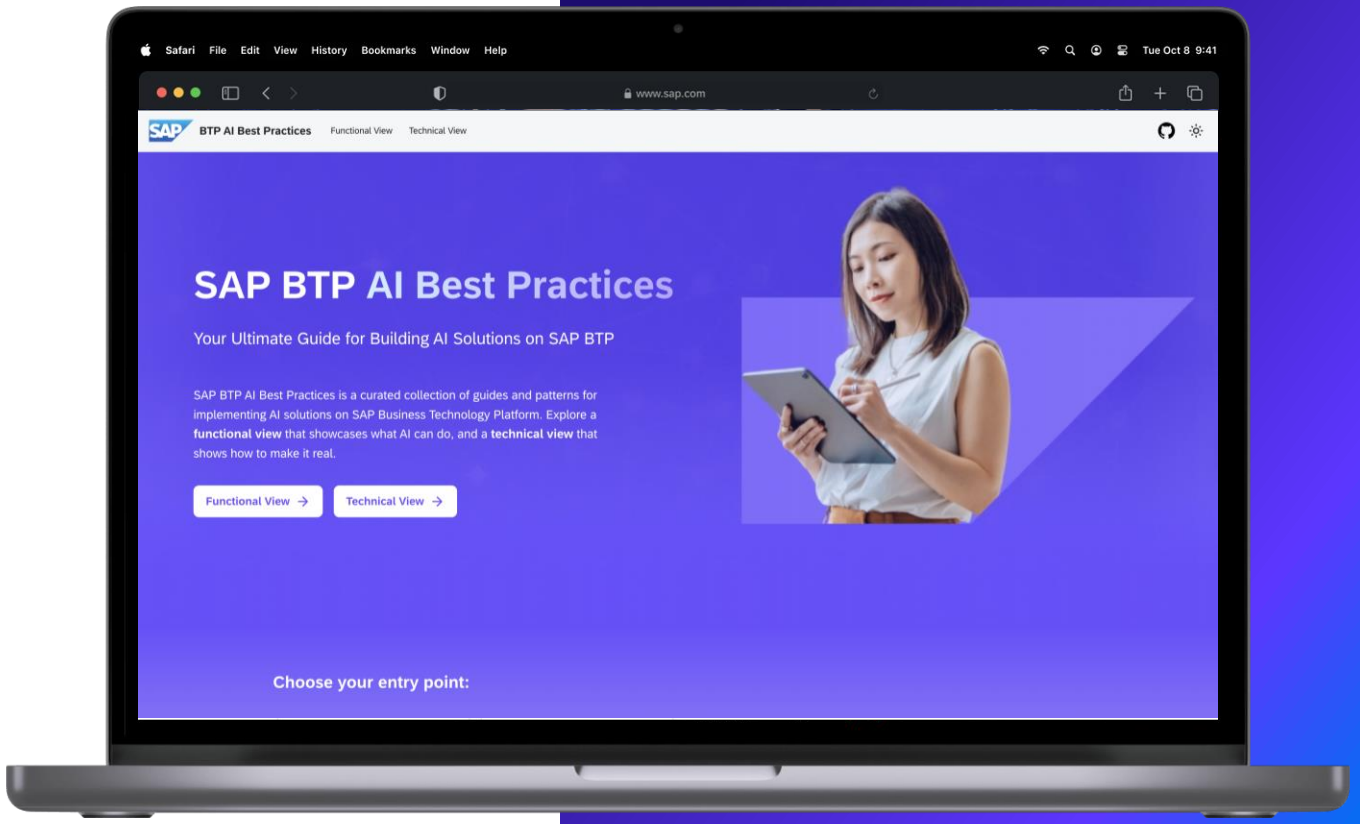
SAP BTP AI Best Practices

Prompt templating

A method used to create **structured and reusable prompts** for Large Language Models (LLMs)



BTP AI Services Center of Excellence

12.05.2025

Steps

1 Overview

2 Pre-requisites

3 Key Choices and Guidelines

4 Implementation

Prompt templating

Simple Prompt Reusability for Generative AI Models

Method used to create **structured** and **reusable** prompts **for Large Language Models**

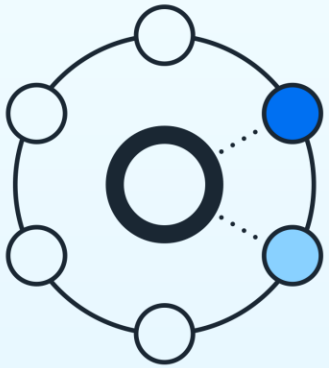
This approach allows **developers** to **program**, **store**, and **reuse** prompts efficiently across different tasks and applications

Expected Outcome

- Create reusable templates for specific tasks
- Ensure performance, reliability and efficiency of LLM responses.

Key Benefits

Why use BTP AI Core instead of direct access?



Consistency and Reusability

Templates ensure that prompts are structured consistently, which helps in achieving reliable responses from LLMs. They can be reused across different models and tasks.

Efficiency

Enabling quick adaptation and reuse of structured instructions across different tasks and contexts without the need for manual rewriting..

Customization

Templates allow for easy modification based on specific requirements or user inputs

Pre-requisites

Business

- SAP AI Core with the “Extended” tier on SAP BTP (Pricing Information)

Technical

1. Set up an SAP Business Technology Platform (SAP BTP) subaccount (Setup Guide)
2. Deploy SAP AI Core with extended service plan (Setup Guide)
3. Configure the Orchestration service in AI Launchpad (Setup Guide)

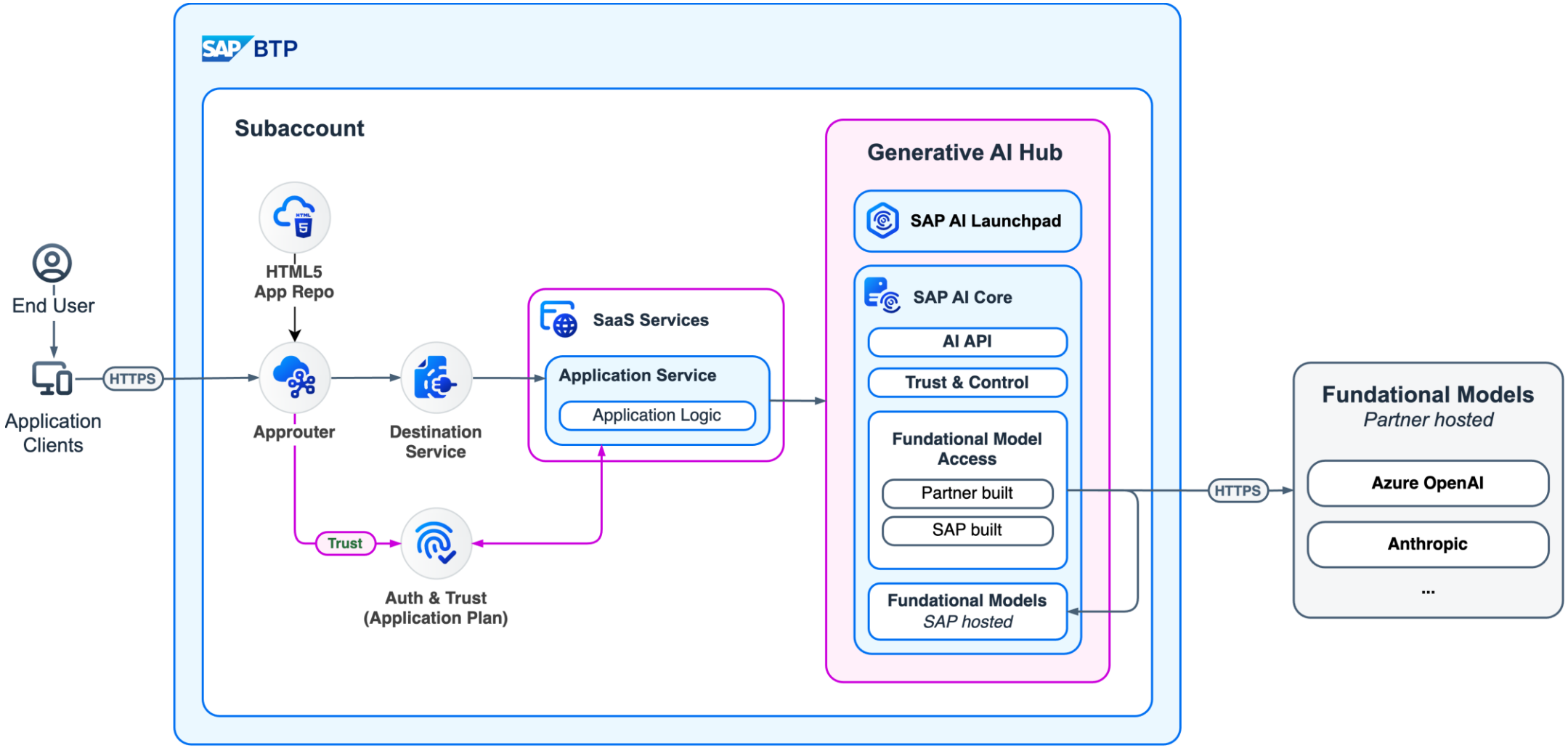
SAP Business Technology Platform (SAP BTP)

- SAP Business Technology Platform (BTP) is an integrated suite of cloud services, databases, AI, and development tools that enable businesses to build, extend, and integrate SAP and non-SAP applications efficiently.

SAP AI Core

- SAP AI Core is a managed AI runtime that enables scalable execution of AI models and pipelines, integrating seamlessly with SAP applications and data on SAP BTP that supports full lifecycle management of AI scenarios.

High-level reference architecture



Key Choices and Guidelines

Prompt Templating

Key Choices and Guidelines

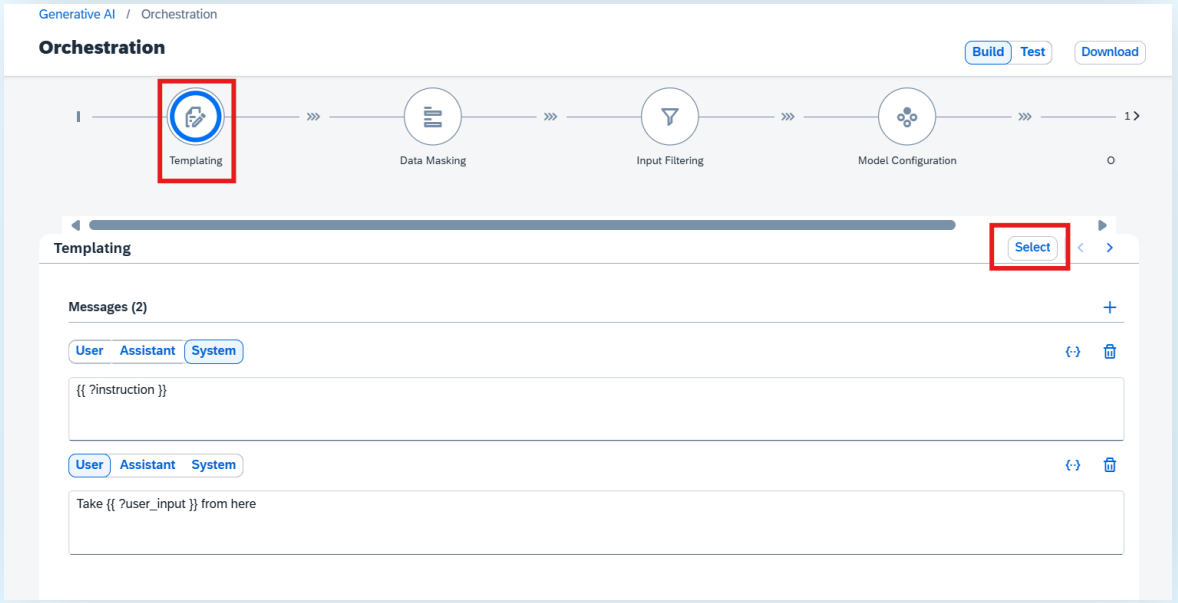
Prompt Templating

Prompt Registry Feature Within Generative AI Hub

If you're using the SAP AI Launchpad, the Prompt Registry is a feature within the Generative AI Hub designed to efficiently manage and reuse prompt templates during orchestration.

It is considered best practice to use this feature whenever possible.

Storing prompts locally or in files should be a last resort.



Tutorials and Learning Journeys

- Leveraging Prompt Registry for Seamless Orchestration
- Prompt LLMs in the generative AI hub in SAP AI Core & Launchpad
  (You can focus on the content from “Step 5: Querying the LLMs” to “Step 7: Prompt Administration")

Key Choices and Guidelines

Prompt Templating

Choose the Right Template Structure

Structured Templates vs. Modular Blocks

Decide early whether you'll use structured templates or modular blocks.

- **Structured Templates:** E.g., parameterized strings like {{?input}}
- **Modular Blocks:** E.g., predefined instructions + dynamic task blocks

Separation of Instructions

Prefer separating static instructions from dynamic input to keep templates clean and reusable.

Design for Flexibility and Reuse

Parameterization

Frequently changing parts such as tone, format, audience, and language should be parameterized. For example: Write a {{?tone}} summary for a {{?audience}} in {{?language}}.

Placeholder Names

Use self-explanatory and domain-consistent placeholder names.

Key Choices and Guidelines

Prompt Templating

Use Instructional Patterns

Some effective patterns:

- **Chain of Thought:** Think step by step
- **Few-shot examples:** Show before asking (More in the Step 15: Few-Shot Prompting of the Prompt LLMs in the generative AI hub in SAP AI Core & Launchpad learning journey)
- **Format enforcement:** Respond in JSON
- **Role prompting:** You are a marketing expert…

Standardize Formatting & Conventions

Define a consistent style for:

- **Placeholder syntax:** {{?variable}}
- **Prompt delimiters:** """, ---
- **Output formatting:** markdown, JSON, tables

Key Choices and Guidelines

Prompt Templating

Test Iteratively

Validation

Validate prompts using edge cases and diverse inputs.

Evaluation Metrics

Evaluate output quality, consistency, length, structure, and hallucination risks. Consider using prompt evaluation checklists or automated metrics.

Handle Error Cases

Fallback Templates

Build fallback templates for scenarios like empty inputs, unparseable outputs, and unexpected answers.

Instructions for Errors

Provide instructions in the prompt such as: “If the input is unclear, respond with ‘Unable to generate response.’”

Key Choices and Guidelines

Prompt Templating

Document Template Usage

Examples and Annotations

- Include examples of input/output for each template.
- Annotate templates with intended use cases, limitations, and customization tips.

Prompt Engineering

Scientific Papers

- Prompt Sketching for Large Language Models
- Enhancing Large Language Model Performance through Prompt Engineering Techniques
- Sensitivity and Robustness of Large Language Models to Prompt Template in Japanese Text Classification Tasks

Articles

- What is prompt engineering? | SAP

Implementation

Programming Model Selection Guidelines

Backend-Only API

Use **Python** (well-maintained) or **JavaScript/TypeScript** (strong async capabilities, Node.js ecosystem).

Full-stack Application (UI & Backend)

Use **CAP App** for optimized performance, scalability, and seamless SAP integration.

Python

SDK

- SAP Generative AI hub SDK (For building apps)
- SAP AI Core SDK and AI API Client SDK (AI Core lifecycle)

Reference Code

- SAP BTP AI Best Practices - Sample Code
- BTP Gen AI Hub SDK Samples (sample #2)

Learning Journeys

- Leveraging Orchestration Capabilities to Enhance Responses

JavaScript/TypeScript

SDK

- SAP Cloud SDK for AI

Reference Code

- SAP BTP AI Best Practices - Sample Code
- SAP Cloud SDK for AI - Sample Code (orchestration file)

Learning Journeys

- Leveraging Orchestration Capabilities to Enhance Responses

CAP App

SDK

- SAP Cloud SDK for AI (Recommended)
- CAP LLM Plugin

Reference Code

- SAP BTP AI Best Practices - Sample Code
- SAP Cloud SDK for AI - Sample Code (orchestration file)

Learning Journeys

- Leveraging Orchestration Capabilities to Enhance Responses

Java

SDK

- SAP Cloud SDK for AI (for Java)

Reference Code

- SAP BTP AI Best Practices - Sample Code
- Sample Spring App example (Service file and Controller file)

Learning Journeys

- Leveraging Orchestration Capabilities to Enhance Responses

Code Sample

JavaScript/TypeScript

```typescript
export default class OrchestrationService {
  async askCapitalOfCountry(req: any) {
    const country = req.data.country;
    if (!country) {
      req.reject(400, 'Country parameter is required');
    }
    const orchestrationClient = new OrchestrationClient({
      llm: {
        model_name: 'gpt-4o',
        model_params: {
          max_tokens: 1000,
          temperature: 0.6,
          n: 1
        }
      },
      templating: {
        template: [
          { role: 'user', content: `What is the capital of {{?country}}?` }
        ]
      }
    });

    const result = await orchestrationClient.chatCompletion({
      inputParams: { country: country }
    });
    return result.getContent();
  }
}
```

Contributors


Stoyanov, Velizar


Antonio, Dan


Marques, Luis

RF
Robledo, Francisco


CIGAINA, MARCO

Thank you