ZENARCHITECTS

Azure OpenAI RAG pattern with Event-driven architecture

Azure OpenAI reference architecture
presented by ZEN Architects

コンテンツ

- シナリオ概要
- アプリUI
- アーキテクチャ
- デプロイ方法
- 考慮事項

シナリオ概要

業種・業務シナリオ

業種:

- 自社のデータを使って AI によるインテリジェントな検索を活用したい企業、官公庁

業務シナリオ:

- 自社で蓄積しているオリジナルのデータに対して、自然言語ベースでの検索を行いたい等のニーズに対して、Azure OpenAI を活用したベクター検索を利用することで、データ活用の高度化を図ることができる。
- ベクター検索の対象となるデータは、入力元を問わずにバックグラウンドかつイベントドリブンでリアルタイム更新されるため、業務運用面においてもメンテナンス効率が高い仕組みを実現できる。

アプリ UI

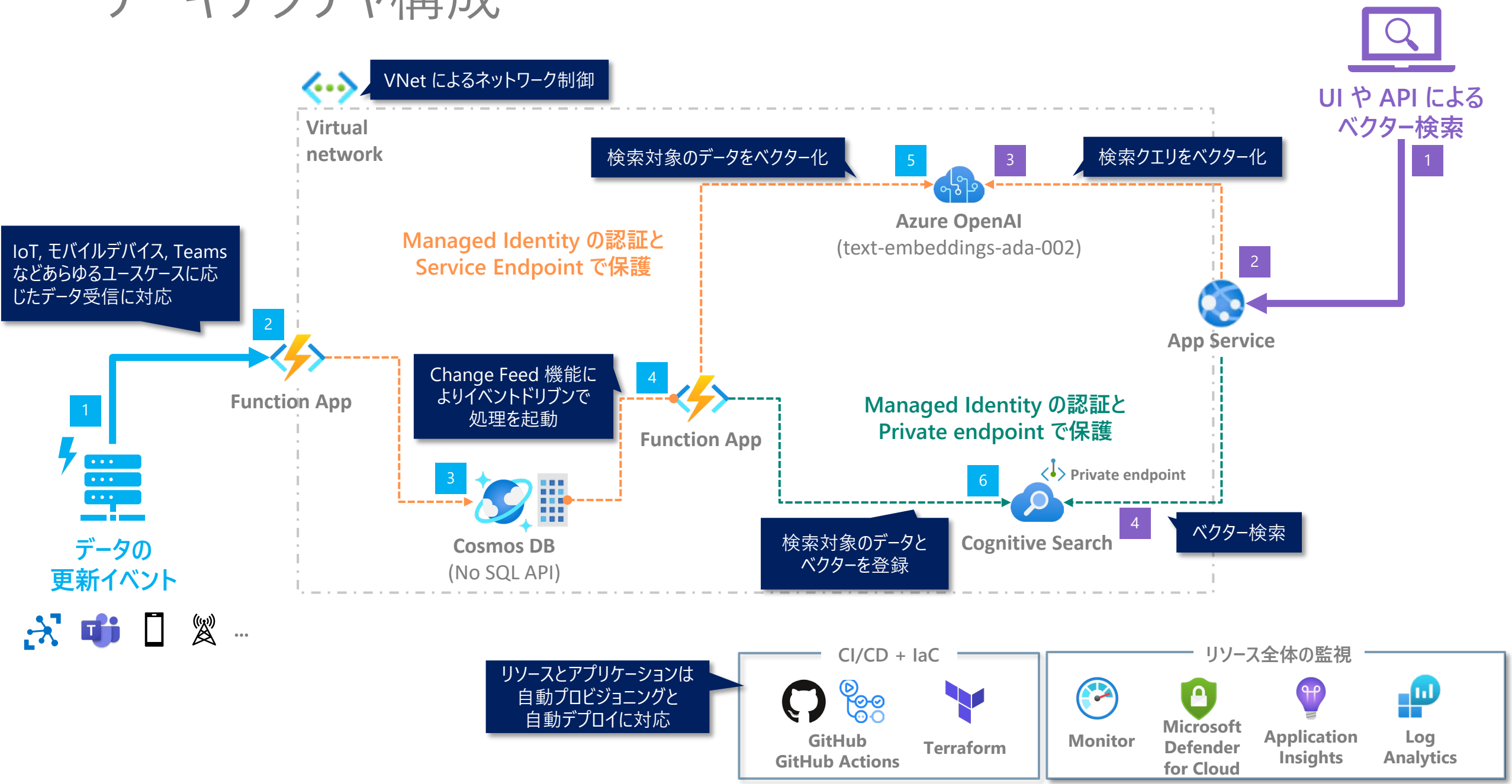
UI について

本アーキテクチャは、自社独自のデータをリアルタイムに更新してベクター検索が可能なデータを構築するためのアーキテクチャ構成と、ベクター検索を可能とする API を構成するためのアーキテクチャのため、UI についてはリファレンス実装を含めていません。

UI は、API をコールすればベクター検索が可能となるため、独自の Web アプリケーション、モバイルアプリをはじめ、Teams や Slack などのチャットインターフェースとの統合が可能です。

アーキテクチャ

アーキテクチャ構成


VNet によるネットワーク制御
Virtual network
UI や API による
ベクター検索
1
検索対象のデータをベクター化
5
3
検索クエリをベクター化
Azure OpenAI
(text-embeddings-ada-002)
Managed Identity の認証と
Service Endpoint で保護
IoT, モバイルデバイス, Teams
などあらゆるユースケースに応
じたデータ受信に対応
2
2
App Service
Function App
Change Feed 機能に
よりイベントドリブンで
処理を起動
4
Function App
Managed Identity の認証と
Private endpoint で保護
1
データの
更新イベント
3
Cosmos DB
(No SQL API)
6
Private endpoint
4
検索対象のデータと
ベクターを登録
Cognitive Search
ベクター検索
...
リソースとアプリケーションは
自動プロビジョニングと
自動デプロイに対応
CI/CD + IaC
GitHub
GitHub Actions
Terraform
リソース全体の監視
Monitor
Microsoft
Defender
for Cloud
Application
Insights
Log
Analytics

アーキテクチャ構成の補足

データの更新イベントから Function App での受信 (1 2 3)

- データの送信元の Webhook やモバイルデバイスから REST API での受信や、Blob storage のファイル更新や IoT Hubs や Event Grid といった Azure のサービスのイベントをトリガーとした受信など、ユースケースに応じて柔軟なデータ受信が可能です。
- Function App では、データのスキーマを補正して Cosmos DB へデータを登録します。
- Cognitive Search へ直接データを保存せずに最初に Cosmos DB へデータを保存することで、後続処理でエラー発生時のリトライの信頼性を高め、データの一貫性を維持しやすくします。

イベントドリブンでの Cognitive Search のインデックス更新 (4 5 6)

- Cosmos DB のデータが更新されると、Change Feed 機能によりイベントドリブンで Cognitive Search のインデックスの更新処理がトリガーされるため、ほぼリアルタイムでの更新を実現します。
- Azure OpenAI の LLM を利用してベクター検索に必要なベクターを生成し、Cognitive Search のインデックスへデータを登録します。

ベクター検索 (1 2 3 4)

- Cognitive Search を使うことで、ベクター検索だけではなくフィルターやファセットによる絞り込み、通常のフルテキストやセマンティック検索とを組み合わせたハイブリッド検索を実現します。

デプロイ方法

GitHub Actions による IaC, CI/CD

GitHub の repository にてプログラム, IaC, CI/CD を含む
ソースコードを公開しています。

- URL: zengeeks/aoai-rag-serverless (github.com)

考慮事項

アーキテクチャのポイント・考慮事項

信頼性

- インフラは Azure の PaaS サービスのみで構成しているため、ゾーン冗長・リージョン冗長や GEO レプリケーションを構成することでビジネスクリティカルなワークロードにも適用が可能です。
- Cosmos DB を利用することで、データ取り込みが高負荷時はオートスケールにより書き込み処理の信頼性を確保し、また Cognitive Search へのデータ更新のトラブルによるリトライ時でも、データの信頼性を確保します。
- Application Insights を組み合わせることでアプリケーションの問題をすばやく検知し、アラートを発行することで対応することが可能です。
- アプリの実装にはネットワーク接続のリトライが組み込みで実装されている Azure SDK を利用しているため、クラウドで避けられないリソース間のネットワーク接続のリスクを最小化します。

セキュリティ

- インフラを Azure の PaaS のサービスで構成しているため、OS やミドルウェアのセキュリティの最適化の責務をクラウドプロバイダーに任せつつ、Microsoft Defender for Cloud の活用により状態の確認や問題の検知・アラートの自動生成を可能とすることで、セキュリティリスクを最小化します。
- Managed Identity を構成することで、Cosmos DB や Cognitive Search、Azure OpenAI へのアクセスの接続文字列などの機密情報の管理を不要とすることで、漏洩のリスクを排除しています。
- 仮想ネットワークで各リソースを保護することにより、ネットワークに内在するセキュリティリスクを軽減します。

アーキテクチャのポイント・考慮事項

コスト最適化

- Azure の PaaS サービスを利用することで、インフラの運用・管理のコストを最小化します。
- ネットワークのセキュリティでは、低コストでセキュアな構成が可能となる Service endpoint を中心に利用し、それが非対応の Cognitive Search のみに Private endpoint を構成することで、コストの最適化をおこなっています。
- Cosmos DB へのデータ蓄積は、TTL を構成して不要なデータの蓄積を自動で削除することで、管理・運用コストを最小化します。

オペレーショナルエクセレンス

- インフラの構築は、Terraform で IaC を構成することで、インフラの作成・変更時の人的オペレーションミスを最小化します。
- アプリケーションの CI/CD は、GitHub Actions による自動化により、人的オペレーションミスを最小化します。
- すべての Azure リソースは Azure Montor と Applications Insights にてモニタリングされています。

パフォーマンス効率

- アプリケーションのインフラに App Service / Function App を採用したことで、ワークロードに合わせてオートスケールを適切に構成することで、高負荷時でもパフォーマンスを最適化することが可能です。
- データストアの書き込みは Cosmos DB を採用することで、ワークロードに合わせて適切なオートスケールを設定することで高負荷時でもパフォーマンスの劣化なく運用が可能です。

ZENARCHITECTS