Smart Generative Chat

株式会社システムサポート
Azure OpenAI
パートナーリファレンスアーキテクチャ
（資料１）

2023年9月15日

システムサポート

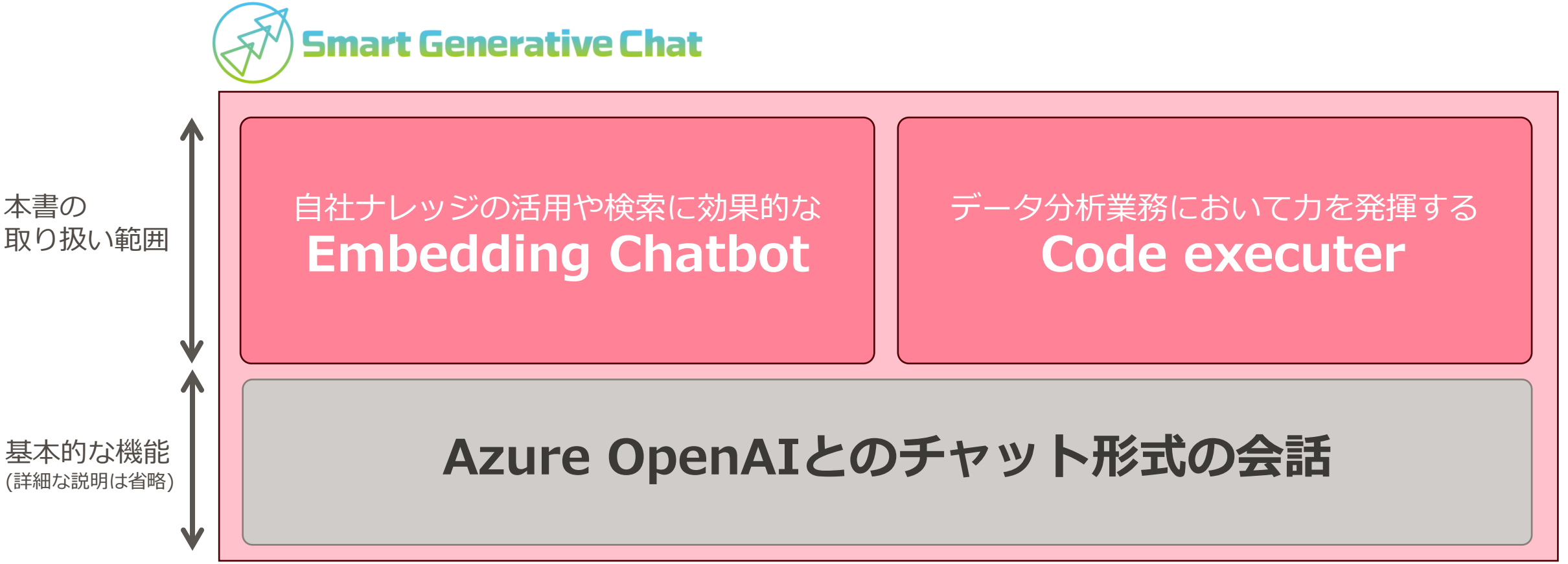
目次

01.はじめに

本書では、株式会社システムサポート（以下、当社と言う）にて提供する、Azure OpenAIを活用したサービス、「**Smart Generative Chat**」に具備する2つの機能について、これに関する業務シナリオ、アーキテクチャ、UI等を紹介します。

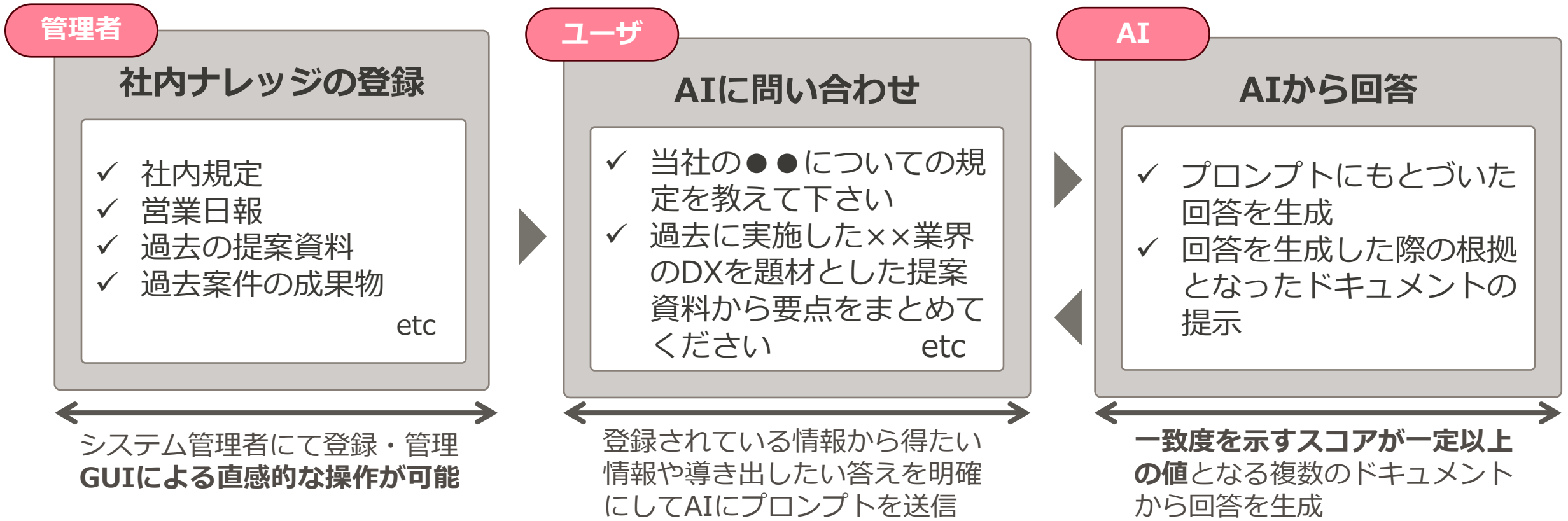
02.業種・業務シナリオ – 全体像

当社が提供する**Smart Generative Chat**では、AIとのチャット機能に加えて、下記の2つの機能を具備しています。いずれの機能も、**業種を問わず、**業務や目的に応じてご活用いただくことが可能です。



02.業種・業務シナリオ – Embedding Chatbot

Embedding Chatbotでは、貴社内の情報やナレッジを画面からデータベース上に登録し、その情報をもとにAIとの会話を行うことができます。AIモデルが知り得ない自社の情報を取り扱うことが可能です。情報やナレッジの登録は、通常、特別な手順を必要としますが、当社の機能では、誰でも直感的に登録を行えるように実装しています。



02.業種・業務シナリオ – Code executer

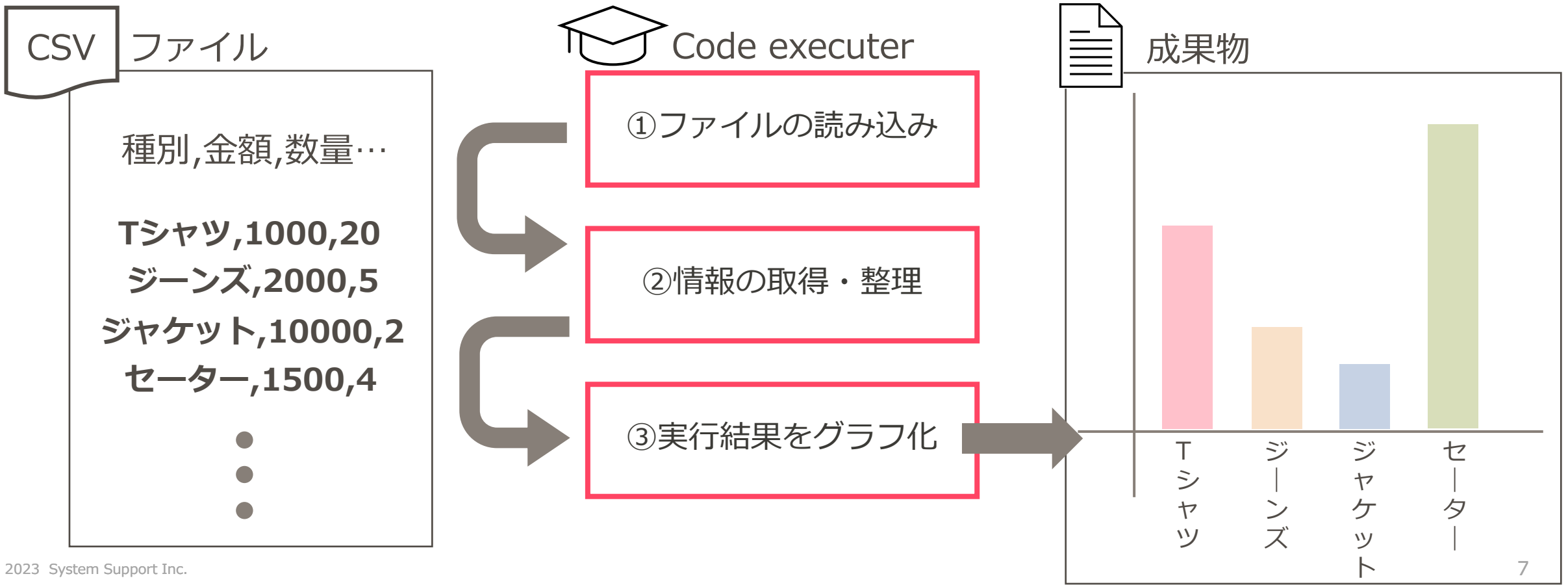
Code executerはプログラミングの知識を持たずとも、Azure OpenAI上でPythonのコードを実行することができる機能です。特にデータ分析の領域で力を発揮します。この機能によりファイルの読み込みや、データベースへの接続してSQLを実行する事が可能になります。

データソース	ファイル (CSV,PDF等)	データベース
AIの動作	ファイルの情報を読み解く	データベースに対してSQLを 実行し結果を取得
アウトプット	得られた結果をもとに指示された方式で分析を行う	

02.業種・業務シナリオ – Code executer

Code executerを活用した簡単な代表的な例として、以下のシナリオも考えられます。

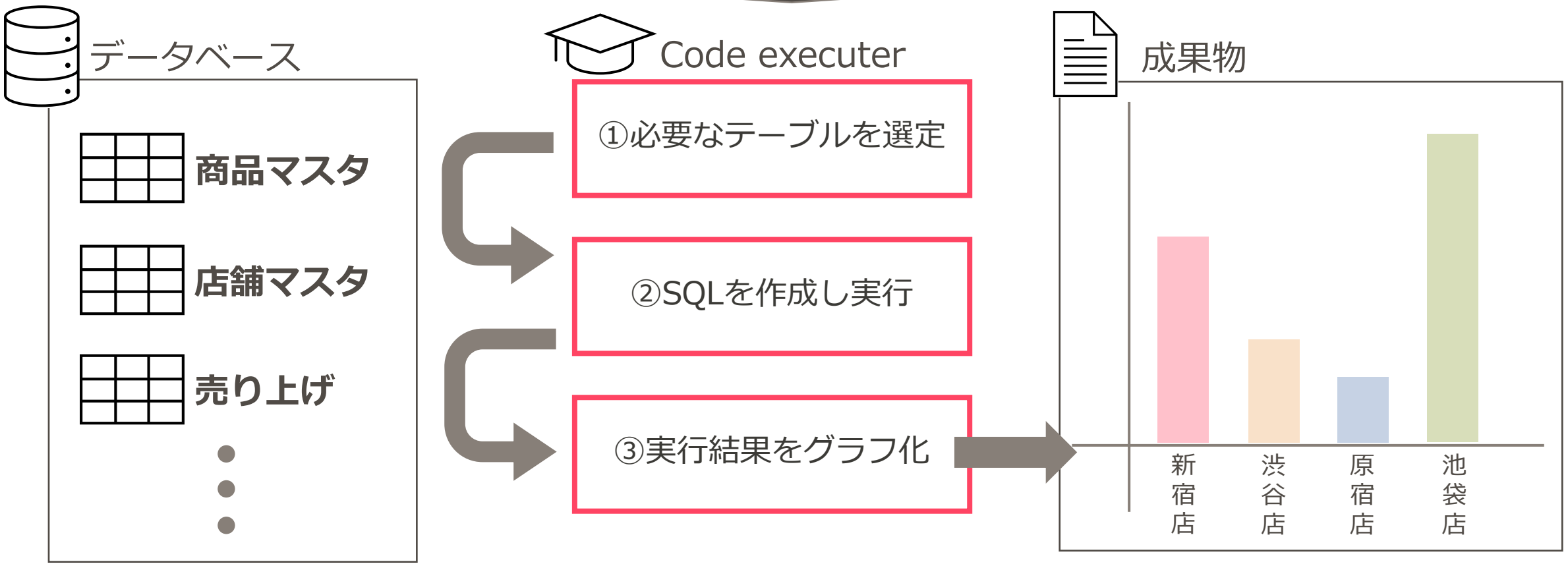
プロンプト：添付のCSVを読み取り、本日分の商品別売上を棒グラフで表現して



02.業種・業務シナリオ – Code executer

ほかにも、Code executerを活用した将来的な展望として、以下のシナリオも考えられます。

プロンプト：データベースに接続し、本日分の商品Aに関する店別売上を棒グラフで表現して



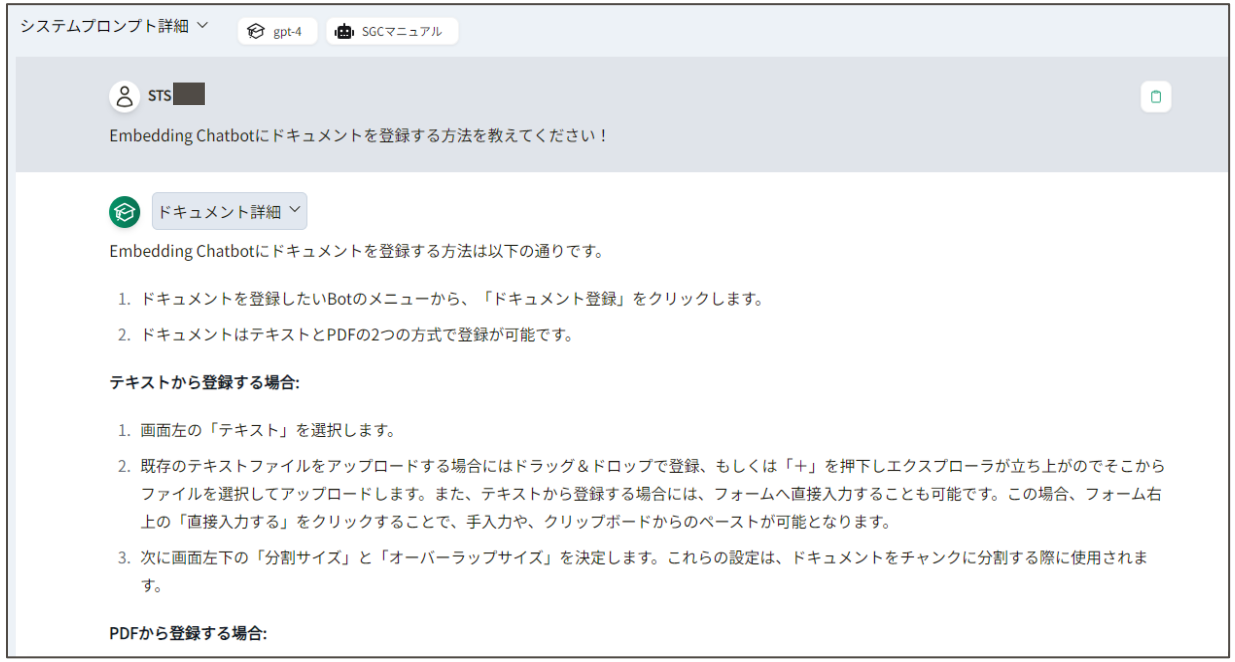
03.アプリUI – Embedding Chatbot

Embedding Chatbotのアプリ UIです。02.業種・業務シナリオで紹介した例を実行しています。

ドキュメントの登録

長大なドキュメントは内容を分割し登録することで、AIがより高い精度でドキュメントの検索・引用が可能に

独自の情報をもつAiとのチャット



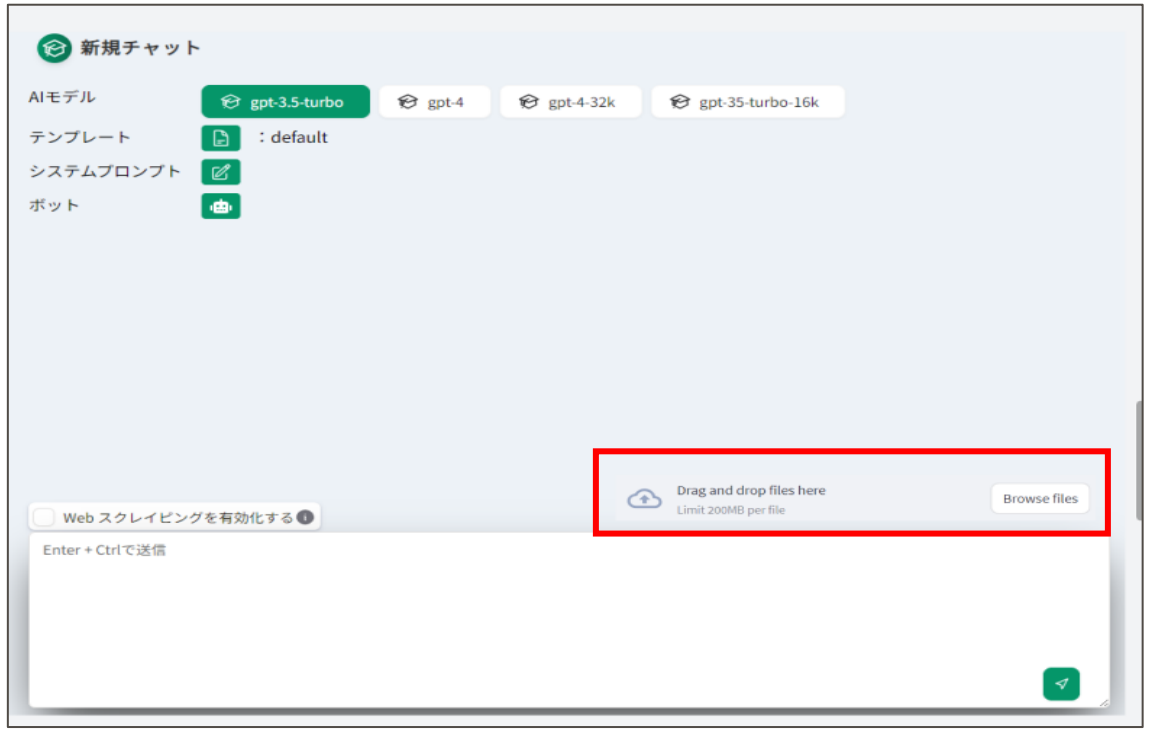
GPTが知るはずがない、自社の情報をチャット形式で気軽に取得することが可能となります。

※画像はイメージであり、導入時には異なっている場合があります。

03.アプリUI – Code executer

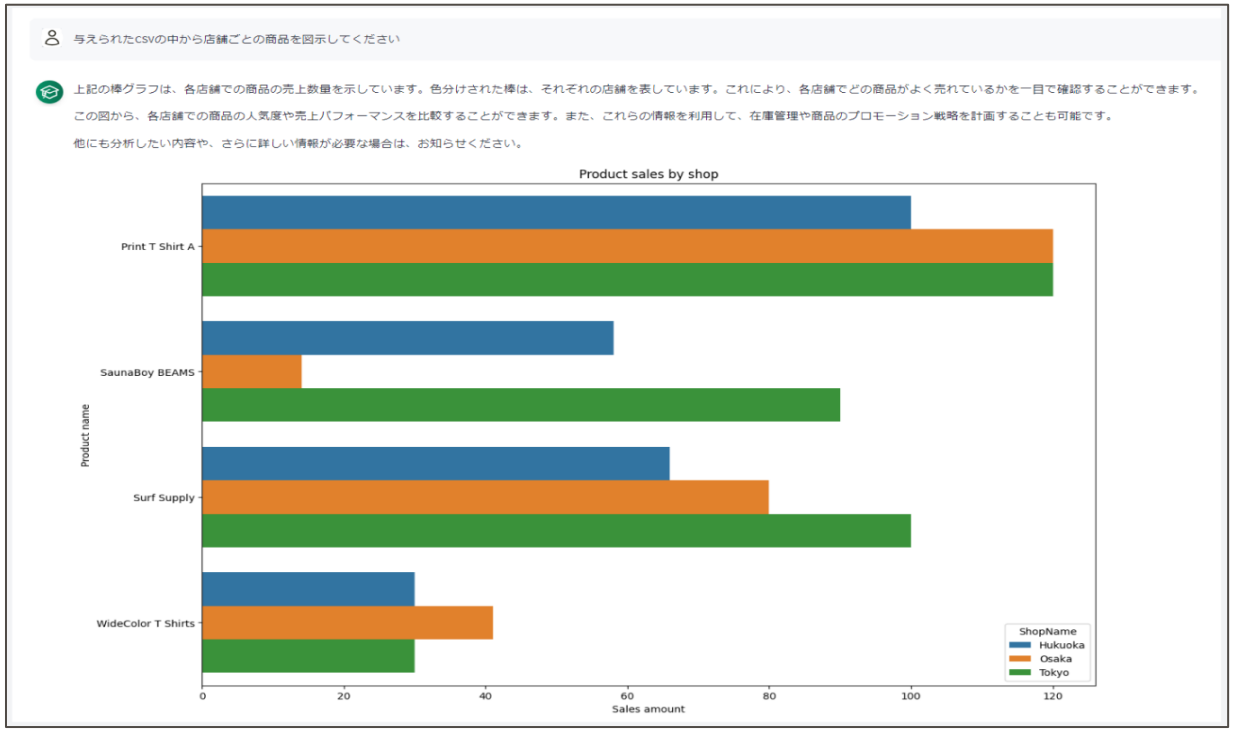
Code executerのアプリUIです。02.業種・業務シナリオで紹介した例を実行しています。

ドキュメントのアップロード



CSVなどのデータをアップロードして、目的の出力をGPTに伝えるだけで利用可能。

自動で与えられた情報を分析/描画

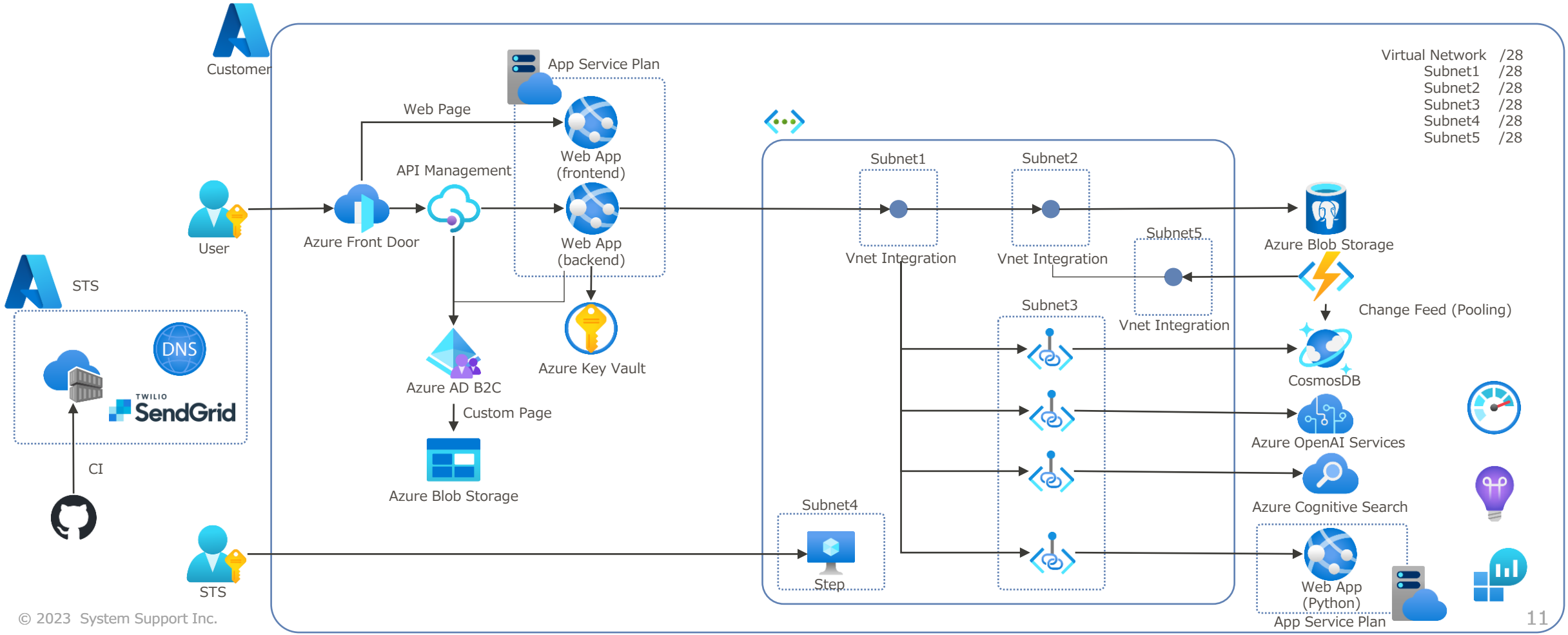


特別な手順はなく、プロンプトだけでグラフなどの描画を実現します。

※画像はイメージであり、導入時には異なっている場合があります。

04.アーキテクチャ図

Smart Generative Chatのアーキテクチャ図です。本書で紹介した2つの機能もこの中に組み込まれています。Smart Generative Chatでは、企業で安全に活用できるように、アーキテクチャを設計しています。



05.考慮事項

本構成はWell-Architected-Frameworkの5つの柱（信頼性、セキュリティ、コスト、オペレーショナルエクセレンス、パフォーマンス効率）に基づき設計を施しています。

信頼性

【例】
システムの安定稼働を目指す実装
障害発生における対策 など

セキュリティ

【例】
外部からの攻撃
情報の漏洩 などに対する対策

コスト

【例】
要件とコストのバランス
想定外の費用発生の防止 など

オペレーショナルエクセレンス

【例】
運用コスト削減に向けた施策
ヒューマンエラーを防止するための自動化 など

パフォーマンス効率

【例】
必要最小のリソース把握
アプリのリソース使用状況監視 など

※枠内記載事項はWell-Architected-Framework説明用の例であり、本構成に必ずしも当てはまるものではありません。

05.考慮事項

信頼性

考慮事項 1

障害に備えた設計

本構成では、各コンポーネントには原則PaaSを採用しています。障害発生時もAzureの復旧さえ完了すれば、アプリケーションもDockerコンテナ上で稼働しているため、特殊な手順を要することなく復旧することが可能です。

考慮事項 2

最小の停止時間を目指したSLA設定

本構成では、システム全体のSLAが**99.48%**です。年間の合計でも停止時間が最大45時間未満に収まることを意味します。パブリッククラウドは、障害の発生を考慮した設計を行う必要がありますが、本構成では可能な限り高いSLAを実現できるような構成を採用しています。

考慮事項 3

柔軟なスケールアウトの設計

本構成では、お客さまの要件に合わせて柔軟なスケールアウトが簡単に実現できる構成を採用しています。また、基本構成には含みませんが、Azure OpenAI Serviceを複数設置して、問い合わせ先を分散させ、システムの信頼性を向上させる仕組みも、オプションとして実現可能です。

05.考慮事項

セキュリティ

考慮事項 1

各種リソースのセキュリティ強化

PaaSに利用におけるネットワークセキュリティの施策として、VNET統合により、PaaSをVNET内に配置。private Linkを採用することで、VNET内のプライベートエンドポイントを参照するように設計しています。また安全性を高めるため「多層防御」の前提のもと、設計しています。

考慮事項 2

Azure Front Doorによるシステム保護

本構成では、Azure Front Doorを採用しています。Azure Front Doorで提供されている、Azureによって定められたポリシーを適用することで、システムを脅威から保護することが可能となります。また、カスタムルールを設定することも可能であり、お客さまの要件に応じて更新が可能です。

考慮事項 3

モダンアプリケーションの採用

本構成のアプリケーションはReact/Next.jsを採用しています。これらの技術は画面からのユーザ入力を適切に処理し不正なコード実行を防ぐことでクロスサイトスクリプティングなどに代表される一般的なWeb攻撃に対するリスクを大幅に軽減してくれます。

05.考慮事項

オペレーショナルエクセレンス

考慮事項 1

継続的なアプリケーション監視

本構成では、Azure Monitor、Application Insight、Logic Appsの各種機能を利用して、アプリケーションのパフォーマンスを常に監視します。

考慮事項 2

自動ビルド/デプロイ

本構成では、GitHub Actionsと連携してアプリケーションの自動ビルド/デプロイ、継続的なインテグレーションが実行します。また運用としてプルリクエスト（アプリのアップデート作業）毎に単体テストが実行されるため、アプリのデグレード、人的なエラーを未然に防ぐことが可能です。

考慮事項 3

IaC技術の採用

本構成では、インフラのコード管理の技術「Terraform」を採用しています。これにより、迅速なインフラ環境のデプロイが実現できるほか、バージョン管理も容易になるため、運用保守の工数を削減できます。

05.考慮事項

パフォーマンス効率

考慮事項 1

水平スケール設計

本構成では、一時的なアクセス集中や、恒常的な利用者数増加にも耐えうるよう設計を施しています。App Service Planをはじめとした、複数のリソースで、容易にスケールイン/アウトを行えます。

考慮事項 2

パフォーマンステストの実施

本構成開発時に、パフォーマンステストを実施しています。これにより、アプリケーションの限界値やリソースの適正なパラメータを把握しています。お客さまのご利用状況や、ご用件に合わせて適切なパラメータ設定を実装します。

コスト

考慮事項 1

利用トークンの制御

期間中の利用トークンが上限に達したら、リクエストを受け付けない機能をアプリケーションに実装しています。時間や性能で金額が定まっているリソースはコスト予測が容易ですが、Azure OpenAI Serviceは利用者のリクエストによって利用金額が変動します。使いすぎによる予算超過を懸念されるお客さまもご安心いただけるような機能も提供しています。

06.デプロイ方法

本書で紹介した機能はいずれも当社のSmart Generative Chatの機能として標準的に具備しています。そのため、貴社の環境に導入いただくことで、すぐにご利用が可能となります。お引き渡しまでの流れや、構築や保守に関する要点を下記に示します。

環境お引き渡しまでの流れ

(※1) 既存のアカウントが存在する場合は、そのアカウントに構築可能です。
(※2) ご要件により、1か月を超える構築期間をいただく場合がございます。

構築・保守に関する要点

要点1

GitHub Actionsとの連携

アプリのビルド/デプロイを自動的に行います。継続的なインテグレーションテストを自動的に実行できるため、アップデート時に起こり得る不具合の早期発見にもつながります。

要点2

Terraformによるデプロイと管理

IaCの技術を活用して、迅速なデプロイが可能です。手動での構築よりも早くご利用開始いただけます。

また、バージョン管理も容易なため、保守性が高く、保守コストを低減できます。

Thank you

STS システムサポート