



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 base per Venis S.p.A, Venezia, Italia

Settembre 2025

JSON USER PROMPT TEMPLATES

- 1 Nuovo *string template* per richiesta definizione da dizionario in formato JSON specificato dall'utente
- 2 Richiesta al portale Gemini di linee guida di eliminazione *markdown* da risposta in JSON
- 3 Modifica *file* GeminiFromClientService.java e GeminiFromClientServiceImpl.java
- 4 Modifica *file* QuestionFromClientController.java
- 5 Test delle funzionalità con Postman/Insomnia

⚠ Il *path* `resources/templates` viene creato automaticamente da Spring AI in fase di creazione del progetto

File `get-json-user-format-definition-prompt.st`

Dammi una definizione del termine `<lemma>`, come riporterebbe un dizionario. Restituisci la risposta in formato JSON, usando la minima formattazione possibile e senza markdown.

Il JSON deve avere la seguente struttura:

- il lemma deve essere inserito come valore della chiave `''lemma''`;
- la definizione in stile formale e conciso deve essere inserita come valore della chiave `''definizione''`;
- esempi di utilizzo del lemma come strutture JSON annidate sotto la chiave `'esempi'`. Le strutture JSON annidate devono riportare come unico valore la `''frase''` di esempio.

Interfaccia servizio Gemini (da *client*)

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

import it.venis.ai.spring.demo.model.Answer;
import it.venis.ai.spring.demo.model.DefinitionRequest;
import it.venis.ai.spring.demo.model.Question;

public interface GeminiFromClientService {

    String getAnswerFromClient(String question);

    Answer getAnswerFromClient(Question question);

    Answer getDefinitionFromClient(DefinitionRequest definitionRequest);

    Answer getCustomFormatDefinitionFromClient(DefinitionRequest definitionRequest);

    Answer getJSONUserFormatDefinitionFromClient(DefinitionRequest definitionRequest);

}
```

Implementazione servizio Gemini (da client) - I

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

import java.util.Map;
import org.springframework.ai.chat.client.ChatClient;
import org.springframework.ai.template.st.StTemplateRenderer;
import org.springframework.beans.factory.annotation.Value;
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.DefinitionRequest;
import it.venis.ai.spring.demo.model.Question;

@Service
public class GeminiFromClientServiceImpl implements GeminiFromClientService {

    private final ChatClient chatClient;

    public GeminiFromClientServiceImpl(ChatClient.Builder chatClientBuilder) {
        this.chatClient = chatClientBuilder.build();
    }

    @Override
    public String getAnswerFromClient(String question) {
        return this.chatClient.prompt()
            .user(question)
            .call()
            .content();
    }

    ...
}
```

Implementazione servizio Gemini (da client) - II

```
...

@Value("classpath:templates/get-json-user-format-definition-prompt.st")
private Resource JSONUserFormatDefinitionPrompt;

@Autowired
ObjectMapper objectMapper;

@Override
public Answer getJSONUserFormatDefinitionFromClient(DefinitionRequest definitionRequest) {
    String chatResponse = this.chatClient.prompt()
        .user(u -> u.text(this.JSONUserFormatDefinitionPrompt)
            .params(Map.of("lemma", definitionRequest.lemma()))
            .templateRenderer(StTemplateRenderer.builder().startDelimiterToken('<')
                .endDelimiterToken('>')
                .build())
            .call()
            .content());
    System.out.println(chatResponse);
    String responseString;
    try {
        JsonNode rootNode = objectMapper.readTree(chatResponse.replace("`", "").replaceFirst("json", ""));
        responseString = objectMapper.writeValueAsString(rootNode);
    } catch (JsonProcessingException e) {
        throw new RuntimeException(e);
    }
    return new Answer(responseString);
}
```


Implementazione controllore REST (da *client*)

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

import org.springframework.web.bind.annotation.RestController;
...
import it.venis.ai.spring.demo.services.GeminiFromClientService;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;

@RestController
public class QuestionFromClientController {

    private final GeminiFromClientService geminiService;

    ...

    @PostMapping("/client/definition")
    public Answer getDefinitionFromClient(@RequestBody DefinitionRequest definitionRequest) {
        return this.geminiService.getDefinitionFromClient(definitionRequest);
    }

    @PostMapping("/client/definition/custom")
    public Answer getCustomFormatDefinition(@RequestBody DefinitionRequest definitionRequest) {
        return this.geminiService.getCustomFormatDefinitionFromClient(definitionRequest);
    }

    @PostMapping("/client/definition/json/user")
    public Answer getJSONUserFormatDefinition(@RequestBody DefinitionRequest definitionRequest) {

        return this.geminiService.getJSONUserFormatDefinitionFromClient(definitionRequest);
    }
}
```

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

Branch:

9-spring-ai-gemini-json-user-format-prompt-templates