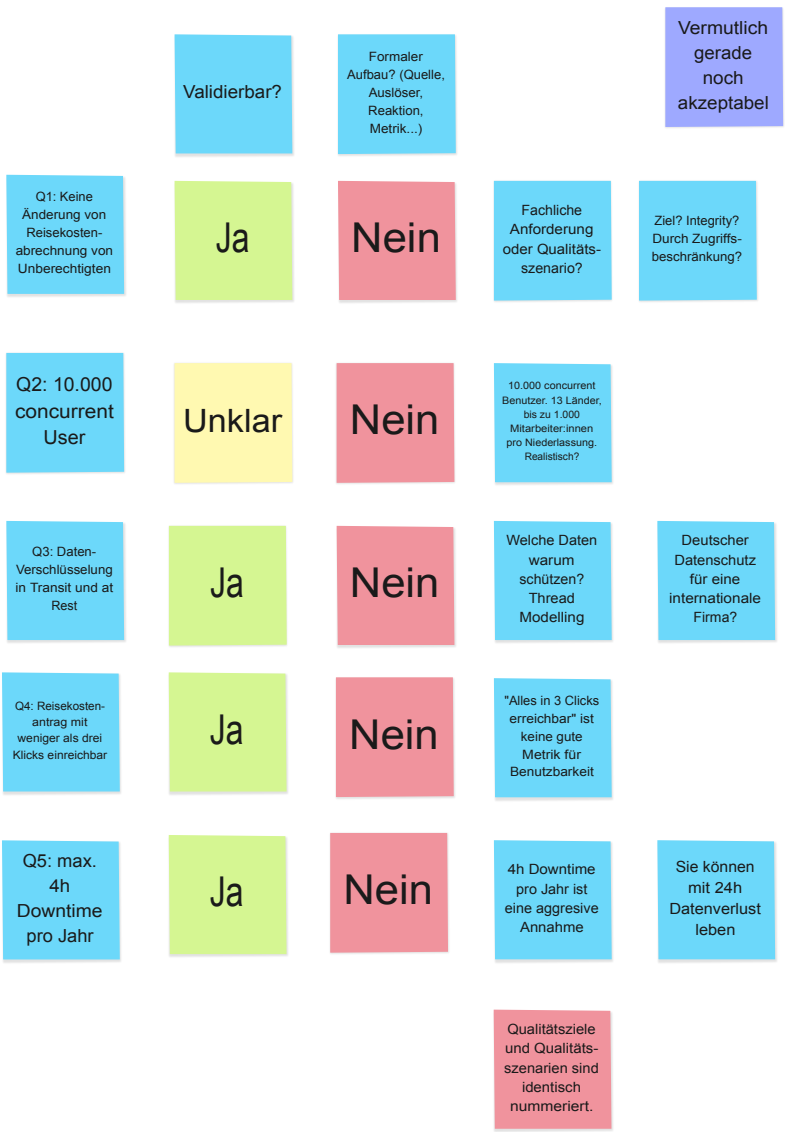
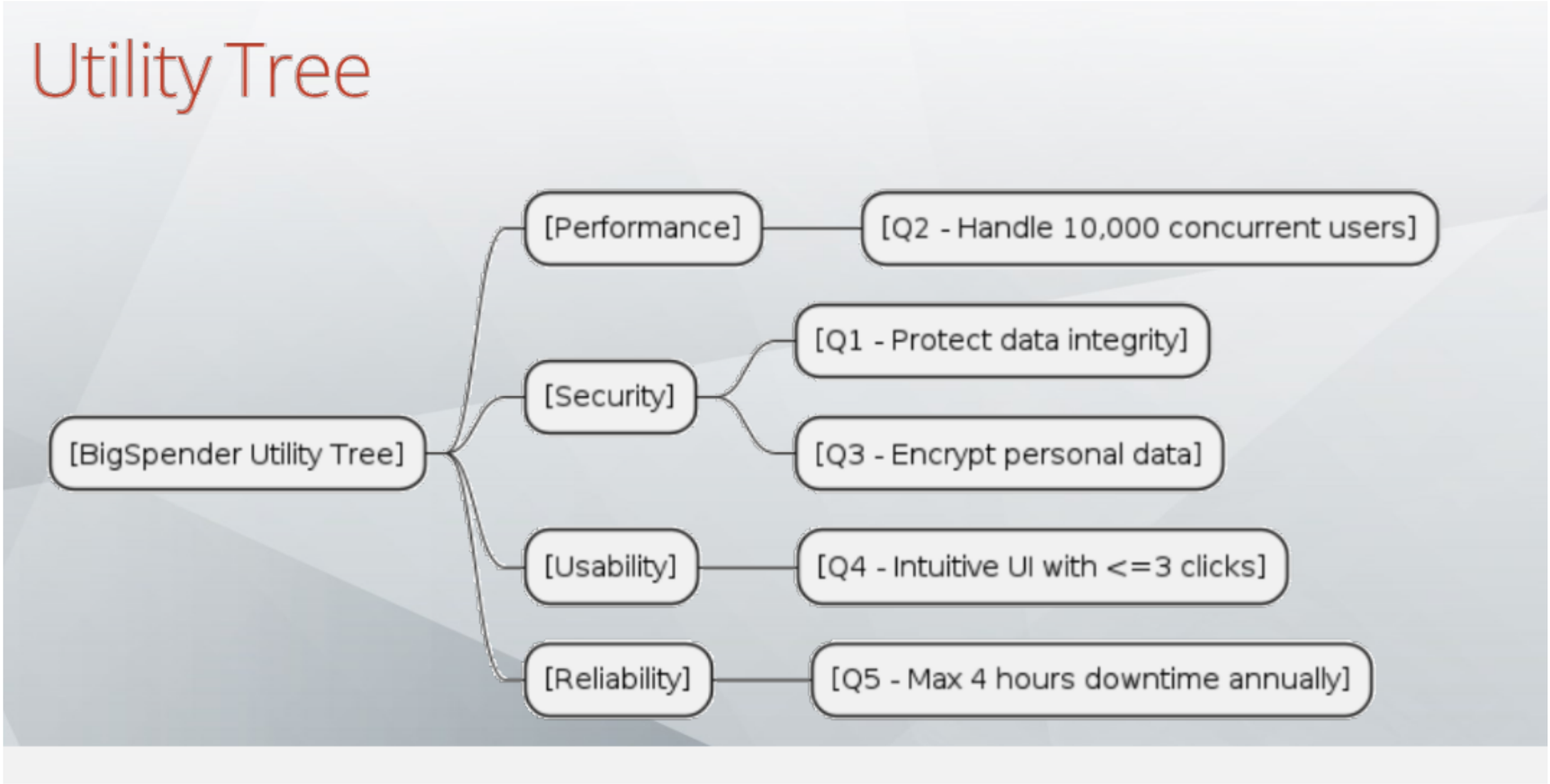
Aufgabe 1 - Qualität

Quality Scenarios

ID	Quality Scenario	Importance
Q1	To maintain a high level of data integrity, the system must ensure that expense entries cannot be altered by unauthorized persons.	High
Q2	The system should handle up to 10,000 users concurrently without performance degradation.	High
Q3	Personal data must be encrypted in transit and at rest to protect against unauthorized access.	High
Q4	The user interface must allow users to submit expenses in no more than three clicks from the dashboard.	Medium
Q5	The system's annual downtime should not exceed 4 hours to ensure continuous business operations.	High



Qualitätsbaum / Qualitätsziele



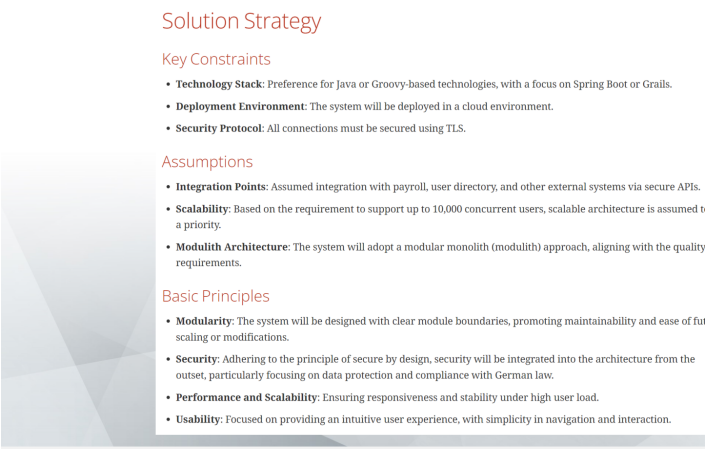
Anpassbarkeit (Länder, Übernahmen, Gesetze) nicht betrachtet.

Korrektheit nicht betrachtet

Diese zwei Ziele sind zusammen mit Benutzerfreund-lichkeit wesentliche Ziele. Daher ist die Lösung eher enttäuschend.

Aufgabe 2 - Lösungsstrategie

Solution Strategy Results



Solution Strategy

Key Constraints

- **Technology Stack**: Preference for Java or Groovy-based technologies, with a focus on Spring Boot or Grails.
- **Deployment Environment**: The system will be deployed in a cloud environment.
- **Security Protocol**: All connections must be secured using TLS.

Assumptions

- **Integration Points**: Assumed integration with payroll, user directory, and other external systems via secure APIs.
- **Scalability**: Based on the requirement to support up to 10,000 concurrent users, scalable architecture is assumed t a priority.
- **Modulith Architecture**: The system will adopt a modular monolith (modulith) approach, aligning with the quality requirements.

Basic Principles

- **Modularity**: The system will be designed with clear module boundaries, promoting maintainability and ease of fut scaling or modifications.
- **Security**: Adhering to the principle of secure by design, security will be integrated into the architecture from the outset, particularly focusing on data protection and compliance with German law.
- **Performance and Scalability**: Ensuring responsiveness and stability under high user load.
- **Usability**: Focused on providing an intuitive user experience, with simplicity in navigation and interaction.

Formal fein

Vermutlich gerade noch akzeptabel

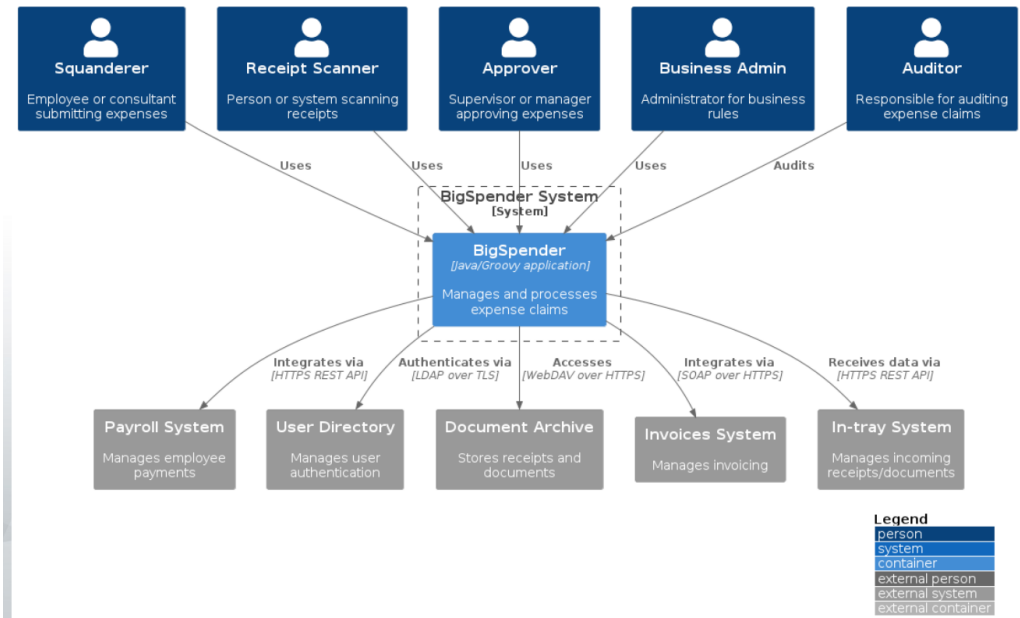
Inhaltlich akzeptabel

Keine Beziehung zu den Qualitäten

Teilweise menschliche Vorgaben

Eigentlich würde man hier ggf. Annahmen erwarten

Aufgabe 3 - technisches Kontextdiagramm



Formal fein: Grafik und Beschreibung vorhannden

Technische Implementierung der UIs fehlen (Web, mobile...)

Vermutlich gerade noch akzeptabel

Zugekaufte Teile bzw. Open Source Frameworks fehlen

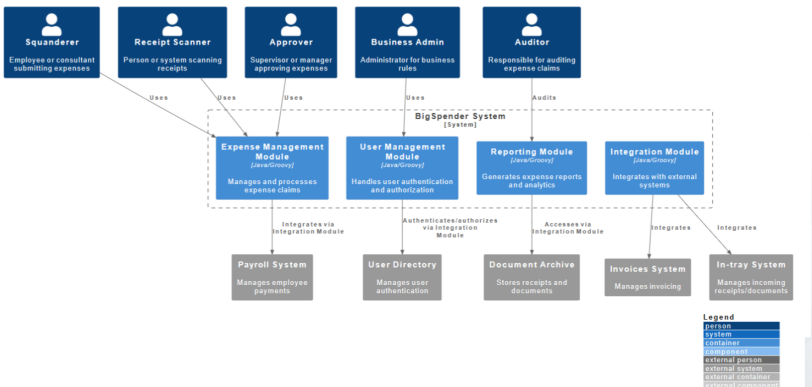
(Man kann diskutieren, was Bestandteile des Systems in einem Kontext-Diagram zu suchen haben.)

Aufgabe 4 - fachliche Strukturierung

You

Let's break our system down to components. Develop and visualize a rough business structuring of the BigSpender system. Use plantUML to do so. Display all external components from sub-task 3 (actors and external systems) in the diagram.

‹ 2 / 2 ›



Formal: Kontext und Aufteilung vermischt - das war aber so vorgegeben.

Spätestens hier ist IMHO eine Überarbeitung nötig. Das ist keine ernsthafte Architektur.

Das System soll Reisekosten verarbeiten. Das ist in einem Modul implementiert. Ist das eine ausreichende Aufteilung?

Die fachlichen Teile sprechen nicht miteinander. Woher bekommt das Reporting seine Daten?

Woher bekommt das In-Tray System die Receipts?

Wie kommen die Expenses in das Invoice System?

IMHO keine sinnvolle fachliche Aufteilung

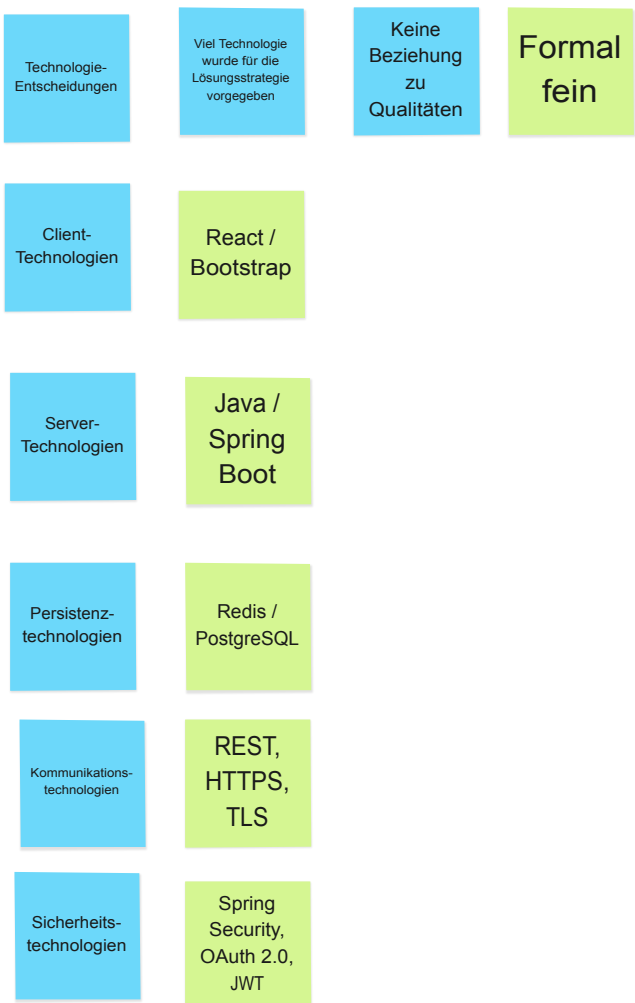
Teilaufgabe 5: Technologie-Entscheidungen

User Interaction: Submitting an Expense Claim

The interaction chosen to demonstrate the technology stack and its alignment with quality goals is "Submitting an Expense Claim."

Technology Stack Overview

- Programming Language: Java with Spring Boot
- Operating System: Linux
- Database Technologies: PostgreSQL and Redis
- Communication Mechanisms: RESTful APIs, HTTPS, and TLS
- Front-End Technologies: React and Bootstrap
- Cloud Infrastructure: AWS
- DevOps Tools: Docker and Jenkins
- Security: Spring Security, OAuth 2.0, JWT
- Monitoring and Logging: ELK Stack



Teilaufgabe 5: Schritt-für-Schritt-Darstellung einer Interaktion

Workflow of Submitting an Expense Claim

What Happens	System Parts Involved	Contribution to Quality Goals
User logs in to submit an expense	React Front-End, Spring Security, User Directory (LDAP over TLS)	Ensures secure access (Security), User-friendly UI (Usability)
User inputs expense details	React Front-End, Bootstrap	Provides an intuitive interface, ensuring data accuracy (Usability)
Expense data submitted to server	Spring Boot Application, RESTful API over HTTPS	Secure transmission of data (Security), Efficient processing (Performance)
Data validated and processed	Spring Boot Application, Business Logic Layer	Ensures data integrity and compliance with business rules (Data Integrity)
Data stored in PostgreSQL	PostgreSQL Database, Transaction Management	Maintains data integrity and consistency (Data Integrity), Reliable data storage (Reliability)
Cache updated in Redis	Redis	Enhances retrieval speed for frequently accessed data (Performance)
Confirmation sent to user	Spring Boot Application, React Front-End	Provides immediate feedback to user (Usability), Ensures system responsiveness (Performance)
Notification sent to Approver	Spring Boot Application, RabbitMQ	Decouples message delivery (Reliability), Asynchronous communication (Performance)
Data available for auditing	Reporting Module, ELK Stack	Facilitates audit processes (Compliance), Effective logging (Reliability)

This workflow demonstrates the "concert of system parts" used to ensure the quality goals. The technology stack has been selected and configured to address the identified quality goals of Security, Usability, Performance, Data Integrity, and Reliability, as defined in subtask 1.

Expense Claim abgeben

Formal fein, schön übersichtlich

Woher kommt RabbitMQ? Ist nicht in der Technologie-Liste

RabbitMQ soll auf ein Qualitätsziel einzahlen.

Kein Knackpunkte

keine wesentlichen Technologien

keine Beziehung zu den drei wesentlichen Qualitätszielen

Teilaufgabe 6 - Bewertung

Top 5 Riskiest and Most Important Quality Scenarios

1. **Data Integrity and Compliance (Q1)**
   - Rationale: Essential for legal compliance and integrity of financial data. Any breach could lead to legal issues and loss of trust.
   - Architectural Decision: Implementing strict access control and audit trails within the system. Using Spring Security for robust authentication and authorization, ensuring that only authorized users can modify data.
   - Trade-offs: This may slightly increase system complexity but is necessary for ensuring compliance and data integrity.
2. **System Performance and Scalability (Q2)**
   - Rationale: Critical for user experience and system reliability, especially during peak times.
   - Architectural Decision: Using a cloud-based infrastructure (AWS) for scalability. Implementing caching mechanisms with Redis to enhance performance.
   - Trade-offs: Additional cost for cloud infrastructure and complexity in cache management, but essential for maintaining performance.
3. **Security of Personal Data (Q3)**
   - Rationale: Data breaches can have severe legal and reputational consequences.
   - Architectural Decision: Encryption of data in transit and at rest, using HTTPS and TLS for secure communication. Implementation of OAuth 2.0 and JWT for secure handling of authentication and authorization tokens.
   - Trade-offs: These security measures may add latency to the system, but they are non-negotiable for ensuring data security.
4. **System Reliability and Availability (Q5)**
   - Rationale: System downtime can disrupt business operations and damage the company's reputation.
   - Architectural Decision: Adopting a microservices architecture for fault isolation, using Docker and Jenkins for consistent deployment, and AWS for high availability.
   - Trade-offs: This approach requires more effort in monitoring and management but is crucial for high availability.
5. **Usability (Q4)**
   - Rationale: Essential for user adoption and productivity. Poor usability can lead to inefficiencies.
   - Architectural Decision: Using React and Bootstrap for creating an intuitive and responsive user interface, ensuring ease of use.
   - Trade-offs: May require more upfront development effort for a user-friendly design, but it is critical for user satisfaction and system adoption.

These scenarios have been selected based on their potential impact on the system's operational effectiveness, legal compliance, user satisfaction, and the business's reputation. The architectural decisions made in each case are aimed at mitigating the risks associated with these scenarios, ensuring the system is robust, compliant, and user-friendly.

Die Lösung bewertet Qualitätsziele keine - szenarien

Die Begrenzung auf fünf ist absurd, weil es nur fünf geben muss.

Da die Qualitätsziele problematisch sind, ist auch die Lösung dieser Aufgabe problematisch.

Die Begründung erscheint mir zu generisch.

Fazit

Die Lösung adressiert wesentliche Herausforderungen (Anpassbarkeit Länder / Gesetze / Merger, Korrektheit) nicht

Dieses Problem bei den Qualitäten in Aufgabe 1 kann im Rest der Architektur nicht mehr korrigiert werden.

Leider erinnert das an zu viele Architektur-entwürfe im wirklichen Leben.

Die Lösung adressiert die wesentliche Herausforderungen Benutzerfreundlichkeit nicht so, dass sie erfolgreich gelöst werden wird.

Die Texte lesen sich alle angenehm und kompetent.