weGROUP DeepAgent Platform - Architektur-Dokumentation

Überblick

Die weGROUP DeepAgent Platform ist eine Multi-Tenant AI-Orchestration Plattform, die auf einer modernen, skalierbaren Architektur basiert.

Multi-Tenant-Architektur

Mandanten-Struktur

Die Plattform unterstützt 8 vorkonfigurierte Mandanten:

1. **weGROUP** (Master-Mandant)
   - Zentrale Verwaltung
   - Super-Admin-Funktionen
   - Mandantenübergreifende Berichte

2. **weANALYTICS**
   - Datenanalyse und Business Intelligence
   - Advanced Analytics Dashboard
   - Predictive Analytics

3. **weFINANCE**
   - Finanzmanagement
   - Budgetierung und Forecasting
   - Expense Tracking

4. **wePROJECT**
   - Projektmanagement
   - Resource Planning
   - Team Collaboration

5. **weHR**
   - Human Resources Management
   - Employee Analytics
   - Performance Tracking

6. **weSALES**
   - Sales Management
   - CRM Integration
   - Sales Analytics

7. **weMARKETING**
   - Marketing Campaigns
   - Lead Management
   - Marketing Analytics

8. **weOPERATIONS**
   - Operational Excellence
   - Process Optimization
   - Supply Chain Management

Datenbank-Schema

```sql
-- Multi-Tenant-Schema
CREATE TABLE tenants (
  id UUID PRIMARY KEY,
  name VARCHAR(255) NOT NULL,
  slug VARCHAR(100) UNIQUE NOT NULL,
  settings JSONB,
  created_at TIMESTAMP DEFAULT NOW()
);

-- Benutzer mit Mandanten-Zuordnung
CREATE TABLE users (
  id UUID PRIMARY KEY,
  email VARCHAR(255) UNIQUE NOT NULL,
  tenant_id UUID REFERENCES tenants(id),
  role user_role NOT NULL,
  permissions JSONB,
  created_at TIMESTAMP DEFAULT NOW()
);

-- Rollen-Hierarchie
CREATE TYPE user_role AS ENUM (
  'SUPER_ADMIN',
  'TENANT_ADMIN',
  'MANAGER',
  'TEAM_LEAD',
  'USER',
  'VIEWER'
);
```

Benutzerrollen-System

Hierarchie (6 Stufen)

1. **SUPER_ADMIN**
   - Vollzugriff auf alle Mandanten
   - System-Administration
   - Mandanten-Management

2. **TENANT_ADMIN**
   - Vollzugriff auf eigenen Mandanten
   - Benutzerverwaltung
   - Mandanten-Konfiguration

3. **MANAGER**
   - Erweiterte Berechtigungen
   - Team-Management
   - Reporting-Zugriff

4. **TEAM_LEAD**
   - Team-spezifische Verwaltung
   - Projekt-Koordination
   - Basis-Reporting

5. **USER**
   - Standard-Benutzerrechte
   - Eigene Daten verwalten
   - Basis-Funktionen

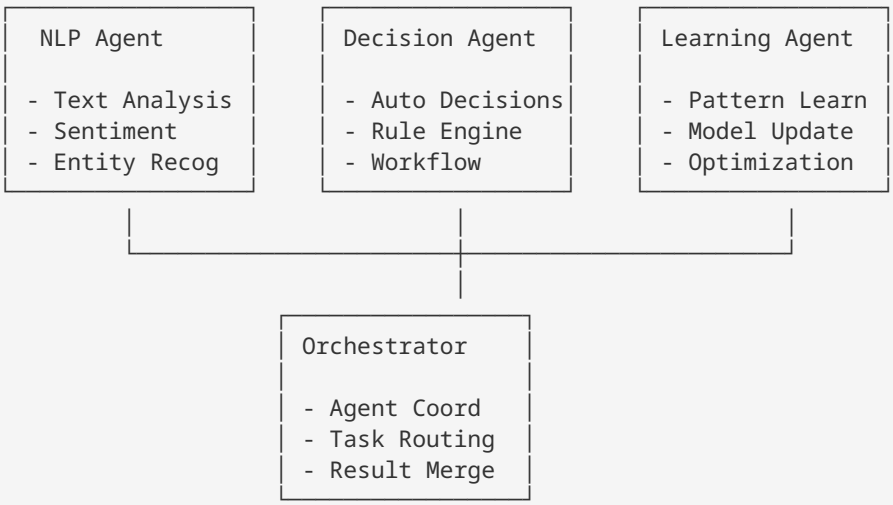
6. **VIEWER**
   - Nur-Lese-Zugriff
   - Dashboard-Ansicht
   - Keine Bearbeitungsrechte

Berechtigungsmatrix

Funktion	SUPER_ADMIN	TENANT_ADMIN	MANAGER	TEAM_LEAD	USER	VIEWER
System-Admin						
Mandant-Admin						
User-Management						
Reporting						
Data-Export						
AI-Features						

AI-Architektur

Multi-Agent-System



AI-Services

1. **Advanced NLP Service**
   - Textverarbeitung
   - Sentiment-Analyse
   - Entity Recognition
   - Sprachübersetzung

2. **Multi-Agent AI Service**
   - Agent-Koordination
   - Task-Distribution
   - Result-Aggregation

3. **Autonomous Decision Service**
   - Regelbasierte Entscheidungen
   - ML-basierte Vorhersagen
   - Workflow-Automatisierung

4. **TensorFlow Client Service**
   - Model-Serving
   - Batch-Prediction
   - Real-time Inference

5. **Voice Command Service**
   - Speech-to-Text
   - Intent Recognition
   - Voice-UI Integration

6. **Self-Learning Service**
   - Continuous Learning
   - Model-Updates
   - Performance-Monitoring

Performance-Architektur

Optimierungsstrategien

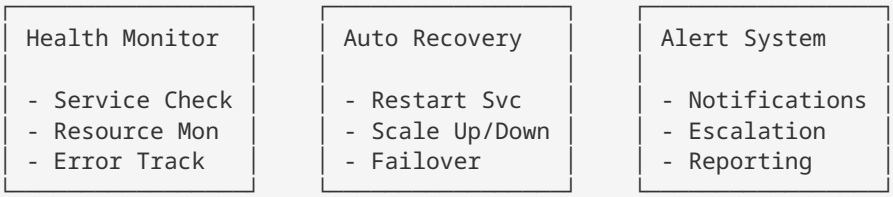
1. **Caching-Layer**
   - Redis für Session-Cache
   - Application-Level Caching
   - Database Query Caching

2. **Load Balancing**
   - Horizontal Scaling
   - Auto-Scaling Groups
   - Health Checks

3. **Database Optimization**
   - Connection Pooling
   - Query Optimization
   - Index Strategies

4. **CDN Integration**
   - Static Asset Delivery
   - Global Distribution
   - Edge Caching

Self-Healing-Mechanismen



Sicherheitsarchitektur

Zero-Trust-Prinzipien

1. **Identitätsverifikation**
   - Multi-Faktor-Authentifizierung
   - Biometrische Authentifizierung
   - Continuous Authentication

2. **Least-Privilege-Access**
   - Rollenbasierte Berechtigungen
   - Just-in-Time-Access
   - Regular Access Reviews

3. **Verschlüsselung**
   - End-to-End-Verschlüsselung
   - Data-at-Rest-Verschlüsselung
   - Transport-Layer-Security

4. **Monitoring & Auditing**
   - Real-time Security Monitoring
   - Audit Logs
   - Compliance Reporting

API-Architektur

RESTful APIs

```
/api/
├── admin/
│   ├── users/
│   ├── tenants/
│   ├── permissions/
│   └── system/
├── ai/
│   ├── multi-agent/
│   ├── nlp/
│   ├── voice-commands/
│   └── autonomous-decisions/
├── multi-tenant/
│   ├── switch/
│   └── current/
└── graphql/
    └── schema/
```

GraphQL-Integration

- Flexible Datenabfragen
- Real-time Subscriptions
- Type-Safe Schema
- Efficient Data Loading

Deployment-Architektur

Container-Strategie

```dockerfile
# Multi-Stage Build
FROM node:18-alpine AS builder
WORKDIR /app
COPY package*.json ./
RUN npm ci --only=production

FROM node:18-alpine AS runner
WORKDIR /app
COPY --from=builder /app/node_modules ./node_modules
COPY . .
EXPOSE 3000
CMD ["npm", "start"]
```

Kubernetes-Deployment

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: wegroup-deepagent
spec:
  replicas: 3
  selector:
    matchLabels:
      app: wegroup-deepagent
  template:
    metadata:
      labels:
        app: wegroup-deepagent
    spec:
      containers:
      - name: app
        image: wegroup/deepagent:v1.2.0
        ports:
        - containerPort: 3000
```

Monitoring & Observability

Metriken

1. **Application Metrics**
   - Response Times
   - Error Rates
   - Throughput

2. **Business Metrics**
   - User Engagement
   - Feature Usage
   - Conversion Rates

3. **Infrastructure Metrics**
   - CPU/Memory Usage
   - Network I/O
   - Database Performance

Logging-Strategie

- Structured Logging (JSON)
- Centralized Log Aggregation
- Real-time Log Analysis
- Alert-based Monitoring

Diese Architektur gewährleistet Skalierbarkeit, Sicherheit und Performance für Enterprise-Anforderungen.