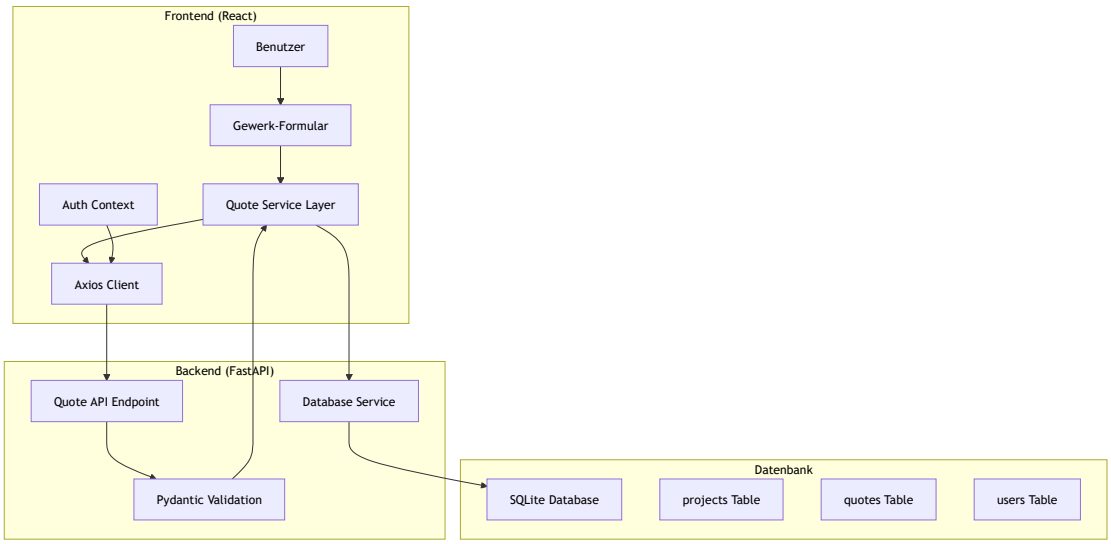
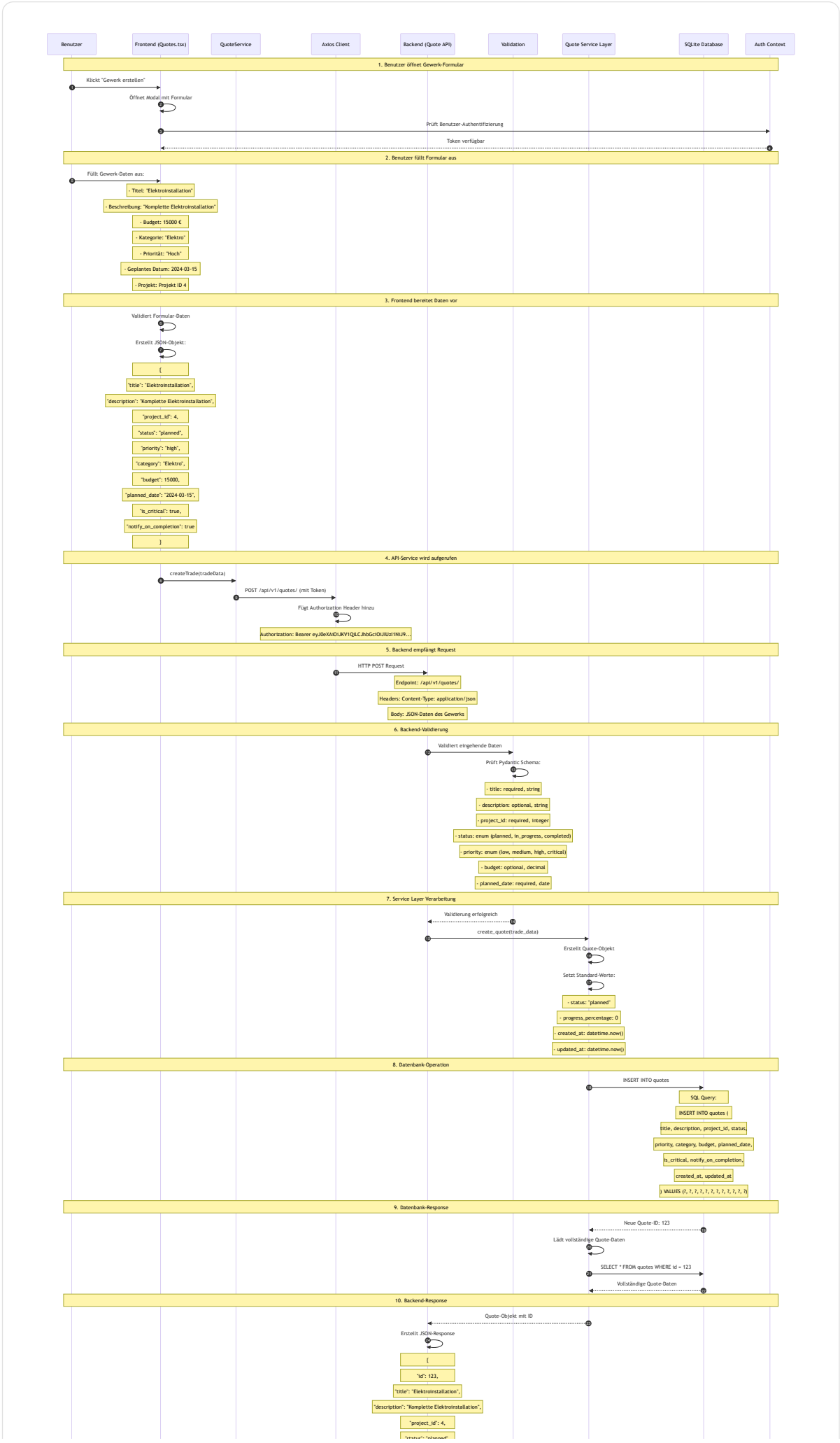
Gewerk-Erstellung: Frontend-Backend Datenfluss

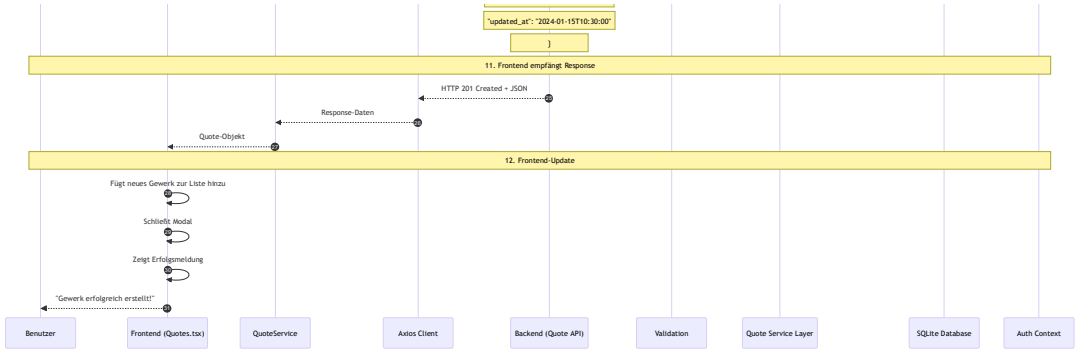
> Diese Dokumentation zeigt den detaillierten Datenfluss beim Anlegen eines Gewerks im BuildWise-System. Von der Benutzerinteraktion im Frontend bis zur Speicherung in der Datenbank.

1. Übersicht des Gewerk-Erstellungsprozesses

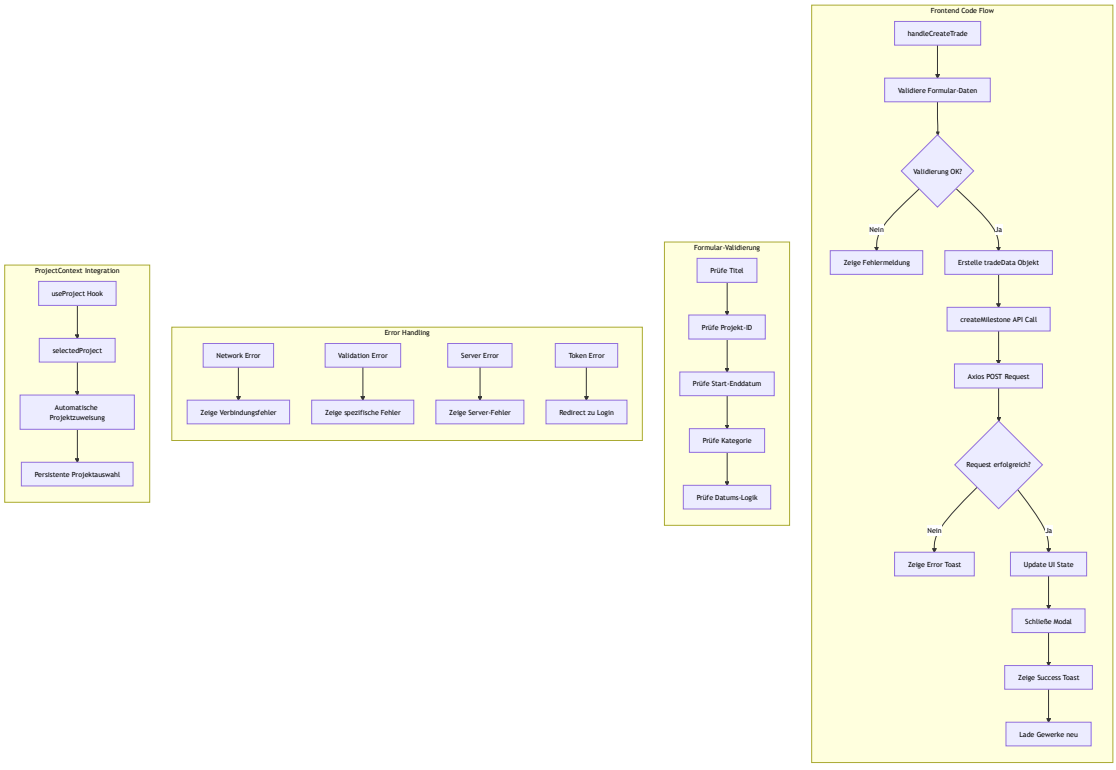


2. Detaillierter Sequenzablauf


Benutzer
Frontend (Quotes.tsx)
QuoteService
Axios Client
Backend (Quote API)
Validation
Quote Service Layer
SQLite Database
Auth Context
1. Benutzer öffnet Gewerk-Formular
Klickt "Gewerk erstellen"
Öffnet Modal mit Formular
Prüft Benutzer-Authentifizierung
Token verfügbar
2. Benutzer füllt Formular aus
Füllt Gewerk-Daten aus:
- Titel: "Elektroinstallation"
- Beschreibung: "Komplette Elektroinstallation"
- Budget: 15000 €
- Kategorie: "Elektro"
- Priorität: "Hoch"
- Geplantes Datum: 2024-03-15
- Projekt: Projekt ID 4
3. Frontend bereitet Daten vor
Validiert Formular-Daten
Erstellt JSON-Objekt:
{
"title": "Elektroinstallation",
"description": "Komplette Elektroinstallation",
"project_id": 4,
"status": "planned",
"priority": "high",
"category": "Elektro",
"budget": 15000,
"planned_date": "2024-03-15",
"is_critical": true,
"notify_on_completion": true
}
4. API-Service wird aufgerufen
createTrade(tradeData)
POST /api/v1/quotes/ (mit Token)
Fügt Authorization Header hinzu
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9...
5. Backend empfängt Request
HTTP POST Request
Endpoint: /api/v1/quotes/
Headers: Content-Type: application/json
Body: JSON-Daten des Gewerks
6. Backend-Validierung
Validiert eingehende Daten
Prüft Pydantic Schema:
- title: required, string
- description: optional, string
- project_id: required, integer
- status: enum (planned, in_progress, completed)
- priority: enum (low, medium, high, critical)
- budget: optional, decimal
- planned_date: required, date
7. Service Layer Verarbeitung
Validierung erfolgreich
create_quote(trade_data)
Erstellt Quote-Objekt
Setzt Standard-Werte:
- status: "planned"
- progress_percentage: 0
- created_at: datetime.now()
- updated_at: datetime.now()
8. Datenbank-Operation
INSERT INTO quotes
SQL Query:
INSERT INTO quotes (
title, description, project_id, status,
priority, category, budget, planned_date,
is_critical, notify_on_completion,
created_at, updated_at
) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
9. Datenbank-Response
Neue Quote-ID: 123
Lädt vollständige Quote-Daten
SELECT * FROM quotes WHERE id = 123
Vollständige Quote-Daten
10. Backend-Response
Quote-Objekt mit ID
Erstellt JSON-Response
{
"id": 123,
"title": "Elektroinstallation",
"description": "Komplette Elektroinstallation",
"project_id": 4,



3. Frontend Code-Ablauf (Quotes.tsx)



Detaillierte Frontend-Implementierung

handleCreateTrade Funktion:

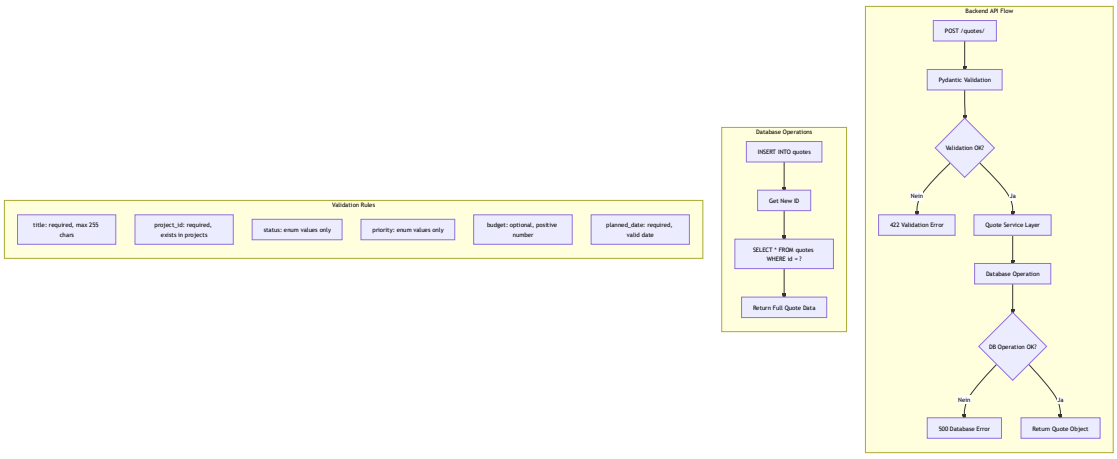
```
const handleCreateTrade = async (e: React.FormEvent) => { e.preventDefault(); // Validierung if (!tradeForm.title.trim()) { setError('Bitte
```

geben Sie einen Titel für das Gewerk ein.'); return; } if (!selectedProject) { setError('Bitte wählen Sie ein Projekt aus.'); return; } // Erstelle tradeData Objekt const tradeData = { project_id: selectedProject, // Aus ProjectContext title: tradeForm.title.trim(), description: tradeForm.description.trim(), status: 'planned', category: category, planned_date: tradeForm.start_date, start_date: tradeForm.start_date, end_date: tradeForm.end_date, priority: tradeForm.priority, is_critical: tradeForm.is_critical, notify_on_completion: true }; // API-Call await createMilestone(tradeData); // UI-Update setShowTradeModal(false); setSuccess('Gewerk erfolgreich erstellt!'); // Lade Gewerke neu const milestonesData = await getAllMilestones(); setTrades(milestonesData); };

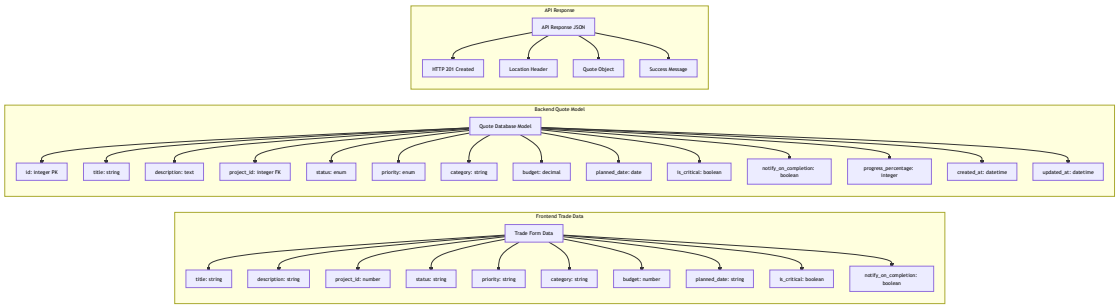
ProjectContext Integration:

const { selectedProject: currentProject } = useProject(); // Für Bauträger: Verwende das aktuell ausgewählte Projekt if (!isServiceProviderUser && currentProject) { console.log('🔧 Using current project from context:', currentProject.id); setSelectedProject(currentProject.id); }

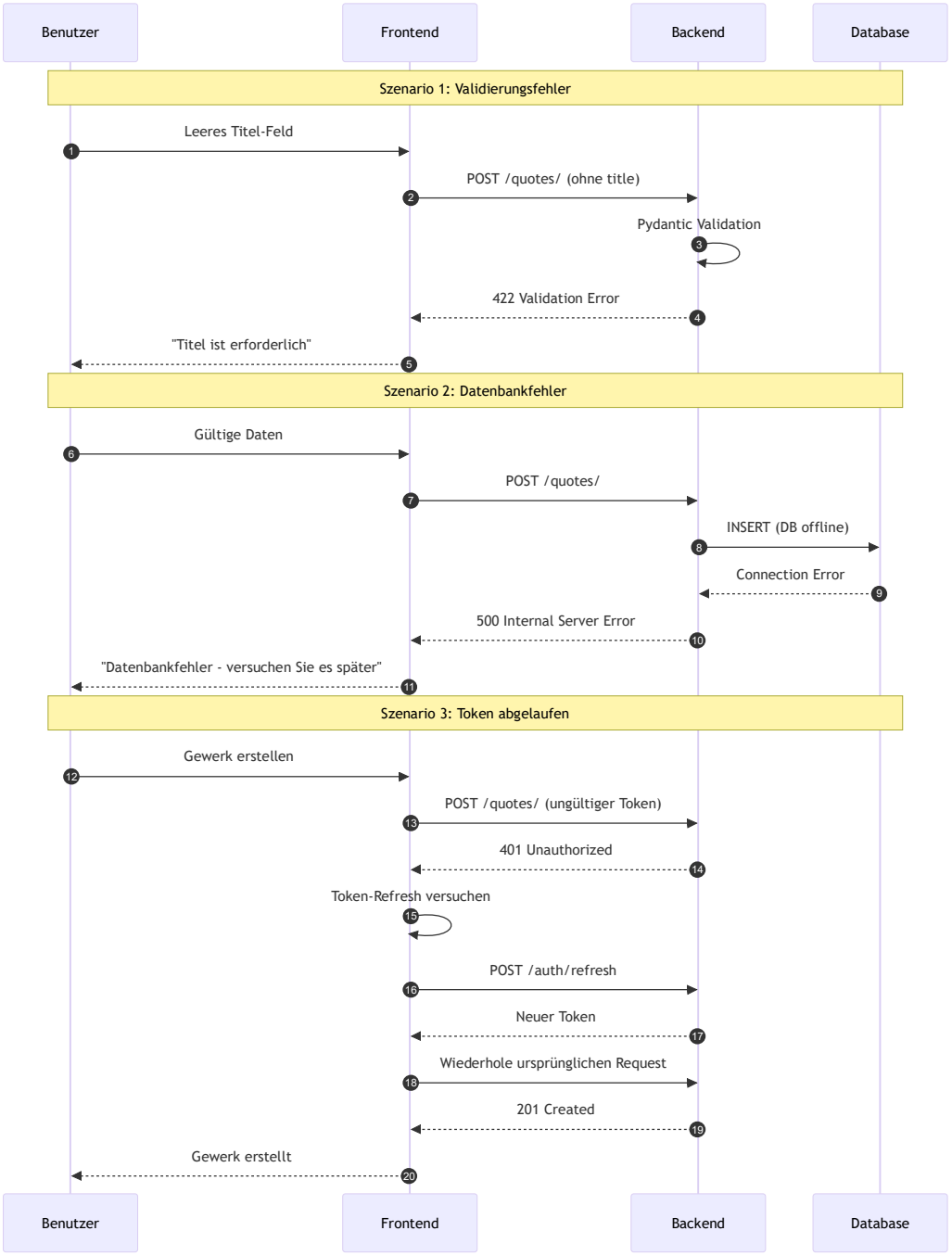
4. Backend Code-Ablauf (Quote API)



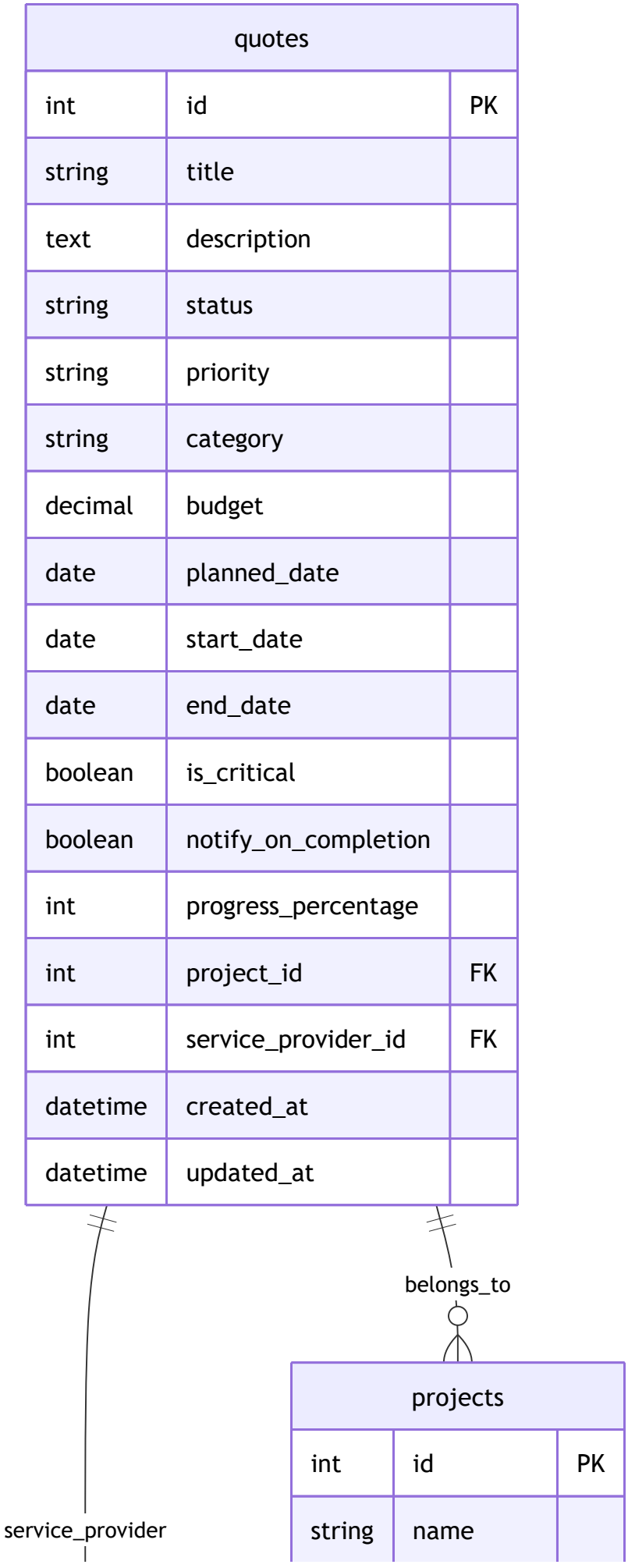
5. Datenstrukturen

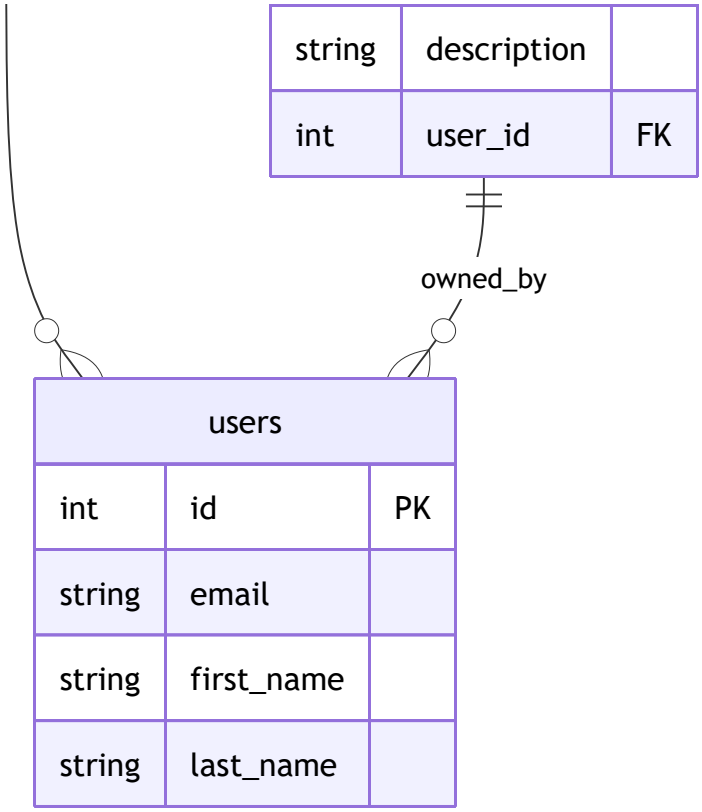


6. Error Handling Szenarien



7. Datenbank-Schema für Quotes


quotes
int id PK
string title
text description
string status
string priority
string category
decimal budget
date planned_date
date start_date
date end_date
boolean is_critical
boolean notify_on_completion
int progress_percentage
int project_id FK
int service_provider_id FK
datetime created_at
datetime updated_at
belongs_to
projects
int id PK
string name
service_provider



Zusammenfassung des Datenflusses

- **Frontend-Initiation:** Benutzer klickt "Gewerk erstellen" → Modal öffnet sich → AuthContext prüft Token → ProjectContext liefert ausgewähltes Projekt
- **Daten-Validierung:** Frontend validiert Formular (Titel, Projekt, Datum, Kategorie) → Erstellt JSON-Objekt → Bereitet API-Request vor
- **ProjectContext Integration:** Automatische Projektzuweisung aus globaler Projektauswahl → Persistente Projektauswahl über Neuladen hinweg
- **API-Kommunikation:** Axios sendet POST-Request mit Token → Backend validiert mit Pydantic → Service Layer verarbeitet
- **Datenbank-Operation:** SQLAlchemy führt INSERT aus → Neue Quote-ID wird generiert → Vollständige Daten werden zurückgegeben
- **Response-Handling:** Backend sendet 201 Created + JSON → Frontend aktualisiert UI-State → Erfolgsmeldung wird angezeigt
- **Error-Handling:** Validierungsfehler (422), Datenbankfehler (500), Token-Fehler (401) mit automatischem Refresh
- **UI-Update:** Modal schließt sich → Gewerke-Liste wird neu geladen → Success-Toast wird angezeigt

Beispiel-Daten

Frontend → Backend Request:

{ "title": "Elektroinstallation", "description": "Komplette Elektroinstallation", "project_id": 4, "status": "planned", "priority": "high", "category": "Elektro", "budget": 15000, "planned_date": "2024-03-15", "is_critical": true, "notify_on_completion": true }

Backend → Frontend Response:

{ "id": 123, "title": "Elektroinstallation", "description": "Komplette Elektroinstallation", "project_id": 4, "status": "planned", "priority": "high", "category": "Elektro", "budget": 15000.00, "planned_date": "2024-03-15", "is_critical": true, "notify_on_completion": true, "progress_percentage": 0, "created_at": "2024-01-15T10:30:00", "updated_at": "2024-01-15T10:30:00" }