipeline**
    - Adapt the training script so that it runs as an **Azure ML job** on a managed compute cluster.
    - Ensure the job accepts input dataset parameters and outputs a registered model to the **Azure ML Model Registry**.
2. **Model Deployment**
    - Deploy the registered model as an **Azure ML Managed Online Endpoint**.
    - The endpoint should expose a REST API for real-time churn predictions.
    - Ensure deployment follows best practices (staging vs. production deployments, traffic splitting, rollbacks).
3. **CI/CD Integration**
    - Use **GitHub Actions** to build an automated pipeline that:
        - Runs unit tests on every PR.
        - Submits training jobs to Azure ML.
        - On success, registers the new model in the registry.
        - Deploys the model automatically to a **staging endpoint**.
        - Provides a manual approval gate before promoting the model to production.
4. **Monitoring & Alerts**
    - Integrate **Application Insights** or **Azure Monitor** with the online endpoint.

    - Track prediction latencies, error rates, and data drift (feature distributions vs. baseline).
    - Provide an alerting plan (e.g., email or Teams notifications) if drift or high error rates are detected.

5. **Governance & Security**
    - Store sensitive configs (keys, connection strings) in **Azure Key Vault**.
    - Document a strategy for safe rollback in case of faulty deployments.

Deliverables

- **Technical Report/PPT (Max 5 slides)** describing:
    - Overall MLOps architecture on Azure ML.
    - Key design decisions (compute, environments, monitoring).
    - Cost optimization and governance/security considerations.
- **GitHub Repository** containing:
    - Modularized training and scoring scripts (parameterized).
    - GitHub Actions workflow (.github/workflows/mlops.yml).
    - Azure ML job definitions (YAML or Python SDK).
    - Unit tests for training, preprocessing, and inference functions.
    - Instructions in README.md for end-to-end execution.
- **Deployed Online Endpoint**:
    - API endpoint showing prediction response.
- **Monitoring Plan**:
    - Evidence of integrated Application Insights/monitoring/Drift Detection