

4 Fallstricke bei der Low-Code-Anwendungsentwicklung:

Die eigene Vorgehensweise überdenken und häufige Fehler vermeiden.

1

Warum ein Low-Code Development-Ansatz wichtig ist.

2

Fallstrick 1: Das Entwicklungsteam ist zu groß und zu sehr spezialisiert.

4

Fallstrick 2: Sie beschäftigen sich nicht eingehend genug mit dem Problem.

5

Fallstrick 3: Feedback-Prozess ist langsam, knapp und verzerrt.

6

Fallstrick 4: Erfolg wird nur beim Onlineschalten gemessen.

7

Wie man Low-Code Development optimieren kann.

8

Über Appian.

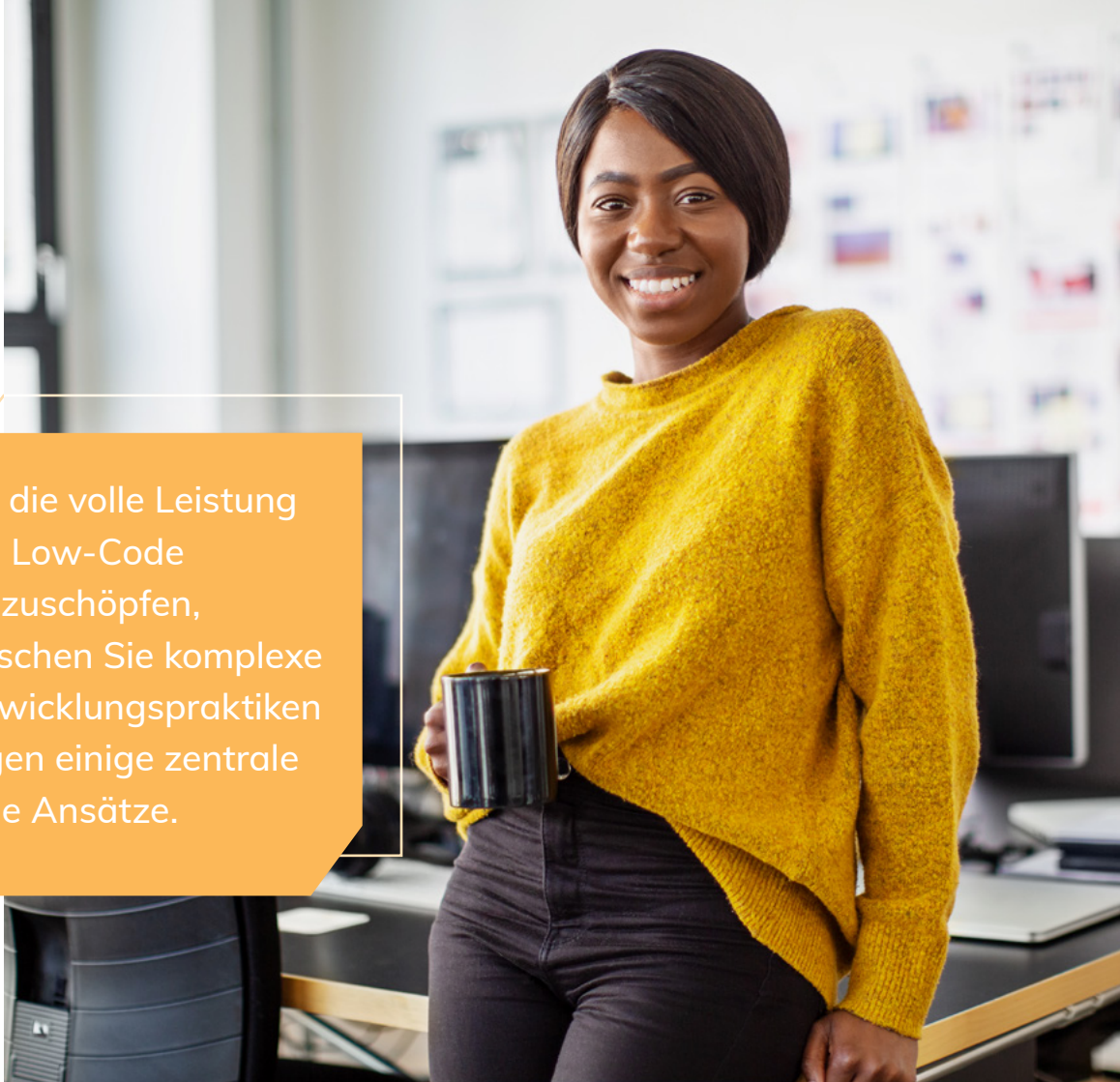
IT-Teams in den unterschiedlichsten Branchen setzen auf Low-Code als neue Methode zur Anwendungsentwicklung.

Der Prozess funktioniert nicht nur schneller (und bringt im Vergleich zur traditionellen Entwicklung leistungsstarke Anwendungen in der 10-fachen Geschwindigkeit oder sogar noch schneller hervor) – wenn Sie richtig vorgehen, können Sie mit Low-Code bessere Funktionalität liefern, mit kleineren, vielseitigeren Entwicklerteams arbeiten und die Zusammenarbeit mit den Benutzern optimieren.

Leider schöpfen viele IT-Teams das volle Potenzial von Low-Code bislang nicht annähernd aus. Der Grund dafür ist, dass sie Low-Code-Projekte unter Anwendung von High-Code-Ansätzen durchführen.

Um Low-Code optimal zu nutzen, können Sie nicht einfach Java durch eine Low-Code-Plattform ersetzen und Ihre gewohnte Vorgehensweise ansonsten beibehalten. Sie müssen auf veraltete, komplexe Entwicklungsansätze verzichten und stattdessen einige wichtige Team- und Produktdesign-Ansätze implementieren.

Entdecken Sie die vier häufigsten Fallstricke bei der Low-Code-Anwendungsentwicklung und finden Sie heraus, wie Sie diese mithilfe eines modernen Low-Code-Ansatzes bekämpfen können.



Um die volle Leistung von Low-Code auszuschöpfen, tauschen Sie komplexe Entwicklungspraktiken gegen einige zentrale agile Ansätze.

Fallstrick 1: Das Entwicklungsteam ist zu groß und zu sehr spezialisiert.



Traditionelle Anwendungsentwicklungsteams sind groß, spezialisiert und es bestehen zahlreiche Abhängigkeiten innerhalb von bzw. zwischen verschiedenen Teams.

Die erfolgreichsten Low-Code-Teams sind in ihrer Arbeitsweise auf Wendigkeit und die Minimierung von Abhängigkeiten ausgerichtet.

Sie sind klein, besitzen aber dennoch alle notwendigen Fähigkeiten, um Ideen umzusetzen und Anwendungen hervorzubringen. Sie arbeiten zudem projektübergreifend zusammen, sodass das Teamwork bei allen Projekten auf Anhieb und ohne neuerliche Lernkurve funktioniert.

Low-Code-Teams sollten kollaborativ und egalitär arbeiten. Das Aufgabenprofil eines jeden Teammitglieds sollte neben der Entwicklertätigkeit noch weitere Kompetenzen umfassen – etwa „Entwickler und Tester“ oder „Entwickler und Scrum Master“. Jeder, vom Teamleiter bis zum Software-Architekten, wird Funktionen entwickeln. Und jeder Entwickler sollte mindestens ein zusätzliches Spezialgebiet haben – z. B. UX-Design, Testen oder Architektur.

Stellen Sie bei der Zusammenstellung Ihres Teams sicher, dass diese sechs Spezialgebiete abgedeckt sind:

Produktinhaber. Dies ist die einzige Rolle, die nicht die Entwicklung von Funktionen umfasst. Der Produktinhaber sollte die geschäftlichen Anforderungen darlegen, den Wert priorisieren und sicherstellen, dass die Anforderungen für alle Beteiligten sichtbar, transparent und klar sind.

UX-Design. Mindestens eine Rolle im Team sollte den Bereich UX-Design abdecken. Diese Person kann sicherstellen, dass bewährte Designprinzipien befolgt werden, und Lösungsentwürfe für wichtige Designentscheidungen erstellen.

Technisches Design und Architektur. Dieser Aufgabenbereich bezieht sich auf die Lenkung der Lösungsarchitektur. Somit soll sichergestellt werden, dass das Team bei der Anwendungserstellung korrekt vorgeht.

Unternehmensanalyst. Diese Person befasst sich eingehend mit den geschäftlichen Anforderungen und den Bedürfnissen der Benutzer. Sie forscht nach, stellt kritische Fragen und denkt jedes Szenario durch, damit ausgehend von komplexen Prozessen eine Lösung kodiert werden kann.

Tester. Neben der Entwicklung von Funktionen umfasst diese Rolle die Durchführung von explorativen Tests, um fehleranfällige Bereiche zu identifizieren, die von Testskripten nicht erfasst werden. Die Verantwortlichen für diesen Bereich müssen zudem Kenntnisse über Tools für automatisierte Tests und Leistungs- und Lasttests mitbringen.

Teamleiter oder Scrum Master. Diese Rolle ermöglicht dem Team schnelle Entwicklungsarbeit. Die verantwortliche Person koordiniert die Entwicklungsaufgaben, beseitigt Hindernisse und fördert agile Praktiken.

Low-Code hilft Ihnen, die Komplexität in allen Bereichen niedrig zu halten – auch in Bezug auf Ihre Anwendungsteams.

Da Low-Code einfacher zu erlernen ist als High-Code, können Spezialisten in Bereichen wie UX, Ethnografie, Testen und Business-Analyse als Low-Code-Entwickler übergreifend geschult werden. Diese neuen „generalisierten Spezialisten“ bringen interdisziplinäre Fähigkeiten mit, die es Ihnen ermöglichen, Entwicklungsteams anders zusammenzustellen und die volle Leistung von Low-Code auszuschöpfen. Teams, die in Hinblick auf Low-Code-Prozesse optimiert wurden, tragen zur Kostensenkung bei – sowohl in Bezug auf den Arbeitsaufwand (weniger Mitarbeiter, die voll ausgelastet sind) als auch auf das Produkt (wirklich funktionsübergreifende und kollaborativ arbeitende Teams entwickeln mit viel höherer Wahrscheinlichkeit ein angemessenes Produkt). Mithilfe von Low-Code-Teams lässt sich die Time-to-Value verkürzen, da dank ihrer vielseitigen Zusammensetzung interne Engpässe beseitigt und externe Abhängigkeiten reduziert werden können. Zum Beispiel ist es nicht mehr notwendig, auf die Verfügbarkeit eines Datenbankspezialisten zu warten, wenn ein Datenbankexperte und Low-Code-Entwickler Teil des Teams ist.



Fallstrick 2: Sie beschäftigen sich nicht eingehend genug mit dem Problem.



Traditionelle Entwicklungsansätze erfordern Unternehmensanalysten, die zwischen Geschäftsanwendern und Technologie-Teams vermitteln. Da diese beiden Bereiche üblicherweise voneinander isoliert sind, arbeiten auch die Personen, die mit dem Problem konfrontiert sind, getrennt von denjenigen, die am besten in der Lage sind, es zu lösen: den Entwicklern. Auch wenn Entwickler versuchen, das Problem vollständig zu erfassen, wird dieser Prozess durch Kommunikationskanäle, die Missverständnisse begünstigen (z. B. E-Mail), fehlerhafte Übersetzungen, Latenzzeiten, übermäßiges Vertrauen in die Dokumentation und – die gravierendste Schwierigkeit – durch fehlenden Zugang zu Endnutzern erschwert.

Anstelle von Programmierern mit einem Jahrzehnt Erfahrung in der Nutzung von Java sollten Sie nach Entwicklern suchen, die mehr daran interessiert sind, Probleme zu lösen, als Code zu schreiben. Diese Personen können überdies ein oder zwei technische Spezialgebiete sowie allgemeine Kenntnisse der Softwareentwicklung und der Geschäftsbereiche, in denen sie arbeiten, haben.

Wenn Sie die richtigen Mitarbeiter finden, sollten Sie Ihr Team auch dazu ermutigen, offene Fragen zu stellen. Anstatt zu überlegen, ob die Einhaltung der Anforderungen ein Problem darstellt, sollten Sie stattdessen lieber erwägen, welche Konsequenzen eine Nichteinhaltung der Anforderungen nach sich zieht. Nehmen Sie Anforderungen nie für bare Münze. Fördern Sie die Gewohnheit Ihrer Entwickler, weiterzudenken, Sachverhalte zu hinterfragen und nichtlineare Lösungen zu entdecken.

Dieser Ansatz sollte sich bereits während der Projektinitiierung, die oft als „Sprint Zero“ bezeichnet wird, anhand von gemeinsamen Sitzungen abzeichnen, an denen die IT-Abteilung, die Geschäftssponsoren und die Endbenutzer teilnehmen. Arbeiten Sie zusammen, um Ziele zu definieren, zu untersuchen, was die Anwendung zur Verwirklichung dieser Ziele leisten muss, und einen Plan zu entwerfen, um ein wertvolles Produkt zu liefern. Es geht nicht darum, jedes Detail

Ihres Projekts zu definieren, sondern so gut zu planen, dass Sie in die richtige Richtung starten.

Sprechen Sie mit Geschäftsanwendern über diese fünf Schlüsselbereiche:

Vision

Was wollen Sie mit der Anwendung erreichen?

Stakeholder

Welche Endanwender und andere Stakeholder werden davon profitieren?

Bedarf

Welches Problem soll diese Anwendung lösen bzw. welcher Bedarf soll dadurch gedeckt werden? Welche Schmerzpunkte und Engpässe zeichnen sich ab und was ist der gewünschte Nutzen?

Anwendung

Was sind die Must-Have-Funktionen und -Merkmale? Warum sind sie Must-Haves? Gibt es einen kausalen Zusammenhang zwischen der Funktion und einem messbaren Geschäftsziel?

Geschäftsziele

Welche messbaren Geschäftsziele gibt es (nach Priorität gereiht)?

Denken Sie auch daran, den Ist-Zustand für Ihre Benutzer zu beurteilen. Da Ihre neue Anwendung die Art und Weise, wie sie arbeiten, verändern wird, müssen Sie verstehen, welche Ziele Ihre Stakeholder verfolgen und was ihnen diesbezüglich im Weg steht. Das funktioniert am besten, indem Sie einen Praxisbezug zu echten Anwendern aufbauen und sie bei ihrer täglichen Arbeit beobachten. Wenn das nicht möglich ist, bitten Sie die Anwender, ihre täglichen Prozesse zu beschreiben und sich darauf zu konzentrieren, warum sie die jeweilige Aktivität ausführen und was ihre größten Schmerzpunkte im Prozess sind.

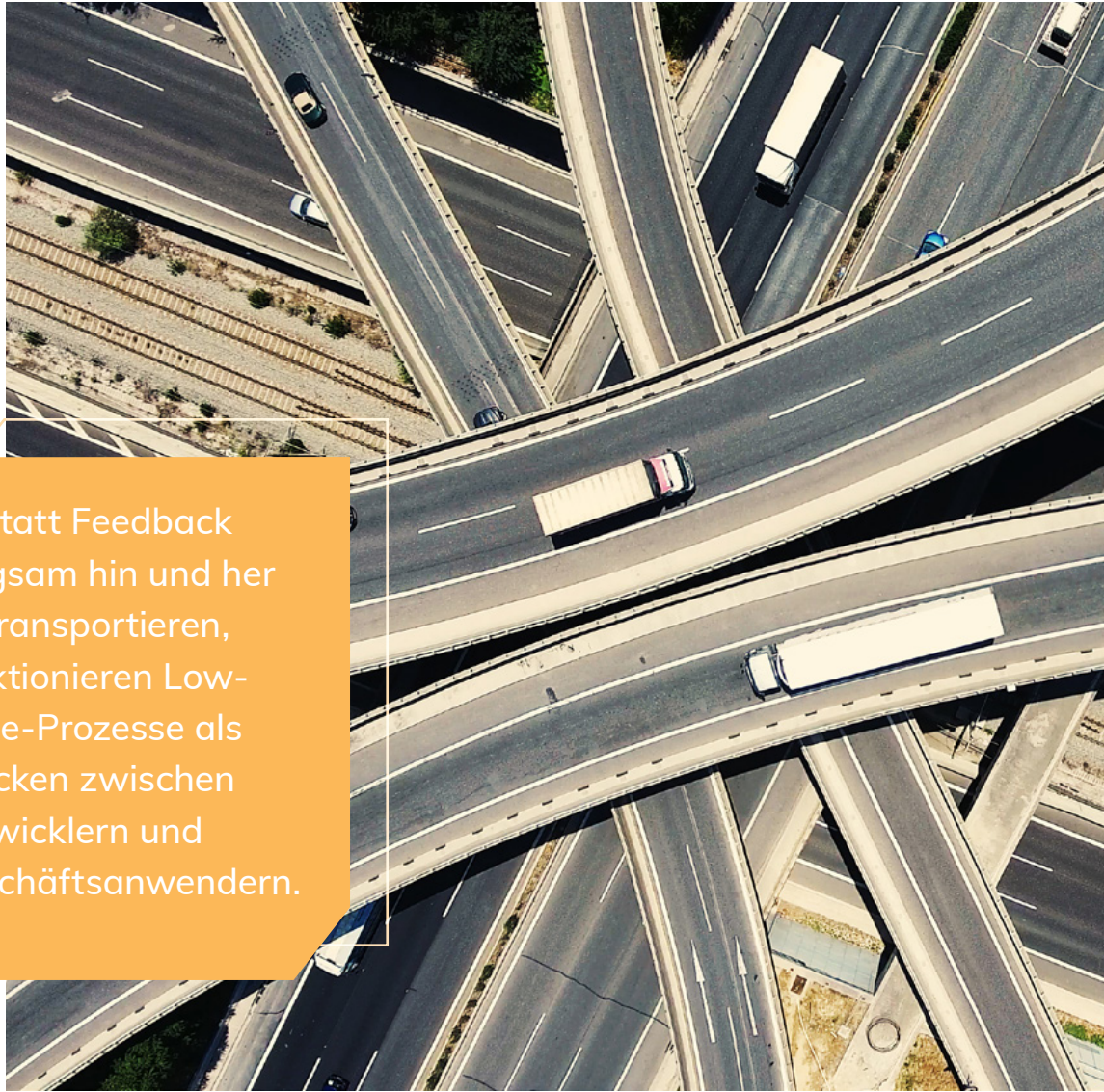
Fallstrick 3: Feedback-Prozess ist langsam, knapp und verzerrt.



Low-Code-Prozesse beseitigen die Hindernisse, die High-Code-Projekte verlangsamen. Bei traditionellen Entwicklungspraktiken sind die Feedback-Schleifen vergleichbar mit einer Fähre, die zwischen zwei Ufern verkehrt. Die Geschäftsanwender befinden sich auf der einen Seite und die Entwickler, die die Anwendung erstellen, auf der anderen Seite. Informationen werden langsam hin und her transportiert. Der Prozess ist dabei von hoher Latenz geprägt und es besteht ein großes Risiko für Verzerrungen.

Die besten Lösungen entstehen, wenn Fachanwender und Entwickler gemeinsam Anwendungen erstellen. Anstatt eine endlose Feedback-Schleife aufrechtzuerhalten, funktionieren Low-Code-Prozesse als Brücke zwischen Entwicklern und Fachanwendern. Eine solche Vorgehensweise vereinfacht die Zusammenarbeit und fördert die regelmäßige Weitergabe von klarem Input, der Ihnen hilft, Annahmen zu testen, während Sie sich an die richtige Lösung herantasten.

Die verstärkte Zusammenarbeit und die iterative Entwicklung helfen Ihnen auch, frühzeitig die Zustimmung der Beteiligten zu erhalten. Die Entwicklung findet somit nicht im Rahmen isolierter Prozesse statt, die erst am Ende des Projekts einen ersichtlichen Mehrwert liefern. Stattdessen können die Endbenutzer die Anwendung validieren und beeinflussen, während sie vor ihren Augen Gestalt annimmt. Prototypen und laufende Builds geben den Anwendern etwas Konkretes, auf das sie reagieren können, und beschleunigen den Designprozess, indem sie sicherstellen, dass Ihr Entwicklungsteam ein klares Verständnis von den Anforderungen hat.



Anstatt Feedback langsam hin und her zu transportieren, funktionieren Low-Code-Prozesse als Brücken zwischen Entwicklern und Geschäftsanwendern.

Die Bereitstellung ist ein bedeutender Meilenstein. Aber wenn Ihre Anwendung nicht genutzt wird oder wenn die Benutzer unzufrieden sind, wird sie nie einen Wert liefern.

Dies wird bereits ab dem Sprint Zero relevant. Während dieser Phase müssen Sie ein tiefgreifendes Verständnis für die Bedürfnisse und Schmerzpunkte Ihrer Benutzer aufbauen, das als Grundlage für Ihre Lösung dient.

Konzentrieren Sie sich während des gesamten Entwicklungslebenszyklus weiterhin auf Ihre Endbenutzer. Identifizieren Sie einige wichtige Stakeholder, die in der Benutzergruppe ein hohes Ansehen genießen, und binden Sie sie so oft und so früh wie möglich in den Prozess ein, um kontinuierlich den Geschäftswert zu demonstrieren. Probieren Sie die folgenden Ideen aus, um Ihre Fürsprecher in die Lösungsfindung einzubeziehen:

- Laden Sie sie ein, am Designprozess teilzunehmen, um Input zu geben und Fragen zu stellen. Bitten Sie sie dann, sich erneut an die breitere Benutzergruppe zu wenden, um über die Lösung, die entwickelt wird, zu sprechen.
- Stellen Sie die Demo-Schlüssel zur Verfügung, damit sie eine Showcase-Demo für die breitere Benutzergruppe durchführen können.
- Holen Sie Feedback von ihnen und anderen Benutzern ein.

Planen Sie Gelegenheiten ein, bei denen die IT-Abteilung in den Hintergrund treten kann, damit Ihre Benutzer im Fokus stehen können. Das Ziel ist es, dass sie ebenso in die Entwicklung der Anwendung involviert sind wie Sie.

Low-Code hilft Ihnen, Anwendungen schnell und gleich beim ersten Mal richtig zu erstellen. Um das volle Potenzial von Low-Code auszuschöpfen, denken Sie über Ihre traditionellen Entwicklungsansätze hinaus, um Folgendes zu erreichen:

- Optimieren Sie die Entwicklung mithilfe eines straffen, agilen Teams.
- Definieren Sie das richtige Problem, indem Sie ein tiefes Verständnis für Ihre Benutzer entwickeln.
- Vereinfachen Sie die Zusammenarbeit mit Geschäftsanwendern.
- Sorgen Sie für erfolgreiche Anwendungen, die auf Akzeptanz und positive Resonanz stoßen.

Es spielt keine Rolle, ob Sie gerade erst mit der Implementierung von Low-Code-Prozessen beginnen oder bereits an der Skalierung über Ihr gesamtes Unternehmen hinweg arbeiten. Indem Sie einige agile Best Practices einführen, können Sie die Entwicklung signifikant beschleunigen und anstatt mittelmäßigen Anwendungen herausragende Lösungen liefern.

Weitere Best Practices, Tools, Vorlagen und andere Ressourcen zur Optimierung Ihrer Low-Code-Anwendungsentwicklung finden Sie in der [Appian Delivery Methodology](#).



Indem Sie einige agile Best Practices einführen, können Sie die Entwicklung signifikant beschleunigen und anstatt mittelmäßigen Anwendungen herausragende Lösungen liefern.

Appian unterstützt Unternehmen mit einer Low-Code-Automatisierungsplattform bei der schnellen Erstellung von Anwendungen und Workflows. Durch die Kombination von Menschen, Technologien und Daten in einem einzigen Workflow kann Appian Unternehmen helfen, Ressourcen zu maximieren und Geschäftsergebnisse zu verbessern. Viele der weltweit größten Unternehmen nutzen Appian-Anwendungen, um das Kundenerlebnis zu verbessern, operative Exzellenz zu erreichen und globales Risikomanagement und Compliance zu vereinfachen.

Die Mitglieder des Appian Customer Success Teams sind die führenden Experten für Appian-Lösungen. Wir legen größten Wert darauf, das Richtige für unsere Kunden zu tun. Unser Fokus liegt darauf, mit unserer Low-Code-Automatisierungsplattform schnell die besten Geschäftsergebnisse für sie zu erzielen.

Machen Sie praktische Erfahrungen mit der [Appian Community Edition](#) oder kontaktieren Sie success@appian.com, um zu erfahren, wie Appian Customer Success Ihnen bei Ihren Low-Code-Projekten helfen kann.



**Was uns am meisten
überrascht hat, ist die
Tatsache, wie rasch wir
eine Anwendung entwickeln
und bereitstellen können.**

Ryder

Weitere Informationen finden
Sie unter de.appian.com
Wenn Sie uns kontaktieren möchten, senden
Sie eine E-Mail an info.de@appian.com



Appian unterstützt Unternehmen bei der schnellen Erstellung von Anwendungen und Workflows mit einer Low-Code-Automatisierungsplattform. Durch die Kombination von Menschen, Technologien und Daten in einem einzigen Workflow kann Appian helfen, Ressourcen zu maximieren und Geschäftsergebnisse zu verbessern. Viele der weltweit größten Organisationen nutzen Appian-Anwendungen, um die Kundenerfahrung zu verbessern, operative Exzellenz zu erreichen und globales Risikomanagement und Compliance zu vereinfachen. Für weitere Informationen besuchen Sie de.appian.com

