Pràctica 1

En l'actualitat, les aplicacions d'IA generativa utilitzen diversos models de llenguatge (LLM) com OpenAI GPT-4, Claude, LLaMA o Mistral, que poden ser allotjats en núvols públics, entorns privats o localment. Quan es construeix una aplicació basada en intel·ligència artificial (IA), cal tenir un bon control sobre quin model s'usa, com s'invoca i amb quina configuració. Per això, cada vegada és més habitual fer servir un **catàleg de models LLM**, un back-end que permet consultar, registrar i configurar els models que pot consumir una aplicació d'IA.

En la **Pràctica 1**, implementarem el back-end RESTful per gestionar un catàleg de models de llenguatge. L'objectiu és crear una API Web per consultar, registrar i gestionar models LLM així com les seves configuracions específiques, pensat per ser utilitzat en entorns en el núvol.

1. Objectius

L'objectiu principal és construir una API Web que exposi els recursos necessaris per mantenir un catàleg de models d'IA generativa, juntament amb metainformació com:

- Proveïdor del model (OpenAI, Anthropic, Mistral…)
- Capacitats ("chat-completion", "code-generation"…)
- Tipus de llicència ("Permissive Open Source Licenses", "Propietary", "Custom"...)
- Característiques del model (longitud màxima del context en tokens, tipus d'entrada ("text", "image"...), tipus de sortida ("text"...)
- Data d'entrenament
- Versions

2. Funcionament general

Com que encara no implementarem cap "front-end" web, només ens haurem de preocupar en aquesta pràctica d'implementar les interfícies RESTful definides en l'API Web. Específicament, l'API Web servirà per exposar el model de dades relacional implementat amb JPA, juntament amb la implementació d'una part de la lògica de negoci de l'aplicació.

Les aplicacions web de catàleg en línia mostren un llistat dels models més populars. Aquest llistat ha d'incloure el **logotip** del proveïdor del model. Al costat dret del logotip, apareix en primera el **nom** del model (p.ex., GPT-4.1-mini) i dessota, un **resum del model de 20-30 paraules**. També, s'hi acostuma a afegir la seva **capacitat principal,** com p. ex., "chat-completion", "code-generation", "audio-generation", "text-to-image", "text-to-speech", etc.

Cal destacar que els models poden ser totalment públics (Mistral, amb llicència Apache 2.0) o només disponibles pels usuaris registrats (GPT-4, amb llicència Custom). Si els models només són accessibles pels usuaris registrats, s'indiquen amb la icona 🔒.

Un exemple, podria ser el següent:

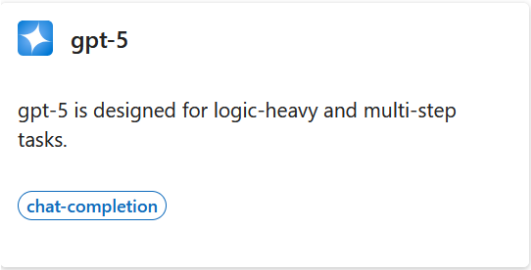


Figura 1. Resum del model.

Quan es clica damunt del logotip o el nom del model, típicament s'obre una nova pàgina amb la descripció completa del model. Per facilitar el desenvolupament de la pràctica, la descripció detallada del model seguirà el següent format:

- Damunt de tot, a l'esquerra tornarà a aparèixer el **logotip** en mida més gran.
- Seguit del **nom** del model i l'**última versió** (p. ex., 2025-06-10) i dessota el nom del **proveïdor** i la data de l'última **actualització del model** (p. ex., July 2025).
- A continuació, s'inclou una **descripció** més llarga del model i **tres capacitats** en les quals destaca el model, p. ex., *Reasoning, Multilingual, Coding*.
- Finalment, s'afegeix un text amb les especificacions més concretes del model.

Un exemple podria ser:

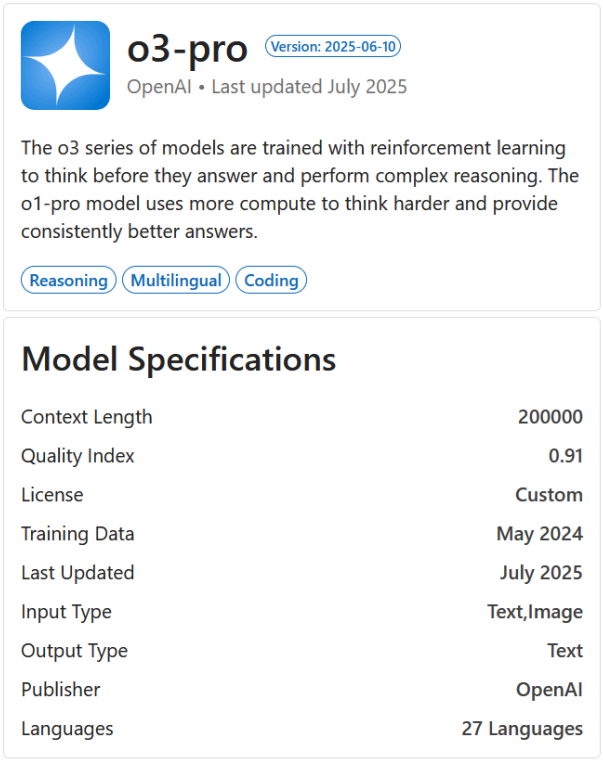


Figura 2. Visualització sencera de la informació del model.

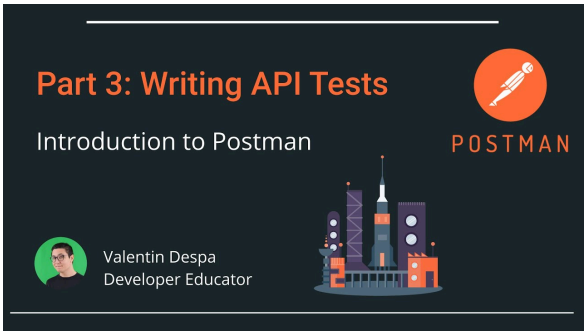
Tota aquesta informació s'haurà de representar correctament mitjançant JPA. En concret, s'hauran de crear com a mínim **dues entitats**: Model i Customer.

- **Un Model** representa un LLM concret amb les seves propietats.
- **Un Customer** és un usuari registrat del catàleg.

3. Enunciat

Haurem de suportar les funcionalitats següents:

1. Definirem una API REST comuna per tothom i que els grups hauran d'implementar a les seves pràctiques.
2. Cal implementar els serveis REST perquè treballin i retornin JSON. Eines que us poden ser d'interès: Gson, JAXB (exemple d'ús)
3. Fer servir versionat per la sostenibilitat de l'API.
4. Cal implementar un client REST per a poder interaccionar i testejar la vostra pràctica. No esteu obligats a fer servir cap tecnologia per implementar els serveis o clients REST, és més, l'essència d'aquest sistema és la simplicitat, i això serà el que més es valorarà de la vostra solució. Suggeriment PostMan API: https://www.postman.com/. Podeu fer servir TDD de manera simple amb Postam també:



.

3.1 API

Els serveis web exposats relatius als tres recursos principals: **models** i **usuaris.** Els tres recursos principals hauran de seguir la següent especificació:

Pels models:

- **GET /rest/api/v1/models?capability=${capability}&provider=${provider}**

Llista tots els models disponibles, ordenats per nom. Es poden afegir filtres **opcionals**:

- **capability=**chat-completion per mostrar només models que suportin xat.
- **provider=**openai per mostrar només models d'un proveïdor concret.
- Es podrà especificar un màxim de dos capacitats en l'atribut **capability**, i per tant la part de consulta de l'URL podria ser del tipus: **capability=${capability1}&capability=${capability2}&provider=${provider}.**
- El filtratge s'ha de fer mitjançant una consulta a la base de dades. No s'acceptarà com a vàlid retornar el llistat de tots els models i filtrar-los de forma programàtica amb Java.

- **GET /rest/api/v1/models/{id}**

Retorna els detalls complets del model, és a dir, **tota la informació indicada en la Figura 2.**

- Si el model és privat, només es podrà retornar si l'usuari està registrat (autentificat).

- **POST /rest/api/v1/models**

Registra un nou model al catàleg a partir d'un JSON proporcionat. S'ha de validar:

- Que el proveïdor indicat existeix
- Que les capacitats siguin vàlides (chat-completion, code-generation, etc.)
- Que els paràmetres tinguin valors raonables

En cas correcte, es retorna **HTTP 201** Created amb l'ID del model.

- Per aquesta operació, cal que l'usuari estigui autentificat.

- **PUT /rest/api/v1/models/{id}**

Modifica la informació d'un model existent. **Opcional**.

- Per aquesta operació, cal que l'usuari estigui autentificat.

- **DELETE /rest/api/v1/models/{id}**

Esborra un model del catàleg. **Opcional**.

- Per aquesta operació, cal que l'usuari estigui autentificat.

Pels usuaris:

- **GET /rest/api/v1/customer**
  - Llistat JSON dels usuaris.
  - Aquesta crida no pot retornar informació confidencial, p. ex., la contrasenya dels usuaris.

- **GET /rest/api/v1/customer/${id}**
  - Retorna la informació de l'usuari amb identificador ${id}.
  - Cal indicar l'**enllaç a l'últim model que va consultar** per suportar el principi de **HATEOAS**. Per exemple, en JSON:

```
"links": {
     "model": "/models/12345"
}
```

  - Aquesta crida no pot retornar informació confidencial, p. ex., la contrasenya d'aquest usuari.

- **PUT /rest/api/v1/customer/${id}**
  - **Opcional!** Modifica les dades del client amb identificador ${id} al sistema amb les dades JSON/XML proporcionades.
  - Per aquesta operació, cal que el client estigui autentificat.

3.2 Autentificació dels usuaris

Per aquells serveis web que requereixen que el client estigui autentificat, s'ha creat una anotació de tipus **Runtime** Java anomenada **@Secured.** Quan anoteu un mètode amb aquesta anotació, el servidor intentarà autentificar a l'usuari amb el mètode d'autenticació d'accés bàsica (en anglès, **HTTP Basic Authentication**), que usa la capçalera HTTP Authorization per passar al servidor les credencials de l'usuari.

Hi ha formes més professionals de realitzar aquesta autentificació i autorització d'operacions REST via JAX-RS. Com a solució simple i sense seguir cap estàndard, podríeu implementar un petit mecanisme d'API Key Authentication. O fer servir altres mecanismes d'autentificació com JWT, etc.

Resumint tot això, tindrem les següents funcionalitats obligatòries a implementar:

De l'API REST:

1. Les crides API REST obligatòries.
2. Versionat en l'API REST.
3. Dades enviades i rebudes en format JSON.
4. L'autentificació de les crides que així ho requereixen.
5. **Ús de codis HTTP adequats als resultats de les operacions de l'API REST.**
6. Proveir un client per l'API REST.
7. Proveir un joc de proves per aquest client.

Nota. Cal que diferencieu els recursos de l'API REST (**model** i **customer**) de les entitats o objectes de domini que necessiteu per poder implementar l'API web.

Es permet l'ús d'eines d'IA generativa, com ara els grans models de llenguatge (LLMs), per ajudar en la generació de codi. **Tanmateix, els estudiants hauran d'assumir la responsabilitat total del contingut generat per aquestes eines, amb el risc que pugui ser considerat plagi si es generen pràctiques amb contingut idèntic entre grups.** Això també inclou la capacitat d'explicar amb detall el contingut generat per les eines d'IA com si hagués estat desenvolupat pel mateix estudiant.

Ampliacions opcionals:

Sempre i quant la part obligatòria estigui ben dissenyada i desenvolupada, amb els següents punts s'aconseguirà pujar nota:

Àmbit actuació REST:

- Permetre que els serveis REST rebin i retornin XML com a l'alternativa a JSON.
- Cuinar codis i missatges HTTP per la gestió de les excepcions remotes (ex,. 404 Not Found, 403 Forbidden, 201 Created, ...).
- Ampliar l'API REST per incorporar les parts opcionals indicades.
- Preparació de **tests unitaris** automàtics amb Postman (https://learning.postman.com/docs/writing-scripts/test-scripts/) o altre programari.
- Sistema d'autentificació diferents a l'HTTP Basic Authentication.

4. Documentació

Juntament amb el codi font caldrà lliurar un informe en format **PDF** amb la següent estructura:

- **Introducció**. Presentació general de la pràctica i dels objectius.
- **Estructura de la pràctica**. Per exemple, caldrà especificar l'estructura del projecte, dels fitxers, les entitats JPA, entre d'altres.
- **Decisions de disseny**. Quines alternatives heu considerat a l'hora de fer el projecte i perquè heu escollit l'alternativa escollida enfront de les altres. Aquí, entre altres coses, hauríeu d'explicar quin model dels vistos a classe heu fet servir, quines parts opcionals o millores addicionals heu fet i com heu decidit implementar-les. Heu emprat eines i recursos externs?
- **Jocs de proves realitzats**. Quines proves heu fet per assegurar-vos que el vostre projecte funciona correctament en tots els casos possibles. És molt important realitzar un joc de proves extensiu, que cobreixi tots els casos, tant els positius com els negatius, com aquells que puguin produir excepcions d'algun tipus. Per tot això, heu aplicat TDD i/o BDD?
- **Conclusions**. A quines conclusions heu arribat en finalitzar aquest primer projecte.
- **Manual d'instal·lació**. La instal·lació s'ha de reduir als següents passos:
  1. Obrir projecte NetBeans;
  2. Donar-li a deploy de la vostra pràctica;
  3. (Opcionalment) Executar scripts per crear la base de dades;
  4. Passos per executar el client REST i petit joc de proves.

  A més, ha d'haver-hi un usuari de prova anomenat **sob** i contrasenya **sob** per poder provar el correcte funcionament del mecanisme de les revisions o reviews. Si necessiteu forçosament alguna altra acció, expliciteu-la adequadament.

  La **nom de la vostra base de dades ha de ser** com segueix: "**sob_grup_NN**", on "**NN**" és el número del vostre grup de pràctiques del Moodle.

5. Lliurament del projecte

El projecte es lliurarà via Moodle abans de la data màxima d'entrega.

Cal que entregueu:

1. Un arxiu **.zip** anomenat SOB_P1_nom1_nom2.zip, on nom1 i nom2 seran el nom (i cognoms) de cada un dels components del grup que entrega la pràctica.
    - Hi ha d'haver una carpeta anomenada **projecte** on hi haurà tots els fitxers que formen el projecte en NetBeans, de manera que es pugui obrir directament en el mateix entorn del laboratori.
    - Cal deixar ben clar l'ús dels passos addicionals que siguin necessaris per posar en funcionament la vostra pràctica.
2. Un fitxer anomenat **documentacio.pdf** amb la documentació de la pràctica que segueixi el format explicat en el punt anterior.

6. Avaluació

Cal fer com a mínim les parts obligatòries que us permetran aconseguir fins a un 9.5 de la nota del projecte.

Si implementeu 1 ampliació opcional correctament, la nota que es podrà aconseguir serà fins a un 10 per aquesta pràctica. És preferible tenir 1 ampliació ben desenvolupada que una quantitat superior i que estiguin incorrectes.

Alguns dels aspectes que es valoraran són:

- Funcionalitat correcta del projecte
- Estructuració de la pràctica
- Decisions de disseny preses
- Elegància de la solució proposada a nivell de programació
- Seguretat i extensibilitat de l'aplicatiu
- Compliment dels estàndards i convencions a l'hora de desenvolupar una aplicació web en Java+Netbeans (Java Code Conventions).
- Qualitat de la documentació presentada.

Darrera modificació: dijous, 13 de novembre 2025, 09:30

◄ Lliurament segona convocatòria

Salta a...

Codi base per la pràctica 1 (JDK 17; GlassFish 6) ►