



Golden Data



DISRUPTIVE ARCHITECTURES:
IOT, IOB & GENERATIVE IA

98774 - Gabriel Previ de Oliveira
(2TDSPF)

97850 - Gustavo Soares Fosaluza
(2TDSPF)

551692 - Mateus Vinicius da Conceição
(2TDSPV)

552000 - Pedro Henrique Figueiredo
(2TDSPV)

550871 - Vitor da Silva Ruas (2TDSPV)

Problema

Vivemos na era da informação, diariamente somos bombardeados com milhares de informações, e é assim que tendências são geradas. A mutabilidade do setor de comércio eletrônico, traz consigo diversos desafios.

I

Divulgação de produtos

- Falta de precisão, ao divulgar anúncios

2

Retenção dos clientes

- A precariedade na compreensão dos clientes no pós compra, os afastam das empresas

3

Fidelização dos clientes

- Muitas empresas tem dificuldade em estabelecer clientes fiéis

Solução

Nossa solução é uma plataforma B2B, nosso objetivo é solucionar a dor das empresas

Visão Geral

Golden Data é uma plataforma que conecta diretamente empresas e clientes para otimizar a personalização dos serviços, especialmente para empresas de comércio eletrônico.

Cliente

Uma parte da plataforma será para o cliente, é nela que coletaremos os dados através de pesquisas que serão respondidas pelo usuário

Personalização

Receba anúncios e descontos que correspondem aos seus interesses e hábitos de consumo.

Descontos exclusivos

Cupons personalizados entregues diretamente por email, facilitando a obtenção de benefícios.

Relevância

Acesso a ofertas e produtos que realmente importam para você.



Mercado Alvo

Para onde está direcionado nosso foco?

Empresas que atuam no setor de e-commerce

O Golden Data pretende atender aos mais diversos segmentos de e-commerce do país

Empresa

Do lado da empresa, teremos uma plataforma por assinatura com avaliação gratuita e dois planos: Plano Beta, em que a empresa pagará o valor integral pelos serviços e o plano Sócios, que oferecerá desconto nos serviços, ter permissão de distribuir seus próprios cupons e ter acesso aos emails dos clientes, além de obter insights mais profundos e detalhados.

Identificação de Tendências

Análise avançada de dados para identificar novas oportunidades de mercado.

Segmentação avançada

Uso Machine Learning e IA para segmentar o público alvo de forma precisa

Aumento nas conversões

Campanhas de marketing direcionadas aumentam as chances de venda.



Bibliotecas, Frameworks e outras tecnologias

Para o desenvolvimento do modelo de Machine Learning, utilizaremos algumas ferramentas

- **Python:** Linguagem principal
- **Pandas e NumPy:** Manipulação e limpeza de dados
- **Scikit-learn:** Modelagem, pré-processamento e avaliação
- **TensorFlow/Keras:** Redes neurais
- **Matplotlib/Seaborn:** Visualização de dados
- **Flask/FastAPI:** APIs RESTful para deploy
- **Docker:** Containerização e orquestração
- **Azure:** Hospedagem em nuvem



Bibliotecas utilizadas no protótipo

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score
```

```
from faker import Faker
import random
```



Bibliotecas utilizadas no protótipo

- pandas: Manipulação e análise de dados em estruturas como DataFrame.
- LabelEncoder (sklearn): Converte dados categóricos em numéricos.
- StandardScaler (sklearn): Padroniza os dados com média 0 e desvio padrão 1.
- train_test_split (sklearn): Divide os dados em conjuntos de treino e teste.
- KNeighborsClassifier (sklearn): Algoritmo de classificação KNN baseado em vizinhos mais próximos.
- DecisionTreeClassifier (sklearn): Algoritmo que cria uma árvore de decisão para classificação.
- accuracy_score, precision_score, recall_score (sklearn): Métricas de desempenho do modelo.
- faker.Faker: Gera dados fictícios para testes.
- random: Gera números e seleções aleatórias.



Composição do protótipo

O nosso modelo protótipo é composto por dois componentes principais, um programa em Python e um Notebook do Pandas, feito no Google Colab

A lógica em Python “puro”, é responsável por gerar nossos dados para testes. utilizando a biblioteca Faker é possível gerar dados fictícios, combinamos com as funcionalidades da lib random para diversificar e randomizar os dados gerados e podemos escolher a quantidade que quisermos, basta alterar o valor da variável. Executamos tudo isso dentro de uma função que posteriormente é chamada passando a quantidade de linhas de dados como parâmetro. Após a geração dos dados eles são armazenados em um dataframe com colunas dedicadas e importado para um arquivo csv.

O arquivo csv gerado é consumido pelo nosso segundo componente, o notebook feito no Google Colab. Dentro dele são feitas todas as análises e treinamentos dos dados.

```
dataset_generator.py > ...
dataset_generator.py > ...
YOU, 4 months ago | 1 author (You)
1 import pandas as pd
2 from faker import Faker
3 import random
4
5 fake = Faker()
6
7 def generate_data(num_records):
8     data = []
9     for _ in range(num_records):
10
11         costumer_id = fake.random_int(min=1000, max=9999)
12         age = fake.random_int(min=18, max=80)
13         gender = random.choice(['M', 'F'])
14         location = random.choice(['SP', 'RJ', 'BA', 'DF', 'MG', 'RS', 'SC', 'PR', 'AM', 'PA'])
15         purchase_amount = round(random.uniform(20.0, 10000.0), 2)
16         product_category = random.choice(['Eletronicos', 'Livros', 'Saúde', 'Moda', 'Esportes', 'Beleza', 'Casa', 'Brinquedos', 'Games'])
17         clicked_ad = random.choice(['Sim', 'Não'])
18         browsing_history = [fake.url() for _ in range(random.randint(1, 10))]
19         interested_score = round(random.uniform(0.5, 1.0), 2)
20         churn_risk = random.choice(['Low', 'Medium', 'High'])
21
22         data.append([costumer_id, age, gender, location, purchase_amount, product_category, clicked_ad, browsing_history, interested_score, churn_risk])
23
24     return data
25
26 num_records = 10000
27 data = generate_data(num_records)
28
29 columns = [
30     'Customer_ID',
31     'Age',
32     'Gender',
33     'Location',
34     'Purchase_Amount',
35     'Product_Category',
36     'Clicked_Ad',
37     'Browsing_History',
38     'Interest_Score',
39     'Churn_Risk'
40 ]
41
42 df = pd.DataFrame(data, columns=columns)
43
44 df.to_csv('costumer_behavior.csv', index=False)
45
```

print('Arquivo gerado: costumer_behavior.csv') You, 4 months ago • adicionando arquivos do projeto

GD-CostumerBehavior.ipynb ★

Arquivo Editar Ver Inserir Ambiente de execução Ferramentas Ajuda Última edição em 19 de maio

+ Código + Texto

{x}

CO

[] import pandas as pd
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score

▼ Importando dados

O arquivo csv estará disponível no repositório do projeto

[] dados = pd.read_csv('costumer_behavior.csv')
dados.head()

	Customer_ID	Age	Gender	Location	Purchase_Amount	Product_Category	Clicked_Ad	Browsing_History	Interest_Score	Churn_Risk
0	1462	21	F	MG	3308.26	Beleza	Não	[http://zuniga.net/, http://www.hoffman-dun...]	0.67	Low
1	4465	57	F	SC	3081.58	Games	Não	[https://martin.com/, https://mills.com/, ...]	0.62	Low
2	2001	27	F	RS	563.87	Eletronicos	Não	[http://price-joyce.com/, https://sims.com/...]	0.85	High
3	8886	66	M	SC	6858.62	Saúde	Não	[http://hudson.org/, http://hernandez.biz/...]	0.60	Medium
4	5215	59	F	AM	7410.20	Games	Sim	[http://ray.com/, https://www.smith.info/, ...]	0.79	High

[] # Tamanho do conjunto de dados
dados.shape
print(f'O conjunto de dados tem {dados.shape[0]} linhas e {dados.shape[1]} colunas.')

O conjunto de dados tem 10000 linhas e 10 colunas.

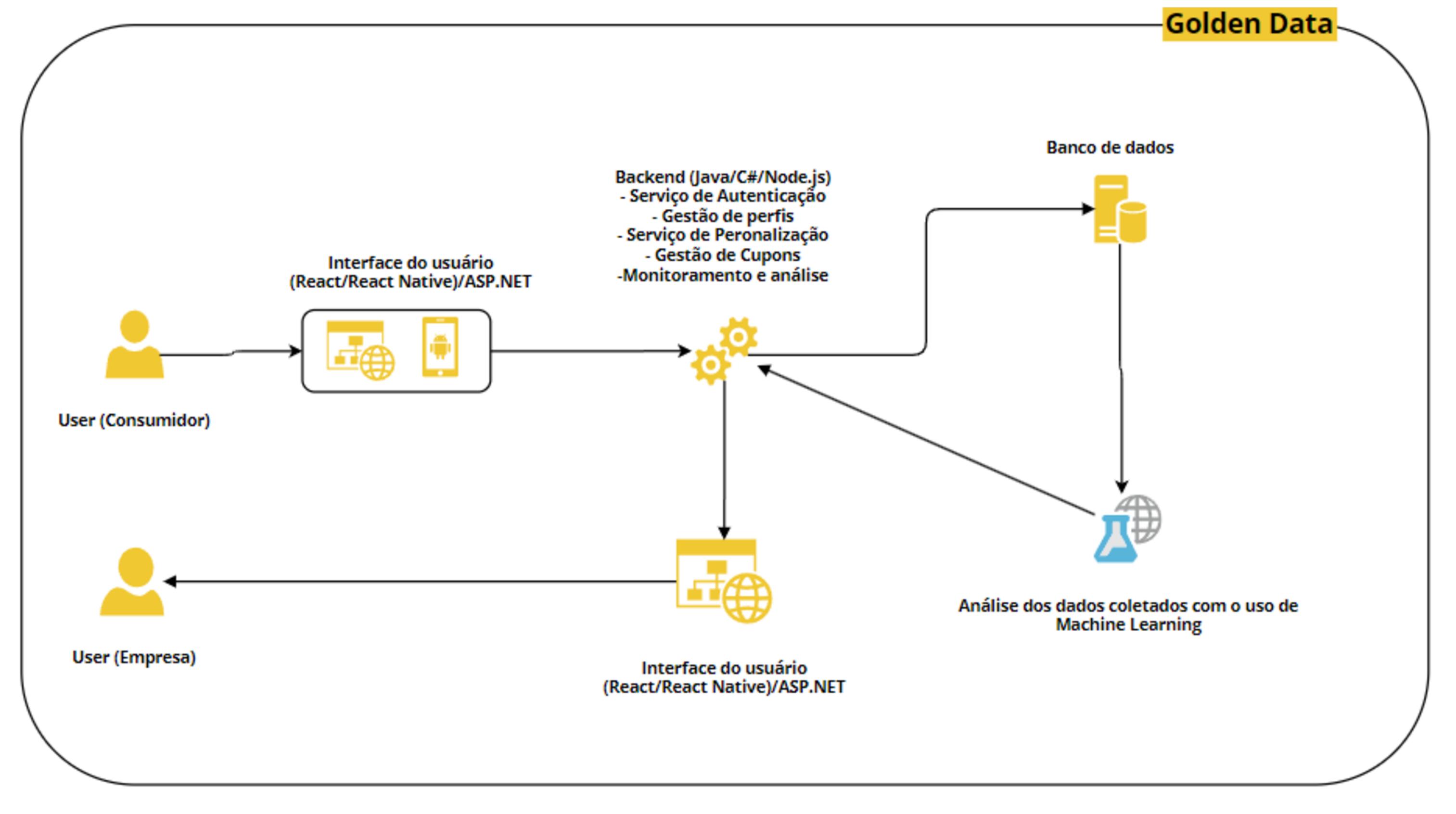
[] # Nome das colunas
dados.columns

[] Index(['Customer_ID', 'Age', 'Gender', 'Location', 'Purchase_Amount', 'Product_Category', 'Clicked_Ad', 'Browsing_History', 'Interest_Score', 'Churn_Risk'], dtype='object')

[] # Estatísticas descritivas das colunas numéricas
dados.describe()

	Customer_ID	Age	Purchase_Amount	Interest_Score
count	10000.000000	10000.000000	10000.000000	10000.000000
mean	5498.503800	49.071400	5016.003426	0.751862

Arquitetura da plataforma



1. Usuários (Consumidor e Empresa)

- Existem dois tipos de usuários:
 - User (Consumidor): Este grupo de usuários interage com a plataforma para indicar seus interesses e receber cupons e anúncios personalizados de acordo com essas preferências.
 - User (Empresa): As empresas utilizam a plataforma para acessar os dados fornecidos pelos consumidores, criando campanhas personalizadas e enviando cupons e ofertas diretamente aos consumidores.

2. Interface do Usuário (React/React Native/Kotlin/ASP.NET)

- Mobile (Kotlin/React Native):
 - A parte mobile da interface do usuário será desenvolvida com Kotlin (para aplicativos nativos Android) e React Native (para uma experiência multiplataforma). A escolha de Kotlin para Android permite aproveitar os recursos nativos da plataforma, garantindo desempenho otimizado.
 - React Native será usado para fornecer uma interface que funcione em diversas plataformas móveis, incluindo iOS e Android.
- Web (ASP.NET):
 - No ambiente web, a interface será desenvolvida usando ASP.NET, fornecendo uma interface robusta para empresas e consumidores interagirem com a plataforma diretamente de navegadores.

3. Backend (Java/C#/Node.js)

- O backend é responsável por toda a lógica de negócios e é desenvolvido com Java, C# ou Node.js. Ele suporta múltiplos serviços, incluindo:
 - Serviço de Autenticação: Garante que apenas usuários registrados possam acessar a plataforma, diferenciando o login de consumidores e empresas.
 - Gestão de Perfis: Armazena e gerencia dados de perfil dos consumidores, como preferências, e os perfis das empresas, incluindo suas campanhas de anúncios.
 - Serviço de Personalização: Baseado nas preferências fornecidas pelos consumidores, ele cria e oferece recomendações e cupons personalizados.
 - Gestão de Cupons: Gerencia o ciclo de vida dos cupons gerados pelas empresas e enviados para os consumidores.
 - Monitoramento e Análise: Este serviço monitora as interações e a atividade dentro da plataforma, fornecendo insights para otimização do serviço.

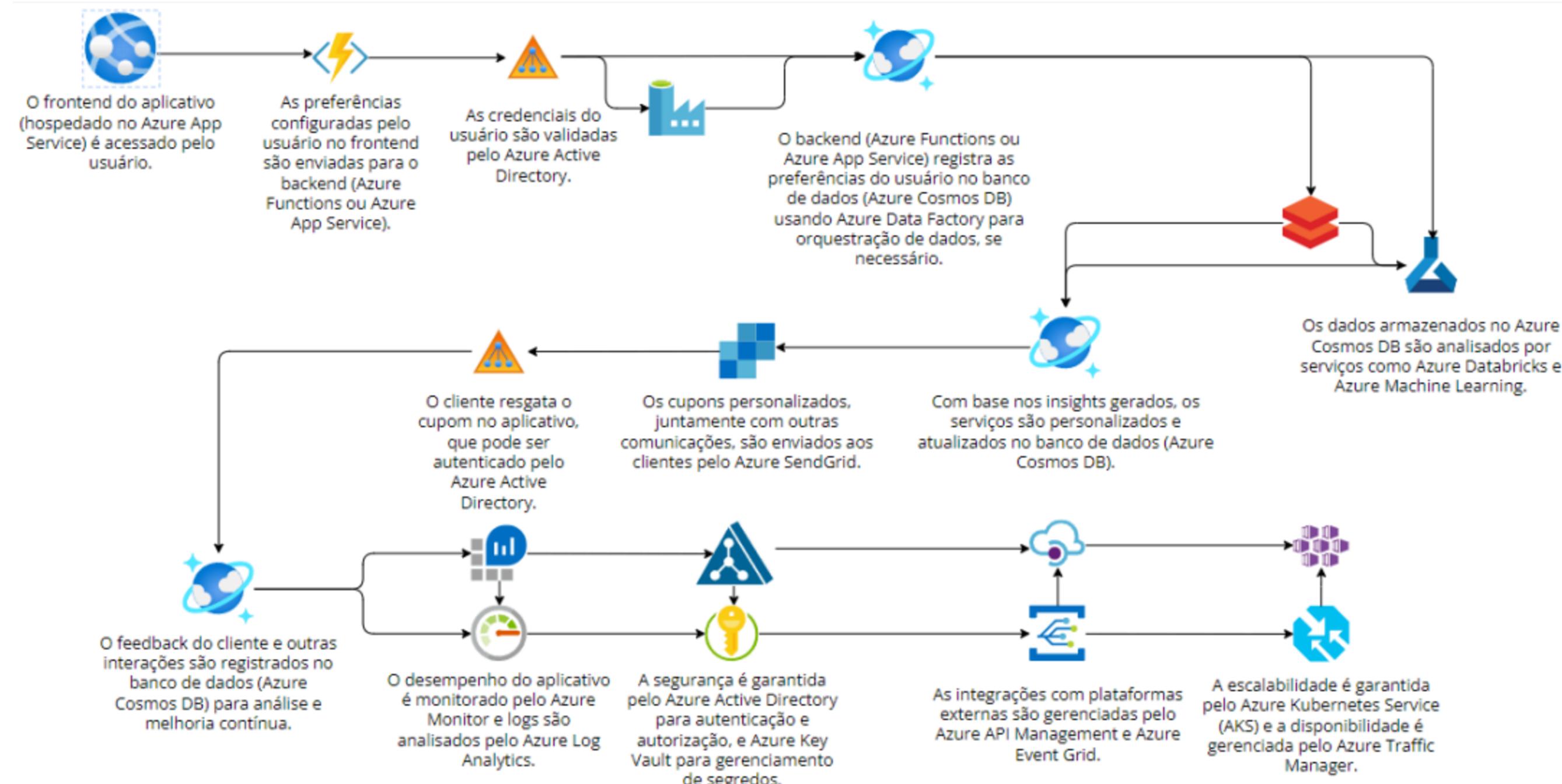
4. Banco de Dados

- O banco de dados armazena as informações dos perfis dos consumidores e das empresas, além de dados relacionados aos cupons e atividades na plataforma. Esse componente é essencial para fornecer as informações necessárias ao backend para que a personalização de serviços seja eficiente.

5. Análise de Dados com Machine Learning

- A análise de dados é realizada com técnicas de Machine Learning que processam as informações coletadas, buscando padrões nos interesses e comportamentos dos consumidores. Essa análise permite que as empresas ofereçam campanhas mais direcionadas e eficazes, com base nos insights gerados.
- A personalização não se limita apenas às preferências explícitas dos consumidores, mas também pode ser ajustada com base nos comportamentos identificados pela análise de dados.

Arquitetura Azure



- Frontend no Azure App Service: O aplicativo é acessado pelo usuário, e as preferências configuradas no frontend são enviadas para o backend.
- Autenticação via Azure Active Directory: As credenciais do usuário são validadas pelo Azure Active Directory, garantindo segurança e controle de acesso.
- Backend no Azure Functions ou App Service: O backend recebe as preferências do usuário e as armazena no banco de dados Azure Cosmos DB, utilizando Azure Data Factory, se necessário, para orquestrar os dados.
- Análise de dados no Cosmos DB: Os dados armazenados no Cosmos DB são analisados por serviços como Azure Databricks e Azure Machine Learning, para gerar insights.
- Personalização e envio de cupons: Com base nos insights gerados, os cupons personalizados são enviados aos clientes via Azure SendGrid.
- Escalabilidade e integração com outros serviços: O sistema é escalável usando Azure Kubernetes Service (AKS), e a disponibilidade é gerenciada pelo Azure Traffic Manager. As integrações com plataformas externas são gerenciadas pelo Azure API Management e Azure Event Grid.
- Monitoramento e segurança: O desempenho do aplicativo é monitorado pelo Azure Monitor e Azure Log Analytics, enquanto a segurança é reforçada com Azure Key Vault para gerenciamento de segredos.
- Feedback do cliente e melhorias contínuas: O feedback do cliente e interações são armazenados no Azure Cosmos DB para análise e melhoria contínua.