SAP SUPPORT CHATBOT

ProjektDokumentation

Aurelius Messmer, Jan Steiner, Robin Meier

2025

Inhalt

[1. Projektinitiierung und Zielsetzung 2](#_Toc199597299)

[1.1 Ausgangslage 2](#_Toc199597300)

[1.2 Scope-Definition 2](#_Toc199597301)

[1.3 Miro-Board 2](#_Toc199597302)

[2. Designprozess und technische Architektur 3](#_Toc199597303)

[2.1 Initiales Design 3](#_Toc199597304)

[2.2 Identifizierte Probleme mit Agent-Blocks 3](#_Toc199597305)

[3. Lösungsansatz und finale Implementierung 4](#_Toc199597306)

[3.1 Migration zu Prompt-basierten Lösungen 4](#_Toc199597307)

[3.2 Pfadsteuerung durch Condition-Blocks 4](#_Toc199597308)

[4. Finale Systemarchitektur 5](#_Toc199597309)

[4.1 Benutzerinteraktion 5](#_Toc199597310)

[4.2 Happy Path Workflow 5](#_Toc199597311)

[4.3 Ticket-Erstellungsprozess 5](#_Toc199597312)

[4.4 High Level Overview 6](#_Toc199597313)

[5. Release und Monitoring 7](#_Toc199597314)

[5.1 Deployment 7](#_Toc199597315)

[5.2 Testing und Validierung 7](#_Toc199597316)

[5.3 Monitoring-Ergebnisse (30. April - 28. Mai 2025) 7](#_Toc199597317)

[6. Fazit und Erkenntnisse 8](#_Toc199597318)

# 1. Projektinitiierung und Zielsetzung

## 1.1 Ausgangslage

Die Entwicklung des Chatbots basiert auf praktischen Erfahrungen im IT-Support-Bereich. Während ein vollständiger Ersatz des IT-Supports durch einen Chatbot unrealistisch ist, können spezifische Teilaufgaben automatisiert oder vereinfacht werden. Diese Erkenntnis entstand aus einem bestehenden Chatbot-Use-Case im IT-Support der Firma (implementiert mit Copilot Studio).

## 1.2 Scope-Definition

Aufgrund der Komplexität eines allgemeinen IT-Support-Chatbots wurde der Anwendungsbereich auf SAP-spezifische Anfragen eingegrenzt. Diese Fokussierung ermöglicht:

* Reduzierung der Problemvarianten
* Konzentration auf ein spezifisches System
* Bessere Kontrollierbarkeit des Konversationsflusses
* Gezieltere Datengrundlage

## 1.3 Miro-Board

Aus dieser Definition wurde das «Miro-Board» mit den entscheidenden Faktoren für die Erstellung eines Chatbots erstellt:

<https://miro.com/welcomeonboard/K1RIUFRqZ1dVQ0pJMnNORlZSTkJsQXEyMkZMTUNUb1p1YlcwUHJWcGdrMFpncjVacnlMaUFkcEZpVGpiK05nQm55UkUwY28wOEtob2dpeFBuNHRWbklqOGV2UVJacmhLTENVWmNmamR1QThZWlFmaEFRa0o3VlpmRjluVHhGbGxNakdSWkpBejJWRjJhRnhhb1UwcS9BPT0hdjE=?share_link_id=393164571735>

# 2. Designprozess und technische Architektur

## 2.1 Initiales Design

**Agent-Block als Kernkomponente:**

* Direkte LLM-Integration für Antwortgenerierung
* Knowledge Search-Funktionalität
* Pfadsteuerung für Konversationsfluss

**Input-Kontrolle:**

* Separater Agent zur Validierung von Benutzeranfragen
* Knowledge Base zur Richtlinienprüfung
* Filterung nicht-SAP-bezogener Anfragen

**Ticket-System:**

* Optionale Ticket-Erstellung bei unzureichenden Antworten
* Nachgelagerte Implementierung

## 2.2 Identifizierte Probleme mit Agent-Blocks

**Technische Herausforderungen:**

1. **Sporadische Input-Anforderungen:** Unvorhersagbare zusätzliche User-Inputs trotz generierter Antworten
2. **Pfadsteuerung:** Inkonsistente Pfadwahl trotz präziser Definitionen
3. **Unvollständige Ausführung:** Pfadwahl ohne entsprechende Agent-Outputs
4. **Variable-Erstellung:** Unmöglichkeit der gezielten Variablensetzung durch Prompt Engineering

# 3. Lösungsansatz und finale Implementierung

## 3.1 Migration zu Prompt-basierten Lösungen

**Vorteile der neuen Architektur:**

* Präzisere Kontrolle des Konversationsflusses
* Automatische Flow-Fortsetzung nach Output
* Eliminierung zusätzlicher User-Input-Anforderungen

## 3.2 Pfadsteuerung durch Condition-Blocks

**Implementierungsstrategie:**

Prompt-Output-Format:

*- Bei gefundener Lösung: "Ich habe eine Lösung gefunden [Output]"*

*- Bei fehlender Lösung: "Ich habe keine Lösung gefunden"*

Condition-Prüfung:

*- Contains "Ich habe eine Lösung gefunden" → Pfad A*

*- Andernfalls → Pfad B*

**Zusätzliche Funktionalitäten:**

* Variablenspeicherung durch "Set"-Block
* Wiederverwendung für Ticket-Erstellung

# 4. Finale Systemarchitektur

## 4.1 Benutzerinteraktion

**Input-Beschränkung:** Nur zwei manuelle Eingaben pro Konversation

1. Problembeschreibung
2. E-Mail-Adresse (bei Ticket-Erstellung)

**Navigation:** Primär button-basiert für strukturierte Führung

## 4.2 Happy Path Workflow

1. **Initialisierung:** Automatische Begrüßung
2. **Problemerfassung:** Benutzer-Input zur Problembeschreibung
3. **Validierung:** SAP-Relevanz-Prüfung
4. **Antwortgenerierung:**
   * Lösungsvorschlag aus Knowledge Base
   * Detaillierungsanfrage bei zu allgemeinen Beschreibungen
   * Identifikation eines Richtlinien Verstoss
5. **Zufriedenheitsabfrage:**
   * Bei Unzufriedenheit: Erweiterte KB-Suche
6. **Fortsetzungsabfrage:** Weitere Anfragen möglich
7. **Abschluss:** Verabschiedung

## 4.3 Ticket-Erstellungsprozess

**Benutzeridentifikation:**

* E-Mail-Eingabe mit Entity Capture
* KB-basierte Mitarbeitervalidierung
* Automatische Informationsextraktion

**Dringlichkeitsbewertung:**

* Drei-Stufen-System (hoch/mittel/niedrig)
* Benutzergesteuerte Auswahl

**Ticket-Generierung:**

* Antragsteller: Mitarbeiterinformationen
* Dringlichkeit: Benutzerauswahl
* Problemfall: Ursprüngliche Beschreibung
* Lösungsvorschlag: System-Output

## 4.4 High Level Overview

Ein Bild, das Text, Screenshot enthält.

KI-generierte Inhalte können fehlerhaft sein.

# 5. Release und Monitoring

## 5.1 Deployment

Der Chatbot wurde über die Sharing-Funktion von Voiceflow veröffentlicht. Eine direkte Integration in eine Webseite war aufgrund fehlender verfügbarer Webinfrastruktur nicht möglich, jedoch ermöglichte die Sharing-URL einen vollständigen Funktionstest in einer produktionsähnlichen Umgebung.

## 5.2 Testing und Validierung

**Entwicklungsbegleitende Tests:** Während des gesamten Designprozesses wurden kontinuierliche Tests durchgeführt, die umfangreiche Monitoring-Daten generierten und wertvolle Einblicke in die Systemperformance lieferten.

**Penetration Testing:** Ein externer Kollege führte gezieltes Stress-Testing durch, bei dem komplizierte und zweideutige Nachrichten verwendet wurden, um die Robustheit des Systems zu testen. Der Chatbot bewies seine Stabilität und hielt allen Manipulationsversuchen stand.

## 5.3 Monitoring-Ergebnisse (30. April - 28. Mai 2025)

Ein Bild, das Text, Screenshot, Diagramm, Software enthält.

KI-generierte Inhalte können fehlerhaft sein.

# 6. Fazit und Erkenntnisse

Das Projekt demonstriert die Bedeutung einer gezielten Scope-Definition und technischen Architekturentscheidungen für erfolgreiche Chatbot-Implementierungen. Die Migration von Agent-Blocks zu Prompt-basierten Lösungen mit Condition-Blocks erwies sich als entscheidend für die Systemstabilität und Benutzerfreundlichkeit.

**Schlüsselerkenntnisse:**

* Fokussierung auf spezifische Anwendungsbereiche erhöht die Erfolgswahrscheinlichkeit
* Technische Limitierungen erfordern flexible Architekturanpassungen
* Strukturierte Benutzerführung verbessert die User Experience
* Systematische Problemlösung durch iterative Designverbesserungen

**Erfolgsfaktoren:**

* Klare Abgrenzung des Anwendungsbereichs (SAP-fokus)
* Robuste technische Implementierung durch Prompt-Engineering
* Benutzerfreundliche Navigation durch Button-basierte Interaktion
* Integrierte Eskalationsmöglichkeiten durch Ticket-System