Richtlinie zum „Software-Sprint“

Akronym – FormsWizard

Schlussbericht

Zuwendungsempfänger:

Lötzsch, Tilsch, Stege & Steinmayer GbR

Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 01IS23S10 gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt beim Autor.

Kurze Darstellung der Aufgabenstellung und Motivation

Was war Deine Motivation?

Die Software baut auf einem vorhergehenden Projekt auf, welches unter großem Zeitdruck für einen einzelnen Anwendungsfall ehrenamtlich gebaut wurde: https://github.com/internet4refugees/beherbergung

Die Anwender jener Software waren sehr dankbar, da sie zu einer deutlichen Effizienzsteigerung gegenüber den vorher verwendeten Lösungen und damit zu einer Entlastung der Nutzerinnen führte. Gleichzeitig konnten wir einen Beitrag leisten, die Datensicherheit gegenüber der vorher verwendeten Softwarelösungen zu verbessern. Mehrere Anwendergruppen gaben uns unabhängig voneinander die Rückmeldung, dass sie sich ein ähnliches Werkzeug für viele andere Aufgaben auch wünschen würden.

Welches Problem wolltest Du mit Deinem Projekt lösen?

Benötigt wurde eine Open-Source-Komplettlösung, die sobald einmal deployed, Endnutzern ermöglicht Formulare selbstständig zu erstellen, zu veröffentlichen und darüber empfangene Datensätze kollaborativ zu verarbeiten.

Besonderer Wert wurde auf möglichst hohe Datensicherheit gelegt, was bestehende Dienste (z.B. Google Forms, Microsoft Forms, Jotform, Airtable) nicht hinreichend erfüllen. Nach unserer Überzeugung sollte der Server nur verschlüsselte Daten verwalten. Deshalb verschlüsseln wir sensible

Informationen bereits auf den Geräten der Nutzer:innen. CryptPad erfüllt diese Anforderung zwar ebenso, ist jedoch für die Erstellung von Formularen nur eingeschränkt nutzbar. Zudem lässt sich das System nur schlecht erweitern.

Für die Verarbeitung der Datensätze wurde zusätzlich gefordert, dass mehrere Nutzer:innen am selben Dokument gleichzeitig arbeiten können. So sollen Änderungen anderer in Echtzeit angezeigt werden.

Kollaboratives Arbeiten soll jedoch auch in Krisengebieten möglich sein, wo eine stabile Netzwerkverbindung nicht immer möglich ist. Daher soll FormsWizard auch offline-Bearbeitung ermöglichen. Änderungen von verschiedenen Nutzer:innen werden bei wiederhergestellter Verbindung automatisch in einen gemeinsamen Stand eingepflegt.

Die Datensätze sollen primär tabellarisch (eine Zeile je abgesendetes Formular) dargestellt werden. Wie bereits im Vorgängerprojekt soll ein effizientes Sortieren und Filtern je Spalte einstellbar sein. Darstellung und Optionen werden für jede Spalte aus dem Datentyp des entsprechenden Formularfeldes abgeleitet.

Unser Vorgängerprojekt besaß zusätzlich zur Tabellenansicht eine Kartendarstellung und ermöglichte beide geeignet zu Kombinieren um Datensätze schnell und übersichtlich zu filtern und sortieren. Wir wollten erproben, in wie weit sich dieser Ansatz auf beliebige strukturierte Datensätze verallgemeinern lässt, welche Visualisierungen für verbreitete Aufgaben dafür besonders hilfreich sind und, in welchem Maß eine automatische Zuordnung von Datentypen (Formularfeldern/ Spaltentypen) zu Visualisierungen sinnvoll und hilfreich ist. Mindestens sollte ein Framework entstehen, das für konkrete Anwendungsfälle Entwicklern erlaubt, Erweiterungen für die Datenverarbeitung einfach zu integrieren.

Wie war die geplante Vorgehensweise zur Problemlösung (auch Angabe der wichtigsten Meilensteine)?

Die grobe Architektur der Software stand frühzeitig fest:

https://github.com/FormsWizard/formswizard/blob/main/doc/src/arch/overview.md

Dem entsprechend konnten Komponenten mit klar definierten Schnittstellen voneinander abgegrenzt und weitestgehend unabhängig voneinander implementiert werden:
https://github.com/FormsWizard/formswizard/wiki/project:components

Da wir mit 4 Teammitgliedern jeweils nur Teilzeit und zu unterschiedlichen Arbeitszeiten tätig waren, legten wir wert darauf, Abhängigkeiten zwischen den Aufgaben der verschiedenen Mitstreiter zu minimieren. Deshalb wurden klare Zuständigkeiten festgelegt, welche EntwicklerInnen an welcher Komponente selbstständig arbeiten sollte.

Während des Förderzeitraums wurde agil, per Scrum mit zweiwöchigen Sprints und einem wöchentlichen Teammeeting gearbeitet. Aufgaben wurden von den Beteiligten selbstständig per Kanban verwaltet. Die Vorgehensweise wurde vor Projektstart besprochen und dokumentiert:
https://github.com/FormsWizard/formswizard/wiki/project:workflow

Um zu einem zufriedenstellenden Gesamtergebnis zu kommen und zu verhindern, dass am Ende Ressourcen für die zuletzt bearbeiteten Komponenten fehlen, wurde eine Iterative Vorgehensweise gewählt. Dafür wurde der Projektzeitraum in 3 Phasen zu jeweils ~2 Monaten aufgeteilt:

Im ersten Monat des Förderzeitraumes wurde die bestehende Anforderungsanalyse durch weitere Gespräche mit künftigen Nutzern ergänzt. Existierende (Teil-)Lösungen, Softwarebibliotheken, Standards und Protokolle wurden inspiziert. Während dessen wurden UI-Entwürfe als Wireframes entworfen. Die Ergebnisse erlaubten eine frühzeitige Definition der Schnittstellen und APIs zwischen den Komponenten.
Im zweiten Monat wurden erste funktionale Prototypen der 3 Frontend-Kernkomponenten (FormsDesigner, Formularrenderer, Processing) implementiert. Besonderer Fokus lag auf dem FormsDesigner, da im Gegensatz zu den anderen Komponenten hier keine Erfahrung aus vorangehenden Projekten existierte.

Im dritten bis vierten Monat wurde die Software iterativ weiterentwickelt und ein Großteil der geplanten Feature umgesetzt. Erste Nutzertests halfen frühzeitig zu erkennen, dass das gewählte Bedienkonzept grundlegend gut funktioniert und motivierten uns. Die gröbsten Fehler wurden zeitig erkannt und konnten entsprechend priorisiert gelöst werden.

Im fünften und sechsten Monat wurde sich auf die Optimierung der Benutzbarkeit für Endnutzer fokussiert. Es wurden Tests und Dokumentation geschrieben.

Beitrag des Projektes zu den Zielen der Förderinitative „Software-Sprint“

Wer ist die Zielgruppe für Deine Lösung? Wie profitiert sie von den Ergebnissen Deines Projekts?

Unsere Hauptanwendung richtet sich an alle Gruppen (z.B. Vereine, Gremien, Projektteams, …), die gemeinsam strukturiert Daten erheben und verarbeiten möchten. Für die meisten Anwendungsfälle von Google Forms könnte FormsWizard eine brauchbare Alternative werden.

Einen Mehrwert existiert für Nutzer:innen, die Wert auf Datenschutz-/Datensicherheit legen. Wir möchten Datensouveränität ermöglichen und geben unser bestes Nutzerdaten nicht ausschließlich rechtlich, sondern technisch zu schützen.

Aus diesem Grund profitieren besonders Gruppen, die besonders sensible Daten verarbeiten, wenn FormsWizard die nötige Reife erreicht und weiter betreut wird. Im Moment kooperieren wir insbesondere mit NGOs.

Die Kollaborative Processing-Komponente mit Offline-Funktionalität richtet sich insbesondere auch an Einsatzkräfte in Krisen-/Katastrophengebieten, wo eine stabile Netzwerkverbindung nicht gewährleistet werden kann.

Neben den Endnutzern der Hauptanwendung gibt es weitere Zielgruppen, die von Anfang des Projektes an bedacht wurden:

Der FormsDesigner ist auch als eigenständiges Werkzeug nützlich. Per Drag und Drop können Formulare einfach generiert werden – ohne nötige Programmierkentnisse. FormsDesigner kann so eigenständig für die Erstellung von Web-Formularen genutzt werden.

Erstellte Formulare werden durch eine deklarative Datenstruktur repräsentiert. Mit Absicht haben wir uns dafür entschieden dafür als Datentyp den Standard JSONSchema zu verwenden. Es gibt bereits einige verbreitete Bibliotheken (z.B. JSONForms), die Formulare auf Basis von JSONSchema generieren. Diese Vorgehensweise hat nicht nur uns viel Arbeit abgenommen, sondern ist potentiell auch spannend für alle, die den gleichen Standard nutzen: https://json-schema.org/implementations.html
FormsDesigner kann zur graphischen Erstellung von JSONSchema genutzt werden. Dies ist insbesondere für Neulinge von JSONSchema nützlich, da es einen intuitiven Zugang erlaubt ,ohne die Sprache lernen zu müssen.

JSONSchema selbst ist ein nützlicher Standard, der Umgang mit JSON erheblich erleichter. JSONSchema kann als optionales Typsystem für JSON-Datenstrukturen genutzt werden und erlaubt damit Validierung von Typsicherheit, was Sicherheit und Stabilität von Anwendungen verbessern kann. Wir hoffen mit FormsDesigners anderen Entwickler den Einstieg in JSONSchema zu vereinfachen.

Während der Implementierung von FormsWizard wurden einige Herausforderungen gelöst, die als eigenständiges Softwarepaket konzipiert wurden. Sobald wir diese finalisiert und dokumentiert haben, wollen wir sie separat veröffentlichen.

Welche Bezüge gibt es zu den Themenfeldern und Zielen des Software Sprints?

FormsWizard bemüht sich um Datensouveränität und Datenschutz.

FormsWizard ermöglicht zivilgesellschaftlichen Akteuren sichere Verarbeitung von sensiblen Daten.

Unsere Hauptzielgruppe sind NGOs, die sich für Solidarität und Menschenrechte engagieren.

Im Rahmen der Entwicklung entstandene Bibliotheken erweitern existierende Open Source Projekte. Der Philosophie „Public Money? Public Code!“ folgend stellen wir der Gesellschaft offene und nachhaltige Technologien zur Verfügung.

Ausführliche Darstellung der Ergebnisse

Welche konkreten Ergebnisse hast Du erzielt?

FormsWizard konnte wie geplant implementiert werden.

Eine Demo des Prototypen ist hier verfügbar:

https://formswizard.github.io/formswizard/new

Eine ausführliche Erklärung der Ergebnisse wurde auf dem Symposium Datenspuren gehalten. Der Videomitschnitt der Präsentation kann hier eingesehen werden:

https://media.ccc.de/v/ds23-256-formswizard-to-be-prepared-for-the-next-crisis

Konnten alle Meilensteine erreicht werden?

Ja.

(Integrations-)Tests und Dokumentation waren etwas umfangreicher geplant, werden aber gerade vervollständigt.

Welche zusätzlichen Erkenntnisse hast Du aus der Projektarbeit gewonnen, auch im Hinblick auf die Begleitung durch die Open Knowledge Foundation?

Die angebotenen Coachings haben dem Projekt durch gezieltes Feedback voran gebracht und viele Erkenntnisse erzielt.

Besonders nützlich für uns war die Vernetzung mit anderen Teilnehmern auf den DemoDays. Wir konnten viel voneinander lernen und stehen mit einigen weiterhin in Kontakt um einander zu unterstützen.

Zielgruppe, Nutzen und mögliche Weiterentwicklungen

Welcher Nutzen ergibt sich für die Zielgruppe aus den Ergebnissen Deines Projekts?

Siehe „Wer ist die Zielgruppe für Deine Lösung? Wie profitiert sie von den Ergebnissen Deines Projekts?“

Für die Zielgruppe unserer Hauptanwendung ergibt sich der Nutzen, sobald wir eine Version veröffentlichen, die wir als bereit für den Produktivbetrieb einschätzen. Das ist im Moment für November 2023 geplant. Bis dahin sind noch eine überschaubare Zahl kleinerer Anpassungen nötig. Wir streben hierfür auch einen Security Audit an.

Im Moment wird die Software von ersten Beta-TesterInnen eingesetzt.

Unsere Bibliotheken können bereits in der Open Source Entwicklung genutzt werden.

Welche weiter-gehenden Effekte ergeben sich aus der Open-Source-Stellung der Ergebnisse?

Sämtliche Projektziele sind ausschließlich durch Open-Source erfüllbar. Auf Closed-Source basierende Dienste gibt es zwar, aber denen können und wollen unsere Nutzer:innen ihre Daten nicht anvertrauen.

Unsere Entwicklung basiert auf und trägt bei zu einer weltweiten FOSS-Community, die sich für nachhaltige und gemeinnützige Softwareentwicklung einsetzt. Dieser ermöglichen wir durch eine freie Lizenz, gute Dokumentation und Standard-kompatible Entwicklung die Weiterverwendung unserer Arbeit.

Ein weiter-gehender Effekt der gelebten Open-Source-Philosophie ergibt sich aus dem freien Wissensaustausch und alltäglichen Weiterbildung im Hackspace und auf Konferenzen, auf denen wir uns mit FormsWizard beteiligen.

Gibt es Ideen für die Weiterentwicklung Deiner Lösung und Pläne zu deren Umsetzung?

Erster Meilenstein ist die Fertigstellung der Funktionalität, die für einen produktiven Einsatz erwartet wird und die Bereitstellung einer kostenlos nutzbaren Instanz durch unseren Hackerspace (geplant für November):

- Vollständige Ableitung der Spaltendefinitionen aus dem JSONSchema
- Bereitstellung von Renderern für Tabellenspalten aller relevanten Typen
- Fertigstellung angefangener Visualisierungen
- Paketierung der Deployments (Nix, Docker, Debian, …)

Als nächstes sind Erweiterungen für FormsWizard geplant, die eine unsere Entwicklung begleitende NGO für ihr Tagesgeschäft benötigt. (geplant bis Jahresende):

- PluginAPI + Webhooks

- weitere Visualisierungen

Für die Erreichung dieser Ziele ist bis Ende des Jahres wöchentlich jeweils ein ehrenamtlicher Arbeitstag eingeplant.

Im Dezember sind bereits zwei Treffen mit einigen anderen NGOs geplant, bei der wir unsere Ergebnisse erneut präsentieren möchten. Da wir für diese Zielgruppe eine besonders hohe Motivation für ehrenamtliche Betätigung aufbringen, sie besonders anspruchsvolle Anforderungen haben und eine regelmäßige enge Zusammenarbeit möglich ist, soll in der Anfangszeit die Kooperation mit diesen Gruppen priorisiert werden.

Wir bemühen uns schrittweise eine Nutzer- und Entwicklercommunity für FormsWizard aufzubauen. Als Ergebnis des ersten Konferenzbeitrags wurden uns im September bereits mehrere Angebote zur Unterstützung und Einladungen auf weitere Veranstaltungen angeboten. Als Open-Source-Projekt lassen wir gerne unsere Mitstreiter entscheiden, wie wir das Projekt gemeinsam weiterentwickeln.

Derzeit ist keine kommerzielle Verwertung geplant. Wir haben jedoch Interesse an weiteren Finanzierungsmöglichkeiten um FormsWizard in künftigen Sprints mit mehr Ressourcen weiterentwickeln zu können, als dies bei rein ehrenamtlicher Tätigkeit möglich ist. Entsprechende Möglichkeiten werden derzeit evaluiert.

Unabhängig davon sollen Schrittweise viele der von uns entwickelten Pakete mit einer stabilen API und angemessener Dokumentation vervollständigt und auf https://www.npmjs.com veröffentlicht werden, damit andere Entwickler sie als Softwarebibliothek verwenden können. Um besser aufgefunden werden zu können, sollen die Bibliotheken sobald vorzeigbar an geeigneten Stellen verlinkt werden (z.B. https://json-schema.org/implementations.html und https://github.com/yjs/yjs).

Hat die Arbeit in dem Projekt Dich in Deiner persönlichen, fachlichen Weiterentwicklung unterstützt?

Ja, sehr.

Als selbstständiger Freiberufler ist jedes längerfristige Projekt im Team eine wertvolle Chance um voneinander fachlich und menschlich zu lernen.

Als aktiver Open-Source-Entwickler und umfangreich ehrenamtlich aktiver Mensch ist es großartig, über einen Zeitraum eine solide Finanzierung für das Projekt zu haben, welches man am liebsten voranbringen möchte.

Die Gelegenheit sich intensiv mit den Technologien auseinander zu setzen, die einen interessieren, ist eine hervorragende Weiterbildungsmöglichkeit. Dafür Werkzeuge zu bauen, ist erfahrungsgemäß eine deutliche Hilfe für folgende Projekte.

Kurze Darstellung der Arbeiten, die zu keiner Lösung geführt haben

Gab es Arbeiten bzw. Lösungsansätze, die nicht weiter verfolgt wurden? Was waren die Hinter-gründe, und wie bist Du alternativ vorgegangen?

Es gab keine Arbeiten, die zu keiner Lösung geführt haben.

Statt dessen gab es aber einige Erkenntnisse in der Recherche-Phase, die dazu führten, dass ursprünglich geplante Lösungsansätze nicht begonnen und statt dessen bessere Lösungen umgesetzt werden konnten.

z.B.:

- https://tanstack.com/table bietet genau die gewünschte Abstraktion, für die wir eine Lösung selber entwerfen wollten. Der Plan, wie in vorherigen Projekten, https://github.com/inovua/reactdatagrid oder https://mui.com/x/react-data-grid/ nutzen zu müssen, konnte verworfen werden.
- Eine Adaption von https://github.com/lscheibel/redux-yjs-bindings gibt uns eine sehr gute Schnittstelle zur Synchronisation/Persistierung der Client-Zustände. Eine ursprünglich geplante Eigenlösung war nicht erforderlich.

Kurze Angabe von Präsentationsmöglichkeiten für mögliche Nutzer

Wo können sich Interessenten detailliert über Deine Projektergebnisse informieren (z.B. Webseite, GHitHub, Veröffentlichungen)?

https://github.com/FormsWizard

https://media.ccc.de/v/ds23-256-formswizard-to-be-prepared-for-the-next-crisis

https://formswizard.github.io/formswizard/new

künftig: https://formswizard.github.io/

Kurze Erläuterung zur Einhaltung der Arbeits- und Kostenplanung

Gab es im Projektverlauf Ereignisse, die eine Anpassung der Planung erforderlich machten – z.B. Mehr- oder Minderaufwand bei der Bearbeitung von Teilaufgaben?

Im Projektverlauf ergaben sich zwei Mehraufwände.

Für die FormsDesigner-Komponente war insgesamt etwa ein Drittel der verfügbaren Arbeitszeit eingeplant. Real wurde etwa die Hälfte der Ressourcen benötigt. Der Aufwand die Komponente zu entwerfen, zu implementieren, mit Nutzern zu testen und auf einen nahezu für den Produktivbetrieb geeignet zu sein, hat mehr als die veranschlagten 8 Entwicklerwochen benötigt. Unerwartet kompliziert hat sich insbesondere die Herausforderung dargestellt, Drag-and-Drop für beliebige Endgeräte (Maus und Touchscreen) umzusetzen. Auch sehr verbreitete Bibliotheken implementieren es in einigen Fällen nicht wie gewünscht. Im UX-Coaching wurde uns bestätigt, dass diese Aufgabe alleine ein eigenes Prototypefund-Projekt rechtfertigen würde.

Das Team war sehr motiviert die Komponente in bestmöglicher Qualität zu bauen. Da der FormsDesigner (in den ersten Entwürfen noch als Wizard bezeichnet) ein sehr entscheidenter Teil des Gesamtprojektes ist und im Bezug auf UI/UX die wichtigste Neuerung gegenüber den Vorgängerprojekten darstellt, wurde entschieden, dieser Komponente mehr Ressourcen zur Verfügung zu stellen.

Der zweite Mehraufwand ergab sich aus dem Wunsch das umfangreiche Projekt in viele kleine Pakete aufzuteilen. Das Paradigma jedes relevante Feature in einem eigenen Paket zu kapseln und einfach wiederverwendbar zu machen, hatte sich bereits in anderen Projekten als sehr zielführend erwiesen. Kleinteilige Modularisierung macht den Code nicht nur übersichtlicher, testbarer und allgemein besser strukturiert. Für jede Lösung einer Herausforderung ein eigenes Softwarepaket zu veröffentlichen erlaubt uns und anderen Entwicklern künftig für ähnliche Teilaufgaben die bestehende Implementierung einfach nutzen zu können und ist daher im Sinne der Philosophie von Free/Libre Open Source Software unbedingt erstrebenswert. In diesem Projekt war Modularisierung auch wichtig, um Angriffsvektoren durch unnötige Abhängigkeiten zu reduzieren und überblicken zu können.

Allerdings hatten wir für Typescript (Sprache in der FormsWizard implementiert ist) noch kein Tooling, mit dem wir richtig zufrieden waren. Separate Pakete bereitzustellen kann zu eingeschänkter Developer experience führen und erhöht die Anforderungen an DevOps. Um Software langfristig ehrenamtlich längerfristig pflegen zu können, ist uns ein hoher Automatisierungsgrad mit geeigneter CI/CD wichtig. Wir haben dieses Projekt genutzt um uns mit https://turbo.build/repo und https://pnpm.io/ vertraut zu machen. Diese Werkzeuge klangen vielversprechend, stellten uns jedoch Teils vor unerwartete neue Herausforderungen. Rückblickend halten wir die Entscheidung zur Nutzung der Tools für richtig und werden sie sehr gerne für künftige Projekte nutzen und weiterempfehlen. Nach unserer Einschätzung lassen sich unsere anfänglichen Probleme durch bessere Dokumentation und Beispiele lösen. Diese wollen wir deshalb gerne entsprechend zu diesen Werkzeugen beisteuern. Ebenso sind wir dabei, eine Integration für https://nixos.wiki/wiki/Nix_package_manager zu veröffentlichen,

um Turbo- und pnpm-Pakete Pakete unkompliziert als Nix-Pakete und damit reproduzierbar zu bauen.

Ein Minderaufwand ergab sich, als in der Recherchephase redux-yjs-bindings gefunden wurde.

Insgesamt konnte der Projektplan mit den ursprünglich geplanten Meilensteinen eingehalten werden. Der verwendete iterative Entwicklungsprozess hat uns geholfen, das Projektziel zu erreichen, indem auf neue Erkenntnisse flexibel reagiert werden konnte. Ebenso war es hilfreich Überraschungen frühzeitig erkennen zu können, indem wir Fragestellungen mit offenem Ergebnis am Anfang des Projektzeitraums eingeplant hatten.

Im Ergebnis sind wir sehr zufrieden. Der FormsDesigner (Wizard) erfüllt unsere Erwartungen im vollen Umfang. Für die Verarbeitungskomponente stand weniger Zeit als veranschlagt zur Verfügung. Deswegen wurde sich bisher auf die Kernfunktionalität beschränkt. Die Architektur ist aus unserer Sicht dennoch sehr überzeugend gelungen und konnte als praktisches Framework für künftige Projekte umgesetzt werden. Erste der geplanten Visualisierungen sind bisher prototypisch umgesetzt und sollen zeitnah vervollständigt werden.

Aus Ressourcengründen wurde die Projektwebsite niedrig priorisiert und ist bisher nicht mehr als ein Platzhalter. Bis zum geplanten Release der fertigen Anwendung (geplant im November) soll sie noch mit den bisher fehlenden Inhalten gefüllt werden.

Kurze Darstellung von etwaigen Ergebnissen bei anderen Stellen

Gab es Entwicklungen anderer Personen oder Institutionen, die Einfluss auf Deine Arbeiten und die Zielsetzung hatten? Wenn ja, worin bestand dieser und wie bist Du damit umgegangen?

https://www.nocodb.com/ haben wir erst unmittelbar vor Beginn des Förderzeitraumes kennengelernt. Uns gefallen viele der UX-Konzepte, jedoch fehlend elementare von unseren Nutzern benötigte Funktionen: https://github.com/FormsWizard/formswizard/wiki/research:nocodb

Der Kontakt zu Entwicklern von NocoDB war sehr angenehm und führte dazu, dass beide Projekte voneinander Inspiriert werden konnten.

Unsere kurzzeitige Überlegung, ob statt einer Eigenentwicklung eine Anpassung von NocoDB sinnvoll sein könnte, wurde schnell verworfen, als sich herausstellte, dass die Architektur von NocoDB zu unflexibel ist, um unsere Anforderungen integrieren zu können. Dennoch möchten wir den Kontakt gerne aufrecht erhalten und können NocoDB an NutzerInnen weiterempfehlen, die bereits jetzt

eine ausgereifte Lösung benötigen, solange der Anwendungsfall ein geringeres Maß an Datensicherheit und keine Kollaboration erfordert.