

# Basisvorlage

# Lastenheft

# zur Softwarebeschaffung

Diese Vorlage dient als Grundlage für die Erstellung eines strukturierten Lastenhefts zur Softwarebeschaffung. Sie muss an die spezifischen Anforderungen und Bedürfnisse Ihres Unternehmens und der zu beschaffenden Software angepasst werden. Der Inhalt dieser Vorlage stellt ein grundlegendes Gerüst dar und ist mit kurzen Erläuterungen zu jedem der Punkte versehen. Die Vorlage erhebt keinen Anspruch auf Vollständigkeit. Es ist essenziell, die Vorlage individuell zu erweitern, um die unternehmensspezifischen Geschäftsprozesse, technischen Voraussetzungen und strategischen Ziele abzubilden.

# Lastenheft

**[Unternehmensname]  
[Firmenlogo]**

**Version:** [xx]  
**Datum:** [xx]  
**Verantwortliche Person(en):** [Name, Position]

### Dokumentversionen

|  |  |  |  |
| --- | --- | --- | --- |
| **Versionsnr.** | **Datum** | **Verfassende Person** | **Anmerkung/Änderungsgrund** |
| 0.1 |  |  | Ersterstellung |
| 0.2 |  |  | Überarbeitung |

## **1. Einleitung**

### **1.1 Projektziel**

**Warum ist das wichtig?**  
Das Projektziel ist der zentrale Orientierungspunkt für die gesamte Softwareauswahl. Es beschreibt, welche Verbesserungen, Effizienzsteigerungen oder strategischen Vorteile durch die Einführung der neuen Software erreicht werden sollen. Eine klare und präzise Beschreibung erleichtert später die Kommunikation mit potenziellen Anbietern und stellt sicher, dass die richtige Softwarelösung ausgewählt wird.

**Was sollte hier stehen?**

* Welche Probleme oder Herausforderungen sollen mit der Software gelöst werden?
* Welche unternehmerischen Ziele werden mit der Software verfolgt (z. B. Automatisierung von Prozessen, Kostenersparnis, bessere Datenanalyse)?
* Gibt es bestimmte KPIs (Key Performance Indicators), anhand derer der Erfolg gemessen werden soll?

➡ **Ausführlichkeit: Sehr wichtig**  
Dieser Punkt sollte präzise und klar formuliert werden, da er die Grundlage für die gesamte Anforderungsdefinition bildet.

### **1.2 Projektumfang**

**Warum ist das wichtig?**  
Der Projektumfang grenzt den Rahmen des Softwareprojekts ab und verhindert, dass während der Evaluierung oder Implementierung unerwartete Anforderungen auftreten. Eine zu weit gefasste Definition kann zu Mehraufwand und unnötigen Kosten führen.

**Was sollte hier stehen?**

* Welche Geschäftsbereiche oder Abteilungen werden die Software nutzen?
* Welche Prozesse und Anwendungsfälle werden abgedeckt?
* Was gehört nicht zum Projektumfang (Abgrenzung ist wichtig, um den Fokus zu behalten)?
* Soll die Software nur in einer bestimmten Abteilung oder unternehmensweit eingesetzt werden?

**➡ Ausführlichkeit:** **Mittel bis hoch**. Die Informationen sollten konkret, aber nicht zu detailliert sein – spätere Abschnitte spezifizieren die Anforderungen weiter.

### **1.3 Hintergrund**

**Warum ist das wichtig?**  
Dieser Abschnitt gibt Anbietern einen Einblick in die Ausgangslage und die Beweggründe für die Softwarebeschaffung. Dadurch können sie besser einschätzen, welche Lösung am besten geeignet ist.

**Was sollte hier stehen?**

* Warum wird eine neue Software benötigt? (z. B. bestehende Lösung ist veraltet, unzureichend oder ineffizient)
* Gibt es bereits eine Software im Einsatz, und warum soll sie ersetzt oder ergänzt werden?
* Welche bisherigen Herausforderungen oder Probleme bestehen im aktuellen System?
* Gibt es besondere Unternehmens- oder Marktbedingungen, die die Auswahl beeinflussen (z. B. gesetzliche Anforderungen, Internationalisierung, Digitalisierung)?

**➡ Ausführlichkeit: Kurz und prägnant**. Eine kompakte Zusammenfassung reicht aus, solange die Hauptgründe für die Softwarebeschaffung klar werden.

## **2. Unternehmensspezifische Anforderungen**

Dieser Abschnitt definiert die unternehmensspezifischen Anforderungen an die Software. Hier wird beschrieben, welche Prozesse und Arbeitsabläufe durch die Software unterstützt werden sollen und welche Funktionen unverzichtbar sind. Die Geschäftsprozesse und Schlüsselfunktionen sollten detailliert beschrieben werden, da sie die Kernanforderungen definieren. Die Benutzergruppen können etwas kompakter gehalten werden, sollten aber dennoch klar strukturiert sein. Eine klare Definition hilft Anbietern, maßgeschneiderte Lösungen anzubieten und unnötige Features oder Kosten zu vermeiden.

### **2.1 Geschäftsprozesse**

**Warum ist das wichtig?**  
Die Software sollte sich an den bestehenden Geschäftsprozessen orientieren und diese optimal unterstützen. Eine präzise Beschreibung verhindert, dass die Software später wichtige Anforderungen nicht erfüllt oder aufwändige Anpassungen nötig sind.

**Was sollte hier stehen?**

* Welche Geschäftsprozesse sind für das Unternehmen essenziell und müssen durch die Software abgebildet werden?
* Gibt es standardisierte oder branchenspezifische Abläufe, die unterstützt werden müssen?
* Welche Prozessschritte sind besonders zeitkritisch oder fehleranfällig und sollten automatisiert oder optimiert werden?
* Welche Prozesse sind bereits in Planung, die durch die neue Software abgebildet werden müssen?

**➡ Ausführlichkeit: Sehr wichtig**. Eine detaillierte Beschreibung ist essenziell, um sicherzustellen, dass die Software nahtlos in den Arbeitsalltag integriert werden kann.

### **2.2 Schlüsselfunktionen**

**Warum ist das wichtig?**  
Hier werden die wichtigsten Funktionen definiert, die die Software zwingend bieten muss. Eine klare Priorisierung verhindert, dass Anbieter Lösungen mit unnötigem Funktionsumfang vorschlagen, die den Budgetrahmen sprengen oder unnötig komplex sind.

**Was sollte hier stehen?**

* Welche Kernfunktionen sind für den täglichen Betrieb unerlässlich? (z. B. Berichterstellung, Automatisierung, Datenanalyse)
* Gibt es spezifische Funktionalitäten, die über den Standard hinausgehen?
* Gibt es Anforderungen an die Modularität der Software, damit sie später erweitert bzw. ergänzt werden kann?

**➡ Ausführlichkeit: Sehr wichtig.** Eine detaillierte Auflistung ist essenziell, um passende Angebote zu erhalten und Missverständnisse zu vermeiden.

### **2.3 Benutzergruppen**

**Warum ist das wichtig?**  
Nicht alle Nutzer haben die gleichen Anforderungen an die Software. Eine klare Definition der Benutzergruppen hilft dabei, die Benutzeroberfläche, Berechtigungen und Funktionen gezielt auf deren Bedürfnisse abzustimmen.

**Was sollte hier stehen?**

* Welche Benutzergruppen gibt es (z. B. Administratoren, Sachbearbeiter, externe Nutzer)?
* Welche spezifischen Anforderungen hat jede Benutzergruppe? (z. B. eingeschränkte Zugriffsrechte, spezielle Dashboards)
* Wie viele Nutzer pro Gruppe sind voraussichtlich aktiv?

**➡ Ausführlichkeit: Mittel bis hoch.** Eine präzise Definition erleichtert die spätere Implementierung, kann aber kompakt gehalten werden.

## **3. Funktionale Anforderungen**

In diesem Abschnitt werden die konkreten Funktionen definiert, die die Software bieten muss. Die Kernfunktionen und Integrationsanforderungen sind besonders wichtig und sollten detailliert beschrieben werden. Anbieter benötigen diese Angaben, um eine geeignete Lösung vorzuschlagen. Die Anpassbarkeit sollte je nach Unternehmensbedarf präzisiert werden, kann aber bei geringem Anpassungsbedarf kürzer ausfallen.

### **3.1 Kernfunktionen**

**Warum ist das wichtig?**  
Dieser Punkt beschreibt die essenziellen Funktionen, ohne die die Software nicht sinnvoll eingesetzt werden kann. Je klarer die Kernfunktionen definiert sind, desto gezielter kann eine passende Softwarelösung ausgewählt werden.

**Was sollte hier stehen?**

* Welche Hauptaufgaben muss die Software erfüllen? (z. B. Rechnungsstellung, Lagerverwaltung, Berichterstellung)
* Gibt es branchenspezifische Funktionen, die erforderlich sind?
* Müssen bestimmte Prozesse automatisiert oder standardisiert werden?

**➡ Ausführlichkeit: Sehr wichtig.** Eine detaillierte Beschreibung ist notwendig, um sicherzustellen, dass die Software die entsprechenden Prozesse unterstützt.

### **3.2 Anpassbarkeit**

**Warum ist das wichtig?**  
Nicht jede Standardsoftware passt perfekt zu den bestehenden Unternehmensprozessen. Eine flexible Lösung sollte sich an spezifische Anforderungen anpassen lassen, um langfristig effizient zu bleiben.

**Was sollte hier stehen?**

* Muss die Software individuelle Workflows oder Felder etc. unterstützen?
* Sind benutzerdefinierte Einstellungen oder Konfigurationsmöglichkeiten erforderlich?
* Ist eine Erweiterung durch Plugins oder individuelle Entwicklungen notwendig?
* Ist ein Customizing an ein bestimmtes Corporate Design etc. erforderlich?

**➡ Ausführlichkeit: Mittel.** Falls Anpassbarkeit wichtig ist, sollte sie detailliert beschrieben werden. Falls Standardlösungen ausreichen, kann dieser Punkt kürzer gehalten werden.

### **3.3 Integration**

**Warum ist das wichtig?**  
Die Software sollte sich nahtlos in die bestehende IT-Landschaft einfügen. Eine mangelhafte Integration kann zu Dateninkonsistenzen und ineffizienten Arbeitsabläufen führen.

**Was sollte hier stehen?**

* Welche bestehenden Systeme müssen angebunden werden? (z. B. ERP, CRM, Buchhaltung)
* Sind bestimmte Schnittstellen (APIs) oder Datenformate erforderlich?
* Soll die Software mit Cloud-Diensten oder On-Premise-Systemen kommunizieren?

**➡ Ausführlichkeit: Sehr wichtig.** Eine klare Definition der Integrationsanforderungen verhindert spätere technische Herausforderungen und Kostensteigerungen.

## **4. Nicht-funktionale Anforderungen**

Nicht-funktionale Anforderungen definieren, wie die Software funktionieren soll. Sie betreffen Aspekte wie Benutzerfreundlichkeit, Leistung, Skalierbarkeit und Sicherheit. Diese Anforderungen sind entscheidend für die langfristige Nutzbarkeit der Software und sollten sorgfältig durchdacht werden.

Die Performance-, Sicherheits- und Skalierbarkeitsanforderungen sind besonders wichtig und sollten detailliert beschrieben werden, da sie schwer nachträglich optimiert werden können. Die Benutzerfreundlichkeit kann kompakter gehalten werden, sollte aber grundlegende Erwartungen an eine intuitive Bedienung beinhalten. Ein gut ausgearbeiteter Abschnitt sorgt für eine leistungsfähige, sichere und zukunftsfähige Softwarelösung.

### **4.1 Benutzerfreundlichkeit**

**Warum ist das wichtig?**  
Eine intuitive und gut strukturierte Benutzeroberfläche steigert die Akzeptanz der Software und reduziert Schulungsaufwand und Fehlerquellen.

**Was sollte hier stehen?**

* Muss die Software für alle Nutzergruppen einfach bedienbar sein?
* Gibt es bestimmte Anforderungen an die Navigation, das Design oder die Barrierefreiheit?
* Soll die Software eine mobile Nutzung (Smartphone, Tablet) unterstützen?

**➡ Ausführlichkeit: Mittel.** Benutzerfreundlichkeit ist wichtig, aber oft schwer messbar. Kurze, klare Anforderungen genügen, um später gezielt Tests durchzuführen.

### **4.2 Performance**

**Warum ist das wichtig?**  
Die Software muss zuverlässig und schnell arbeiten, auch unter hoher Last. Eine schlechte Performance kann zu Produktivitätsverlusten führen.

**Was sollte hier stehen?**

* Gibt es Anforderungen an die Reaktionszeiten der Software? (z. B. „Ladezeiten unter 2 Sekunden“)
* Muss die Software auch bei hoher Nutzerzahl oder großen Datenmengen performant bleiben?
* Gibt es spezielle Anforderungen an Verfügbarkeit und Ausfallsicherheit?

**➡ Ausführlichkeit: Sehr wichtig.** Performance-Anforderungen sollten klar messbar sein, da spätere Optimierungen oft teuer sind.

### **4.3 Skalierbarkeit**

**Warum ist das wichtig?**  
Die Software sollte mit dem Unternehmen mitwachsen können, um zukünftige Investitionen zu schützen.

**Was sollte hier stehen?**

* Muss die Software problemlos auf mehr Benutzer oder größere Datenmengen erweiterbar sein?
* Ist eine Cloud-Lösung gewünscht, die dynamisch skaliert werden kann?
* Wird eine modulare Erweiterbarkeit der Software benötigt?

**➡ Ausführlichkeit: Mittel bis hoch.** Skalierbarkeit sollte beschrieben werden, vor allem wenn zukünftiges Wachstum absehbar ist.

### **4.4 Sicherheit**

**Warum ist das wichtig?**  
IT-Sicherheit und Datenschutz sind essenziell, um Unternehmens- und Kundendaten zu schützen und gesetzliche Vorgaben einzuhalten.

**Was sollte hier stehen?**

* Welche Verschlüsselungsstandards sind erforderlich?
* Welche Zugriffskontrollen (z. B. Rollen- und Rechtekonzepte, Multi-Faktor-Authentifizierung) sind notwendig?
* Sind gesetzliche Sicherheitsvorgaben (z. B. DSGVO, ISO 27001) einzuhalten?

**➡ Ausführlichkeit: Sehr wichtig.** Sicherheitsanforderungen sollten präzise definiert sein, um spätere Risiken und Compliance-Probleme zu vermeiden.

## **5. Technische Anforderungen**

Technische Anforderungen definieren die Voraussetzungen für den Einsatz der Software in der bestehenden IT-Infrastruktur. Eine gute Abstimmung mit der IT-Abteilung ist essenziell, um sicherzustellen, dass die Software problemlos integriert und betrieben werden kann. Fehlerhafte oder unklare technische Anforderungen können zu hohem Mehraufwand bei der Implementierung führen.

### **5.1 Systemintegration**

**Warum ist das wichtig?**  
Die Software sollte sich nahtlos in die bestehende IT-Landschaft einfügen. Eine unzureichende Integration kann zu ineffizienten Workflows und doppeltem Datenmanagement führen.

**Was sollte hier stehen?**

* Mit welchen bestehenden Systemen (z. B. ERP, CRM, Buchhaltung) muss die Software verbunden werden?
* Welche Schnittstellen (APIs, Webhooks, Datenformate wie XML/JSON) müssen unterstützt werden?
* Soll die Software On-Premises, in der Cloud oder als hybride Lösung betrieben werden?
* Welche Anforderungen an Schnittstellen und Kommunikationsprotokolle (REST, SOAP, GraphQL) bestehen?
* Welche Datenbanken (z. B. MySQL, PostgreSQL, Oracle) sollen unterstützt werden.

**➡ Ausführlichkeit: Sehr wichtig**. Eine präzise Beschreibung der Integrationsanforderungen verhindert spätere technische Probleme und Zusatzkosten.

### **5.2 Datenmigration**

**Warum ist das wichtig?**  
Bestehende Daten müssen oft in das neue System übernommen werden. Eine fehlerhafte oder unvollständige Migration kann zu Datenverlust oder Inkompatibilitäten führen.

**Was sollte hier stehen?**

* Welche Daten (z. B. Kunden-, Produkt-, Transaktionsdaten) müssen übernommen werden?
* In welchem Format liegen die Daten vor?
* Gibt es Anforderungen an die Datenbereinigung oder -transformation?
* Welche Systeme dienen als Datenquelle?

**➡ Ausführlichkeit: Mittel bis hoch.** Die Migration ist oft komplex, daher sollte sie so konkret wie möglich beschrieben werden, um Aufwand und Risiken zu minimieren.

### **5.3 Technologie**

**Warum ist das wichtig?**  
Die Software muss mit der bestehenden Infrastruktur kompatibel sein, um unnötige Anpassungen oder Hardware-Upgrades zu vermeiden.

**Was sollte hier stehen?**

* Welche Betriebssysteme müssen unterstützt werden? (Windows, macOS, Linux, mobile Betriebssysteme)
* Sind bestimmte Hardware-Anforderungen zu berücksichtigen?
* Welche Datenbanken und Entwicklungsumgebungen müssen unterstützt werden?

**➡ Ausführlichkeit: Wichtig, aber knapp.** Eine kurze Liste mit den wesentlichen technischen Anforderungen reicht meist aus.

## **6. Regulatorische und rechtliche Anforderungen**

Regulatorische und rechtliche Anforderungen sind entscheidend, um gesetzliche Vorgaben einzuhalten und rechtliche Risiken zu vermeiden. Besonders in stark regulierten Branchen (z. B. Gesundheitswesen, Finanzen, öffentliche Verwaltung) müssen Unternehmen sicherstellen, dass die Software den geltenden Vorschriften entspricht.Unternehmen sollten sich frühzeitig mit ihrer Rechts- oder Compliance-Abteilung abstimmen, um sicherzustellen, dass alle relevanten Vorgaben berücksichtigt werden.

### **6.1 Compliance**

**Warum ist das wichtig?**  
Die Software muss alle relevanten gesetzlichen und branchenspezifischen Vorschriften erfüllen. Eine fehlende Compliance kann zu Bußgeldern oder rechtlichen Konsequenzen führen.

**Was sollte hier stehen?**

* Welche gesetzlichen Vorgaben gelten für das Unternehmen? (z. B. DSGVO, ISO 27001, HIPAA)
* Gibt es branchenspezifische Vorschriften, die die Software erfüllen muss?
* Welche Nachweise oder Zertifizierungen werden vom Softwareanbieter erwartet?

**➡ Ausführlichkeit: Sehr wichtig.** Die Anforderungen sollten klar definiert sein, um rechtliche Risiken zu vermeiden.

### **6.2 Datenschutz**

**Warum ist das wichtig?**  
Datenschutz ist essenziell, um personenbezogene und unternehmensinterne Daten vor unbefugtem Zugriff und Missbrauch zu schützen. Eine Nichteinhaltung kann hohe Strafen und Imageschäden zur Folge haben.

**Was sollte hier stehen?**

* Welche Datenschutzrichtlinien müssen eingehalten werden? (z. B. DSGVO-konforme Datenspeicherung und -verarbeitung)
* Wie soll mit personenbezogenen Daten umgegangen werden? (Speicherung, Löschung, Zugriffskontrolle)
* Gibt es Anforderungen an Verschlüsselung, Backups oder Anonymisierung?

**➡ Ausführlichkeit: Sehr wichtig**. Datenschutzanforderungen müssen detailliert beschrieben werden, um spätere Risiken zu minimieren.

## **7. Projektmanagement und Implementierung**

Eine strukturierte Planung der Implementierung ist entscheidend für eine erfolgreiche Einführung der Software. Eine unzureichende Vorbereitung kann zu Verzögerungen, höheren Kosten und Akzeptanzproblemen führen. In diesem Abschnitt werden Zeitpläne, Ressourcenbedarf, Schulungskonzepte und Supportstrukturen definiert. Sie sollten gut durchdacht sein, um Verzögerungen und unerwartete Probleme zu vermeiden. Die Schulungskonzepte können flexibler gestaltet werden, sollten aber zumindest grundlegend geplant sein.

### **7.1 Zeitplan**

**Warum ist das wichtig?**  
Ein klar definierter Zeitplan stellt sicher, dass die Einführung der Software strukturiert und effizient abläuft. Ohne eine realistische Planung kann es zu Verzögerungen und unerwarteten Kosten kommen.

**Was sollte hier stehen?**

* Welche Meilensteine gibt es (z. B. Auswahlphase, Testphase, Rollout)?
* Wie lange soll die Implementierung dauern?
* Gibt es Fristen oder externe Abhängigkeiten (z. B. Jahresabschluss, gesetzliche Änderungen)?

**➡ Ausführlichkeit: Sehr wichtig**. Der Zeitplan sollte realistisch und detailliert sein, um Verzögerungen zu vermeiden.

### **7.2 Ressourcen**

**Warum ist das wichtig?**  
Die Einführung der Software benötigt Personal und Budget. Ein klares Bild über die benötigten Ressourcen hilft, Engpässe frühzeitig zu erkennen.

**Was sollte hier stehen?**

* Welche internen und externen Ressourcen werden benötigt? (z. B. IT-Abteilung, externe Berater)
* Welche Rollen und Verantwortlichkeiten gibt es im Projektteam?
* Welches Budget ist für die Implementierung vorgesehen?

**➡ Ausführlichkeit: Mittel bis hoch.** Die Ressourcenplanung sollte gut durchdacht sein, aber nicht übermäßig detailliert.

### **7.3 Schulungen**

**Warum ist das wichtig?**  
Eine gut geschulte Belegschaft erhöht die Akzeptanz der Software und reduziert Fehler. Schulungen sind oft ein entscheidender Erfolgsfaktor bei der Einführung neuer Systeme.

**Was sollte hier stehen?**

* Welche Nutzergruppen müssen geschult werden?
* Welche Schulungsformate sind geplant? (z. B. Präsenzschulungen, E-Learning, Handbücher)
* Wird ein kontinuierliches Schulungskonzept für neue Mitarbeitende benötigt?

**➡ Ausführlichkeit: Wichtig.** Schulungen sollten geplant, aber nicht zu detailliert beschrieben werden – Anpassungen sind oft erforderlich.

### **7.4 Support und Wartung**

**Warum ist das wichtig?**  
Nach der Einführung muss sichergestellt sein, dass Probleme schnell gelöst und Updates regelmäßig durchgeführt werden.

**Was sollte hier stehen?**

* Welche Supportstufen sind erforderlich? (z. B. 1st-Level-, 2nd-Level-Support)
* Welche Reaktionszeiten sind notwendig?
* Sind regelmäßige Wartungsarbeiten oder Updates erforderlich?

**➡ Ausführlichkeit: Sehr wichtig.** Eine klare Definition der Support- und Wartungsanforderungen stellt sicher, dass die Software langfristig reibungslos funktioniert.

## **8. Erfolgskriterien und Messgrößen**

Die Definition von Erfolgskriterien und Messgrößen stellt sicher, dass die Einführung der Software objektiv bewertet werden kann. Durch klare Vorgaben kann später überprüft werden, ob die Software die gewünschten Verbesserungen bringt. Eine klare Definition in diesem Abschnitt erleichtert die spätere Bewertung und sorgt für Transparenz bei allen Beteiligten.

### **8.1 KPIs (Key Performance Indicators)**

**Warum ist das wichtig?**  
KPIs sind messbare Werte, die den Erfolg der Softwareeinführung objektiv bewerten. Ohne KPIs ist es schwierig zu beurteilen, ob die Software die gewünschten Ergebnisse liefert.

**Was sollte hier stehen?**

* Welche Kennzahlen sind relevant? (z. B. Zeitersparnis pro Prozess, Reduktion von Fehlern, Nutzerzufriedenheit)
* Wie werden die KPIs gemessen? (z. B. automatische Berichte, Umfragen, Systemanalysen)
* Welche Zielwerte sollen erreicht werden?

**➡ Ausführlichkeit: Sehr wichtig.** KPIs sollten klar definiert und messbar sein, um eine objektive Erfolgskontrolle zu ermöglichen.

### **8.2 Erfolgskriterien**

**Warum ist das wichtig?**  
Erfolgskriterien legen fest, wann das Softwareprojekt als erfolgreich gilt. Sie sind eine Mischung aus technischen, wirtschaftlichen und organisatorischen Faktoren.

**Was sollte hier stehen?**

* Wann gilt das Projekt als erfolgreich? (z. B. „Mindestens 90 % der Nutzer sind mit der neuen Software zufrieden.“)
* Welche qualitativen Kriterien sind wichtig? (z. B. einfache Bedienung, weniger manuelle Arbeitsschritte)
* Gibt es Akzeptanzkriterien, die von den Endnutzern erfüllt sein müssen?

**➡ Ausführlichkeit: Mittel bis hoch.** Erfolgskriterien sollten gut durchdacht, aber nicht zu detailliert sein, da sie sich im Laufe des Projekts leicht anpassen lassen müssen.

## **9. Anbieteranforderungen**

Dieser Abschnitt beschreibt, welche Anforderungen der Softwareanbieter erfüllen muss. Neben der technischen Leistungsfähigkeit spielen auch Erfahrung, Support-Angebote und vertragliche Regelungen eine Rolle.

### **9.1 Erfahrungen und Referenzen**

**Warum ist das wichtig?**  
Ein erfahrener Anbieter mit relevanten Referenzen kann die Implementierung effizienter gestalten und branchenspezifische Herausforderungen besser bewältigen.

**Was sollte hier stehen?**

* Hat der Anbieter Erfahrung in der eigenen Branche?
* Gibt es Referenzkunden, die ähnliche Anforderungen hatten?
* Werden Best Practices oder Beratung zur Prozessoptimierung angeboten?

**➡ Ausführlichkeit: Mittel.** Eine kurze Liste mit wesentlichen Kriterien reicht aus, da detaillierte Prüfungen später erfolgen können.

### **9.2 Service Level Agreements (SLAs)**

**Warum ist das wichtig?**  
SLAs legen vertraglich fest, welche Leistungen der Anbieter garantieren muss, z. B. Verfügbarkeit, Reaktionszeiten und Supportbedingungen. Ohne klare SLAs kann es später zu Problemen bei Support und Wartung kommen.

**Was sollte hier stehen?**

* Welche Verfügbarkeitszeiten (z. B. 99,9 % Uptime) sind erforderlich?
* Welche Reaktionszeiten gelten für Supportanfragen?
* Welche Konsequenzen gibt es bei Nichteinhaltung der SLAs?

**➡ Ausführlichkeit: Sehr wichtig.** SLAs müssen präzise definiert werden, um eine verlässliche Servicequalität sicherzustellen.

### **9.3 Kostenstruktur**

**Warum ist das wichtig?**  
Die Kostenstruktur muss transparent und nachvollziehbar sein, um spätere Überraschungen zu vermeiden. Neben Lizenzkosten sind auch Implementierung, Wartung und potenzielle Zusatzkosten zu berücksichtigen.

**Was sollte hier stehen?**

* Fragen nach einmaligen oder laufenden Kosten (z. B. Lizenz, Abonnement, nutzungsabhängige Gebühren) und Zusatzkosten (z. B. Schulungen, Support, Erweiterungen)?
* Gibt es eine Kostenübersicht für verschiedene Module oder Nutzeranzahlen?

**➡ Ausführlichkeit: Sehr wichtig**. Eine klare Kostenstruktur hilft bei der Budgetplanung und beim Vergleich verschiedener Anbieter.

## **10. Anforderungen an die Dokumentation**

Eine umfassende Dokumentation ist essenziell für eine effiziente Nutzung und Verwaltung der Software. Sie stellt sicher, dass Nutzer schnell eingearbeitet werden können und IT-Abteilungen alle notwendigen Informationen zur Wartung und Weiterentwicklung haben. Die technische Dokumentation ist besonders wichtig und sollte detailliert beschrieben werden, um spätere IT-Probleme zu vermeiden.

### **10.1 Nutzerhandbücher**

**Warum ist das wichtig?**  
Eine gut strukturierte Benutzer­dokumentation reduziert Schulungsaufwand und Support-Anfragen.

**Was sollte hier stehen?**

* Welche Art von Dokumentation wird benötigt? (z. B. Handbücher, Online-Hilfen, interaktive Tutorials)
* Gibt es Anforderungen an die Sprache oder Verständlichkeit der Anleitungen?
* Soll die Dokumentation regelmäßig aktualisiert werden?

**➡ Ausführlichkeit: Mittel.** Eine allgemeine Anforderung reicht aus, Details können später mit dem Anbieter geklärt werden.

### **10.2 Technische Dokumentation**

**Warum ist das wichtig?**  
Die IT-Abteilung benötigt eine detaillierte technische Dokumentation, um das System zu verwalten, anzupassen und bei Problemen schnell reagieren zu können.

**Was sollte hier stehen?**

* Welche Informationen sind erforderlich? (z. B. API-Dokumentation, Datenbankstruktur, Installationsanleitungen)
* Muss der Quellcode oder eine Entwickler-Dokumentation bereitgestellt werden?
* Gibt es spezifische Anforderungen für Integrationen oder individuelle Anpassungen?

**➡ Ausführlichkeit: Sehr wichtig**. Eine klare Anforderung an die technische Dokumentation vermeidet spätere Probleme bei Wartung und Weiterentwicklung.