



ICT Training Center



Il tuo partner per la Formazione e la Trasformazione digitale della tua azienda





SPRING AI

GENERATIVE ARTIFICIAL INTELLIGENCE CON JAVA

Simone Scannapieco

Corso avanzato per Venis S.p.A, Venezia, Italia

Novembre 2025



RETRIEVAL AUGMENTED GENERATION

PARTE 1

 Storico della conversazione valida per singola sessione

- ➔ Configurazione *JDBC H2 chat memory* per ChatClient Ollama
 - 1 Modifica al `pom.xml` per dipendenze Spring AI JDBC, H2 e *devtools*
 - 2 Modifica ad `application.yml`
 - 3 Creazione schema H2
 - 4 Modifica ai bean ChatClient per Ollama
 - 5 Test delle funzionalità con Postman/Insomnia

AMBIENTE DI SVILUPPO

CREAZIONE ACCOUNT HUGGINGFACE

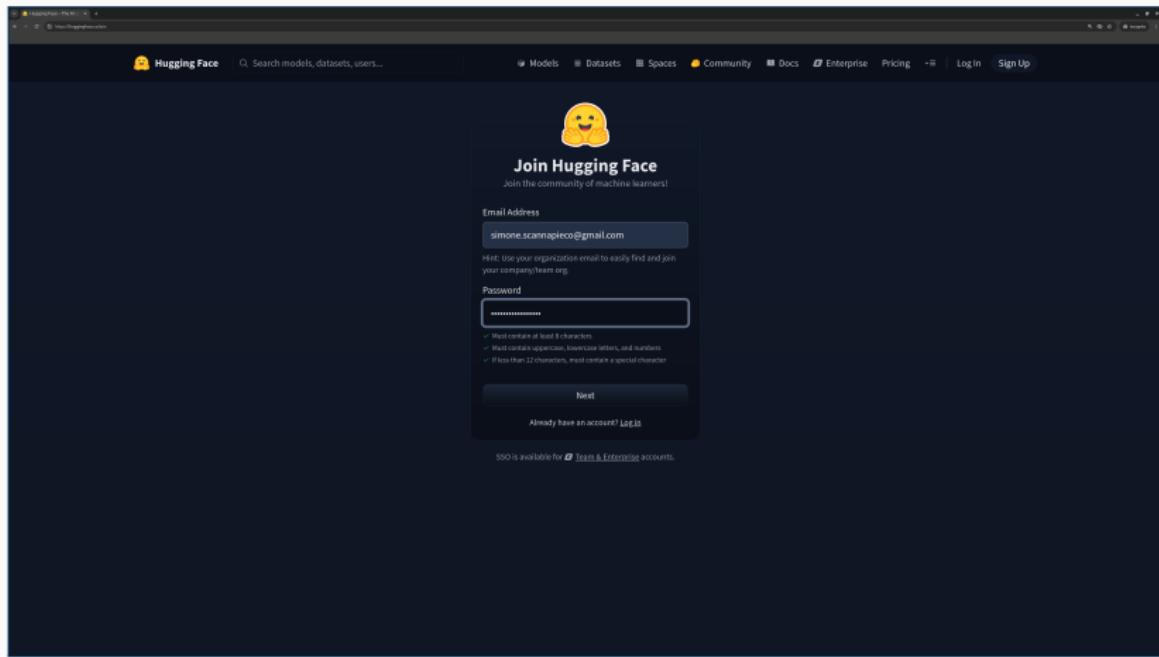
- 1 Accedere al portale <https://huggingface.co/>

The screenshot shows the Hugging Face homepage. At the top, there's a navigation bar with links for Models, Datasets, Spaces, Community, Docs, Enterprise, Pricing, Log In, and Sign Up. A search bar is also present. The main content area features a large emoji of a smiling face, followed by the text "The AI community building the future." Below this, there's a section titled "This platform where the machine learning community collaborates on models, datasets, and applications." At the bottom left, there are buttons for "Explore AI Apps" and "Browse 1M+ models". On the right side, there's a sidebar with categories like Multimodal, Computer Vision, Natural Language Processing, Audio, and Tabular. The main right panel displays a list of trending models, such as "meta-llama/Llama-2-7B", "stabilityai/stable-diffusion-xml-base", "openai/llmchat", "llyasaili/convbert", "carrapacho/mazoscope_v2_26", "meta-llama/Llama-2-13B", "tiiuae/Falcon-40B-Instruct", "KlausMWeisz/CoTMB-v2.0", "CompVis/stable-diffusion-v1-4", "stabilityai/stable-diffusion-v1-2", and "Salesforce/qwen-7b-davinci". Each model entry includes a thumbnail, name, description, and metrics like "Test Generation" and "Updated 2 days ago".

AMBIENTE DI SVILUPPO

CREAZIONE ACCOUNT HUGGINGFACE

- Creare una nuova utenza premendo il pulsante Sign up



AMBIENTE DI SVILUPPO

CREAZIONE ACCOUNT HUGGINGFACE

3 Accedere al portale e selezionare Access tokens nel menu in alto a destra

The screenshot shows the Hugging Face platform interface. On the left, there's a sidebar with options like 'Profile', 'Inbox (0)', 'Settings', 'Billing', and 'Get PRO'. Below that are sections for 'Organizations' and 'Resources'. The main area shows a 'Following' list with several AI models and their descriptions. On the right, there's a sidebar for 'Profile' (Simone.Scannapieco) and a 'Trending' section. The top navigation bar includes links for 'Models', 'Datasets', 'Spaces', 'Community', 'Docs', 'Enterprise', and 'Pricing'. A blue circle highlights the 'Access Tokens' option in the 'Settings' dropdown menu, which is part of the 'Hugging Face PRO' section. A pink box at the bottom highlights the 'Max2.2 34B Fast' model entry.

AMBIENTE DI SVILUPPO

CREAZIONE ACCOUNT HUGGINGFACE

4 Premere sul pulsante Create new token

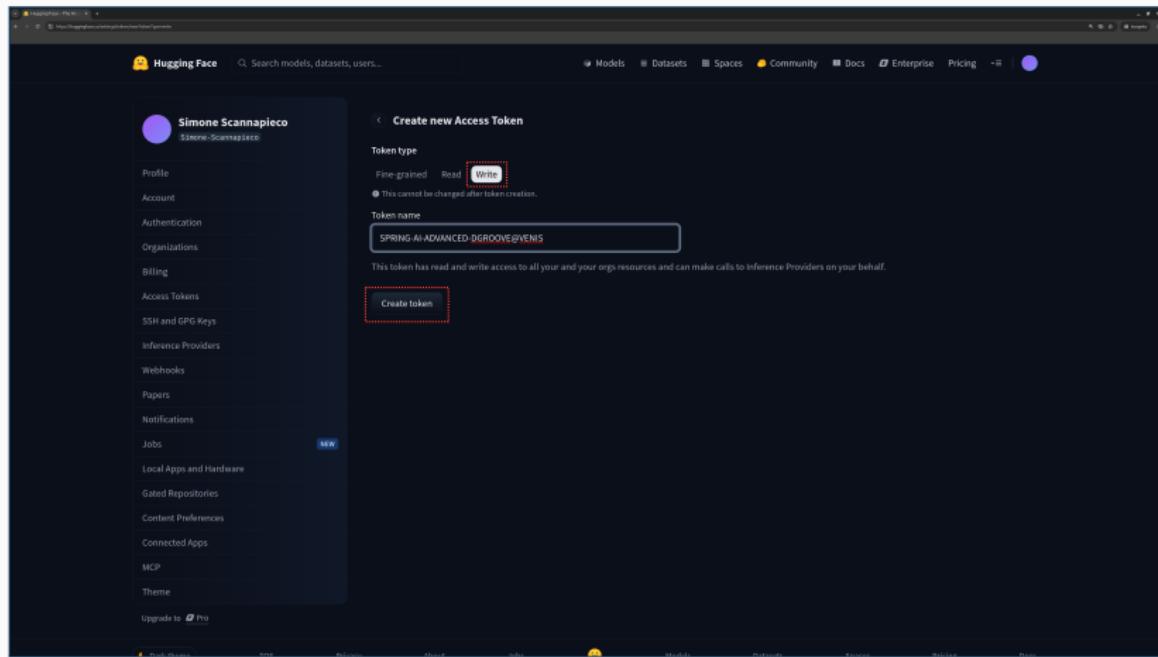
The screenshot shows the Hugging Face Hub interface. On the left, there's a sidebar with various account management options like Profile, Account, Authentication, Organizations, Billing, and several sections under Access Tokens. The 'Access Tokens' section is currently selected. At the top right of this section, there's a button labeled '+ Create new token' which is highlighted with a red dashed box. Below this button, there's a note about access tokens and a warning to not share them. A table lists existing access tokens with columns for Name, Value, Last Refreshed Date, Last Used Date, and Permissions (with 'WRITE' checked). The table includes entries such as 'SPRING-AI-ADVANCED-DGROOVE@VENIS', 'ITS-LAST COOKIE-2024', 'CCAI_AIPERUMANUM', 'SENTENCE-TRANSFORMERS-TEST-2025', and 'ITS-LAST-SUSE-2024'. At the bottom of the page, there are links for Local Apps and Hardware, Gated Repositories, Content Preferences, Connected Apps, MCP, and Theme, along with an 'Upgrade to Pro' button.

Name	Value	Last Refreshed Date	Last Used Date	Permissions
SPRING-AI-ADVANCED-DGROOVE@VENIS	H_L_HFYX	about 1 hour ago	-	WRITE
ITS-LAST COOKIE-2024	H_L_cEp	13 days ago	Jan 9	READ
CCAI_AIPERUMANUM	H_L_wHFU	Jul 23	-	READ
SENTENCE-TRANSFORMERS-TEST-2025	H_L_XRW	May 26	May 26	READ
ITS-LAST-SUSE-2024	H_L_mbQL	Dec 2, 2024	Dec 6, 2024	READ

AMBIENTE DI SVILUPPO

CREAZIONE ACCOUNT HUGGINGFACE

- 5 Selezionare token di tipo Write, denominare il token e premere Create token



AMBIENTE DI SVILUPPO

CREAZIONE ACCOUNT HUGGINGFACE

6 Cercare il modello di *embedding*, leggendone la card

The screenshot shows the Hugging Face platform interface. At the top, there is a search bar with the query "Simone Scannapieco". Below the search bar, there are sections for "Models", "Datasets", "Spaces", "Community", "Docs", "Enterprise", "Pricing", "Log In", and "Sign Up".

The "Models" section displays a list of models related to "Simone Scannapieco". One model is highlighted: "Simone-Scannapieco/sentence-bert-base-italian-v2", which has a "See 1 model result for 'Simone Scannapieco'" link. Other models listed include "meta-llama/Llama-2-7B", "stable-diffusion/stable-diffusion-xl-base-0.9", "openchat/openchat", "llyasviel/Gretelinet-v1.1", "carlsberg/zenosmoke_v2_7L", "meta-llama/Llama-2-13b", "tiiuae/Falcon-4B-1.1-instruct", "MiroslavM/VisatCoke-15B-V2.0", "CompVis/latent-diffusion-v1-4", "stabilityai/stable-diffusion-2-insta", and "Salesforce/qwen-7b-disk-12bit".

The main content area features a large yellow emoji of a smiling face with hands up, and the text "The AI community building the future." Below this, it says "The platform where the machine learning community collaborates on models, datasets, and applications." There are buttons for "Explore AI Apps" and "Browse 1M+ models".

At the bottom, there is a "Trending on 🤖 this week" section.

AMBIENTE DI SVILUPPO

CREAZIONE ACCOUNT HUGGINGFACE

6 Cercare il modello di *embedding*, leggendone la card

The screenshot shows the Hugging Face Model Card for the model `sentence-bert-base-italian-xxl-uncased-F32-GGUF`. The card includes the following details:

- Model Card:** A link to the detailed model card.
- Downloads last month:** 16
- GGUF:** Model size: 8.18 params, Architecture: bert.
- Hardware compatibility:** 32-bit, F32 (441.85)
- Inference Providers:** Sentence Similarity
- Base model:** sickipack/sentence-bert-base-italian-xxl-uncased
- Quintiles:** (link)
- Dataset used to train:** PhillipMay/stsb_wmt11_it
- Viewer:** Updated May 14, 2024 - 01:06:34 ± 3.3K v 0.67

File launch.json

```
{  
    // Use IntelliSense to learn about possible attributes.  
    // Hover to view descriptions of existing attributes.  
    // For more information, visit: https://go.microsoft.com/fwlink/?linkid=830387  
    "version": "0.2.0",  
    "configurations": [  
        {  
            "type": "java",  
            "name": "Launch Current File",  
            "request": "launch",  
            "mainClass": "${file}"  
        },  
        {  
            "type": "java",  
            "name": "DemoApplication",  
            "request": "launch",  
            "mainClass": "it.venis.ai.spring.demo.DemoApplication",  
            "projectName": "demo",  
            "env": {  
                "GOOGLE_AI_API_KEY": "...",  
                "HUGGING_FACE_HUB_TOKEN": "..."  
            }  
        }  

```

File settings.json

```
{  
    "java.compile.nullAnalysis.mode": "disabled",  
    "java.configuration.updateBuildConfiguration": "interactive",  
    "java.test.config": {  
        "env": {  
            "GOOGLE_AI_API_KEY": "...",  
            "HUGGING_FACE_HUB_TOKEN": "..."  
        }  
    }  
}
```

File docker-compose.yml

```
services:  
  ...  
  spring-ai-vector-store:  
    image: qdrant/qdrant:${QDRANT_VERSION:-latest}  
    hostname: spring-ai-vector-store  
    container_name: spring_ai_vector_store  
    ports:  
      - ${QDRANT_HTTP_PORT:-6333}:6333  
      - ${QDRANT_GRPC_PORT:-6334}:6334  
    volumes:  
      - spring_ai_vector_store:/qdrant/storage  
    restart: unless-stopped  
  
volumes:  
  ...  
  spring_ai_vector_store:  
    name: spring_ai_vector_store
```

File spring-ai.env

```
...  
# qdrant configuration  
QDRANT_VERSION=v1.13.0  
QDRANT_HTTP_PORT=6333  
QDRANT_GRPC_PORT=6334  
  
# default: latest  
# default: 6333  
# default: 6334
```

Dipendenze di sistema aggiuntive

```
...
<dependency>
    <groupId>org.springframework.ai</groupId>
    <artifactId>spring-ai-rag</artifactId>
</dependency>
<dependency>
    <groupId>org.springframework.ai</groupId>
    <artifactId>spring-ai-advisors-vector-store</artifactId>
</dependency>
<dependency>
    <groupId>org.springframework.ai</groupId>
    <artifactId>spring-ai-starter-vector-store-qdrant</artifactId>
</dependency>
...
...
```

Configurazione applicativo

```
spring:
  ...
  autoconfigure:
    exclude:
      - org.springframework.ai.vectorstore.qdrant.autoconfigure.QdrantVectorStoreAutoConfiguration
        # We must disable Vector Store auto-configuration because of two different EmbeddingModel beans
        # (OpenAI-Gemini and Ollama). The goal is to have two different vector collections, one for each
        # family of LLM.
  ai:
    ollama:
      ...
      embedding:
        model: hf.co/Simone-Scannapieco/sentence-bert-base-italian-xxl-uncased-F32-GGUF
  openai:
    ...
    embedding:
      options:
        model: gemini-embedding-001
  vectorstore:
    qdrant:
      initialize-schema: true
      host: 172.17.0.1
      port: 6334
  ...
```

File erase_llm_volumes.sh

```
#!/bin/bash

volume_name=spring_ai_llm

docker volume rm $volume_name
```

File erase_vector_store_volumes.sh

```
#!/bin/bash

volume_name=spring_ai_vector_store

docker volume rm $volume_name
```

File get-rag-data-system-ita-prompt.st

Sei un assistente AI in grado di rispondere alle domande dell'utente solo in base al contesto fornito dalla sezione DOCUMENTI.

Se la risposta non è presente nella sezione DOCUMENTI, informa l'utente di non sapere la risposta.

DOCUMENTI: <documenti>

File get-rag-data-system-eng-prompt.st

You are a helpful assistant, answering questions based on the given context in the DOCUMENTS section and no prior knowledge.

If the answer is not in the DOCUMENTS section, then reply that you cannot answer to the question.

DOCUMENTS: <documenti>

Configurazione Vector Store - I

```
package it.venis.ai.spring.demo.config;

import org.springframework.ai.embedding.EmbeddingModel;
import org.springframework.ai.openai.OpenAiEmbeddingModel;
import org.springframework.ai.vectorstore.VectorStore;
import org.springframework.ai.vectorstore.qdrant.QdrantVectorStore;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;

import io.qdrant.client.QdrantClient;
import io.qdrant.client.QdrantGrpcClient;

@Configuration
public class RAGConfig {

    @Autowired
    public OpenAiEmbeddingModel openAiEmbeddingModel;

    @Autowired
    public EmbeddingModel ollamaEmbeddingModel;

    @Value("${spring.ai.vectorstore.qdrant.host:#{null}}")
    private String qdrantHost;
    @Value("${spring.ai.vectorstore.qdrant.port:#{null}}")
    private String qdrantPort;
    @Value("${spring.ai.vectorstore.qdrant.use-tls:#{null}}")
    private String useTls;

    ...
}
```

Configurazione Vector Store - II

```
...
@Bean
public QdrantClient qdrantClient() {
    QdrantGrpcClient.Builder grpcClientBuilder = QdrantGrpcClient.newBuilder()
        .qdrantHost == null ? "localhost" : qdrantHost,
        .qdrantPort == null ? 6334 : Integer.valueOf(qdrantPort),
        .useTls == null ? false : Boolean.valueOf(useTls));
    return new QdrantClient(grpcClientBuilder.build());
}

@Value("${spring.ai.vectorstore.qdrant.collection-name.gemini:#{null}}")
private String qdrantCollectionNameGemini;
@Value("${spring.ai.vectorstore.qdrant.collection-name.ollama:#{null}}")
private String qdrantCollectionNameOllama;
@Value("${spring.ai.vectorstore.qdrant.initialize-schema:#{null}}")
private String qdrantInitializeSchema;

@Bean
public VectorStore geminiVectorStore(QdrantClient qdrantClient,
    @Qualifier("openAiEmbeddingModel") EmbeddingModel geminiEmbeddingModel) {
    return QdrantVectorStore.builder(qdrantClient, geminiEmbeddingModel)
        .collectionName(qdrantCollectionNameGemini == null ? "vector_store_gemini" : qdrantCollectionNameGemini)
        .initializeSchema(qdrantInitializeSchema == null ? false : Boolean.valueOf(qdrantInitializeSchema))
        .build();
}

@Bean
public VectorStore ollamaVectorStore(QdrantClient qdrantClient,
    @Qualifier("ollamaEmbeddingModel") EmbeddingModel ollamaEmbeddingModel) {
    return QdrantVectorStore.builder(qdrantClient, ollamaEmbeddingModel)
        .collectionName(qdrantCollectionNameOllama == null ? "vector_store_ollama" : qdrantCollectionNameOllama)
        .initializeSchema(qdrantInitializeSchema == null ? false : Boolean.valueOf(qdrantInitializeSchema))
        .build();
}
```

Componente popolamento Qdrant

```
package it.venis.ai.spring.demo.rag;

import java.util.List;
import java.util.stream.Collectors;

import org.springframework.ai.document.Document;
import org.springframework.ai.vectorstore.SearchRequest;
import org.springframework.ai.vectorstore.VectorStore;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.stereotype.Component;

import jakarta.annotation.PostConstruct;

@Component
public class TextDataLoader {

    private final VectorStore geminiVectorStore;
    private final VectorStore ollamaVectorStore;

    public TextDataLoader(@Qualifier("geminiVectorStore") VectorStore geminiVectorStore,
        @Qualifier("ollamaVectorStore") VectorStore ollamaVectorStore) {
        this.geminiVectorStore = geminiVectorStore;
        this.ollamaVectorStore = ollamaVectorStore;

    }

    @PostConstruct
    public void loadVenisInfoIntoVectorStore() {
        List<String> venisInfo = List.of(
            "VENIS SpA, acronym for Venice Informatics and Systems, is the ICT services company and local electronic commun
```

Interfaccia servizio

```
package it.venis.ai.spring.demo.services;

import it.venis.ai.spring.demo.model.Answer;
import it.venis.ai.spring.demo.model.QuestionRequest;

public interface RAGService {

    public Answer getGeminiRAGTextToVectorStoreAnswer(QuestionRequest request);

    public Answer getOllamaRAGTextToVectorStoreAnswer(QuestionRequest request);

}
```

Implementazione servizio

```
package it.venis.ai.spring.demo.services;

import java.util.List;
import java.util.Map;
import java.util.stream.Collectors;

import org.springframework.ai.chat.client.ChatClient;
import org.springframework.ai.chat.memory.ChatMemory;
import org.springframework.ai.document.Document;
import org.springframework.ai.template.st.StTemplateRenderer;
import org.springframework.ai.vectorstore.SearchRequest;
import org.springframework.ai.vectorstore.VectorStore;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.context.annotation.Configuration;
import org.springframework.core.io.Resource;
import org.springframework.stereotype.Service;

import it.venis.ai.spring.demo.model.Answer;
import it.venis.ai.spring.demo.model.QuestionRequest;

@Service
@Configuration
public class RAGServiceImpl implements RAGService {

    private final ChatClient geminiChatClient;
    private final ChatClient ollamaChatClient;
    private final ChatClient ollamaMemoryChatClient;
    private VectorStore geminiVectorStore;
    private VectorStore ollamaVectorStore;

    public RAGServiceImpl(
        @Qualifier("geminiChatClient") ChatClient geminiChatClient,
        @Qualifier("ollamaChatClient") ChatClient ollamaChatClient,
        @Qualifier("ollamaMemoryChatClient") ChatClient ollamaMemoryChatClient,
        @Qualifier("ollamaVectorStore") VectorStore ollamaVectorStore,
```

Implementazione controllore REST

```
package it.venis.ai.spring.demo.controllers;

import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RestController;

import it.venis.ai.spring.demo.model.Answer;
import it.venis.ai.spring.demo.model.Question;
import it.venis.ai.spring.demo.model.QuestionRequest;
import it.venis.ai.spring.demo.services.QuestionService;
import it.venis.ai.spring.demo.services.RAGService;

@RestController
public class QuestionController {

    private final QuestionService service;
    private final RAGService ragService;

    public QuestionController(QuestionService service, RAGService ragService) {

        this.service = service;
        this.ragService = ragService;
    }

    @PostMapping("/gemini/ask")
    public Answer geminiAskQuestion(@RequestBody Question question) {

        return this.service.getGeminiAnswer(question);
    }

    @PostMapping("/ollama/ask")
    public Answer ollamaAskQuestion(@RequestBody Question question) {
        return this.service.getOllamaAnswer(question);
    }
}
```

1 Verificare la dashboard Qdrant (<http://172.17.0.1:6333/dashboard>)

The screenshot shows the Qdrant dashboard interface. At the top, there's a search bar labeled "Search Collection". Below it, a table lists two collections:

Name	Status	Points (Approx)	Segments	Shards	Vectors Configuration (Name, Size, Distance)	Actions
vector_store_gemini	green	18	8	1	default 3072 Cosine	⋮
vector_store_oilama	green	17	8	1	default 768 Cosine	⋮

At the bottom left, it says "v1.19.0".



<https://github.com/simonescannapieco/spring-ai-advanced-dgroove-venis-code.git>

Branch: 7-spring-ai-gemini-ollama-rag-text-to-vector-store