www.trafonetz.de

WEITERBILDUNGSMODUL

SOFTWARE-ENTWICKLUNG

Anwendungsnahe Lernbausteine für Zukunftsthemen
in Ihrem Unternehmen

Transformationsnetzwerk
Nordschwarzwald

Ein Projekt der
REGION NORDSCHWARZWALD
Wirtschaftsförderung

Lernpfad: Kernaufgaben und Lernbausteine

SOFTWARE-ENTWICKLUNG

In der digitalen Welt mit schnell wechselnden Marktanforderungen stehen Unternehmen, und hierunter auch die kleinen und mittleren Unternehmen (KMU) in der Region, regelmäßig vor der Herausforderung, Software zu integrieren oder bestehende Software an neue Bedürfnisse anzupassen. Entscheidend dabei ist, die Anforderungen der Mitarbeitenden in den Fachabteilungen zu berücksichtigen und entsprechend zu analysieren. Zudem erfordern technologische Fortschritte eine kontinuierliche Weiterentwicklung und Optimierung der bestehenden Software im Unternehmen.

Perspektiven und Lerninhalte

Das Weiterbildungsmodul befähigt, neue Softwareprojekte im Unternehmen erfolgreich umzusetzen. Die Mitarbeitenden erschließen sich Kompetenzen u. a. aus den Bereichen:

- Anforderungsanalyse für neue Software(-funktionen) im Fachbereich
- Auswahl eines Dienstleisters
- Integration, Überprüfung und Abnahme neuer Software(-funktionen)
- Konzeption von Schulungen für neue Software

Die Mitarbeitenden qualifizieren sich durch die Weiterbildungsbausteine, bestehende Software im Unternehmen anzupassen. Dazu erschließen sie sich Kompetenzen u. a. aus den Bereichen:

- Erfassung der Bedarfe an Anpassungen von Software
- No-Code und Low-Code Programmierung
- Management von Softwarefehlern

Lernformat

Das Lernen findet selbstgesteuert am Arbeitsplatz statt. Die Mitarbeitenden erschließen sich die Kompetenzen schrittweise anhand von Lernaufgaben.

Voraussetzungen und Zielgruppe

Das Weiterbildungsmodul besteht aus verschiedenen Lernbausteinen und ermöglicht dadurch individuelle Lernpfade. Der Awareness-Lernbaustein bietet einen ersten, niederschwelligen Einstieg für alle Mitarbeitenden. Die weiteren Lernbausteine adressieren abhängig vom individuellen Wissensstand im Bereich Softwareentwicklung Anfänger, Fortgeschrittene oder Experten.

Das Weiterbildungsmodul qualifiziert Mitarbeitende in den folgenden Rollen:

N Nutzer: Alle Mitarbeitende, die Software im Unternehmen nutzen.

P Planer: Erfahrene Mitarbeitende und Führungskräfte, welche eine Softwareentwicklung im eigenen Unternehmen planen und steuern.

Übersicht und Struktur

Der Lernpfad gibt einen Überblick über die Kernaufgaben im Bereich Softwareentwicklung für produzierende KMU. Anhand der thematisch zugehörigen Lernbausteine können sie abgleichen, wie das eigene Unternehmen im Bereich Softwareentwicklung aufgestellt ist und mögliche Kompetenzlücken im Unternehmen identifizieren.

Weitere Informationen

Weitere Informationen finden Sie auf unserer Homepage: **trafonetz.de/weiterbildungsmodule**

Gefördert durch:

Bundesministerium für Wirtschaft und Energie

aufgrund eines Beschlusses des Deutschen Bundestages

Stuttgart, den 30.05.2025

Fraunhofer IAO

1.1 Grundlagen der Softwarebedienung

Niveaustufe 1 2 3

Rolle	**N Nutzer**
Lernziel	Die Lernenden haben ein vertieftes Verständnis für die Bedienung der Software, die sie regelmäßig nutzen.
Lernschritte / Vorgehensweise / Inhalte	**Verschaffen Sie sich einen Überblick über die Softwareanwendungen, mit denen Sie regelmäßig arbeiten.** Erstellen Sie eine Übersicht, für welche Arbeitsaufgaben Sie welche Software benutzen, z. B. für die Überwachung von Produktionsanlagen, die Bestellung von Werkzeug oder die Dokumentation von Produktionszahlen. **Beobachten Sie Ihre Arbeitsschritte mit den Softwareanwendungen in Ihrem Arbeitsbereich.** Beispielsweise das Empfangen von Arbeitsaufträgen, die Eingabe von Daten oder das Überwachen von Produktionsparametern. Beobachten Sie dabei, welche Arbeitsschritte Ihnen schnell von der Hand gehen und bei welchen Arbeitsschritten, Sie sich besonders konzentrieren müssen. Stellen Sie fest, ob es Stolpersteine bei der Softwarebedienung gibt, und dokumentieren Sie diese. **Erstellen Sie für sich eine Übersicht über die Funktionen der Software, die für Sie relevant sind und wozu Sie Fragen haben.** Identifizieren Sie mögliche Wissenslücken oder Verständnisprobleme bei der Bedienung der verschiedenen Softwareanwendungen in Ihrem Arbeitsbereich. Notieren Sie sich Funktionen, für die Sie noch eine Anleitung benötigen. **Informieren Sie sich über die Bedienung der Software, um ggf. Fragen zu klären.** Überprüfen Sie, ob es Anleitungen oder Handbücher für die Softwareanwendungen in Ihrem Arbeitsbereich gibt und ob Sie damit selbstständig mögliche Wissenslücken schließen können. Recherchieren Sie weitere Infomaterialien wie Video-Tutorials oder FAQs, um mögliche Verständnisprobleme zu lösen. **Besprechen Sie mit Ihren Kollegen die Bedienung der Software in Ihrem Arbeitsbereich.** Fragen Sie Ihre Kollegen, welche Erfahrungen sie mit den Softwareanwendungen im Arbeitsbereich haben. Reflektieren sie gemeinsam, welche Arbeitsschritte ihnen leichtfallen und für welche sie (zu Beginn) eine Anleitung oder Einweisung benötigt haben. Fragen Sie Ihre Kollegen bei Bedarf nach Unterstützung, um mögliche Verständnisprobleme bei der Softwarebedienung zu beseitigen.
Lernmedien	• Anleitungen und Handbücher zur Bedienung der Softwareanwendungen im Arbeitsbereich • Video-Tutorials und FAQs für Standardsoftware
Empfohlene Vorkenntnisse	**Awareness Softwareentwicklung**
Dauer	**4 Stunden in 2 Wochen**

1.2 Eigenen Bedarf an Anpassungen von Software erkennen

Niveaustufe 1 2 3

Rolle	**N Nutzer**
Lernziel	Die Lernenden können für ihnen bekannte Software einen Bedarf an Anpassungen erkennen und benennen.
Lernschritte / Vorgehensweise / Inhalte	**Dokumentieren Sie wiederkehrende technische Probleme der Softwareanwendungen in Ihrem Arbeitsbereich.** Beispielsweise Fehlermeldungen, Softwareabstürze oder Störungen. Dokumentieren Sie die technischen Probleme möglichst genau (z. B. mit Fotos von der Fehlermeldung). Beschreiben Sie, wie groß die Auswirkungen auf die Produktion sind und bewerten Sie, wie dringend Sie eine Lösung für das wiederkehrende technische Problem benötigen. Falls in Ihrem Unternehmen vorhanden, nutzen Sie Vorlagen für die Dokumentation. **Überprüfen Sie, ob Ihre Arbeitsabläufe von den Softwareanwendungen in Ihrem Arbeitsbereich optimal unterstützt werden.** Achten Sie beispielsweise auf die Reaktionszeit einer Software, z. B. beim Starten eines Programms oder bei der Datenspeicherung. Überprüfen Sie u. a., ob die Software Ihnen ausreichend Feedback gibt, z. B. eine visuelle oder akustische Rückmeldung, die Ihre Eingabe oder eine erfolgreiche Aktion bestätigt. **Entwickeln Sie Vorschläge zur Anpassung von Software, um die Bedienung zu erleichtern.** Notieren Sie sich, welche Änderungen Ihre Arbeit erleichtern würde, z. B. eine Funktion zur Markierung, ob ein Auftrag bereits in Bearbeitung ist. Beschreiben Sie die Vorschläge zur Anpassung möglichst konkret. **Erstellen Sie eine Liste mit Ihren Anpassungsbedarfen.** Beschreiben Sie jeweils das Problem (z. B. Programm hängt sich immer wieder auf) und das Ziel der Anpassung (z. B. Programm muss nicht mehrfach neu gestartet werden). Nennen Sie weitere Informationen, um das Problem zu konkretisieren (z. B. bei welchen Funktionen hängt sich das Programm auf). Erklären Sie den Nutzen der Anpassung. **Besprechen Sie die Anpassungsbedarfe mit Ihrem Vorgesetzten und übergeben Sie ihm Ihre Dokumentation.** Besprechen Sie mit Ihrem Vorgesetzten wiederkehrende technische Probleme sowie Ihre Herausforderungen bei der Softwarebedienung. Übergeben Sie ihm die Liste mit Ihren Anpassungsbedarfen.
Lernmedien	• Newsletter über Software-Updates • Handreichungen zur bestehenden Software im Unternehmen • Webinare zu Benutzerfreundlichkeit, Infomaterial zu neuer Technologie, z. B. Künstliche Intelligenz in der Produktion • Anleitung zum Umgang mit Fehlermeldungen
Empfohlene Vorkenntnisse	**Awareness Softwareentwicklung, 1.1 Grundlagen der Softwarebedienung**
Dauer	**12 Stunden in 2 Monaten**

1.3 Methoden zur Erhebung von Nutzerbedarfen für Software

Niveaustufe 1 2 3

Rolle	**Planer**
Lernziel	Die Lernenden kennen verschiedene Methoden zur Erhebung von Nutzerbedarfen und wissen, worauf Sie bei der Planung einer Erhebung achten müssen.
Lernschritte / Vorgehensweise / Inhalte	**Erarbeiten Sie sich ein grundlegendes Verständnis, welche Anforderungen Nutzende an Software haben können.** Recherchieren Sie dazu, was Nutzerbedarfe sind, und informieren Sie sich, welche unterschiedlichen Arten von Anforderungen Nutzende haben können (z. B. technische Probleme beheben, übersichtliche Benutzeroberfläche, Parameterüberwachung automatisieren). **Informieren Sie sich, welche unterschiedlichen Daten Sie zur Beantwortung einer Fragestellung erheben können.** Z. B. könnte es Sie interessieren, wie fehleranfällig die Bedienung einer Software ist. Dafür könnten Sie entweder die Anzahl der Fehler (quantitative Daten) erheben oder sie erheben für eine tiefere Einsicht qualitativ, wie und bei welchen Arbeitsschritten die Fehler passieren. **Recherchieren Sie Methoden zur Erhebung von Nutzerbedarfen.** Informieren Sie sich über verschiedene Methoden und zu welcher Situation und Fragestellung sie passen. Dabei werden Sie u. a. folgende Methoden kennenlernen: • **Befragungen,** um quantitative Daten zu sammeln (z. B. Umfragen zu Softwarefunktionen). • **Interviews,** um qualitative Einblicke zu erhalten (z. B. persönliche Gespräche mit Nutzenden). • **Beobachtungen,** um Arbeitsprozesse zu verstehen (z. B. Beobachtung von Mitarbeitenden bei der Softwarebedienung). • **Usability-Tests,** um Feedback von den Nutzenden zu erhalten (z. B. Testlauf einer neuen Funktion mit Feedback-Runde). • **Feedback-Formulare,** um im laufenden Betrieb Feedback von den Nutzenden zu sammeln (z. B. Eingabemaske für Feedback nach Nutzung einer Funktion). **Recherchieren Sie Achtungspunkte und Good-Practice-Beispiele.** Informieren Sie sich, worauf Sie bei der Planung einer Erhebung von Nutzerbedarfen achten müssen (z. B. Datenschutz bei Befragungen, Objektivität bei Interviews und Beobachtungen, realistische Nutzungsszenarien bei Usability-Tests). Recherchieren Sie, wie andere Nutzerbedarfe zur Verbesserung von Softwareanwendungen erhoben haben. **Erstellen Sie eine Übersicht, welche Methoden zur Erhebung der Nutzerbedarfe in Ihrem Unternehmen passend wären.** Notieren Sie sich dazu aktuelle Fragestellungen zu bestehender Software im Unternehmen. Wählen Sie passende Methoden zur Beantwortung der einzelnen Fragestellungen aus. Ergänzen Sie die Vor- und Nachteile der einzelnen Methoden. **Informieren Sie sich, was Sie bei den verschiedenen Methoden zur Erhebung von Nutzerbedarfen in Ihrem Unternehmen beachten müssen.** Beispielsweise könnte es erforderlich sein, dass die Zustimmung des Betriebsrates für eine Befragung oder Beobachtung der Mitarbeitenden erforderlich ist.
Lernmedien	• Infomaterial und Sachbücher über Nutzerbedarfe und Anforderungen an Software • Ratgeber und Good-Practice-Beispiele zur Durchführung einer Erhebung von Nutzerbedarfen • Unternehmensinterne Richtlinien und Vorgaben
Empfohlene Vorkenntnisse	**Awareness Softwareentwicklung**
Dauer	**8 Stunden in 2 Wochen**

4.1 Zuständigkeiten für Support

Niveaustufe 1 2 3

Rolle	**N Nutzer**
Lernziel	Die Lernenden kennen für die Software in ihrem Arbeitsbereich ihre Ansprechpersonen für Support im Fall von Software-Ausfällen, Fehlermeldungen, oder Fragen zur Softwarebedienung.
Lernschritte / Vorgehensweise / Inhalte	**Überprüfen Sie Ihr Wissen zu den aktuellen unternehmensinternen Supportstrukturen.** Reflektieren Sie, wie Sie bislang Unterstützung im Fall von Fehlermeldungen oder Bedarf an Support in Anspruch genommen haben. Überlegen Sie, was gut und was nicht so gut funktioniert hat. Vergegenwärtigen Sie, welche Kommunikationswege es zwischen Support und anderen Unternehmensbereich es gibt, ob es beispielsweise eine Anleitung zur Weiterleitung von Fehlermeldungen gibt oder ein Ticket-System mit vorgegebenen Formularen. **Erkundigen Sie sich nach Dokumenten, in denen Ansprechpartner für den Support von Software in Ihrem Bereich gelistet sind.** Erkundigen Sie sich nach vorhandenen Kontaktverzeichnissen oder anderen Informationsquellen wie z. B. Adressetiketten an den Maschinen oder Angaben in Handbüchern in räumlicher Nähe der Maschinen. **Prüfen Sie, ob die Kontaktangaben in den Verzeichnissen oder Handbüchern aktuell sind.** Fragen Sie ggf. bei Ihren Vorgesetzten oder Ihre Kollegen nach, ob Informationen ggf. überprüft, ergänzt oder aktualisiert werden sollten. Soweit nicht vorhanden – erstellen Sie ein Verzeichnis mit internen und externen Kontaktpersonen, die für den IT-Support in Ihren Arbeitsbereich relevant sind. Unterscheiden Sie interne Ansprechpartner im Unternehmen und Ansprechpartner bei externen Dienstleistern. Vereinbaren Sie mit Kollegen wie Sie eine laufende Aktualisierung der Kontaktlisten gewährleisten können. **Denken Sie an Fehlermeldungen bei der Nutzung von Software, mit denen Sie in den letzten drei Monaten konfrontiert waren.** Denken Sie über Ihre eignen Erfahrungen zu folgenden Aussagen nach: • Mir war immer sofort klar, an wen ich mich zur Behebung des Fehlers wenden kann. • Es war für mich ersichtlich, ob es sich um eine kritischen oder um einen leichten Schweregrad des Fehlers gehandelt hat, bzw. welche Konsequenzen dieser Fehler zur Folge hat. • Ich weiß Bescheid, wie ich die Fehlermeldung beschreiben sollte und auf welchem Weg und wohin ich diese zur Bearbeitung einer Behebung schicken sollte. **Diskutieren Sie Ihre Erfahrungen und offene Fragen mit Ihren Kollegen und Vorgesetzten.**
Lernmedien	• Interne Dokumente zur Regelung des Supports • Verzeichnisse mit Kontaktdaten von internem Support und externen Support-Dienstleistern
Empfohlene Vorkenntnisse	**Awareness Softwareentwicklung, 1.1 Grundlagen der Softwarebedienung, 1.2 Eigenen Bedarf an Anpassungen von Software erkennen**
Dauer	**4 Stunden in 2 Wochen**

1.4 Anforderungen für die Weiterentwicklung von Software formulieren

Niveaustufe 1 2 3

Rolle	**N Nutzer**
Lernziel	Die Lernenden können Anforderungen für die Weiterentwicklung bestehender Software im Unternehmen präzise formulieren.
Lernschritte / Vorgehensweise / Inhalte	**Recherchieren Sie, was die Beschreibung einer Anforderung an Software beinhalten sollte.** Dabei werden Sie erfahren, dass Sie u. a. das Problem, das Ziel und den Nutzen beschreiben sollten. Informieren Sie sich, ob es Infomaterial oder Formatvorlagen in Ihrem Unternehmen zur Beschreibung von Anforderungen an Software gibt. Recherchieren Sie, worauf Sie beim Formulieren einer Anforderung achten sollten, und suchen Sie nach Good-Practice-Beispielen. **Informieren Sie sich, ob es einen Bedarf an neuen Funktionen der bestehenden Software in Ihrem Arbeitsbereich gibt.** Überprüfen Sie oder fragen Sie nach, ob es ggf. Aufträge in Ihrem Arbeitsbereich gibt, die aktuell nicht bearbeitet werden können, weil die bestehende Software dafür die Anforderungen nicht erfüllt. Recherchieren Sie, ob es ggf. eine neue Softwareversion gibt und welche neuen Funktionen diese enthält. **Erarbeiten Sie eigene Vorschläge zur Weiterentwicklung bestehender Softwareanwendungen entsprechend Ihrer Bedarfe.** Notieren Sie sich, ob es Funktionen gibt, die Sie bei der von Ihnen genutzten Software vermissen und die Ihre Arbeit erleichtern würden, z. B. eine Archivierungs- oder Löschfunktion für alte Arbeitsaufträge. **Erstellen Sie eine Liste, welche Arbeitsschritte ggf. durch neue Softwarefunktionen automatisiert werden könnten.** Identifizieren Sie dazu Arbeitsschritte, die fehleranfällig sind (z. B. Berechnungen, manuelle Dateneingaben oder -übertragungen). Ein Vorschlag für eine neue Softwarefunktion könnte beispielsweise die Überwachung eines Produktionsparameters sein, d. h. die Software gibt automatisiert eine Warnmeldung, sollte der Parameter einen festgelegten Wert erreichen. **Definieren Sie die Ziele der gewünschten Weiterentwicklungen und beschreiben Sie die neuen Funktionen möglichst konkret.** Beschreiben Sie klar und eindeutig, welches Problem eine Weiterentwicklung lösen soll oder welches Ziel damit erreicht werden soll, z. B. „Ein Produktionsparameter wird aktuell oft übersehen und soll deshalb größer dargestellt werden“ oder „Die Dateneingabe soll ohne Fehlermeldung funktionieren“. Konzentrieren Sie sich bei der Beschreibung darauf, was entwickelt werden soll. Die Frage, wie die neue Funktion technisch umgesetzt wird, müssen Sie nicht beantworten. **Beschreiben Sie die aktuelle Situation in Bezug auf die Software und begründen Sie Ihre Anforderungen für die Weiterentwicklung.** Stellen Sie die aktuelle Situation dar (z. B. „Wenn ich diese Dateneingabe mache, kommt jene Fehlermeldung.“, „Ich kann den neuen Arbeitsauftrag nicht in die Maschine eingeben, weil mir ein zusätzliches Eingabefenster fehlt.“). Ergänzen Sie ggf. Ihre Beschreibung mit Bildern oder Screenshots zur Veranschaulichung der Situation. Begründen Sie Ihre Anforderung in dem Sie die Auswirkungen des Problems nennen oder die Folgen, sollte die Weiterentwicklung einer Software nicht erfolgen (z. B. „Der Produktionsparameter wird oft übersehen, wodurch es zu Fehlproduktionen kommt.“). **Erstellen Sie eine Liste mit Ihren Anforderungen.** Nennen Sie für jede Anforderung Ihr zuvor definiertes Ziel. Ergänzen Sie die bereits erstellte Beschreibung der aktuellen Situation sowie Ihre Begründung. Beschreiben Sie für jede Anforderung Ihre gewünschten neuen Funktionen der Software. Nennen Sie alle technischen Details, die Ihnen bekannt sind und die für die Weiterentwicklung der Software relevant sein könnten (z. B. Betriebssystem, Softwareversion). Fügen Sie ggf. Beispiele hinzu, die Ihre Anforderungen verdeutlichen. Bewerten Sie die Dringlichkeit und Wichtigkeit Ihrer Anforderungen. **Übergeben Sie Ihrem Vorgesetzten Ihre Liste und besprechen Sie die Anforderungen mit ihm.** Beantworten Sie ggf. offene Fragen und besprechen Sie die Dringlichkeit und Wichtigkeit Ihrer Anforderungen.
Lernmedien	• Infomaterial und Formatvorlagen zur Beschreibung von Anforderungen an Software • Webinare zur Anforderungsanalyse • Leitfäden und Good-Practice-Beispiele zur Formulierung von Anforderungen
Empfohlene Vorkenntnisse	**Awareness Softwareentwicklung, 1.1 Grundlagen der Softwarebedienung, 1.2 Eigenen Bedarf an Anpassungen von Software erkennen**
Dauer	**6 Stunden in 3 Wochen**

1.5 Anforderungsanalyse für die Weiterentwicklung bestehender Software

Niveaustufe 1 2 3

Rolle	**Planer**
Lernziel	Die Lernenden können eine Anforderungsanalyse für die Weiterentwicklung bestehender Software im eigenen Unternehmen planen.
Lernschritte / Vorgehensweise / Inhalte	**Recherchieren Sie den Unterschied zwischen funktionalen und nicht-funktionalen Anforderungen an Software.** Dabei werden Sie u. a. erfahren, dass funktionale Anforderungen beschreiben, welche Funktionen eine Software haben sollen, z. B. eine Funktion zur Anmeldung eines Nutzenden oder zur Speicherung von Daten. Nicht-funktionale Anforderungen beschreiben hingegen, wie eine Software seine Funktionen ausführen soll, z. B. die Reaktionszeit bei der Anmeldung oder die Sicherheit von gespeicherten Daten. **Informieren Sie sich, wie Anforderungen an Software formuliert sein sollten.** Dabei werden Sie erfahren, dass u. a. das Problem, das Ziel und der Nutzen beschrieben sein sollte. Informieren Sie sich, ob es Infomaterial oder Formatvorlagen in Ihrem Unternehmen gibt zur Beschreibung von Anforderungen an Software. Recherchieren Sie, worauf bei der Formulierung einer Anforderung geachtet werden sollte, und suchen Sie nach Good-Practice-Beispielen. **Planen Sie die Erhebung von Nutzerbedarfen in Ihrem Unternehmen.** Wählen Sie eine geeignete Methode zur Erhebung der Nutzerbedarfe aus (z. B. Interviews, Beobachtungen, Usability-Tests). Planen Sie eine passende Zeit und Umgebung für die Erhebung sowie die Kommunikation mit den Mitarbeitenden. (Hinweis: Sollte es einen Betriebsrat in Ihrem Unternehmen geben, müssen Sie ggf. die Erhebung mit ihm abstimmen.) **Tragen Sie die Ergebnisse der Erhebung zusammen und kategorisieren Sie die Nutzerbedarfe.** Kategorisieren Sie, welche der Nutzerbedarfe funktionale und welche nicht-funktionale Anforderungen sind. Lassen Sie von einem Fachkollegen (aus der eigene IT-Abteilung oder von einem externen Dienstleister) grob einschätzen, wie aufwändig die Umsetzung der Nutzerbedarfe ungefähr wäre und welche Kosten damit verbunden wären. **Lassen Sie von (internen oder externen) Experten überprüfen, ob es zusätzlich zu den Nutzerbedarfen weitere Anforderungen an die bestehende Software in Ihrem Unternehmen gibt.** Beispielsweise Anforderungen an die Performance, die Zuverlässigkeit oder die Sicherheit. Bei der Überprüfung sollten auch unternehmensinterne Richtlinien sowie gesetzliche Vorgaben (z. B. zur Verarbeitung personenbezogener Daten) beachtet werden. **Erstellen Sie ein Dokument mit den Anforderungen für die Weiterentwicklung bestehender Software, die erfasst wurden.** Jede Anforderung sollte dabei ein definiertes Ziel der Anpassung, eine Beschreibung der aktuellen Situation, eine Begründung, eine Beschreibung der Anpassungen oder neuen Funktionen, ggf. technischen Details und eine Bewertung der Dringlichkeit und Wichtigkeit enthalten. **Bereiten Sie eine Entscheidung darüber vor, welche Anforderungen umgesetzt werden sollen.** Erstellen Sie dafür eine übersichtliche Zusammenfassung der dringlichsten und wichtigsten Anforderungen an die bestehende Software im Unternehmen. Lassen Sie von den Entscheidungsträgern abwägen, ob die Beschaffung einer neuen Software ggf. kosteneffizienter wäre. **Erarbeiten Sie eine Übersicht mit allen Anforderungen, die umgesetzt werden sollen.** Dokumentieren Sie die Entscheidung zur Weiterentwicklung der bestehenden Software.
Lernmedien	• Infomaterial über funktionale und nicht-funktionale Anforderungen an Software • Ratgeber zur Formulierung von Anforderungen, Formatvorlagen für Anforderungsbeschreibungen • Good-Practice-Beispiele für Anforderungsanalysen • Richtlinien im Unternehmen für Erhebungen, z. B. Mitarbeiterbefragungen • Gesetzliche Vorgaben für Software, z. B. Datenschutz, Informationssicherheit • Empfehlungen von Experten, z. B. Cybersecurity, Performance • Webinare zur Priorisierung von Anforderungen • Leitfäden zur Erstellung eines Lastenheftes für eine Softwareentwicklung
Empfohlene Vorkenntnisse	**Awareness Softwareentwicklung, 1.3 Methoden zur Erhebung von Nutzerbedarfen für Software**
Dauer	**15 Stunden in 3 Monaten**

2.1 Projektierung einer Softwareentwicklung für das eigene Unternehmen

Niveaustufe 1 2 3

Rolle	**Planer**
Lernziel	Die Lernenden können die Umsetzung einer Softwareentwicklung für das eigene Unternehmen planen und projektieren.
Lernschritte / Vorgehensweise / Inhalte	**Erarbeiten Sie sich grundlegendes Wissen über Projektmanagementmethoden, die Softwareentwickler/Dienstleister anwenden.** Informieren Sie sich über verschiedene Methoden des Projektmanagements (z. B. klassisch oder agil) und für welche Arten von Projekten diese passend sind. Informieren Sie sich, welche Auswirkungen die Managementmethode des Softwareentwicklers/Dienstleisters auf die Zusammenarbeit hat (z. B. Form und Häufigkeit von Besprechungen). **Informieren Sie sich über den Bedarf an Softwareentwicklungen in Ihrem Unternehmen.** Bringen Sie in Erfahrung, welche Softwareentwicklung umgesetzt werden soll. Informieren Sie sich über die Ergebnisse der Anforderungsanalyse für die Software. Sollten diese nicht vorliegen, planen Sie eine Anforderungsanalyse. **Definieren Sie das Projektziel der Softwareentwicklung, für die ein Bedarf in Ihrem Unternehmen besteht.** Halten Sie in ein bis zwei Sätzen fest, was mit der Softwareentwicklung erreicht werden soll. Als Grundlage können Sie die bereits definierten Ziele der Anforderungen für die Weiter- oder Neuentwicklung der Software verwenden. Überprüfen Sie mittels der SMART-Methode, ob das Projektziel spezifisch, messbar, akzeptiert, realistisch und terminiert ist. **Sollten Sie einen Dienstleister beauftragen, erstellen Sie einen Budgetplan für die Softwareentwicklung.** Informieren Sie sich, ob es in Ihrem Unternehmen Vorlagen für Budgetpläne gibt. Lassen Sie von einer Fachperson einschätzen, wie viele Mittel für die geplante Softwareentwicklung benötigt werden. Fordern Sie dabei die Aufzählung einzelner Kostenpunkte zur Berechnung der Gesamtkosten ein. Recherchieren Sie Referenzen und prüfen Sie die Gesamtkosten auf Plausibilität. Informieren Sie sich, wie viele Mittel in Ihrem Unternehmen für das Projekt zur Verfügung stehen. Stellen Sie die verfügbaren Mittel und die geplanten Kosten gegenüber. **Planen Sie die personalen Ressourcen in Ihrem Unternehmen für die Umsetzung der Softwareentwicklung.** Informieren Sie sich über die Nutzergruppen der geplanten Softwareentwicklung und welche Mitarbeitenden in Ihrem Unternehmen zu diesen gehören. Erarbeiten Sie eine Entscheidung, ob alle Nutzenden oder nur Vertreter von Nutzergruppen bei der Umsetzung der Softwareentwicklung einbezogen werden. Stellen Sie fest, welche Personen in Ihrem Unternehmen darüber hinaus einbezogen werden müssen, z. B. Entscheidungsträger. Lassen Sie bei einer internen Softwareentwicklung die benötigten personalen Ressourcen von einer Fachperson in Ihrem Unternehmen einschätzen. **Planen Sie die Zusammensetzung eines Projektteams in Ihrem Unternehmen mit allen relevanten Stakeholdern und planen Sie die regelmäßige Kommunikation.** Relevante Stakeholder sind beispielsweise Entscheidungsträger (z. B. Geschäftsführung, Abteilungsleitungen), Nutzende, bzw. Vertretungen der Nutzergruppen sowie Beratende (z. B. Datenschutzbeauftragte) in Ihrem Unternehmen.
Lernmedien	• Sachbücher und Webinare zu Projektmanagement und Management-Methoden • Anleitungen und Online-Tutorials für die SMART-Methode • Vorlagen und Praxisbeispiele für einen Budgetplan
Empfohlene Vorkenntnisse	**Awareness Softwareentwicklung, 1.5 Anforderungsanalyse für die Weiterentwicklung bestehender Software, 1.6 Anforderungsmanagement zur Planung neuer Software für das eigene Unternehmen**
Dauer	**20 Stunden in 2 Wochen (variiert in Abhängigkeit von der Projektgröße)**

2.2 Auswahl einer bestehenden Software auf dem Markt oder eines Dienstleisters für die Entwicklung neuer Software

Niveaustufe 1 2 3

Rolle	**Planer**
Lernziel	Die Lernenden können eine Vorgehensweise zur Bewertung von Angeboten für eine Software(-entwicklung) und zur Auswahl eines Dienstleisters erstellen.
Lernschritte / Vorgehensweise / Inhalte	**Überprüfen Sie, welche Voraussetzungen Ihr Unternehmen für eine Software oder einen Dienstleister hat.** Informieren Sie sich beispielsweise, welche Rechte Sie an der Software benötigen. **Erstellen Sie eine Leistungsbeschreibung für eine Softwareanwendung.** Informieren Sie sich über die Inhalte einer Leistungsbeschreibung und recherchieren Sie Good-Practice-Beispiele. Tragen Sie dafür alle Anforderungen an die Software und ggf. an den Dienstleister zusammen. **Erarbeiten Sie Kriterien zur Bewertung und Auswahl von Software, bzw. Dienstleister.** Recherchieren Sie, worin sich Angebote für Softwareentwicklungen unterscheiden können. Dabei werden Sie erfahren, dass ein Kriterium die rechtlichen Rahmenbedingungen sein können (z. B. Rechte an der Software). **Planen Sie eine Marktanalyse.** Lassen Sie von einer Fachperson (aus der eigene IT-Abteilung oder von einem externen Dienstleister) überprüfen oder überprüfen Sie selbst, ob es bestehende Softwarelösungen auf dem Markt gibt, die die für Ihr Unternehmen ermittelnden Anforderungen an eine neue Software erfüllen. **Erstellen Sie eine Übersicht über passende Softwarelösungen bzw. Dienstleister.** Nennen Sie, welche Anforderungen von der jeweiligen Softwarelösung bzw. dem Dienstleister erfüllt werden sowie mögliche Vor- und Nachteile. Stellen Sie Gemeinsamkeiten und Unterschiede gegenüber. **Holen Sie für die Umsetzung einer Softwareentwicklung in Ihrem Unternehmen Angebote von Dienstleistern ein, die passende Lösungen anbieten.** **Überprüfen Sie die Angebote hinsichtlich Ihrer zuvor festgelegten Kriterien.** Überprüfen Sie beispielsweise die Integrationsfähigkeit der angebotenen Software in die im Unternehmen bestehenden Systeme. **Vergleichen Sie die Dienstleister und ihre Angebote beispielsweise hinsichtlich:** • **der Gesamtkosten, bzw. das Preis-Leistungs-Verhältnis:** z. B. Entwicklungskosten, Lizenzgebühren oder Wartungskosten) • **der Flexibilität und Anpassungsfähigkeit:** z. B. die Möglichkeit während der Projektlaufzeit Anforderungen anzupassen oder zu ergänzen • **der Sicherheitsaspekte:** z. B. Unternehmensgröße des Dienstleisters • **der Unternehmenskultur des Dienstleisters:** z. B. die Passung zur eigenen Unternehmenskultur • **der Testmöglichkeiten:** z. B. Demos oder Testversionen der Software für Sie als Kunden • **der angebotenen Unterstützung:** z. B. die Verfügbarkeit von Support und Schulungsangeboten durch den Anbieter nach der Implementierung • **der Wartungsbedingungen:** z. B. Regelungen nach der Implementierung **Recherchieren Sie nach Erfahrungsberichten vorheriger Auftraggeber sowie Referenzen, möglichst von Kunden aus Ihrer Branche.** Informieren Sie sich dazu beispielsweise auf der Website des Dienstleisters. Achten Sie bei den Berichten, u. a. auf die Einschätzung der Zuverlässigkeit oder der Erreichbarkeit des Dienstleisters.
Lernmedien	• Unternehmensinterne Richtlinien für Software, z. B. IT-Sicherheit, Barrierefreiheit. • Infomaterial und Good-Practice-Beispiele für Leistungsbeschreibungen für Software. • Ratgeber zur Auswahl von Software. • Webinare und E-Learnings für Marktanalysen. • Internetauftritte von möglichen Dienstleistern. • Referenzen von anderen Unternehmen.
Empfohlene Vorkenntnisse	**Awareness Softwareentwicklung, 1.3 Methoden zur Erhebung von Nutzerbedarfen für Software, 1.5 Anforderungsanalyse für die Weiterentwicklung bestehender Software, 1.6 Anforderungsmanagement zur Planung neuer Software für das eigene Unternehmen, 2.1 Projektierung einer Softwareentwicklung für das eigene Unternehmen**
Dauer	**8 Stunden in 2 Wochen**

3.1 Anpassung oder Erstellung von Software mit No-Code Programmierung

Niveaustufe 1 2 3

Rolle	**Nutzer**
Lernziel	Die Lernenden können bestehende Software im eigenen Arbeitsbereich ohne Programmierkenntnisse anpassen oder kleine Software-Applikationen neu erstellen.
Lernschritte / Vorgehensweise / Inhalte	**Erarbeiten Sie sich ein Verständnis, was No-Code Programmierung bedeutet und welche Vorteile und Nachteile bzw. Grenzen diese hat.** No-Code Programmierung wird als eine Entwicklungsmethode bezeichnet, bei der ohne die Verwendung von Code und ohne technische Programmierkenntnisse Software sowohl neu erstellt werden kann (z.B. Entwicklung einer mobilen App zur Schichtplanung und Ressourcenverwaltung) als auch bestehende Software angepasst werden kann (z.B. Erweiterung des ERP um ein Modul zur Überwachung der Produktionsqualität und Erstellung von benutzerdefinierten Berichten). • **Vorteile von No-Code Programmierung** sind u.a. eine schnelle Entwicklung oder Anpassung von Software ohne Programmierkenntnisse mit unternehmensinternen Ressourcen. Benutzer können schnell auf sich ändernde Anforderungen reagieren und selbst Anpassungen vornehmen. • **Nachteile bzw. Grenzen von No-Code Programmierung** sind u.a. begrenzte Anpassungsoptionen im Vergleich zu maßgeschneiderter Software, Abhängigkeit von Funktionen und Updates von No-Code Plattformanbietern, Schwierigkeiten bei der Skalierung und Performance, Sicherheitsrisiken aufgrund mangelnder Kontrolle über die zugrunde liegende Infrastruktur. **Machen Sie sich den Unterschied bewusst zwischen der Anpassung von im Unternehmen vorhandener Software und der Erstellung neuer Software unter Einsatz von No-Code Programmierung.** Überlegen Sie, welche Softwareanwendungen in Ihrem Bereich relevant sind und ggf. angepasst oder ergänzt werden sollten: • **Anpassung von im Unternehmen vorhandener Software:** ERP-Systeme, CRM-Systeme, Projektmanagement-Tools, Datenanalyse-Tools, Workflow-Automatisierung u.a. • **Erstellung von neuer Software:** Webseiten, Mobile Apps, Datenbanken, Umfragen und Formulare, Interaktive Prototypen. **Tauschen Sie sich mit Kollegen zu Möglichkeiten und Grenzen des Einsatzes von No-Code Programmierung aus.** Eruieren Sie mögliche Einsatzbereich in Ihrem Arbeitsumfeld. Beispiel zu einer typischen Situation in produzierenden KMU: Verwendung mehrerer Excel-Tabellen zur Verwaltung der Produktionsaufträge, was zu ineffizienten Prozessen und Schwierigkeiten bei der Nachverfolgung von Aufträgen führt. Es besteht der Bedarf, einen zentralen Ort zu schaffen, an dem alle Produktionsinformationen verwaltet werden können. Die Erstellung eines dafür geeigneten Produktions-Workflow-Management-Tools ist über No-Code Programmierung machbar. Überlegen Sie mögliche Einsatzbereiche zur Erstellung oder Anpassung von Software mittels No-Code Programmierung in Ihrem Arbeitsumfeld. **Erkunden Sie, welche gängigen No-Code Plattformen es gibt und worin sich diese unterscheiden.** Listen Sie fünf der gängigen Plattformen auf und notieren Sie, für welchen Zweck diese jeweils geeignet sind. Beispielsweise gibt es Plattformen, die eher auf Web-Entwicklung (z. B. Webflow) oder andere Plattformen, die eher auf App-Entwicklung (z. B. Adalo) oder Automatisierung (z. B. Zapier) spezialisiert sind. **Erkunden Sie, welche gängigen Methoden der No-Code Programmierung es gibt und für welchen Anwendungsbereich sich diese eignen.** Beispielsweise „Drag-and-Drop Builder" für Webseitenerstellung, Landing Pages oder Mobile Apps. „Visuelle Programmierung" für die Verarbeitung und Analyse von Daten, „Formular- und Umfrage-Tools" für die Erstellung interaktiver Formulare und Umfragen zur Datensammlung. Finden Sie heraus, welche weiteren Methoden und ihre Anwendungsbereiche für No-Code Programmierung nutzbar sind. **Schauen Sie sich genau an, in welchen Schritten eine Software mit Einsatz von No-Code Programmierung angepasst wird.** In der Regel gibt es zu allen Methoden der No-Code Programmierung ausführliche Schritt-für-Schritt Anleitungen. Diese finden sich u. a. auf den Plattformen der Tools, auf YouTube, in Blog-Posts oder in Community-Foren oder auf KI-Plattformen. Ein Beispiel zur Methode „Visuelle Programmierung": • **Benutzeroberfläche:** Zugriff auf eine grafische Benutzeroberfläche (Graphic User Interface), die alle verfügbaren Funktionen und Elemente zur No-Code Programmierung abbildet. • **Elemente auswählen:** Auswahl der Elemente (z. B. Buttons, Formulare, Datenbanken) aus einer Bibliothek oder Palette. • **Drag-and-Drop:** Platzieren der Elemente per Drag-and-Drop an die gewünschte Stelle auf der Arbeitsfläche, um das Layout der Software zu gestalten. • **Einstellungen anpassen:** Konfiguration aller Elemente, indem Einstellungen und Eigenschaften über Menüs oder Formulare angepasst werden (z. B. Textänderungen, Farben, Aktionen). • **Logik definieren:** Verbinden von Elementen, um Interaktionen zu bestimmen (z. B. wenn-dann Bedingungen), oft durch einfaches Klicken/Ziehen.

3.1 Anpassung oder Erstellung von Software mit No-Code Programmierung

	• **Vorschau und Test:** Anzeigen der Vorschau und Durchführung von Tests, um sicherzustellen, dass alles wie gewünscht funktioniert. • **Veröffentlichung:** Nach der Fertigstellung kann die Anwendung veröffentlicht und zugänglich gemacht werden. **Listen Sie Arbeitsschritte in ähnlicher Weise, für andere No-Code Tools Ihrer Wahl auf.** Suchen Sie eine Software in Ihrem Arbeitsbereich aus, die Sie mit No-Code Programmierung anpassen möchten. Beispiele für mögliche Anpassungen eines Wartungsmanagement-Systems: • **Dokumentation:** Speichern von Wartungsprotokollen und Anleitungen für Maschinen direkt im System. • **Benachrichtigungen:** Automatisierte Erinnerungen für bevorstehende Wartungen oder Überprüfungen. • **Analysen:** Erstellung von Berichten zur Analyse von Wartungshäufigkeit und -kosten. **Stimmen Sie sich vor einer Umsetzung mit Ihrem Vorgesetzten oder Ihren Kollegen hinsichtlich Auswahl und Umsetzung der Softwareanpassung ab.** Stellen Sie sicher, dass eine gemeinsame Priorisierung und Auswahl von umzusetzenden No-Code-Anpassungen erfolgt. **Wählen Sie ein passendes No-Code Tool aus und setzen Sie die Anpassung oder Weiterentwicklung damit um.** **Dokumentieren Sie alle Schritte Ihres Vorgehens.** Stellen Sie sicher, dass Ihre Anpassungen für Dritte über eine Dokumentation ersichtlich und verständlich sind. **Lassen Sie die Software von Ihren Kollegen testen.** Stellen Sie sicher, dass die Änderungen technisch funktionieren und benutzerfreundlich, bzw. selbsterklärend gestaltet sind. **Erläutern Sie Ihren Kollegen die von Ihnen vorgenommene Software-Anpassung, deren Vorteile sowie ggf. die veränderte Softwarebedienung.** Stellen Sie zudem dar, wie eine veränderte oder neue Softwarelösung in das System und die Arbeitsabläufe integriert wurde. Erstellen von Infomaterial zu neuen Möglichkeiten und veränderten Softwarebedienung.
Lernmedien	• Plattformen für No-Code Tools • Blog-Posts zu Erfahrungen mit No-Code Programmierung • Branchespezifische Community-Foren oder KI-Plattformen zu Erfahrungen, Voraussetzungen und zum Einsatz der No-Code Programmierung • Video-Tutorials zu No-Code Tools
Empfohlene Vorkenntnisse	**Awareness Softwareentwicklung, 1.1 Grundlagen der Softwarebedienung, 1.2 Eigenen Bedarf an Anpassungen von Software erkennen, 1.4 Anforderungen für die Weiterentwicklung von Software formulieren**
Dauer	**16 Stunden in 4 Wochen**

4.2 Verhalten bei Fehlermeldungen im Arbeitsprozess

Niveaustufe 1 2 3

Rolle	**N Nutzer**
Lernziel	Die Lernenden können mit Software-Fehlermeldungen verantwortungsvoll umgehen.
Lernschritte / Vorgehensweise / Inhalte	**Machen Sie sich bewusst, welche Schweregrade für Fehlermeldungen es gibt.** Ein gängiges System zur Kategorisierung ist beispielsweise das nachfolgende: • **Kritisch:** Fehler, der das System oder die Anwendung vollständig zum Absturz bringt und den Betrieb stark beeinträchtigt. Höchste Priorität für sofortige Maßnahmen. Beispiel: „SYSTEMFEHLER: Produktionsdatenbank nicht erreichbar! Bitte überprüfen Sie die Netzwerkverbindung und die Datenbankservereinstellungen. Kontaktieren Sie den IT-Support." • **Hoch:** Fehler, der wesentliche Funktionen beeinträchtigt, jedoch einen Workaround (vorübergehende improvisierte Lösung) zulässt. Hohe Priorität für die Behebung. • Beispiel: „WARNUNG: Materialbestand kritisch niedrig…Bitte überprüfen Sie die Bestellanforderungen und leiten Sie die Bestellung umgehend in die Wege." • **Mittel:** Fehler, der Einschränkungen verursacht, aber nicht kritisch ist. Behebung ist wichtig, kann aber warten. Beispiel: „HINWEIS: Drucker nicht verfügbar! Bitte überprüfen Sie die Druckereinstellungen oder wählen Sie einen anderen Drucker aus, um fortzufahren." • **Niedrig:** Fehler, der geringe Auswirkungen hat oder kosmetischer Natur ist. Kann in zukünftigen Releases behoben werden.Beispiel: „INFO: Neue Softwareversion verfügbar!... Bitte beachten Sie, dass die Installation optional ist und derzeit keine Auswirkungen auf den laufenden Betrieb hat." • **Informational:** Hinweise oder Verbesserungsvorschläge, die keine Fehler darstellen, aber zur Optimierung beitragen können. Beispiel: „INFO: Neuer Produktionsauftrag erstellt…. Weitere Informationen finden Sie in der Auftragsübersicht. Es sind keine weiteren Maßnahmen erforderlich." **Überlegen Sie sich, welchen Schweregrad vergangene Fehlermeldungen in Ihrem Arbeitsbereich hatten.** Ergänzen Sie ggf. welche der Fehlermeldungen Sie in der Vergangenheit falsch und welche Sie hinsichtlich Schweregrad richtig eingeschätzt hatten. Tauschen Sie sich mit Erfahrungen von Kollegen zu ihren Erfahrungen mit Fehlermeldungen aus. **Machen Sie sich vertraut mit der effektiven Dokumentation von Software-Fehlern.** Vergegenwärtigen Sie sich die nachfolgende Schritt-für-Schritt Anleitung. Soweit nicht bereits vorhanden -regen Sie Ihren Vorgesetzten oder einen IT-Kollegen dazu an, entlang dieser Schritte eine Vorlage (Template) zu erstellen, die für Ihren Bereich zur Erfassung von Fehlern passt. • **Fehlerbeschreibung erfassen:** Beschreiben Sie den Fehler klar und präzise. Notieren Sie die Schritte, die den Fehler ausgelöst haben. Geben Sie an, was erwartet wurde und was tatsächlich passiert ist. Fügen Sie Screenshots oder Videos hinzu, wenn möglich. • **Umgebungsbedingungen dokumentieren:** Geben Sie die Softwareversion, das Betriebssystem und andere relevante Umgebungsdetails an (z.B. Hardware-Spezifikation, Benutzereinstellungen, Datum und Uhrzeit, wann de Fehler aufgetreten ist). • **Priorität und Schweregrad festlegen:** Bewerten Sie die Dringlichkeit und den Einfluss des Fehlers auf den Arbeitsbereich und auf das Unternehmen. • **Betroffene Komponenten auflisten:** Geben Sie an, welche Teile der Software oder Systeme betroffen sind. • **Status verfolgen:** Halten Sie den Fortschritt der Fehlerbehebung und -lösung fest (z.B. offen, in Bearbeitung, gelöst). • **Dokumentation abschließen:** Aktualisieren Sie die Dokumentation, wenn der Fehler behoben ist, und fügen Sie die Informationen zur Art der Lösungsfindung hinzu. **Erarbeiten Sie Vorschläge zur Verbesserung der Kommunikation von Fehlermeldungen.** Vergleichen Sie die Kommunikation von Fehlermeldungen im Unternehmen mit Best Practice Beispielen. Überprüfen Sie, ob Kommunikationstools, z. B. Ticket-Systeme eingesetzt werden, ob es Vorlagen zur Dokumentation gibt und wie diese ggf. verbessert werden könnten.
Lernmedien	• Interne Dokumente zur Regelung des Supports im Fall von Softwarefehlern • Unternehmensinterne Richtlinien zum Umgang mit Softwarestörungen, ggf. Code of Conduct zur Lern- u. Fehlerkultur
Empfohlene Vorkenntnisse	**Awareness Softwareentwicklung, 1.1 Grundlagen der Softwarebedienung, 1.2 Eigenen Bedarf an Anpassungen von Software erkennen, 1.4 Anforderungen für die Weiterentwicklung von Software formulieren, 4.1 Zuständigkeiten für Support**
8auer	**8 Stunden in 2 Wochen**

1.6 Anforderungsmanagement zur Planung neuer Software für das eigene Unternehmen

Niveaustufe 1 2 3

Rolle	**Planer**
Lernziel	Die Lernenden können für ihr Unternehmen eine Anforderungsanalyse für neue Software planen unter Berücksichtigung unterschiedlicher Nutzergruppen.
Lernschritte / Vorgehensweise / Inhalte	**Informieren Sie sich, was Nutzergruppen von Softwareanwendungen sind.** Dabei werden Sie erfahren, dass die Nutzenden mit denselben Rollen und Aufgaben (z. B. Produktion, Qualitätsmanagement, IT-Support) einer Nutzergruppe zugeordnet werden, denn in der Regel haben sie dieselben Anforderungen an eine Software. Unterschiedliche Nutzergruppen können hingegen verschiedene funktionale Anforderungen an eine Software haben. Beispielsweise benötigen die Mitarbeitenden aus der Produktion die Softwarefunktion zur Bedienung einer Maschine, wohingegen die Mitarbeitenden des Qualitätsmanagements mit derselben Software die Qualität der Produkte messen und überwachen möchten. Bei einer Anforderungsanalyse ist es daher entscheidend alle Nutzergruppen einzubeziehen. **Erstellen Sie für eine neue Software, für die Sie eine Anforderungsanalyse planen, eine Liste aller relevanter Nutzergruppen.** Identifizieren Sie die Nutzergruppen einer neuen Software in Ihrem Unternehmen, die unterschiedliche funktionale Anforderungen an dieselbe Software haben. Ein ERP-System könnte beispielsweise folgende Nutzergruppen haben: Management, Einkauf, Produktion und IT-Support. **Planen Sie die Erhebung der Anforderungen von verschiedenen Nutzergruppen.** Achten Sie darauf, dass alle relevanten Nutzergruppen bei der Anforderungsanalyse berücksichtigt werden. Überprüfen Sie, ob bei der Erhebung die unterschiedlichen Rollen und Aufgaben der Nutzergruppen berücksichtigt werden. Beispielsweise sollten die Nutzenden nur zu den Funktionen befragt werden, die sie benötigen. **Tragen Sie die Anforderungen aller Nutzergruppen zusammen.** Achten Sie darauf, dass die Anforderungen präzise und vollständig formuliert sind. Überprüfen Sie, welche Schnittmengen es zwischen den Anforderungen gibt. Achten Sie auch auf potenzielle Konflikte, z. B. wenn eine Anforderung eine andere beschneidet. Eine Anforderung könnte beispielsweise sein, dass die Anmeldung möglichst einfach ist und eine Anforderung, dass die Software vor Angriffen geschützt wird. **Besprechen Sie die zusammengetragenen Anforderungen mit allen Nutzergruppen.** Stellen Sie Überschneidungen vor und leiten Sie bei gegensätzlichen Anforderungen eine gemeinsame Lösungsfindung an. **Priorisieren Sie die Anforderungen in Abstimmung mit den Nutzergruppen.** Indem Sie überprüfen, welche Anforderungen unerlässlich sind, um die Software nutzen zu können und welche Anforderungen wünschenswert sind. **Planen Sie die Analyse nicht-funktionaler Anforderungen an neue Software.** Dafür können Sie beispielsweise Richtlinien im Unternehmen, gesetzliche Vorgaben oder Empfehlungen von Experten nutzen. Nicht-funktionale Anforderungen sind beispielsweise Kompatibilität (z. B. Schnittstellen zu anderen Systemen), Sicherheit (z. B. Datenverschlüsselung) oder Skalierbarkeit (z. B. Software kann beim Wachstum des Unternehmens mitwachsen). **Erstellen Sie ein Dokument mit allen Anforderungen an eine neue Software.** Dazu gehört die Beschreibung der funktionalen Anforderungen, z. B. die Beschreibung der Aufgaben, die die Software erfüllen muss, aber auch die Dokumentation der nicht-funktionalen Anforderungen, z. B. wie gut eine Datenbank vor potenzielle Hackerangriffe geschützt sein muss. Achten Sie bei der Formulierung der Anforderungen darauf, zu betonen, wie wichtig die Anforderungen sind, z. B. durch Formulierungen wie „Die Software muss…“, Die Software soll…“ oder „Die Software kann…“.
Lernmedien	• Infomaterial zu Nutzergruppen • Tools für Anforderungsanalysen • Leitfäden zur Formulierung von Anforderungen an Software • Webinar über nicht-funktionale Anforderungen an Software • Unternehmensinterne Richtlinien, z. B. Mitarbeiterbefragungen • Gesetzliche Vorgaben für Software, z. B. Datenschutz, Informationssicherheit • Vorlagen zur Dokumentation von Anforderungen
Empfohlene Vorkenntnisse	**Awareness Softwareentwicklung, 1.3 Methoden zur Erhebung von Nutzerbedarfen für Software, 1.5 Anforderungsanalyse für die Weiterentwicklung bestehender Software**
Dauer	**30 Stunden in 3 Monaten**

2.3 Projektkoordination einer Softwareentwicklung

Niveaustufe 1 2 3

Rolle	**Planer**
Lernziel	Die Lernenden kennen Vorgehensweisen und können diese anwenden, um die Umsetzung einer Softwareentwicklung zu begleiten.
Lernschritte / Vorgehensweise / Inhalte	**Organisieren Sie die regelmäßige Kommunikation mit dem Softwareentwickler/Dienstleister.** Halten Sie wichtige Entscheidungen in Protokollen fest. Planen Sie die Kommunikation zwischen dem Softwareentwickler/Dienstleister mit dem (internen) Projektteam. **Beziehen Sie die Nutzenden der Software aktiv in den Entwicklungsprozess ein.** Organisieren Sie beispielsweise Feedbackrunden mit den Nutzenden. **Überprüfen Sie, ob die Software personenbezogene Daten verarbeitet.** Informieren Sie sich, was personenbezogene Daten sind. Sollte die Software personenbezogene Daten verarbeiten, wenden Sie sich an eine Fachperson (z. B. einen Datenschutzbeauftragten). Planen Sie ggf. eine Datenschutzfolgeabschätzung. **Planen Sie Zwischenziele/Meilensteine zur Überwachung des Projektfortschritts.** Überprüfen Sie, ob für das Projekt bereits Zwischenziele definiert wurden (z. B. im Angebot des Dienstleisters). Ansonsten planen Sie in Abstimmung mit dem Softwareentwickler die Zwischenziele für das Projekt. Überprüfen Sie mittels der SMART-Methode, ob die Zwischenziele spezifisch, messbar, akzeptiert, realistisch und terminiert sind. **Erstellen Sie einen Zeitplan zur Umsetzung der Softwareentwicklung.** Informieren Sie sich, ab wann die Software den Mitarbeitenden in Ihrem Unternehmen zur Verfügung stehen soll. Planen Sie davon ausgehend, bis wann der Softwareentwickler/Dienstleister die definierten Zwischenziele/Meilensteine erreichen sollte. Nutzen Sie als Grundlage die Ressourcenplanung. Berücksichtigen Sie bei der Erstellung des Zeitplans, dass es ggf. Vorlagen Ihres Unternehmens dafür gibt, und nutzen Sie diese. Lassen Sie von einer Fachperson den Zeitplan hinsichtlich seiner Realisierbarkeit überprüfen. **Besprechen Sie den Zeitplan für das Projekt mit dem Softwareentwickler/Dienstleister und dem Projektteam.** Informieren Sie alle Beteiligten am Projekt über die geplanten Zwischenziele bzw. Meilensteine. Stellen Sie den Zeitplan dem Softwareentwickler/Dienstleister und dem Projektteam vor und besprechen Sie den geplanten Prozess. **Dokumentieren Sie den Projektfortschritt.** Aktualisieren Sie regelmäßig den Zeitplan und halten Sie fest, welche Zwischenziele/Meilensteine bereits erreicht wurden. Geben Sie für die Zwischenziele, die noch nicht erreicht wurden an, wie nah man dem Ziel ist, z. B. in Prozentangaben. **Überprüfen Sie bei der Zusammenarbeit mit einem Dienstleister regelmäßig, ob der Budgetplan eingehalten wird.** Behalten Sie dafür die Ausgabe an den Dienstleister im Auge. Überprüfen Sie, ob die Leistungen, die der Dienstleister abrechnen möchte, erbracht wurden. **Setzen Sie sich mit Risiken auseinander, die Ihr Projekt gefährden könnten.** Identifizieren Sie mögliche Risiken (z. B. Ausfall von Mitarbeitenden im Projekt), die zu Verzögerungen im Projekt führen können und die Produktionsstabilität gefährden. Setzen Sie sich damit auseinander, wie Sie damit umgehen, wenn Vereinbarungen nicht eingehalten werden und beispielsweise Leistung nicht den Erwartungen entsprechen. Entwickeln Sie Strategien zur Risikominderung (z. B. Pufferzeiten einplanen, Testung der Software).
Lernmedien	• Webinare oder Infomaterial zur SMART-Methode • Vorlagen für Zeit- bzw. Meilensteinpläne • Unternehmensinterne Richtlinien • Infomaterial und Anleitungen zur Erstellung von Zeitplänen • Good-Practice-Beispiele zur Überwachung eines Projektes • Internetauftritte von möglichen Dienstleistern
Empfohlene Vorkenntnisse	**Awareness Softwareentwicklung, 1.3 Methoden zur Erhebung von Nutzerbedarfen für Software, 1.5 Anforderungsanalyse für die Weiterentwicklung bestehender Software, 1.6 Anforderungsmanagement zur Planung neuer Software für das eigene Unternehmen, 2.1 Projektierung einer Softwareentwicklung für das eigene Unternehmen, 2.2 Auswahl einer bestehenden Software auf dem Markt oder eines Dienstleisters für die Entwicklung neuer Software**
Dauer	**8 Stunden in 2 Wochen**

2.4 Organisation der Integration, Überprüfung und Abnahme neuer Software(-funktionen)

Niveaustufe 1 2 3

Rolle	**Planer**
Lernziel	Die Lernenden können die Integration, Überprüfung und Abnahme neuer Software(-funktion) planen und organisieren.
Lernschritte / Vorgehensweise / Inhalte	**Planen Sie die Integration neuer Software(-funktionen).** Besprechen Sie dafür alle relevanten Schritte mit dem Softwareentwickler/Dienstleister. Stellen Sie sicher, dass die verschiedenen zusammengeführten Softwarekomponenten und -module zusammenarbeiten. **Erarbeiten Sie sich ein Grundverständnis, was bei einer Datenmigration beachtet werden muss.** Dabei werden Sie beispielsweise erfahren, dass eine Bestandsaufnahme der vorhandenen Daten notwendig ist sowie eine Aufbereitung der Daten vor der Migration. Des Weiteren werden Sie erfahren, dass die Daten nach der Migration überprüft werden sollten. **Planen Sie bei Bedarf die Migration bestehender Daten in eine neue Software.** Überprüfen Sie die bestehenden Daten und sichern Sie diese. Organisieren Sie die Bereinigung und Vereinheitlichung der Daten für die Migration. Legen Sie fest, welche Daten in das neue System übertragen werden. **Sollte eine Datenmigration stattgefunden haben, dokumentieren Sie den Migrationsprozess und überprüfen Sie die migrierten Daten.** Halten Sie die getätigten Schritte schriftlich fest. Organisieren Sie die Validierung der Daten im neuen System auf Vollständigkeit und Richtigkeit. **Erstellen Sie eine Checkliste mit den Kriterien Ihres Unternehmens für die Abnahme einer Softwareentwicklung.** Informieren Sie sich, worauf bei der Abnahme einer Software geachtet werden sollte, und recherchieren Sie nach Kriterien sowie vorhandenen Checklisten. Berücksichtigen Sie bei der Checkliste die Anforderungen der verschiedenen Nutzergruppen. Weitere Kriterien neben der Funktionalität und Benutzerfreundlichkeit sind beispielsweise die Leistung, Sicherheit oder Kompatibilität. **Planen Sie Tests zur Sicherstellung der Funktionalität und Zuverlässigkeit der Softwareanwendung.** Organisieren Sie dafür Testaccounts für die Nutzenden der Software in Ihrem Unternehmen. Konzipieren Sie eine Rahmenstruktur für das Feedback der Nutzenden. Planen Sie Last- und Penetrationstest in Abstimmung mit dem Softwareentwickler (Dienstleister) zur Sicherstellung der Funktionalität und Zuverlässigkeit der Software. **Veranlassen Sie, dass der (internen) Projektgruppe die Ergebnisse dieser Tests und das Feedback der Nutzenden präsentiert werden.** Lassen Sie sich die Ergebnisse der Nutzertests und ggf. eines Last- und Penetrationstests zeigen. Organisieren Sie eine Entscheidung der Projektgruppe, welche Fehler und Probleme, die beim Testen festgestellt wurden, für eine Abnahme noch behoben werden müssen. **Überprüfen Sie die Behebung von Fehlern und Problemen, die beim Testen der neuen Software(-funktionen) festgestellt wurden.** Planen Sie die Kommunikation mit dem Softwareentwickler/Dienstleister. Überprüfen Sie, ob erneute Tests erforderlich sind. **Planen Sie die Kommunikation mit den Nutzenden.** Informieren Sie die Nutzenden im Unternehmen darüber, wann die neue Software zur Verfügung steht. Organisieren Sie einen Support, falls Nutzende Fragen zur Softwarebedienung haben.
Lernmedien	• Leitfäden für Datenmigration • Checklisten und Vorlagen im Unternehmen oder von Branchenverbänden zur Abnahme von Software • Infomaterial zur Testung von Software, z. B. Usability-Tests, Last- und Penetrationstests • Feedbackbögen für Nutzende einer Software
Empfohlene Vorkenntnisse	**Awareness Softwareentwicklung, 1.3 Methoden zur Erhebung von Nutzerbedarfen für Software, 1.5 Anforderungsanalyse für die Weiterentwicklung bestehender Software, 1.6 Anforderungsmanagement zur Planung neuer Software für das eigene Unternehmen, 2.1 Projektierung einer Softwareentwicklung für das eigene Unternehmen, 2.2 Auswahl einer bestehenden Software auf dem Markt oder eines Dienstleisters für die Entwicklung neuer Software, 2.3 Projektkoordination einer Softwareentwicklung**
Dauer	**4 Stunden in einer Wochen**

2.5 Konzeption von Schulungen für neue Software(-funktionen)

Niveaustufe 1 2 3

Rolle	**Planer**
Lernziel	Die Lernenden können für unterschiedliche Zielgruppen Schulungen zur Bedienung von Software im Unternehmen konzipieren.
Lernschritte / Vorgehensweise / Inhalte	**Informieren Sie sich über verschiedene Lernformate und ihre Integrationsfähigkeit in den Arbeitsalltag.** Erarbeiten Sie sich einen Überblick über Formate in Präsenz (z. B. Seminare, Workshops), digitale Formate (z. B. Webinare, Lernplattformen) und Kombinationen davon (z. B. Blended Learning). Informieren Sie sich, worin sich Lernformate noch unterscheiden, z. B., ob Lernen selbstgesteuert stattfindet (z. B. E-Learnings), Lernen durch KI geleitet wird oder ein Trainer oder Coach das Lernen anleitet und unterstützt. Bringen Sie in Erfahrung, welche Lernformate sich bestmöglich in den Arbeitsalltag der Mitarbeitenden in Ihrem Unternehmen integrieren lassen. Informieren Sie sich über die Vor- und Nachteile der verschiedenen Lernformate. Lesen Sie Ratgeber zur Konzeption informativer und motivierender Schulungen für Mitarbeitende. **Stellen Sie fest, welche Weiterbildungsstrategie Sie im Unternehmen verfolgen.** Informieren Sie sich über die kurz- und langfristigen Weiterbildungsziele im Unternehmen. Ermitteln Sie, welche Möglichkeiten zur Bedarfsanalyse Sie haben und welche Infrastruktur für Lernen im Unternehmen zur Verfügung steht (z. B. Lernplattformen). Stellen Sie fest, wie Sie im Unternehmen die Wirksamkeit von Weiterbildungen überprüfen. **Ermitteln Sie den Schulungsbedarf für eine Softwareanwendung in Ihrem Unternehmen.** Recherchieren Sie, welche Möglichkeiten Sie zur Identifikation von Qualifikationslücken und Anforderungen Sie im Unternehmen haben. Planen Sie eine Bedarfsanalyse, um festzustellen, welche Schulungsmaßnahmen für welche Nutzer(-gruppen) der Software notwendig sind. **Informieren Sie sich, welches Budget für die Weiterbildung im Unternehmen zu Verfügung steht.** Stellen Sie fest, wie viele Ressourcen Ihr Unternehmen für Weiterbildung bereitstellt und welcher Anteil davon für Schulungen zur Softwarebedienung eingeplant ist. **Informieren Sie sich, welche Schulungen für die neue Software(-funktionen) bereits auf dem Markt angeboten werden.** Z. B. könnte der Dienstleister Schulungen anbieten, der die Software(-funktionen) entwickelt hat. **Überprüfen Sie ggf., welche Möglichkeiten Sie im Unternehmen haben, günstig einfache Schulungsmaterialien zu erstellen.** Informieren Sie sich, ob es beispielsweise Vorlagen für Seminarunterlagen, Programme zur Aufzeichnung von Bildschirmvideos, Tools zu Erstellung von kleinen E-Learnings oder Lernplattformen zur Durchführung von Webinaren im Unternehmen gibt. **Konzipieren Sie bei Bedarf Schulungen für die Nutzenden zur Softwarebedienung.** Berücksichtigen Sie dabei alle Nutzer(-gruppen) und planen Sie ggf. unterschiedliche Schulungsangebote mit verschiedenen Inhalten und angepassten Anforderungen. Konzipieren Sie Schulungen, die sich in den Arbeitsalltag integrieren lassen. **Planen Sie eine Evaluation der Schulungen im Unternehmen.** Überprüfen Sie die Wirksamkeit der Lernformate und passen Sie die Schulungen ggf. an.
Lernmedien	• Infomaterial über Lernformate • Ratgeber zur Konzeption von Schulungen • Lernplattformen, z. B. E-Learnings, Webinare • Unternehmensinterne Dokumente zum Thema Weiterbildung • Ratgeber und Video-Tutorials zur Konzeption von Schulungen
Empfohlene Vorkenntnisse	**Awareness Softwareentwicklung, 1.3 Methoden zur Erhebung von Nutzerbedarfen für Software, 1.5 Anforderungsanalyse für die Weiterentwicklung bestehender Software, 1.6 Anforderungsmanagement zur Planung neuer Software für das eigene Unternehmen, 2.1 Projektierung einer Softwareentwicklung für das eigene Unternehmen, 2.2 Auswahl einer bestehenden Software auf dem Markt oder eines Dienstleisters für die Entwicklung neuer Software, 2.3 Projektkoordination einer Softwareentwicklung, 2.4 Organisation der Integration, Überprüfung und Abnahme neuer Software(-funktionen)**
Dauer	**12 Stunden in 3 Wochen**

3.2 Anpassung oder Erstellung von Software mit Low-Code Programmierung

Niveaustufe 1 2 3

Rolle	**N Nutzer**
Lernziel	Die Lernenden kennen die Grundlagen von Low-Code Programmierung und können damit bestehende Software anpassen sowie neue kleinere Software-Applikationen entwickeln.
Lernschritte / Vorgehensweise / Inhalte	**Erarbeiten Sie sich ein Verständnis dazu, was Low-Code Programmierung bedeutet.** Low-Code Programmierung ist eine Entwicklungsmethode, die es Nutzern ermöglicht, Anwendungen mit minimalem Programmieraufwand zu erstellen, indem sie visuelle Entwicklungswerkzeuge und vorgefertigte Komponenten nutzen. Diese Methode richtet sich sowohl an technisch versierte Entwickler als auch an Benutzer mit geringer Programmiererfahrung. **Machen Sie sich den Unterschied bewusst zwischen der Anpassung von im Unternehmen vorhandener Software und der Erstellung neuer Software unter Einsatz von Low-Code Programmierung.** Überlegen Sie, welche Systeme/Software in Ihrem Bereich relevant ist und ggf. angepasst oder ergänzt werden sollte: • **Anpassung von im Unternehmen vorhandener Software:** ERP-Systeme, CRM-Systeme, Manufacturing Execution Systems (MES), Supply Chain Management (SCM-Systeme), Product Lifecycle. • **Erstellung von neuer Software:** Produktionsplanungstools, Lagerverwaltungssysteme, Qualitätsmanagement-Systeme, Wartungsmanagementsysteme, Schulungs- und E-Learning Tools. **Erkunden Sie die Unterschiede zwischen No-Code und Low-Code Programmierung.** No-Code Programmierung richtet sich an Anwender ohne technische Kenntnisse, während Low-Code einige Programmierkenntnisse erfordert und durch eine benutzerdefinierte Logik mehr Flexibilität bietet. Benötigt werden Grundkenntnisse in Programmierung (z.B. Logik, Scripting), Grundverständnis von Datenbanken und API-Schnittstellen und die Fähigkeit, mit visuellem Design- und Programmierkomponenten zu arbeiten. Entscheidungskriterien zwischen No-Code und Low-Code Programmierung sind u.a. • **Komplexität der Anwendung:** Low-Code Programmierung bei komplexeren Anwendungen, die mehr Anpassungen benötigen. No-Code Programmierung bei einfachen Anwendungen oder spezifischen Aufgaben. • **Technische Fähigkeiten der Nutzer:** Low-Code Programmierung, wenn die Nutzer grundlegende Programmierkenntnisse haben oder Entwickler verfügbar sind. No-Code Programmierung, wenn die Nutzer keine Programmierkenntnisse haben. • **Anpassungsbedarf:** Low-Code Programmierung, wenn individuelle Anpassungen oder Integrationen erforderlich sind. No-Code Programmierung, wenn Standardfunktionen ausreichen. • **Entwicklungszeit:** Low-Code Programmierung, wenn die Flexibilität und Anpassung wichtiger sind als die sofortige Bereitstellung. No-Code-Programmierung, wenn eine schnelle Lösung erforderlich ist. Initiieren Sie einen Austausch, wer in Ihrem Team welche der beiden Methoden ggf. von welchen Kollegen erlernt bzw. eingesetzt werden kann. **Machen Sie Sich bewusst, welche Herausforderungen bei Low-Code Programmierungen zu beachten sind.** Auch wenn für Low-Code Programmierung nur wenig Programmierkenntnisse erforderlich sind, ist ein technisches Grundwissen zu Programmierung und Datenbanken hilfreich. Es sollte die richtige Low-Code-Plattform mit entsprechenden Anforderungen und Funktionen ausgewählt werden. Herausforderungen sind zudem die Gewährleistung von Sicherheit (z. B. über Richtlinien und Standards, Schulungen, Zugriffssteuerung) und eine sichere Integration in bestehende Systeme (z. B. über API-Schnittstellen, Middleware-Lösungen, Testumgebungen, Dokumentation). Überprüfen Sie, ob und wie diese Herausforderungen im eigenen Arbeitsbereich gelöst werden können. **Wählen Sie eine Software-Anwendung in Ihrem Bereich, die Sie über Low-Code Programmierung anpassen möchten.** Mögliche Beispiele sind: • **Produktionsplanungstool:** Eine Anwendung, die es ermöglicht, Produktionsaufträge zu verwalten, Ressourcen zuzuweisen und Engpässe zu identifizieren. Mitarbeitende können Anpassungen vornehmen und Berichte erstellen. • **Qualitätsmanagement-System:** Eine Software zur Erfassung und Nachverfolgung von Qualitätsprüfungen, Abweichungen und Rückmeldungen. Mitarbeitende können Formulare erstellen, um Prüfberichte zu erfassen und automatisierte Benachrichtigungen bei Problemen zu erhalten. • **Mitarbeiter-Feedback-App:** Eine Plattform, über die Mitarbeitende Feedback zu Prozessen und Produkten geben können. Die App kann Umfragen und Bewertungsformulare enthalten, um die Zufriedenheit und Verbesserungsvorschläge der Mitarbeitenden zu sammeln. • **Schichtplanungsanwendung**: Eine Anwendung zur Planung und Verwaltung von Schichten für Mitarbeitende, die die Verfügbarkeit, Urlaubsanträge und Schichtwechsel berücksichtigt. **Besprechen Sie Ihre Ideen mit Ihrem Vorgesetzten und Kollegen.** Finden Sie gemeinsam eine Priorisierung, welche Software-Anwendungen Sie über Low-Code Programmierung verbessert/erweitert werden soll.

3.2 Anpassung oder Erstellung von Software mit Low-Code Programmierung

	Setzen Sie die Low-Code Programmierung mit Hilfe eines geeigneten Low-Code Tools um. Auflisten der Schritte zur Umsetzung einer Veränderung. Am Beispiel Schichtplanerstellung: z. B. Erstellung einer Datenquelle, Erstellung einer neuen App und Anpassung des Layouts, Einrichtung von Funktionen wie Filter, Sortierung oder Benachrichtigungen. **Lassen Sie die Software über Kollegen testen.** Stellen Sie sicher, dass alle Funktionen wie gewünscht arbeiten und die Anwendung selbsterklärend und nutzerfreundlich gestaltet ist. **Erstellen Sie eine Dokumentation für Ihre Low-Code Programmierung.** Prüfen sie entlang der Punkte nachfolgender Checkliste die Inhalte Ihrer Dokumentation und ergänzen Sie ggf. relevante Punkte: • **Technische Umsetzung:** Beschreibung der verwendeten Low-Code-Plattform und Schnittstellen zu anderen Systemen und Datenquellen. • **Datenmodell:** Struktur der Datenbank und ggf. Erläuterung der Datenbeziehungen und -felder • **Entwicklungsprozess:** Schritt-für-Schritt Vorgehen der Umsetzung • **Benutzeranleitungen:** Anleitungen zur Nutzung der Anwendung für Endbenutzer und FAQs. • **Sicherheitsrichtlinien:** Maßnahmen zur Gewährleistung von Sicherheit und Datenschutz sowie Zugriffs- und Berechtigungskontrollen. • **Teststrategie:** Dokumentation zu den durchgeführten Tests. • **Wartung und Support:** Informationen zur Wartung der Anwendung. Kontaktinfos für technischen Support. **Erläutern Sie Ihren Kollegen die von Ihnen vorgenommene Software-Anpassung, deren Vorteile sowie ggf. die veränderte Softwarebedienung.** Stellen Sie dar, wie eine veränderte oder neue Softwarelösung in das System oder/und die Arbeitsabläufe integriert wurde. Erstellen Sie Infomaterial zu neuen Möglichkeiten und veränderten Softwarebedienung. Machen Sie die Dokumentation zugänglich für alle Nutzer der veränderten/neuen Software-Anwendung.
Lernmedien	• Plattformen für Low-Code Tools • Blog-Posts zu Voraussetzungen und zu Einsatzbereichen von Low-Code Programmierung • Community-Foren zur Auswahl von Tools und zum Einsatz von Low-Code Programmierung • Video-Tutorials zur Ergänzung/zum Aufbau ggf. noch fehlender Programmierkenntnisse
Empfohlene Vorkenntnisse	**Awareness Softwareentwicklung, 1.1 Grundlagen der Softwarebedienung, 1.2 Eigenen Bedarf an Anpassungen von Software erkennen, 1.4 Anforderungen für die Weiterentwicklung von Software formulieren, 3.1 Anpassung oder Erstellung von Software mit No-Code Programmierung**
Dauer	**40 Stunden in 3 Monaten**

3.3 Entscheidungsgrundlagen und Richtlinien zur Anpassung oder Erstellung von Software

Niveaustufe 1 2 3

Rolle	**Planer**
Lernziel	Die Lernenden können eine Entscheidungsgrundlage erarbeiten, ob vorhandene Software im Unternehmen mit internen Ressourcen angepasst oder erweitert werden kann und welche unternehmensspezifischen Richtlinien dabei einzuhalten sind.
Lernschritte / Vorgehensweise / Inhalte	**Informieren Sie sich in Ihrem oder anderen Teams zu bekannten Schwachstellen bei der Nutzung von Software.** Erkundigen Sie sich, ob ggf. bereits Anforderungsanalysen aufgrund von Engpässen und wiederkehrenden Problemen in bestehenden Software-Lösungen durchgeführt wurden. Erarbeiten Sie sich über Gespräche oder Anfragen per E-Mail, einen Gesamtüberblick zu offenen Potenzialen in unterschiedlichen Bereichen. **Priorisieren Sie, an welchen Maschinen bzw. Softwareanwendungen Anpassungen am dringlichsten durchzuführen sind.** Tauschen Sie sich dazu mit Teamleitungen und Vorgesetzten aus. Schätzen Sie den erwarteten Zeit- und Kostengewinn ein, der durch die Durchführung einer Anpassung zu erwarten ist. Erstellen Sie eine Diskussionsvorlage als Basis für einen Austausch mit relevanten Stakeholdern im Unternehmen. **Überlegen Sie, ob die Durchführung einer geplanten Anpassung oder Erweiterung mit internen Ressourcen bzw. internen Kapazitäten und Kompetenzen (Szenario 1) oder über externe Dienstleister (Szenario 2) machbar ist.** Erkunden Sie sich nach internen Kompetenzen und Kapazitäten und sprechen Sie sich mit Vorgesetzten oder entsprechenden Fachexperten im Unternehmen ab. Listen Sie Vor- und Nachteile und kalkulieren sie Kosten/Nutzen als Entscheidungsgrundlage. Grundlegende Vorgehensweisen zur Anpassung standardisierte Software sind: • **Soft Customization:** Die Anpassung kann durch Konfigurationseinstellungen, Benutzeroberflächenanpassungen oder durch die Verwendung von Plugins und Modulen durchgeführt werden. Die grundlegende Softwarestruktur bleibt intakt, da der Quellcode nicht verändert wird. • **Hard Customization:** bezieht sich auf tiefgreifende Anpassungen von Software mit Änderungen des Quellcodes. Wartung und Aktualisierungen der Software werden möglicherweise erschwert. **Szenario 1: Anpassung oder Erweiterung einer Software mit internen Ressourcen. Klären Sie rechtliche Vorgaben und Vertragsbedingungen bei einer unternehmensinternen Software-Anpassung.** Prüfen Sie u. a. folgende Punkte: • **Lizenzbedingungen:** Überprüfen Sie die Softwarelizenz, um sicherzustellen, dass Anpassungen erlaubt sind. • **Urheberrecht:** Achten Sie darauf, dass Sie die Urhaberrechte der Software respektieren. Eine Anpassung ohne Erlaubnis des Rechteinhabers kann rechtliche Konsequenzen haben. • **Datenschutz:** Beachten Sie die geltenden Datenschutzgesetze (z. B. DSGVO), insbesondere wenn persönliche Daten involviert sind. • **Vertragsbedingungen:** Stellen Sie sicher, dass die Anpassungen nicht gegen vertragliche Vereinbarungen verstoßen. • **Support und Wartung:** Klären Sie, ob und wie Support und Wartung für die angepasste Software gewährleistet sind. • **Dokumentation:** Halten Sie alle Änderungen und Anpassungen schriftlich fest, um im Falle von rechtlichen Fragen oder Problemen nachweisen zu können, was geändert wurde. **Szenario 2: Vergabe der Umsetzung einer Anpassung/Erweiterung von Software an einen externen Dienstleister.** Stellen Sie sicher, dass Anforderungsanalyse und Zielsetzung der Anpassung dem Durchführenden bekannt sind und stimmen Sie einen Projektplan gemeinsam ab. **Stellen Sie alle Informationen zu vorhandenen unternehmensspezifischen Richtlinien zu einer Software-Anpassung zusammen.** Vermittlung der Inhalte an den ausführenden Fachexperten. Ergänzung der unternehmensspezifischen Richtlinien um geltende gesetzliche und regulatorische Vorgaben für die Anpassung von Software (beispielsweise DIN-Vorgaben zur Gebrauchstauglichkeit von Software), Best Practices zur Sicherstellung von Datenschutz und Compliance und Sicherheitsrichtlinien (z. B. Zugriffskontrollen).
Lernmedien	• Unterlagen zu unternehmensinternen Richtlinien • Verträge mit Software-Lieferanten • Datenschutzgrundverordnung (DSGVO) • Webseiten zu gesetzlichen Vorgaben zu Software-Engineering, z. B. www.iso.org oder www.din.de
Empfohlene Vorkenntnisse	**Awareness Softwareentwicklung, 1.3 Methoden zur Erhebung von Nutzerbedarfen für Software, 1.5 Anforderungsanalyse für die Weiterentwicklung bestehender Software, 2.1 Projektierung einer Softwareentwicklung für das eigene Unternehmen, 3.1 Anpassung oder Erstellung von Software mit No-Code Programmierung, 3.2 Anpassung oder Erstellung von Software mit Low-Code Programmierung**
Dauer	**40 Stunden in 3 Monaten**

4.3 Management von Softwarefehlern

Niveaustufe 1 2 3

Rolle	**Planer**
Lernziel	Die Lernenden können einen kontinuierlichen Verbesserungsprozess für Softwarequalität und zum Umgang mit Softwarefehlern entwickeln.
Lernschritte / Vorgehensweise / Inhalte	**Erstellen Sie eine Übersicht zu Berichten über Softwarefehler in Ihrem Unternehmen/Ihrem Arbeitsbereich.** Beachten Sie den Unterschied zwischen Softwarefehlern und Softwarestörungen: Ein Softwarefehler (Bug) ist ein spezifisches Problem im Code, der zu unerwartetem Verhalten oder falschen Ergebnissen führt. Eine Softwarestörung bezieht sich auf eine Unterbrechung oder den Ausfall einer Softwareanwendung, die deren Nutzung beeinträchtigt oder unmöglich macht. **Machen Sie sich vertraut mit gängigen Methoden der Ursachenanalyse.** Prüfen Sie, ob Ihnen eine der folgenden gängigen Methoden bekannt ist und machen Sie sich ggf. kundig, welche Methode für welche Zielsetzung geeignet ist, z.B. Ishikawa-Diagramm, Fehlerbaumanalyse, Pareto-Analyse, SWOT-Analyse. **Wählen Sie eine der Methoden aus und wenden Sie dies auf ein Ihnen bekanntes Software-Problem an.** Eine einfache typische Methode für technische Probleme ist die „5-Why-Analyse“, diese hilft dabei, die Ursachen von Problemen systematisch zu analysieren und zu beheben. Ein Beispiel für ein Problem zu einem Softwarefehler: • **Problem:** Die Software stürzt beim Speichern von Daten ab. • **1. Frage:** Warum stürzt die Software beim Speichern von Daten ab? Antwort: Weil ein Fehler im Speichermodul auftritt. • **2. Frage:** Warum tritt ein Fehler im Speichermodul auf? Antwort: Weil die Datenbankverbindung unterbrochen wurde. • **3. Frage**: Warum wurde die Datenbankverbindung unterbrochen? Antwort: Weil der Server überlastet waren. • **4. Frage:** Warum war der Server überlastet? Antwort: Weil zu viele gleichzeitige Anfragen von Benutzern gestellt wurden. • **5. Frage:** Warum konnte eine hohe Zahl von gleichzeitigen Anfragen nicht bewältigt werden? Ergebnis: Die Software hält die hohe Benutzerlast nicht aus. **Erarbeiten Sie sich eine Übersicht zu Kriterien bei der Priorisierung von Fehlerbehebungen.** Soweit nicht bereits vorhanden - erstellen Sie auf Basis der nachfolgend vorgeschlagenen Kriterien eine Liste für Ihr Unternehmen/Ihren Arbeitsbereich. • **Schweregrad des Fehlers:** Wie gravierend ist der Fehler? Beeinträchtigt er die Funktionalität oder Sicherheit? • **Häufigkeit des Auftretens:** Wie oft tritt der Fehler auf? Je häufiger, desto höher die Priorität. • **Auswirkungen auf Benutzer:** Wie stark beeinflusst der Fehler die Benutzererfahrung oder den Geschäftsbetrieb? • **Geschäftliche Auswirkungen:** Welche finanziellen oder operationellen Konsequenzen hat der Fehler für das Unternehmen? • **Komplexität der Behebung:** Wie schnell muss der Fehler behoben werden, um negative Auswirkungen zu minimieren? **Füllen Sie das erstellte Template mit einem der letzten Fehlermeldungen beispielhaft aus.** **Schauen Sie sich Best Practice Beispiele zur Fehlerbehebung an und erstellen Sie eine Fehlerhistorie.** Eruieren Sie, wie bisherige Softwarefehler bzw. Softwarestörungen in Ihrem Unternehmen behoben wurden. Beispielsweise über interne Fachexpertise oder einen externen Dienstleister oder über Workshops mit Kollegen. Finden Sie die Erfolgsfaktoren von bisherigen Fehlerbehebungen heraus (z.B. schnelle Problemerfassung und -Weiterleitung Support). Nutzen Sie die Erstellung einer solchen Fehlerhistorie, um wiederkehrende Probleme schneller zu erkennen und zu adressieren. Machen Sie sich kundig, wie Sie Monitoring-Tools für eine laufende Überwachung und Analyse von Softwareleistung einsetzen können, um potenzielle Probleme frühzeitig zu identifizieren. **Richten Sie einen Ablageort ein (digital oder analog), an dem alle Informationen zu bisherigen Software-Fehlern strukturiert und umfassend dokumentiert sind.** Tauschen Sie sich mit Ihren Vorgesetzten und /oder Kollegen dazu aus, wie Sie zukünftig das Fehlermanagement gemeinsam verbessern können.

4.3 Management von Softwarefehlern

	Überlegen Sie gemeinsam mit Vorgesetzten und Kollegen, Maßnahmen zur zum möglichst effektiven Umgang mit Softwarefehlern und -störungen. Organisieren Sie einen Workshop in Ihrem Unternehmen/Ihrem Team zum Umgang mit Softwarefehlern und -störungen. Überlegen Sie, inwieweit dieser Umgang der sonstigen Fehlerkultur im Unternehmen entspricht. Vermitteln Sie die wichtigsten Grundregeln zum Umgang mit Software-Fehlern. Tauschen Sie sich in einem geschützten Raum aus, mit welchen präventiven Maßnahmen Sie in Zukunft Softwarefehler reduzieren und eventuelle Folgeschäden minimieren können. **Erstellen Sie einen Code of Conduct (Verhaltensrichtlinien) für den Umgang mit Softwarefehlern.** Ein Code of Conduct kann helfen, den Umgang mit Softwarefehlern zu standardisieren und eine Kultur der kontinuierlichen Verbesserung zu fördern. Es ist ratsam, spezifische Richtlinien für das eigene Unternehmen zu entwickeln oder anzupassen. Ziehen Sie bei der Erstellung des Code of Conduct folgende Punkte ein: • **Transparenz:** Offene Kommunikation über erkannte Fehler und deren Status. • **Verantwortung:** Klare Zuweisung von Verantwortlichkeiten für die Fehlerbehebung. • **Priorisierung:** Festlegung von Prioritäten für die Behebung von Fehlern basierend auf Schweregrad und Auswirkungen. • **Dokumentation:** Sorgfältige Dokumentation von Fehlern, deren Ursachen und den durchgeführten Maßnahmen. • **Feedback:** Förderung von Rückmeldungen von Benutzern zur Verbesserung der Software. • **Lernkultur:** Ermutigung zur Analyse von Fehlern, um zukünftige Probleme zu vermeiden und das Lernen aus Fehlern zu fördern. • **Sicherheitsbewusstsein:** Berücksichtigung sicherheitsrelevanter Aspekte bei der Fehlerbehebung.
Lernmedien	• Leitfäden und Best Practice Beispiele zum Thema Fehler- und Lernkultur • Online-Kurse zur Kategorisierung und zum Umgang mit Software-Fehlern • Diskussionsplattformen zu Erfahrungen im Umgang und bei der Behebung von Software-Fehlern
Empfohlene Vorkenntnisse	**Awareness Softwareentwicklung, 4.1 Zuständigkeiten für Support, 4.2 Verhalten bei Fehlermeldungen im Arbeitsprozess**
Dauer	**32 Stunden in 6 Monaten**