Multimodal Reasoning AI Agent 🧠

Upload an image and provide a reasoning-based task for the AI Agent. The AI Agent will analyze the image and respond based on your input.

Upload Image

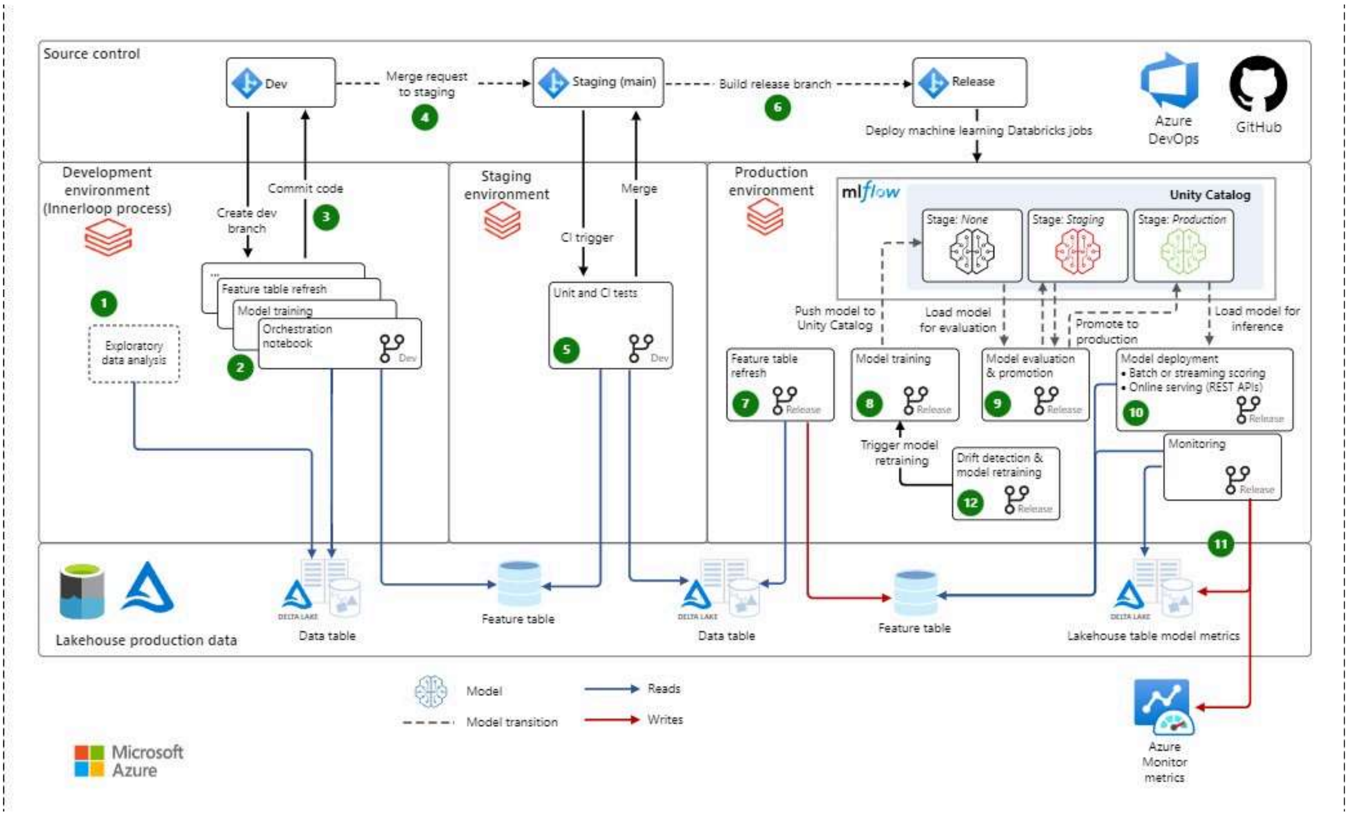
Drag and drop file here

Limit 200MB per file • JPG, JPEG, PNG

Browse files

MLOps.png 77.6KB

×



Uploaded Image

Enter your task/question for the AI Agent:

explain this architecture

Analyze Image

AI Response:

This architecture diagram illustrates a **Machine Learning Operations (MLOps) pipeline** for developing, deploying, and monitoring machine learning models using a lakehouse architecture. It outlines the flow of code, data, and models across different environments, from development to production.

Here's a breakdown of the different stages and components:

1. Source Control (Top Section):

- **Tools:** Azure DevOps and GitHub are used for version control of the code.
- **Workflow:**
  - **(3) Create dev branch:** Developers create branches from the main branch (Staging) to work on new features or bug fixes.
  - **(3) Commit code:** Developers commit their code changes to their respective development branches.
  - **(4) Merge request to staging:** Once development is complete, a merge request is created to merge the development branch into the staging (main) branch.
  - **(6) Build release branch:** When ready for production deployment, a release branch is created from the staging branch.

2. Development Environment (Inner Loop Process - Left Section):

- **Purpose:** This is where data scientists and engineers develop and iterate on models and features.
- **Steps:**
  - **(1) Exploratory data analysis:** Initial investigation and understanding of the data.
  - **(2) Feature table refresh, Model training, Orchestration notebook:** This step encompasses several key activities:
    - **Feature Table Refresh:** Updating the feature tables with the latest data.
    - **Model Training:** Training machine learning models using the refreshed features.
    - **Orchestration Notebook:** Using a tool like Databricks Notebooks to orchestrate the feature engineering and model training processes.
- **Data Interaction:** The development environment reads data from the "Lakehouse production data" (Data table) and writes to "Feature table".

3. Staging Environment (Middle Section):

- **Purpose:** A pre-production environment used for testing and validation before deploying to production.
- **Process:**
  - **Merge:** Code from the development branch is merged into the staging branch.
  - **CI trigger:** This merge action triggers a Continuous Integration (CI) pipeline.
  - **(5) Unit and CI tests:** Automated tests are run to ensure code quality and functionality.
- **Data Interaction:** The staging environment might interact with a separate set of data or a subset of the production data for testing.

4. Production Environment (Right Section):

- **Purpose:** The live environment where the machine learning models are deployed and used for inference.
- **Key Components:**
    - **MLflow:** A platform for managing the ML lifecycle, including tracking experiments, packaging code into reproducible runs, and deploying models.
    - **Unity Catalog:** A unified governance solution for data and AI assets in Databricks, used here for model management.
- **Workflow:**
    - **(6) Deploy machine learning Databricks jobs:** The release branch is deployed to the production environment, likely using Databricks jobs for scheduled or triggered execution.
    - **Model Registry (Unity Catalog):**
        - **Stage: None:** The initial stage of a trained model.
        - **Push model to Unity Catalog:** The trained model is registered in Unity Catalog.
        - **Load model for evaluation:** The model is loaded into the staging stage within Unity Catalog for further assessment.
        - **Stage: Staging:** The model resides in the staging stage, indicating it's under evaluation.
        - **Promote to production:** If the model passes evaluation, it's promoted to the production stage.
        - **Stage: Production:** The model is now the active version for inference.
        - **Load model for inference:** The production model is loaded for serving predictions.
    - **(7) Feature table refresh:** Feature tables in the production environment are refreshed.
    - **(8) Model training:** Scheduled or triggered retraining of the model using the latest production data.
    - **(9) Model evaluation & promotion:** Evaluating the retrained model and promoting it to production if it performs better.
    - **(10) Model deployment:** Deploying the model for various inference scenarios:
        - **Batch or streaming scoring:** Processing data in batches or continuously.
        - **Online serving (REST APIs):** Providing real-time predictions through APIs.
    - **(12) Drift detection & model retraining:** Monitoring the model's performance for data drift or concept drift, and triggering retraining when necessary.

5. Data Layer (Bottom Section):

- **Lakehouse production data:** The central repository for all production data, likely built on a technology like Delta Lake on a data lake.
- **Data table:** Raw or processed data within the lakehouse.
- **Feature table:** Precomputed features used for model training and inference. These are often optimized for performance.

- **Lakehouse table model metrics:** A table storing metrics related to the performance of the deployed models.
- **Data Flow:**
    - Blue lines indicate reading data.
    - Red lines indicate writing data.

6. Monitoring (Bottom Right):

- **(11) Monitoring:** Continuously monitoring the performance of the deployed models and the overall system.
- **Azure Monitor metrics:** A cloud-based monitoring service used to collect and analyze performance data.

Key Technologies:

- **Azure DevOps/GitHub:** For source control and CI/CD.
- **Databricks:** A unified analytics platform often used for data engineering, data science, and machine learning, likely hosting the notebooks, model training, and deployment jobs.
- **MLflow:** For managing the machine learning lifecycle.
- **Unity Catalog:** For data and AI governance, including model registry.
- **Delta Lake:** An open-source storage layer that brings ACID transactions to Apache Spark and big data workloads, enabling the lakehouse architecture.
- **Azure Monitor:** For monitoring the deployed application and infrastructure.

In Summary:

This architecture represents a robust and scalable MLOps pipeline that emphasizes automation, collaboration, and continuous improvement. It leverages the capabilities of a lakehouse architecture and various Azure services to streamline the development, deployment, and monitoring of machine learning models in a production environment.