Documentació de casos d'ús

Guia per desenvolupar casos d'ús de GenAI

CTTI

10 abril 2025 | Versió del document 1.00

Finançat per
Unió Europea Fons Europeu Next Generation
Gobierno de España
Plan de Recuperación, Transformación y Resiliencia
Next Generation Catalunya
Generalitat de Catalunya

Taula de continguts:

Finançat per
Unió Europea Fons Europeu Next Generation
Gobierno de España
Plan de Recuperación, Transformación y Resiliencia
Next Generation Catalunya
Generalitat de Catalunya

Llista d'abreviatures

Abbreviation	Meaning
GenAI	Intel·ligència Artificial Generativa (*Generative Artificial Intelligence*)
MLOps	Operacions de Machine Learning
LLMOps	Operacions de Models de Llenguatge Grans
LLM	Model de Llenguatge Gran
CI	Integració continua
CD	Desplegament continu
DES / PRE / PRO	Desenvolupament / Preproducció / Producció
RAG	Generació Augmentada de Recuperació
DAB	Databricks Asset Bundles
PR	Pull Request
SLA	Service Level Agreement
Workflow	Flux de treball
Workspace	Espai de treball
Features	Característiques
Schedule	Programació

Història documental

Assumpte	Data	Comentaris	Pàgines afectades
0.0.1	26/09/2024	Original	Totes
0.0.2	03/02/2025	Cookiecutter ja no és necessari. Totes les variables provenen del fitxer databricks.yaml.	Totes
0.0.3	06/03/2025	Actualitzat CI/CD amb documentació SIC+	Apartat CI/CD

Finançat per
Unió Europea Fons Europeu Next Generation
Gobierno de España
Plan de Recuperación, Transformación y Resiliencia
Next Generation Catalunya
Generalitat de Catalunya

1. Introducció

La següent documentació proporciona una guia completa per implementar un xatbot amb integració de Generació Augmentada de Recuperació (RAG), i càrrega de documentació en temps real, allotjat a Azure Databricks. Seguint les millors pràctiques de Databricks per a científics de dades, enginyers de ML i equips de DevOps que busquen racionalitzar tot el cicle de vida dels models GenAI, es detalla un procés de desplegament estructurat des dels entorns de desenvolupament fins a la producció.

1.1. Visió general de l'arquitectura aplicada a LLMOps

La gestió del cicle de vida d'un **cas d' ús de GenAI** requereix organitzar i mantenir diferents entorns. Aquests entorns ajuden a estructurar el desenvolupament, les proves i la implementació de models LLM alhora que garanteixen l'escalabilitat, la governança i el control de costos. L'arquitectura definida a Azure Databricks permet els punts següents.

- **Segmentació dels entorns**
    - El **desenvolupament**, les **proves** i la posada en **producció** se separen en entorns diferents: **desenvolupament (DES)**, **preproducció (PRE)** i **producció (PRO)**.
    - Això ajuda a **aïllar els problemes** abans de la implementació a producció.
- **Gestió del cicle de vida del model**
    - **Repositori de codi i pipelines de CI/CD:** El proveïdor utilitzarà un repositori per al projecte o cas d'ús, que tindrà configurades les *pipelines* de CI/CD que permetran fer el pas entre entorns.
    - **Entorn de desenvolupament (DES):** Dins de l'entorn de DES, el proveïdor desenvoluparà el model de ML a l'espai de treball (workspace) corresponent al domini d'informació del projecte o cas d'ús.
    - **Entorn de preproducció (PRE):** Una vegada el model està desenvolupat, s'utilitzen les *pipelines* de CI/CD per fer les proves sobre l'entorn de PRE. Les proves unitàries es realitzaran sobre les *pipelines* de CI/CD i les proves d'integració s'executaran sobre l'espai de treball (workspace)de l'entorn de PRE corresponent al domini d'informació del projecte o cas d'ús. En aquest entorn es realitzaran totes les validacions necessàries abans de passar a producció.
    - **Entorn de producció (PRO):** Un cop passades les proves i les validacions, s'utilitzen les *pipelines* de CI/CD per passar el codi a PRO, dins l'espai de treball corresponent al domini d'informació del projecte o cas d'ús.
- **Gestió de Costos i Recursos**
    - Cada **proveïdor** tindrà un espai de treball separat a l'entorn de **Desenvolupament** per garantir un seguiment clar dels costos.
    - Els espais de treball estan alineats amb els dominis d'informació de la Generalitat de Catalunya, garantint una governança eficient i una òptima utilització dels recursos.

Aquest enfocament estructurat garanteix que els projectes GenAI segueixin les millors pràctiques **d'escalabilitat, seguretat i governança,** alhora que dóna suport a una transició suau del **desenvolupament al desplegament**.

1.2. Visió general de LLMOps

Per tal d'executar i estandarditzar els processos de LLMOps, s'ha desenvolupat un codi font que servirà com a plantilla i que estarà accessible en un repositori.

Aquest document serveix com a manual per als professionals que busquen aprofitar els models de GenAI,centrant-se en la utilització d'extrem a extrem del codi font proporcionat, des del moment en què es clona del repositori fins al desplegament.

Aquesta guia proporciona instruccions pas a pas per a:

- Configurar l'entorn, cobrint totes les dependències, configuracions i permisos necessaris.
- Adaptar la plantilla de codi per satisfer les necessitats específiques dels casos d'ús mitjançant **Databricks Asset Bundles**.
- Desplegament dels models en entorns de preproducció i producció, automatització de *pipelines* per a la integració contínua (CI) i el desplegament continu (CD).
- Establir processos de seguiment i avaluació per garantir que els models es mantinguin robustos i fiables al llarg del temps.

Aquesta guia té com a objectiu explicar el procés d'integració dins de la plataforma LLMOps, permetent als usuaris construir, desplegar i mantenir solucions ML escalables i d'alt rendiment amb facilitat, a partir d'un codi de plantilla. El proveïdor només gestionarà l'espai de treball de DES. Els altres entorns i espais de treball (PRE i PRO) són gestionats pel proveïdor de la PTD, i les *pipelines* de CI/CD pel SIC+.

A continuació es mostra el diagrama amb els passos a seguir per a desenvolupar, testejar i desplegar un projecte. Aquest procés de LLMOps pot estar subjecte a canvis, per exemple, es preveu l'ús del SIC+ o la lectura de dades de PRO.

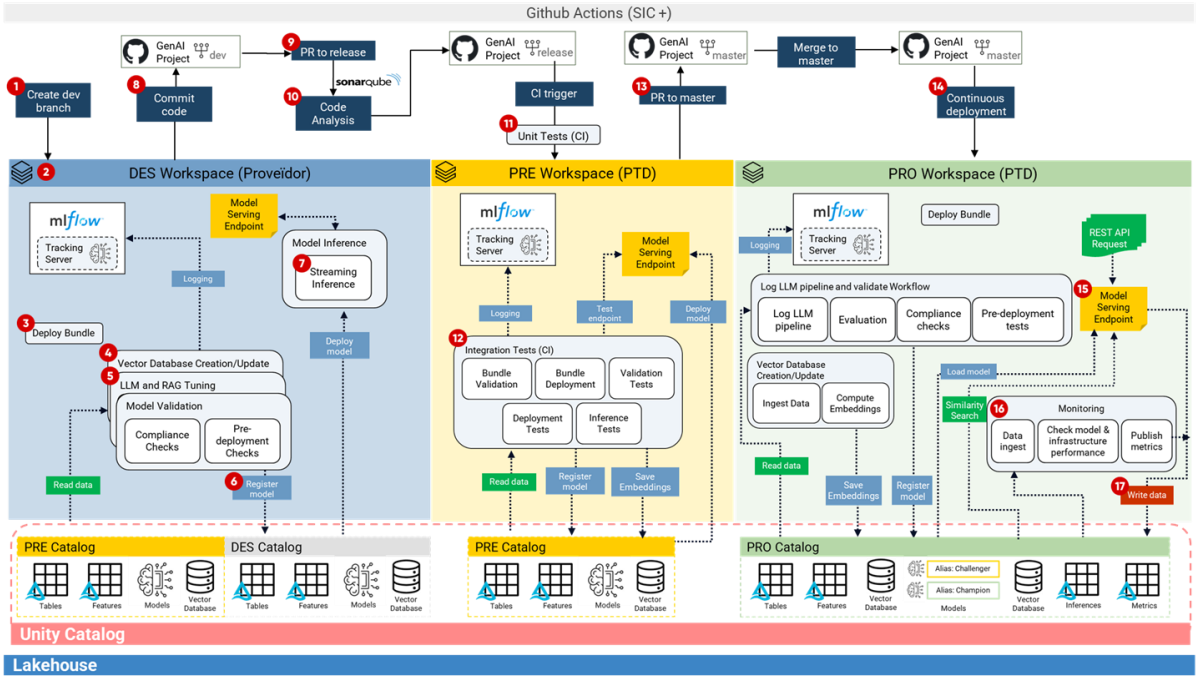

Figura 1 – Flux de treball LLMOps entre entorns Proveïdor DES i PTD PRE i PRO

Finançat per
Unió Europea Fons Europeu Next Generation
Gobierno de España
Plan de Recuperación, Transformación y Resiliencia
Next Generation Catalunya
Generalitat de Catalunya

La plantilla cobreix els elements més comuns per construir un xatbot amb la incorporació d'una base de coneixement de documents i RAG per millorar el rendiment del model. El cas del xatbot RAG té dues línies de treball principals.

- **Processament i ingesta de documents:** Descriu els passos per processar documents no estructurats (tipus .pdf, .docx, .md i .txt) i ingerir-los en una base de dades vectorial.
- **Construcció de cadenes per a interaccions i inferències de models:** Descriu el procés de construcció de la cadena d'interacció i inferència per permetre la integració i el funcionament perfectes dels models GenAI.

A continuació, trobareu els passos detallats per treballar per replicar o adaptar aquesta solució a les vostres necessitats:

Pas	Descripció
1	Cloneu la plantilla des del repositori a la vostra branca de desenvolupament. Modifiqueu el codi per alinear-lo amb el cas d'ús específic. Trobareu mes detall a l'apartat **Primers passos.**
2	Assegureu-vos que l'espai de treball estigui configurat d'acord amb les directrius. Autenticar la CLI del Databricks per habilitar l'accés. Seguiu els passos de les seccions 2.1.1 Databricks CLI i 2.1.2Autenticació del Databricks
3	En aquest pas, configureu el codi de la plantilla DAB en funció del cicle de vida del model. Implementeu-lo a l'espai de treball de DES juntament amb el flux de treball (workflow) i els recursos configurats. Per a més detalls, consulteu **Recursos del Databricks.** ***Nota**: A l'entorn DES, és probable que aquest pas es repeteixi diverses vegades a mesura que itereu en els fluxos de treball. Els passos següents descriuen les tasques necessàries per a cada iteració del flux de treball.*
4	Comenceu per establir el corpus de documentació reunint tots els documents i continguts rellevants que voleu emmagatzemar i recuperar posteriorment. Assegureu-vos que la documentació estigui en un format estructurat per que es pugui processar (e.g., text files, PDFs, etc.). A continuació, definiu el procés de *chunking* i *embedding*. Finalment, carregueu les *embeddings* de vectors a la vostra base de dades vectorial per indexar-los i poder-los cercar.

Pas	Descripció
5	Aquest pas és fonamental i dóna forma al rendiment de tota la pipeline RAG. Elaboreu una sol·licitud i/o una cadena per guiar la recuperació i la generació, incorporant context. Implementeu mesures similars a la base de dades vectorial per permetre una recuperació eficient de dades. Ajusteu els llindars i els paràmetres per obtenir resultats rellevants. Enriquiu les dades recuperades amb informació addicional. Milloreu la comprensió i la precisió. Aprofiteu les dades i el context recuperats per generar una resposta LLM. Assegureu-vos de l'alineació amb el format indicat i desitjat. Finalment, proveu el sistema amb consultes variades. Refineu les sol·licituds basades en els resultats per optimitzar la precisió.
6	Quan el model funciona bé i ha superat el pas d'avaluació, podeu registrar la nova versió del model (si el model ja existia) al catàleg DES.
7	En aquest pas, heu de configurar el vostre *Serving Endpoint* del Databricks[1]) per tal d'exposar el vostre model a l'accés extern.
8	Actualitzeu el codi del paquet amb la configuració específica de la base de dades vectorial, l'enginyeria de prompts, la inferència i la lògica de validació. A continuació, envieu el codi a la branca de desenvolupament del vostre repositori sol·licitat a SIC+, detallat a **Git Repositori & CI/CD Pipelines.**
9	Quan estigueu preparats per a realitzar les proves, envieu el codi a la branca de llançament mitjançant una sol·licitud d'extracció (PR) a la plataforma CI/CD.
10	Utilitzeu la *pipeline* del flux de treball detallat a la secció **Git Repositori & CI/CD Pipelines** per realitzar l'anàlisis de codi mitjançant SonarQube i les proves.
11	En aquest pas s'executen les proves unitàries configurades prèviament. ***Nota**: Si us plau, trobeu més detalls sobre les proves unitàries a la secció* **Proves unitàries**

[1] Més detalls sobre el punt final de servei del model Databricks a Crear punts finals de servei de models personalitzats - Azure Databricks | Microsoft Learn

Pas	Descripció
12	També s'executen proves d'integració prèviament configurades en DES. Aquestes proves s'executen abans de desplegar el paquet l'espai de treball de PRE des de la PTD. Comproveu amb el proveïdor de la PTD els requisits per migrar a PRE i adapteu el codi en conseqüència (especialment variables en el fitxer databricks.yml). ***Nota**: Si us plau, trobeu més detalls sobre les proves d'integració a la secció* **Proves d'integració**
13	Després de l'execució correcta de totes les proves i validacions en el pipeline PRE CI/CD, s'ha d'aprovar la *Pull Request (PR)* cap a la branca master i el codi del paquet es fusiona amb la branca *master*.
14	Comproveu amb el proveïdor de la PTD els requisits per migrar a Anàlisi SonarQube**:** Utilitza SonarQube per analitzar el codi a la recerca d'errors, vulnerabilitats i "code smells" (indicadors de problemes en el codi). Utilitza SonarQube de CTTI o la teva pròpia instància. • **Unit Testing**: Executa proves unitàries amb *pytest* per validar el funcionament correcte de funcions i mòduls individuals de manera aïllada. • **Proves d'integració:** Després del desplegament, es poden executar proves d'integració per assegurar que els diferents components funcionen correctament junts en l'entorn de preproducció. Aquestes proves s'han de dissenyar i afegir segons les necessitats específiques. Producció (PRO) i adapteu el codi en conseqüència (especialment variables en el fitxer databricks.yml). El proveïdor de la PTD allibera el codi validat i el desplega a l'espai de treball de PRO des de PTD. Després de la migració de codi amb èxit, el paquet es desplega a l'espai de treball de PRO.
15	En aquest pas, és important configurar el ***Model Serving Endpoint*** per complir els *Service Level Agreement (*SLA*)* de l'entorn PRO (per exemple, latència, escalabilitat).
16	Quan està en producció, és important controlar el rendiment del model. Algunes mètriques s'emmagatzemen per defecte i podeu definir altres mètriques que preferiu.
17	Analitzeu les mètriques del pas anterior per garantir el rendiment i l'estabilitat del model a l'entorn de producció (per exemple, totes les inferències del model, mètriques de rendiment).

Finançat per
Unió Europea Fons Europeu Next Generation
Gobierno de España
Plan de Recuperación, Transformación y Resiliencia
Next Generation Catalunya
Generalitat de Catalunya

2. Primers passos

2.1. Prerequisits

Aquest projecte té diversos requisits previs essencials per al seu correcte funcionament. Com que la plataforma està desenvolupada dins de Databricks, que executa **fluxos de treball** i tasques automatitzades en *notebook*s de Python, tenir Python instal·lat és crucial. A més, per desplegar recursos a Databricks, necessitareu accés a la CLI (*command-line interface*) de Databricks, així com els permisos d'Azure adequats per interactuar amb els **espais de treball** de Databricks necessaris.

2.1.1 Databricks CLI

Per a aquest projecte, instal·leu la versió 0.205.0 o posterior de la CLI del Databricks. Suggerim dos enfocaments, però altres es poden trobar a Install or update the Databricks CLI - Azure Databricks | Microsoft Learn.

Windows:

Des del símbol del sistema, executeu les dues ordres *winget* següents per instal·lar la CLI i, a continuació, reinicieu el símbol del sistema:

winget search databricks

winget install Databricks.DatabricksCLI

Per finalitzar correctament la instal·lació, reinicieu el terminal on esteu treballant. Confirmeu si la CLI de Databricks està instal·lada correctament. Per fer-ho, visualitzeu la versió de l'executable de la CLI de Databricks utilitzant l'opció -v o executant l'ordre version.:

databricks -v

Linux:

Comenceu instal·lant el Subsistema de Windows per a Linux (WSL) des de la Microsoft Store, específicament la versió Ubuntu 22.04.3 LTS. Un cop configurat WSL, utilitzeu les ordres següents per instal·lar la CLI de Databricks. En aquest document, la instal·lació es realitza mitjançant Homebrew:

brew tap databricks/tap

brew install databricks

Si Homebrew encara no està instal·lat en el vostre entorn WSL, podeu instal·lar-lo utilitzant l'ordre següent:

/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"

Després de la instal·lació, assegureu-vos que la CLI del Databricks estigui actualitzada:

brew upgrade databricks

2.1.2 Autenticació del Databricks

A continuació, autentiqueu la CLI del Databricks per habilitar la implementació de recursos a l'àrea de treball del Databricks de desenvolupament designada.

Comenceu generant un token d'accés personal. Navegueu al vostre **espai de treball** de Databricks i feu clic, a l'extrem superior dret on es troba el vostre usuari. (Figura 2 – Configuració d'usuari de la interfície d'usuari de l' espai de treball de Databricks)

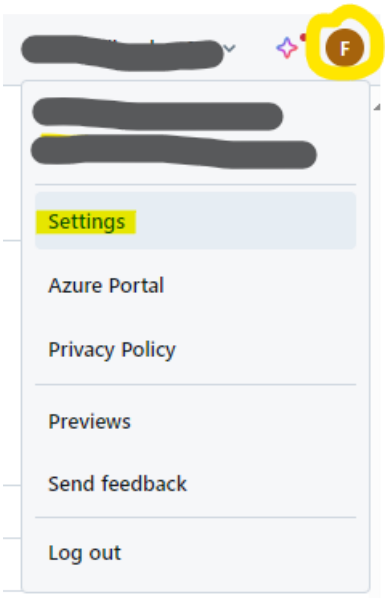

Figura 2 – Configuració d'usuari de la interfície d'usuari de l'espai de treballde Databricks.

Després, feu clic a **Configuració**, seleccioneu **Desenvolupador** i després **Tokens d'accés**, i feu clic a **Gestionar**. Creeu un nou token i emmagatzemeu-lo de manera segura. (Figura 3 -)

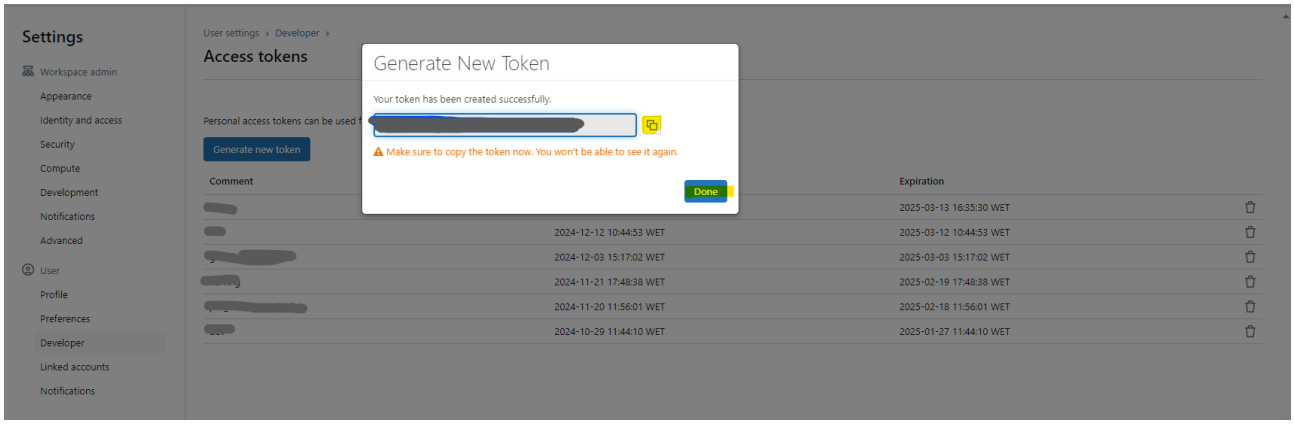

Figura 3 - Generació d'un nou token a la interfície d'usuari de Databricks

Al vostre terminal, executeu l'ordre següent per configurar la CLI del Databricks:

databricks configure

Quan se us demani, hauríeu de tenir la informació següent:

- **Databricks Host** - correspon a l'adreça URL de l'espai de treball del Databricks.
- ***Token* d'accés personal** - proporcioneu el *token* d'accés personal que acabeu de generar per al vostre usuari.

Per confirmar que el vostre perfil s'ha creat correctament, executeu el codi següent:

databricks auth profiles

Assegureu-vos també de validar el vostre perfil amb YES:

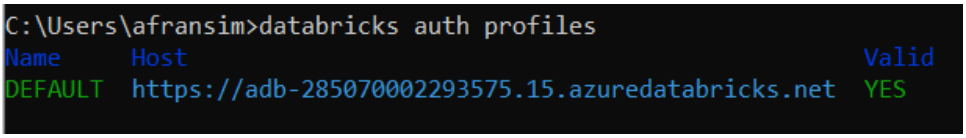

Figura 4 – Perfil, per defecte, de Databricks validat l'espai de treball

Això completa el procés de configuració, permetent el desplegament i la gestió sense interrupcions dels recursos de Databricks. Després, tindreu el perfil de configuració corresponent afegit al vostre fitxer .databrickscfg i a partir d'aquí podreu desplegar els recursos de Databricks a l'espai de treball de destinació.

Per obtenir informació més detallada, consulteu https://learn.microsoft.com/en-us/azure/databricks/dev-tools/auth/azure-cli.

2.2. Instruccions de configuració

Després de configurar els requisits previs del mòdul anterior, ara podem fer un seguiment de la configuració del projecte de plantilla GenAI per crear el vostre cas d'ús personalitzat. El primer pas és clonar el repositori de plantilles.

Per sol·licitar accés a aquest repositori, contacteu amb ACOCLDSIC:

- S'ha de demanar accés a través de **JIRA** amb el servei de **ACOCLDSIC**
- Repositori:https://github.com/ctti-dev/3670.00-sta-genai_template-databricks/tree/feature/template

Per accedir al repositori de plantilles:

- Els usuaris han de tenir el correu electrònic de GICAR.
- Aneu a https://github.com/ctti-dev/3670.00-sta-genai_template-databricks/tree/feature/template.

Després d'obtenir els permisos d'accés necessaris, podeu clonar el repositori de plantilles a la vostra màquina local.

1. Copieu l'URL del repositori proporcionat per a la plantilla.
2. Navegueu al directori desitjat a la vostra màquina local on voleu clonar el repositori.
3. Executeu la següent ordre al terminal:

   git clone https://github.com/ctti-dev/3670.00-sta-genai_template-databricks/tree/feature/template

   En cas que trobeu un error de certificat SSL, podeu solucionar-lo temporalment amb l'ordre:

git -c http.sslVerify=false clone https://github.com/ctti-dev/3670.00-sta-genai_template-databricks/tree/feature/template

Nota:

- *L'URL del repositori proporcionat està destinat únicament a accedir a la plantilla.*
- *IMPORTANT: Durant el desenvolupament del vostre cas d'ús específic, l'equip de casos d'ús ha de crear i utilitzar el seu propi repositori dedicat.*

2.3. Estructura del codi

L'estructura del codi segueix l'enfocament de Databricks MLOps Stacks, navegueu a MLOps Stacks: model development process as code - Azure Databricks | Microsoft Learn com a referència. Després de clonar el repositori de plantilles GenAI, el directori hauria de tenir la carpeta arrel *sta-{use_case}* i organitzar-se de la següent manera:

- **.gitignore:** Aquest fitxer especifica quins fitxers i directoris Git ha d'ignorar quan fa el seguiment dels canvis. Normalment, inclou fitxers temporals, dades sensibles, registres i configuracions d'entorn que no s'han de versionar.

- **README.md:** L'arxiu de documentació principal del projecte. Proporciona una visió general del projecte, instruccions de configuració, directrius d'ús i qualsevol altra informació rellevant per als usuaris i col·laboradors.
- **test-requirements.txt:** Una llista de dependències específicament necessàries per executar proves d'integració i unitàries. Garanteix que s'instal·lin les biblioteques i eines correctes per validar la funcionalitat del projecte.

- **sta-chatbot:** El directori de la plantilla que es reemplaça pel nom real del projecte. Inclou tots els fitxers i subdirectoris específics per als Databricks Asset Bundles.
  - **assemble_application/:** Aquest directori conté els *scripts* i les eines relacionades amb la creació d'una instància del xatbot al Unity Catalog i la seva inscripció a MLflow.
    1. **Assemble_Application.py:** Aquest *notebook* automatitza el muntatge de l'aplicació del xatbot, especificant algunes de les configuracions de RAG i registrant el model a MLflow, així com registrant-lo al Databricks Unity Catalog.
    2. **chain_history.py:** Fitxer de Python que conté les funcions d'ajuda de la tasca de l'aplicació Assemble, així com la lògica per implementar l'historial de xat per a RAG.
  - **create_objects/:** Aquest directori conté els *scripts* per crear els objectes de dades necessaris a les ubicacions d'emmagatzematge adequades.
    1. **notebooks/CreateObjects.py:** Aquest *notebook* crea objectes de dades com els esquemes de bronze, plata i or; taules de prova, seguiment i *chunk*s; i el volum extern per contenir els fitxers utilitzats per construir la base de dades vectorial.
    2. **utils.py:** Aquest fitxer conté les funcions d'ajuda que es criden durant l'execució del notebbok anterior.
  - **build_document_index/:** Aquest directori conté els *script*s per accedir i preparar les dades per utilitzar-les amb l'accelerador de bot QA.
    1. **notebooks/Build_Document_Index.py:** Aquest és el *notebook* principal per crear els recursos necessaris per al xatbot. Aquest *notebook* instancia els esquemes, volums, taules i *endpoint* necessaris per què la plataforma funcioni. Les taules per a text, chunks i seguiment s'instancien i es poblen en aquesta tasca. Els noms dels *endpoint*s d'*embedding*, cerca vectorial i LLM també s'instancien en aquest *notebook*.

2. **utils.py:** Aquest fitxer conté les funcions d'ajuda que es criden durant l'execució del quadern (*notebook*) anterior.

- **deploy_application/:** Aquest directori conté els *script*s per implementar el model personalitzat que es va registrar a MLflow durant la tasca de muntatge de l'aplicació.

  1. **notebooks/Deploy_Application,py:** Aquest *notebook* és el fitxer principal per desplegar l'aplicació del xatbot desplegant el model als *endpoint*s de *serving* de Databricks, permetent un enfocament containeritzat perquè les aplicacions frontend accedeixin fàcilment al model en producció.
  2. **utils.py:** Aquest fitxer conté les funcions d'ajuda que es criden durant l'execució del *notebook* anterior.

- **index_update/: Aquest directori conté els scripts per actualitzar l'índex del document cada vegada que es carreguen fitxers nous**.

  1. **notebooks/UpdateDocIndex_genai_model.py:** Aquest *notebook* és el fitxer principal per al flux de treball d'actualització de l'índex de documents, que conté només aquesta tasca. Automatitza l'actualització de l'índex i el contingut de les taules amb informació que prové de la càrrega de fitxers temporals per part de l'usuari, posant-los a disposició com a context per al LLM, durant un període de temps temporal.
  2. **utils.py:** Aquest fitxer conté les funcions d'ajuda que es criden durant l'execució del *notebook* anterior.

- **remove_files/:** Aquest directori conté els *script*s per eliminar les dades dels fitxers temporals.

  1. **notebooks/RemoveDocumentIndex.py:** Aquest *notebook* és el fitxer principal que estableix el flux de treball per eliminar els fitxers temporals, que conté només aquesta tasca. Automatitza l'eliminació programada dels fitxers temporals que van ser afegits pels usuaris del xatbot. Per evitar mantenir informació no essencial als índexs, aquest flux de treball s'executa per netejar els recursos de les metadades addicionals i les dades temporals incloses pel flux de treball d'actualització de l'índex de documents.
  2. **utils.py:** Aquest fitxer conté les funcions d'ajuda que es criden durant l'execució del *notebook* anterior.

- **tests/:** Aquest directori conté les proves unitàries i d'integració per a cada flux de treball i tasca de la plataforma LLMOps, concretament::

  1. **integration_tests/:** Carpeta que conté les proves d'integració per a la plataforma LLMOps, que prova tots els endpoints que es creen i es criden per a aquesta plataforma.

     i. **__init__.py:** (Buit) Fitxer d'inicialització, necessari perquè les proves s'executin com s'esperava.
     ii. **test_chatbot_app.py:** Prova l'*endpoint* de l'aplicació del xatbot.
     iii. **test_chatbot_llm.py:** Prova l'*endpoint* LLM del xatbot.
     iv. **test_embeddings.py:** Prova l'*endpoint* d'*embedding*s.
     v. **test_vector_search_index.py:** Prova l'*endpoint* d'índex de cerca vectorial.

  2. **unit_tests/:** Carpeta que conté les proves unitàries per a la plataforma LLMOps, que prova totes les funcions d'ajuda per a cada flux de treball i tasca.

     i. **__init__.py:** (Buit) Fitxer d'inicialització, necessari perquè les proves s'executin com s'esperava.

ii. **test_build_document_index.py:** Prova les funcions d'ajuda de construcció de l'índex de documents.
iii. **test_deploy_application.py:** Prova les funcions d'ajuda de desplegament de l'aplicació.
iv. **test_evaluate_model.py:** Prova les funcions d'ajuda d'avaluació del model.
v. **test_remove_deleted_files_info.py:** Prova les funcions d'ajuda d'eliminació de la informació dels fitxers eliminats.
vi. **test_update_index.py:** Prova les funcions d'ajuda d'actualització de l'índex de documents.

- **resources/:** Emmagatzema els fitxers de configuració YAML necessaris per configurar els fluxos de treballs del Databricks.
  1. **chatbot-rag-resource.yml:** Aquest fitxer de configuració YAML configura la tasca de Databricks per construir l'aplicació Chatbot Rag, definint un nou clúster amb un *worker* i configuracions específiques de Spark, juntament amb permisos per a l'accés d'usuari. Aquesta tasca consisteix en les tasques següents: construir l'índex de documents, muntar l'aplicació i desplegar l'aplicació, la cadena de les quals es defineix en aquest fitxer per a la configuració completa d'aquest flux de treball. A més, és en aquest fitxer que s'instancien les variables necessàries per a cada tasca.
  2. **create-objects-resource.yml:** Aquest fitxer YAML configura una tasca de Databricks per crear objectes de dades com esquemes, taules i volums. Defineix un clúster de tasques amb un *worker*, configuracions personalitzades de Spark i permisos d'usuari. La tasca inclou una sola tasca, CreateObjects, amb totes les variables necessàries instanciades dins del fitxer per a l'execució del flux de treball.
  3. **index-update-resource.yml:** Aquest fitxer de configuració YAML configura la tasca de Databricks per construir la tasca d'actualització de l'índex de documents. En aquest fitxer, hem definit aspectes similars als treballs anteriors (variables, clúster, accés d'usuari i configuracions de Spark) i hem definit la tasca única que s'executa quan s'executa aquesta tasca. A més, aquesta tasca s'executa amb un disparador de *file_arrival*, per tant, el directori per escoltar l'arribada de fitxers i el temps mínim entre disparadors també es defineixen dins de l'estructura..
  4. **remove-files-resource.yml:** Finalment, aquest fitxer de configuració YAML configura la darrera tasca de Databricks per a aquest cas d'ús, la tasca d'eliminació de la informació dels fitxers eliminats. En aquest fitxer, tornem a configurar les configuracions necessàries. Aquest flux de treball es dispara de manera programada segons l'expressió cron definida en aquest fitxer YAML.

Per a una exploració més profunda de GenAI mitjançant Databricks, navegueu a https://ai-cookbook.io/.

2.4. Configuració

El primer pas per configurar el projecte per a el vostre cas d'ús és configurar el fitxer *databricks.yml* [2].

Aquest fitxer serveix com a fitxer de configuració que configura fluxos de treball, clústers, permisos, secrets i espais de treball. Cada variable es pot sobreescriure als espais de treball de PRE i PRO segons les seves especificitats. Assegureu-vos que tots els valors s'omplen correctament per garantir la generació adequada del projecte.

A continuació es mostra la llista de variables incloses en el fitxer databricks.yml:

- **experiment_name:** Nom de l'experiment per al xatbot rag.

[2] For more detail on *bundle* configuration please check Databricks Asset Bundle configuration - Azure Databricks | Microsoft Learn

- **finops_projecte:** L'etiqueta del projecte Finops hauria de seguir la nomenclatura sta-{use_case}.
- **use_case:** Nom del cas d'ús, utilitzat per anomenar tots els recursos relacionats amb el cas d'ús. Consultar amb PTD com han anomenar.
- **catalog:** Nom del catàleg del domini, on s'emmagatzemen les dades a Unity Catalog.
- **volume:** Volum de jerarquia per a l'emmagatzematge de fitxers.
- **volume_permanent_files:** Carpeta de volum per a fitxers permanents.
- **volume_temporary_files:** Carpeta de volum per a fitxers temporals.
- **table_functional_name:** Part del nom de la taula per instanciar les taules necessàries.
- **managed_location:** Ubicació administrada per al cas d'ús, al compte d'emmagatzematge de l'Azure.
- **external_location:** Ubicació externa per al cas d'ús, al compte d'emmagatzematge de l'Azure.
- **creator_service_principal:** Servei principal creador per al cas d'ús (específic per al desplegament de paquets)
- **runner_service_principal:** Servei principal executor per al cas d'ús (específic per al desplegament de paquets)
- **runner_service_principal_token_secret:** Ubicació secreta del *token* d'accés personal del servei principal de Databricks. Comproveu la secció *Secrets d'Azure i Databricks* per obtenir més detalls sobre els secrets.
- **openai_api_base:** URL base de l'API OpenAI.
- **openai_api_key_secret:** Ubicació secreta de la clau de l'API OpenAI.
- **monitoring_group:** Nom del grup d'usuaris per a l'accés a la supervisió de recursos.
- **monitoring_group_vs_permission:** Permís per al grup de monitoratge de la cerca vectorial.
- **monitoring_group_endpoints_permission:** Permís per al grup de monitoratge als *endpoints de serving*.
- **machine_group:** Nom del grup d'usuaris per a les màquines que executen fluxos de treball.
- **machine_group_vs_permission:** Permís per al grup de màquines a la cerca vectorial.
- **machine_group_endpoints_permission:** Permís per al grup de màquines.
- **users_group:** Nom del grup d'usuaris per als usuaris que visualitzen els recursos.
- **users_group_vs_permission:** Permís per al grup d'usuaris a la cerca vectorial.
- **users_group_endpoints_permission:** Permís per al grup d'usuaris als *endpoints de serving*.
- **notification_email:** Adreça de correu electrònic per a notificacions**.**

2.5. Bones pràctiques per personalitzar fluxos de treball a Databricks

La plantilla proporcionada està dissenyada per cobrir els passos més comuns d'un cas d'ús típic de LLM + RAG, oferint una base sòlida sobre la qual construir. Tanmateix, cada cas d'ús és únic i probablement es requerirà algun nivell de personalització per alinear-se amb requisits específics o lògica empresarial. Per garantir que el vostre flux de treball es mantingui eficient, mantenible i escalable, us recomanem seguir aquestes bones pràctiques.

2.5.1 Llibreries

Quan es treballa amb Databricks Workflows, instal·lar llibreries de manera eficient és crucial per al rendiment, la reproduïbilitat i l'estabilitat. En lloc d'instal·lar llibreries a nivell de *notebook*, la millor pràctica és instal·lar llibreries dins de cada tasca per garantir l'aïllament i evitar conflictes de dependències.

Aquest enfocament:

- Evita conflictes de dependència entre diferents feines que s'executen al mateix clúster.
- Redueix el temps d'inici ja que les biblioteques només s'instal·len quan es necessita.
- Garanteix la reproductibilitat, ja que cada tasca obté les versions exactes de la biblioteca necessàries.

Per aconseguir-ho, heu de definir la versió de la llibreria que s'instal·larà en la definició del flux de treball mitjançant el seu fitxer yml específic.

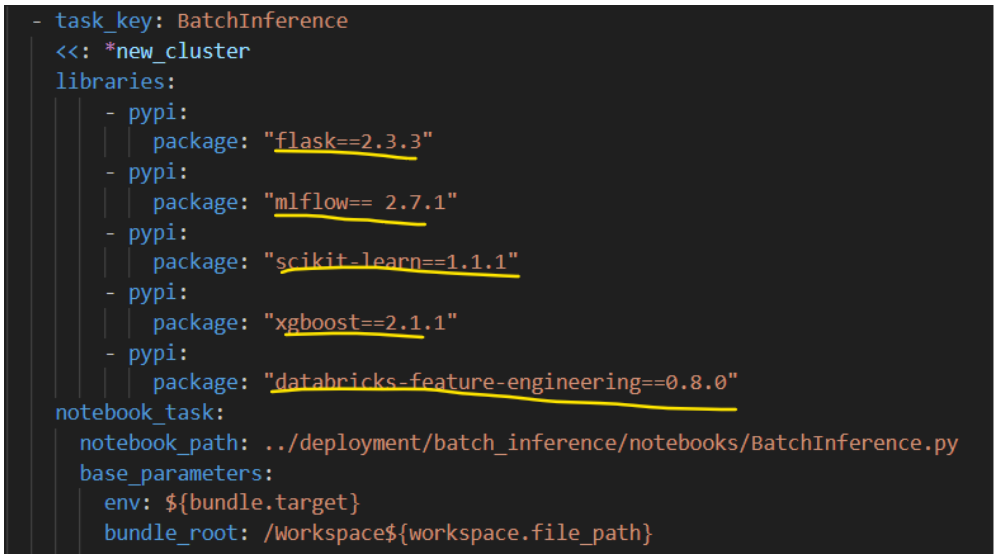

Figura 7 – Definició de cinc llibreries a nivell de tasca en el flux de treball batch inference

2.5.2 Centralitzar el codi reutilitzable

Crea un arxiu utils.py per emmagatzemar totes les funcions auxiliars i reutilitzables. En importar aquest mòdul als vostres *notebook*s o *script*s, elimineu la duplicació de codi i agilitzeu les actualitzacions als fluxos de treball

2.5.3 Parametritza els notebooks

Utilitza dbutils.widgets per passar paràmetres dinàmics als vostres *notebook*s. Això fa que els vostres fluxos de treball siguin flexibles i reutilitzables en diferents conjunts de dades, entorns o configuracions. Exemple:

```
dbutils.widgets.text("input_path", "/path/to/data")
input_path = dbutils.widgets.get("input_path")
```

2.5.4 Fluxos de treball modulars

Dividiu els vostres fluxos de treball en *notebook*s o fitxers de Python separats, cadascun responsable d'una funcionalitat específica, com ara la ingesta de dades, la transformació o l'entrenament de models. Això millora la llegibilitat, la depuració i l'escalabilitat.

2.5.5 Control de versions amb Git

Integra el teu projecte de Databricks amb un repositori Git per habilitar el control de versions i la col·laboració. Confirma els canvis regularment, utilitza branques de funcionalitats per a les actualitzacions i assegura't que tots els *script*s estiguin sincronitzats amb el repositori.

Per obtenir més detalls sobre com sol·licitar el repositori Git i les *pipelines* de CI/CD associades, consulteu la secció **Git Repositori & CI/CD Pipelines.**

2.5.6 Recomanacions de desenvolupament

Recomanem utilitzar un entorn de desenvolupament integrat (IDE) com **VSCode** per escriure i gestionar el vostre codi. Aquesta configuració sincronitza el vostre projecte amb el repositori Git, fent que el control de versions i la col·laboració siguin més fluids.

Per a qualsevol fitxer de Python enllaçat a les tasques de tasques de Databricks, assegureu-vos que la primera línia de cada fitxer inclogui el comentari següent:

```
# Databricks notebook source
```

Això garanteix que el fitxer s'interpreti com un *notebook* de Databricks quan s'importa. També fa que la depuració o el treball exploratori siguin més fàcils d'utilitzar dins de la interfície d'usuari de Databricks.

En adherir-vos a aquestes pràctiques i aprofitant un IDE, podeu personalitzar els fluxos de treball de manera efectiva mentre manteniu un procés de desenvolupament robust i eficient.

3. Recursos del Databricks

Després de desplegar el codi font de la plantilla, ja es pot consultar tota l'estructura del projecte al fitxer README.md que es troba a la nova carpeta del projecte

Ara es pot desplegar els recursos a l'entorn **del Databricks DES**. A la línia d'ordres, simplement executeu l'ordre següent:

databricks bundle deploy -p <configuration-profile-name>

Això desplegarà els recursos de Databricks seguint les instruccions especificades al fitxer databricks.yml. Aquest fitxer especifica els recursos que s'han de desplegar a Databricks (dins de la carpeta de recursos) i els espais de treball de destinació existents (DES, PRE i PRO). En executar aquesta ordre, estarem instanciant els quatre fluxos de treball per a la plantilla..

3.1. Databricks Asset Bundles

Els Databricks Asset Bundles (DAB)[3] agilitzen la gestió i el desplegament de recursos de Databricks, com ara fluxos de treball, clústers, *notebook*s, etc., empaquetant-les en *bundle*s estructurats. Aquests paquets permeten una integració perfecta amb *pipelines* CI/CD.

DAB alinea el desenvolupament d'aplicacions basades en dades amb els principis moderns de l'enginyeria de programari, automatitzant els passos clau del cicle de vida que anteriorment requerien esforç manual. Aquest enfocament estructurat simplifica la governança, millora la fiabilitat i accelera el cicle de vida del desenvolupament, fent de DAB una eina essencial per als projectes que utilitzen Databricks.

Els passos següents descriuen com començar a utilitzar DAB al terminal:

1. **Inicialitzar la gestió de perfils amb la CLI de Databricks**

   Executeu l'ordre següent per configurar l'autenticació per a cada espai de treball de destinació:

   databricks auth login --host <workspace-url>

   Seguiu les indicacions per desar la configuració com a perfil de la CLI de Databricks. Premeu Intro per acceptar el nom de perfil per defecte o especifiqueu un nom personalitzat.

2. **Llista i inspecció de perfils**:

   Per visualitzar els perfils existents:

   databricks auth profiles

   Per visualitzar la configuració existent d'un perfil específic, executeu l'ordre:

   databricks auth env --profile <profile-name>

3. **Autenticació completa**

   Al vostre navegador web, seguiu les instruccions que apareixen a la pantalla per iniciar sessió a l'espai de treball d'Azure Databricks especificat.

4. **Desplegar recursos amb Databricks Asset Bundles**

[3] DAB's Official Documentation What are Databricks Asset Bundles? - Azure Databricks | Microsoft Learn

Un cop configurada l'autenticació, podeu desplegar recursos al l'espai de treball Databricks DES de Databricks utilitzant el framework DAB.

databricks bundle deploy -p <profile-name> -t <target>

Exemple, desplegar DAB en preproducció mitjançant el perfil per defecte:

databricks bundle deploy -p DEFAULT -t pre

Això desplegarà els recursos de Databricks seguint les instruccions especificades al fitxer databricks.yml. Aquest fitxer especifica els recursos que s'han de desplegar a Databricks (dins de la carpeta de recursos) i els espais de treball de destinació existents (DES , PRE i PRO). En executar aquesta ordre, estarem instanciant els quatre **fluxos de treball** de la plantilla. Aquests s'expliquen a l'apartat següent.

3.2. Fluxos de treball

Per a aquest projecte, hi ha quatre fluxos de treball diferents, configurats mitjançant fitxers YAML, a la carpeta de recursos. Tot i que la plantilla proporcionada cobreix els passos més comuns d'un cas d'ús de GenAI, es possible que el vostre cas d'ús concret requereixi un cert grau de personalització.

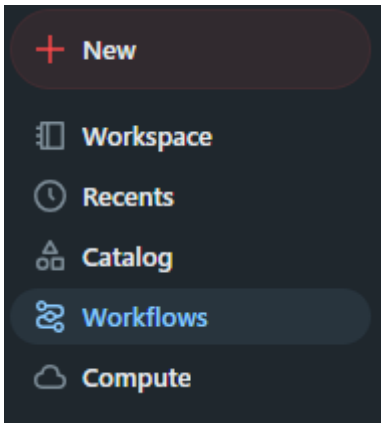

Figura 8: Fluxos de treball d'accés desplegats al Databricks

Més endavant en aquest capítol documentarem la finalitat i les funcionalitats de les tasques, els *notebook*s i les proves d'aquest projecte. A la barra lateral esquerra, navegueu a **Workflows**, després trobeu els fluxos de treball i executeu-los per esbrinar les configuracions futures necessàries.

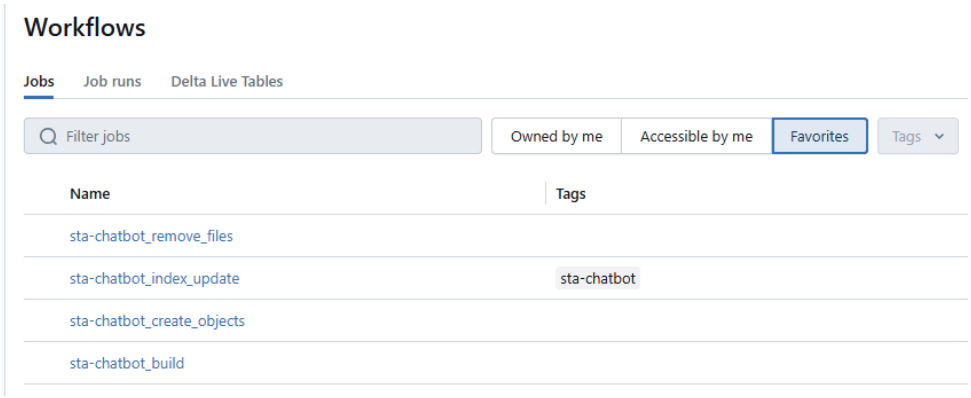

Figura 9 – Llistar els fluxos de treball creats pel bundle

3.2.1 Crea un flux de treball d'objectes

Es el primer flux de treball que s'ha d'executar. Té només una tasca que crea tots els objectes de dades buits que utilitzarà el cas d'ús. El fitxer *create-objects-workflow-resource.yml* configura una tasca de Databricks per crear objectes de dades com esquemes, taules i volums. Defineix un clúster de tasques amb un *worker*, configuracions personalitzades de Spark i permisos d'usuari. La tasca inclou una sola tasca, CreateObjects, amb totes les variables necessàries instanciades dins del fitxer per a l'execució del flux de treball.

Nota: La creació del catàleg és gestionada pel proveïdor de la PTD.

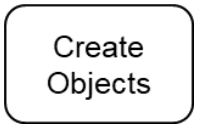

Figura 10 – Crear diagrama de flux d'objectes

El codi d'aquesta tasca es pot trobar al fitxer:

create_objects\notebooks\CreateObjects.py

3.2.2 Flux de treball Chatbot Rag

Aquesta configuració YAML configura una tasca de Databricks per construir l'índex vectorial, invocar el model LLM, el model d'*embeddings* i construir la cadena del xatbot com a model. Defineix un nou clúster amb un *worker* i configuracions específiques de Spark, juntament amb permisos per a l'accés d'usuari. El clúster està etiquetat per al seguiment de FinOps amb un nom de projecte específic.

La tasca consta de diverses tasques, cadascuna amb dependències i paràmetres definits. Es poden configurar notificacions per correu electrònic per a l'èxit o el fracàs de la tasca, assegurant un seguiment adequat del flux de treball.

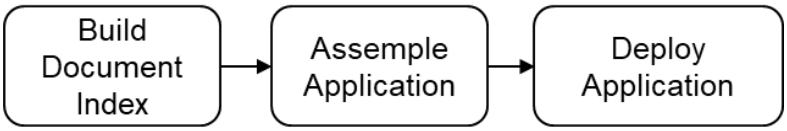

Figura 51: Diagrama del flux de treball Chatbot Rag

El flux de treball del model es defineix al fitxer **chatbot-rag-resource.yml**, dins de la carpeta de recursos. Comprèn 3 tasques seqüencials..

Construir l'index de documents

El flux de treball Chatbot Rag comença amb la tasca **Construir l'índex de documents**. El seu propòsit és accedir i preparar les dades per utilitzar-les amb el xatbot.

En executar el *notebook*, configurem l'*endpoint* d'*embeddings*, així com l'índex de cerca vectorial i el seu *endpoint* respectiu, basant-nos en els fitxers originals inclosos al volum *hierarchy_volume_core_folder*. També és responsable d'instanciar les taules necessàries de text, *chunk*s, seguiment i índex, que són requisits dels fluxos de treball restants.

Actualment, admet tipus de documents .pdf. Per a altres documents, s'han de canviar *build_document_index/utils.py* i *index_update/utils.py*, concretament la funció *extract_text.*

El codi d'aquesta tasca es pot trobar al fitxer:

build_document_index\notebooks\Build_Document_index.py

Muntar l'aplicació

La tasca Muntar l'aplicació al Flux de treball Chatbot Rag Flux de treball Chatbot Rag és la segona tasca i és responsable d'integrar el model entrenat a l'entorn operatiu de la plataforma. Aquesta tasca registra el model al Unity Catalog respectiu, garantint un seguiment i una integració adequats amb Databricks, alhora que registra el model a MLflow per al seguiment i el control de versions. En fer-ho, permet una gestió perfecta del model, posant-lo a disposició per a futurs desplegaments, monitoratge i experimentació dins del framework LLMOps.

A més del procés d'instanciació i registre, aquesta tasca configura els paràmetres essencials del model, assegurant que el model estigui configurat correctament per al seu ús al pipeline de producció. Conclou enviant una consulta de prova per validar la funcionalitat del model, confirmant que el model està integrat correctament i llest per al desplegament o l'avaluació addicional. Totes les funcions es descriuen extensament al fitxer.

El codi d'aquesta tasca es pot trobar al fitxer:

Assemble_application\notebooks\Assemble_Application.py

Desplegar l'aplicació

La tasca **Desplegar l'aplicació** al flux de treball se centra en desplegar el model personalitzat que es va registrar a MLflow durant la tasca anterior. Utilitza Databricks *Model Serving* per desplegar el model en un entorn containeritzat, proporcionant una opció de desplegament eficient i escalable. Aprofitant aquestes capacitats de *serving*, el model es fa accessible perquè les aplicacions autenticades hi interactuïn.

Aquest mètode de desplegament simplifica el procés perquè els equips de LLMOps gestionin i integrin models en diversos entorns de producció. Proporciona un monitoratge i un control robustos del rendiment del model, alhora que garanteix l'escalabilitat i la seguretat. La tasca garanteix que el model estigui llest per a una integració perfecta amb aplicacions i serveis externs, agilitzant el flux de treball des del desenvolupament del model fins al desplegament en producció. Totes les funcions es descriuen extensament al fitxer.

El codi d'aquesta tasca es pot trobar al fitxer:

deploy_application\notebooks\Deploy_Application.py

3.2.3 Flux de treball d'actualització de l'índex de documents

El Flux de treball d'actualització de l'índex de documentsestà dissenyat per agilitzar el procés d'actualització de l'índex de documents sempre que es carreguen fitxers nous per lots o en temps real. S'activa automàticament quan es col·loca un fitxer al volum *hierarchy_volume_user_temporary_folder*, assegurant que qualsevol informació nova s'incorpori ràpidament al sistema. Aquest flux de treball ajuda a mantenir la integritat i la precisió de l'índex vectorial, permetent que la plataforma es mantingui actualitzada amb les dades més recents carregades per l'usuari.

Aquest flux de treball es compon d'una **sola tasca**, també anomenada **Actualitzar l'índex de documents**, que se centra a accedir al nou fitxer a la carpeta temporal i actualitzar els components rellevants del sistema. Concretament, actualitza l'índex vectorial i les taules de base de dades relacionades (taules de text, *chunk*s i seguiment) amb el contingut dels fitxers carregats. Això garanteix que la informació dels documents es

processi i indexi correctament, fent-la cercable i accessible per a aplicacions o processos posteriors, en aquest cas l'aplicació Chatbot. Totes les funcions es descriuen extensament al fitxer.

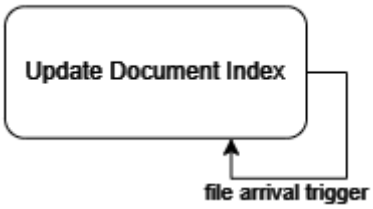

Figura 62: Flux de treball d'actualització de l'índex de documents

El codi d'aquesta tasca es pot trobar al fitxer:

index_update \notebooks\Index_Update.py

3.2.4 Flux de treball d'eliminació de fitxers suprimits

L'Eliminació de fitxers suprimits és un flux de treball programat que elimina la informació dels fitxers temporals de les taules, els volums i l'índex. Activada segons un calendari, aquest flux de treball garanteix que tots els fitxers i dades temporals carregats anteriorment durant el procés **d'actualització de l'índex de documents** s'eliminin correctament. En fer-ho, evita l'acumulació de dades no utilitzades o redundants, mantenint l'índex vectorial, els volums i les taules associades optimitzats i eficients. Això és essencial per gestionar l'emmagatzematge i garantir que la plataforma només conservi dades rellevants i utilitzades activament.

La **tasca única** dins d'aquest flux de treball, també anomenada **Eliminar fitxers suprimits**, se centra a accedir i purgar els fitxers temporals i les seves entrades corresponents de l'índex vectorial i les taules de base de dades (text, chunks i seguiment). Això garanteix que qualsevol dada transitòria, que ja no sigui necessària després del procés d'indexació de documents, s'elimini del sistema. Mitjançant la neteja diària, aquesta tasca ajuda a optimitzar el rendiment del sistema i garanteix que la base de dades vectorial es mantingui actualitzada només amb informació bàsica pertinent, donant suport així al manteniment només de dades rellevants per a futures consultes d'usuari. Totes les funcions es descriuen extensament al fitxer.

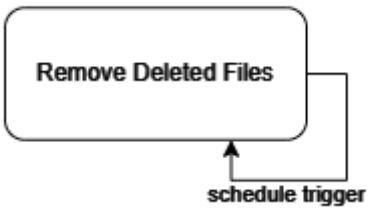

Figura 73:Flux de treball d'eliminació de fitxers suprimits.

El codi d'aquesta tasca es pot trobar al fitxer:

index_update \notebooks\Index_Update.py

3.3. Secrets d'Azure i Databricks

Aquest document recomana utilitzar secrets principalment per emmagatzemar de manera segura claus d'API, serveis principals i credencials d'usuari. Les convencions de nomenclatura per als àmbits i claus secrets són establertes i proporcionades pel proveïdor de la PTD.

Per tractar la gestió de secrets obrir un tiquet:

- Tiquet: Remedy
- Aplicació: Plataforma Transversal de Dades de la Generalitat de Catalunya

En desenvolupament, han de seguir la mateixa nomenclatura que als fluxos de treball de Databricks de preproducció i producció gestionats pel proveïdor de la PTD.

Hi ha dos tipus de gestió de secrets:

- **Azure Key Vault-Backed Secrets -** Permet fer referència a secrets emmagatzemats a l'Azure Key Vault. Aquests àmbits proporcionen una interfície només de lectura a l'Azure Key Vault, el que significa que els secrets s'han d'administrar directament a l'Azure Key Vault.
- **Databricks-Backed Secrets -** Aquests àmbits emmagatzemen secrets en una base de dades xifrada administrada per l'Azure Databricks. A diferència dels àmbits recolzats per Azure Key Vault, els àmbits secrets recolzats per Databricks permeten la gestió de secrets directament dins del Databricks.

La recomanació és utilitzar Secrets amb suport d'Azure Key Vault, tenint en compte que proveïdor de la PTD utilitza aquesta opció per a PRE i PRO. D'aquesta manera, tots els secrets utilitzats a Azure es poden centralitzar a Key Vault.

Per obtenir informació més detallada, consulteu Secret management - Azure Databricks | Microsoft Learn.

4. Proves

Les proves són essencials per garantir que cada part d'un sistema funciona com s'espera, tant per si sola com quan s'integra amb altres components. En un cas d'ús de LLMOps, això ajuda a assegurar que el rendiment del model, les *pipelines* de dades i la funcionalitat general siguin precises i fiables. En executar proves regularment, es poden detectar problemes amb anticipació, evitar que els errors arribin a producció i garantir un alt nivell de confiança en la base de codi. S'ha desenvolupat un conjunt de proves, però se'n poden afegir d'altres d'acord amb les funcions afegides als fluxos de treball.

4.1. Proves unitàries

Les proves unitàries són proves per a components o funcions individuals aïllades. Per exemple, les proves unitàries verifiquen que una funció que gestiona la ingesta de dades processa els fitxers correctament o que el pas de preparació de dades d'un model elimina les columnes esperades. Les proves unitàries se centren en peces de funcionalitat petites i ben definides per garantir que es comporten tal com s'espera. Aquestes es troben a la carpeta **tests/unit_tests.**

4.2. Proves d'integració

Aquestes proves se centren en verificar que diversos components funcionen correctament junts. Per exemple, les proves d'integració poden ajudar a garantir que un model entrenat es pot servir des d'un *endpoint* de Databricks i respon correctament a les sol·licituds externes. L'objectiu és simular interaccions del món real entre diferents parts del sistema. Les proves d'integració es poden col·locar a la carpeta **tests/integration_tests.**

Finançat per
Unió Europea Fons Europeu Next Generation
Gobierno de España
Plan de Recuperación, Transformación y Resiliencia
Next Generation Catalunya
Generalitat de Catalunya

5. Git Repositori & CI/CD Pipelines

Aquesta secció descriu els processos automatitzats per integrar i desplegar aplicacions als entorns de PRE i PRO. Gestionats per l'equip de SIC+, aquests *pipelines* estan dissenyades per agilitzar la gestió del cicle de vida, millorar l'automatització i garantir processos de desplegament fiables.

Per a més detalls consulteu la documentació oficial de CTTI:

https://canigo.ctti.gencat.cat/plataformes/ghec/

Per sol·licitar un repositori Git i les *pipelines* CI/CD requerides, es necessari obrir un tiquet a Suport Cloud amb la informació següent:

General application information

Department code	Entity code	Maintenance lot	Application Code	Component code	Acronim name	*Environments	*Cloud
PRE	CTTI				STA	dev,pre,pro	azure

List of repositories

Technic component name	*Repository type	*Category	*Engine	*Technology & version	*Builder & version
	databricks	databricks	N/A	python	python 3.x

Figura 16: Exemple de sol·licitud d'aplicació de CI/CD

On:

- S'han de validar els valors del lot de manteniment, el codi de l'aplicació (Application Code) i el codi del component (Component Code).
- El nom del component tècnic ha de seguir la nomenclatura SIC+ i PTD.
- Els altres valors haurien de mantenir-se igual, però confirmeu-ho segons el vostre cas específic.

Les subseccions següents ofereixen una breu descripció dels passos de cada pipeline.

5.1. Desenvolupament (DES)

S'automatitza el procés CI per a l'entorn de desenvolupament (branca DES). Està dissenyat per garantir que el codi compleix els estàndards de qualitat i està lliure de vulnerabilitats de seguretat abans del desplegament. A més, aplica validació per desplegar el *bundle*. El flux de treball (https://canigo.ctti.gencat.cat/plataformes/ghec/workflows/gh-definicio-workflows/) hauria d'estar configurat per realitzar les següents tasques clau:

1. **Validació i desplegament a Databricks DES:** Valideu el projecte a l'entorn de preproducció de Databricks, simulant un estat proper a la producció.
2. **Anàlisi SonarQube:** Utilitzeu SonarQube per analitzar el codi a la recerca d'errors, vulnerabilitats i "code smells" (indicadors de problemes en el codi). Utilitzeu SonarQube de CTTI o la vostra pròpia instància.

5.2. Preproducció (PRE)

Aquest flux de treball és un pas crític previ a la producció, ja que permet provar el comportament del sistema en un entorn controlat similar a la producció. A més, executa les proves unitàries configurades en el codi i verifica les proves d'integració després de desplegar el *bundle*. El flux de treball hauria d'estar configurat per dur a terme les següents tasques clau:

- **Anàlisi SonarQube**: Utilitza SonarQube per analitzar el codi a la recerca d'errors, vulnerabilitats i "code smells" (indicadors de problemes en el codi). Utilitza SonarQube de CTTI o la teva pròpia instància.
- **Unit Testing**: Executa proves unitàries amb *pytest* per validar el funcionament correcte de funcions i mòduls individuals de manera aïllada.
- **Proves d'integració:** Després del desplegament, es poden executar proves d'integració per assegurar que els diferents components funcionen correctament junts en l'entorn de preproducció. Aquestes proves s'han de dissenyar i afegir segons les necessitats específiques.

5.3. Producció (PRO)

La pipeline de Producció (PRO) és un component crucial del procés CI/CD, dissenyat per garantir el desplegament sense problemes, fiable i escalable dels models d'aprenentatge automàtic a l'entorn de producció. Aquesta pipeline és gestionada i monitoritzada pels equips de SIC+ per mantenir alts estàndards de rendiment i fiabilitat.

Seguint aquests processos estructurats, les *pipelines* CI/CD gestionen eficaçment el cicle de vida del desplegament, assegurant aplicacions d'alta qualitat, segures i de rendiment tant en entorns de preproducció com de producció.