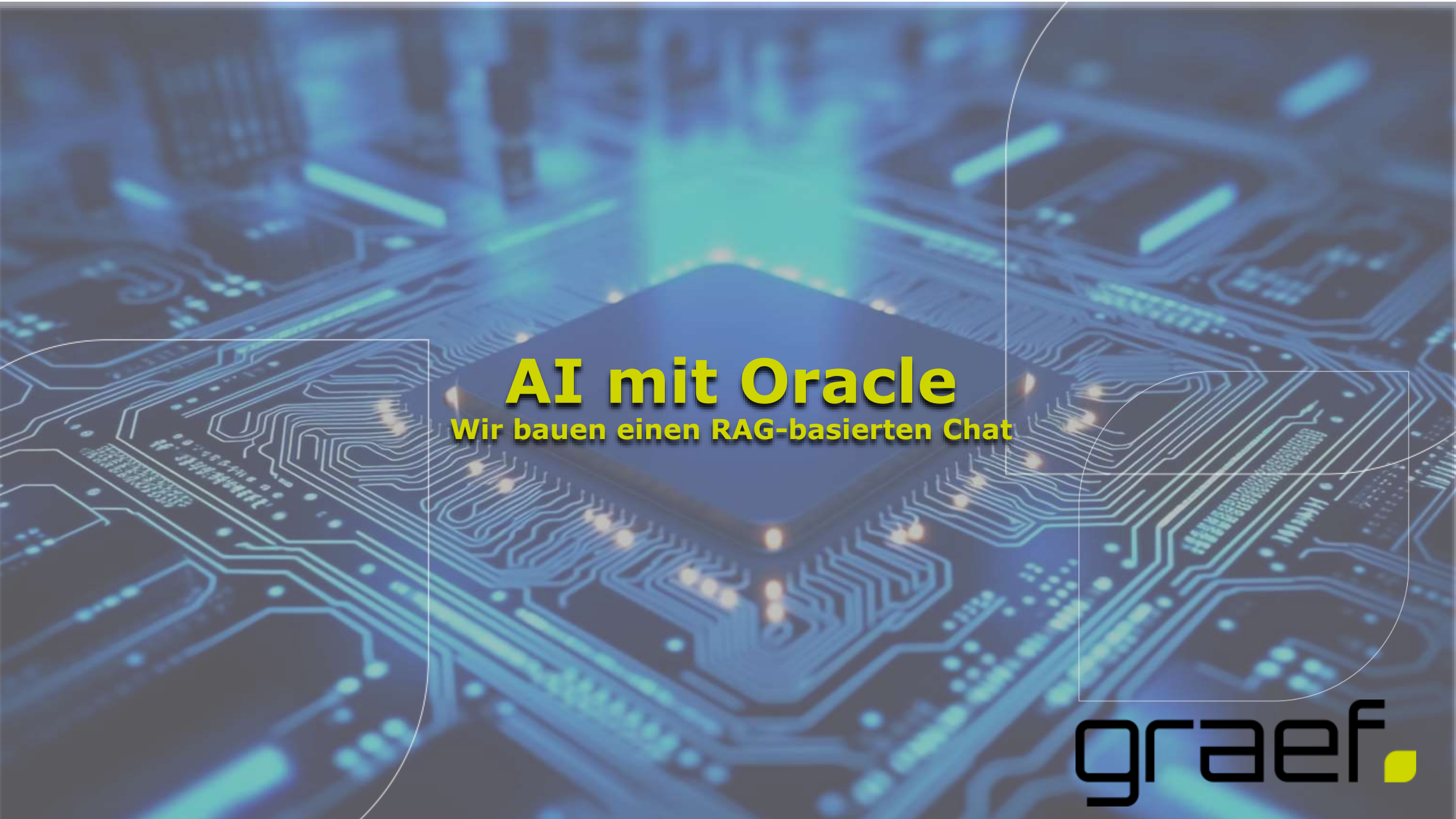
AI mit Oracle
Wir bauen einen RAG-basierten Chat
graef.

Warum noch einer?

... wo es schon viele Demos u.a. von Oracle gibt?

- Bisherige Lösungen setzen meistens auf Frontend-Frameworks auf
  - Integrierbarkeit der Lösungen in andere, ggf. schon bestehende Frameworks schwierig (SSO, separate Webseiten, Fat Clients)
  - Verwendung der Oracle oft nur als Vektor-Datenspeicher für Vektor-Metrik
- Backend Implementation als PL/SQL
  - Unabhängig von Frontend-Plattform
  - Minimaler Technologie Stack
  - Keine zusätzlichen Frameworks nötig, die in bestehende Frontends integriert werden müssten
  - Keine weiteren vermeidbaren* Lizenz- oder Hostingkosten
    (* die OpenAI-API-Calls sind unvermeidbar kostenpflichtig)
  - Einfache Integrierbarkeit in bestehende Lösungen mit bestehender Oracle-DB
  - **Keine** Cloud-Lösung, alles (außer ChatGPT) on Prem möglich
- Eingesetzte Systeme und Techniken:
  - Oracle 23AI (free, on Prem) mit PL/SQL
  - OpenAI Account für ChatGPT-API-Nutzung
  - Optional: Python, Docker

graef

Basics - Wir reden mit der KI

Zwei mögliche Implementationen:

1. Mit dbms_vector_chain.utl_to_generate_text
   (setzt Oracle 23AI voraus)

   + Einfache Implementation, schnelle Umsetzung

   - Wenig Flexibilität

   - Keine Telemetriedaten

2. Mit UTL_HTTP implementiertes REST-Interface

   - Komplexere Implementation; aber nur einmal

   + Maximale Flexibilität

   + Telemetriedaten

   + Funktioniert auch auf älteren Versionen wie Oracle 19c

graef.

Variante 1: Implementation mit Oracle 23AI Bordmitteln

Vorarbeiten:

- **OpenAI Account anlegen, Projekt erstellen und API-Key erzeugen**
- **DBA: ACL für Zugriff auf API von OpenAI**
- **DBA: GRANT create credential TO <user>;**
- **Credential aus API-Key erzeugen**

Implementation:

- **Parametertabelle für LLM Modelle anlegen und befüllen**
- **Implementation Funktion mit UTL_TO_GENERATE_TEXT (Beispiel auf oracle.com)**

Test und Bewertung

- Funktioniert nur mit Oracle 23AI
- Funktioniert nur mit von Oracle vorgesehenen LLMs
- Keine „Spezialitäten" der APIs

graef.

Variante 2: Verwendung von UTL_HTTP

Vorarbeiten: **(* ggf. Schon in Variante 1 erledigt)**

- **OpenAI Account mit Projekt und API-Key besorgen***
- **DBA: ACL für Zugriff auf API von OpenAI***
- **DBA: Wallet erzeugen**

Implementation:

- **Parametertabelle für LLM Modelle anlegen und befüllen***
- **Implementation Funktion wie in https://oracle-base.com/articles/misc/utl_http-and-ssl mit Anpassungen für OpenAI-JSON Content (siehe https://platform.openai.com/docs/overview)**

Test und Bewertung

- Funktioniert auch z.B. mit Oracle 19c
- Individuelle Anpassungen an LLMs möglich, z.B. Telemetriedaten über Tokenverbrauch

graef.

Was kommt als nächstes?

Bisherige Implementation unflexibel:

- **Nachschärfungen der Anfragen nur durch Umformulierungen**
- **Kein „Dialog“**
- **Ergebnisse zum Teil nicht reproduzierbar, daher keine Weiterentwicklung der Ergebnisse**

Nächste Schritte:

- Implementation eines „Chats“ durch Historienverwendung

Graef Computer GmbH · Fallgatter 5 · 44369 Dortmund
Telefon: +49 231 222 429 - 99 · info@graef.com