

Abschlussprüfung Sommer 2025

Fachinformatiker fr. Anwendungsentwicklung

Dokumentation zur betrieblichen Projektarbeit

Entwicklung eines Odoo-Moduls zur

**ZIP-komprimierten Dateiverwaltung** im

**Dokumentenmanagementsystem** (DMS)

Abgabetermin: 26. Mai 2025

**Diese Dokumentation wurde erstellt von:**

Mark Albers

Tienrade 1, 21031 Hamburg

E-Mail: [albers.mark02@gmail.com](mailto:albers.mark02@gmail.com)

Prüflingsnummer: (131)- 54068

**Ausbildungsbetrieb:**

BFW-Berufsförderungswerk Hamburg GmbH

Marie-Bautz-Weg 16, 22159 Hamburg

Ansprechpartner: Dr. Olaf Kubillus

**Betrieb für die Praktische Ausbildung:**

Sirum GmbH

Am Sandtorkai 32, 20457 Hamburg

Ansprechpartner: Dipl. Ing. (FH) Georg Notter

E-Mail: [Georg.Notter@Sirum.de](mailto:Georg.Notter@Sirum.de)

Inhaltsverzeichnis

[1. Einleitung - 1 -](#_Toc198059159)

[1.1. Projektumfeld - 1 -](#_Toc198059160)

[1.2. Projektbeschreibung - 1 -](#_Toc198059161)

[1.3. Projektschnittstellen - 1 -](#_Toc198059162)

[1.4. Projektbegründung - 1 -](#_Toc198059163)

[1.5. Projektziel - 2 -](#_Toc198059164)

[2. Projektplanung - 2 -](#_Toc198059165)

[2.1. Projektphasen - 2 -](#_Toc198059166)

[2.2. Ressourcenplanung - 2 -](#_Toc198059167)

[2.3. Entwicklungsprozess - 3 -](#_Toc198059168)

[2.4. SMART-Ziele - 3 -](#_Toc198059169)

[3. Analysephase - 3 -](#_Toc198059170)

[3.1. Ist-Analyse - 3 -](#_Toc198059171)

[3.2. Wirtschaftlichkeitsanalyse - 4 -](#_Toc198059172)

[3.2.1. „Make or Buy“-Entscheidung - 4 -](#_Toc198059173)

[3.2.2. Projektkosten - 4 -](#_Toc198059174)

[3.2.3. Amortisationsdauer - 5 -](#_Toc198059175)

[3.3. Nicht-monetäre Aspekte - 5 -](#_Toc198059176)

[3.4. Anwendungsfälle - 5 -](#_Toc198059177)

[3.5. Lastenheft - 6 -](#_Toc198059178)

[4. Entwurfsphase - 6 -](#_Toc198059179)

[4.1. Maßnahmen zur Qualitätssicherung - 6 -](#_Toc198059180)

[5. Kosten/Nutzen-Analyse - 7 -](#_Toc198059181)

[6. Risiko-Berechnung - 8 -](#_Toc198059182)

[7. SMART-Ziele - 9 -](#_Toc198059183)

[8. SWOT-Analyse - 10 -](#_Toc198059184)

[9. Durchführung - 11 -](#_Toc198059185)

[10. Qualitätssicherung - 12 -](#_Toc198059186)

[11. Soll-Ist-Vergleich - 13 -](#_Toc198059187)

[12. Fazit - 14 -](#_Toc198059188)

[13. Glossar - 15 -](#_Toc198059189)

[14. Anhang - 16 -](#_Toc198059190)

# Einleitung

In der folgenden Projektdokumentation wird der Ablauf des IHK-Abschlussprojektes, das der Autor im Rahmen seiner Ausbildung zum Fachinformatiker für Anwendungsentwicklung durchgeführt hat, erläutert.

## Projektumfeld

Der Ausbildungsbetrieb ist die Sirum GmbH. Sirum hat sich zum Ziel gesetzt, eine innovative Softwarelösung für die Transportlogistik zu entwickeln. Zurzeit beschäftigt Sirum etwa 30 Mitarbeiter. Während herkömmliche Lösungen oft nur einzelne Aspekte der Logistik abdecken, fokussiert sich Sirum auf eine ganzheitliche Optimierung, um Einschränkungen in der Gesamteffizienz zu überwinden. Mit dem Sirum Transport Management System (TMS), wurde eine umfassende Softwarelösung entwickelt, die auf dem Odoo-Framework basiert. Sirum setzt auf eine Software as a Service (SaaS)-Architektur, die eine transparente und kalkulierbare Kostenstruktur ermöglicht.

## Projektbeschreibung

Im Bereich der Dokumentenverwaltung wird das bestehende Document Management System (DMS) verwendet, in den verschiedenen Dateitypen wie Rechnungen, Excel-Tabellen und weitere geschäftsrelevante Dokumente abgelegt und verwaltet werden.

Das Projekt basiert auf einer konkreten Anforderung eines Kunden. Ziel war es zunächst, ein Modul zu entwickeln, mit dem ausgewählte Dateien aus dem bestehenden ir.attachment-Modell komprimiert und als ZIP-Datei heruntergeladen werden können.

Im weiteren Verlauf äußerte der Kunde zusätzliche Anforderungen: Es sollte eine benutzerfreundliche Oberfläche geschaffen werden, die eine gezielte Filterung von Dokumenten nach Datum (von–bis) und Kategorie ermöglicht. Die Ergebnisse sollten dabei in einer eigenen Benutzeroberfläche (View) angezeigt und als Datensatz gespeichert werden. Dies erlaubt es dem Nutzer, eine bestimmte Dokumentenmenge einmal zu definieren und anschließend bei Bedarf herunterzuladen, ohne den Filterprozess erneut durchlaufen zu müssen.

Das entwickelte Modul stellt somit eine umfassende Erweiterung des bisherigen DMS dar. Es unterstützt die Anwender dabei, effizienter mit großen Mengen an Dokumenten zu arbeiten und relevante Dateien gezielt zusammenzustellen und zu exportieren.

## Projektschnittstellen

Das entwickelte Modul greift auf das bestehende Modell ir.attachment zu, in dem alle im DMS gespeicherten Dokumente verwaltet werden. Es erweitert die Funktionalität dieses Modells um Filter-, Zwischenspeicherungs- und Downloadoptionen. Die Dokumente selbst bleiben im System bestehen und werden nicht verändert, sondern lediglich referenziert, gefiltert und für den Download vorbereitet.

Eine weitere Schnittstelle besteht zur View, über die der Anwender mit dem Modul interagiert. Diese View wurde speziell für das Projekt neu erstellt und nutzt einen Wizard zur Eingabe der gewünschten Filterkriterien (Datum von–bis, Kategorie). Die gefilterten Ergebnisse werden anschließend in einem temporären Record gespeichert, um eine wiederholte Verwendung zu ermöglichen.

## Projektbegründung

Im Arbeitsalltag der Fachabteilungen werden regelmäßig große Mengen an Dokumenten im DMS verwaltet, darunter unter anderem Rechnungen, Excel-Dateien und Verträge. Bisher fehlte jedoch eine Möglichkeit, diese Dokumente gezielt zu filtern und gesammelt herunterzuladen. Stattdessen mussten die benötigten Dateien manuell gesucht, einzeln ausgewählt und heruntergeladen werden, was ein zeitaufwändiger und fehleranfälliger Prozess ist.

Das Modul leistet einen Beitrag zur Standardisierung der Prozesse, indem es eine strukturierte und wiederverwendbare Lösung innerhalb des bestehenden Systems bietet. So wird nicht nur der manuelle Aufwand reduziert, sondern auch die Fehleranfälligkeit im Umgang mit Dokumenten verringert.

## Projektziel

Das Ziel des Projekts ist die Entwicklung eines Moduls zur effizienten Verwaltung und gezielten Bereitstellung von Dokumenten innerhalb des bestehenden DMS. Die Anwendung soll es den Benutzern ermöglichen, Dokumente anhand eines definierten Datumsbereichs sowie bestimmter Kategorien zu filtern, diese in einer übersichtlichen View darzustellen und bei Bedarf gebündelt als ZIP-Datei herunterzuladen.

# Projektplanung

In diesem Kapitel geht es darum, wie das weitere Vorgehen bestimmt wurde. Das Projekt wurde dazu zeitlich geplant, alle benötigten Ressourcen ermittelt und der Entwicklungsprozess bestimmt.

## Projektphasen

Für die Umsetzung des Projekts standen 80 Stunden zur Verfügung. Im Rahmen der Projektplanung wurden die Stunden auf verschiedene Phasen aufgeteilt und somit der gesamte Projektablauf abgebildet. Eine grobe Zeitplanung mit den Hauptphasen kann aus Tabelle 1 entnommen werden. Eine detaillierte Zeitplanung mit den einzelnen Schritten der jeweiligen Phase befindet sich im Anhang [**A.1: Detaillierte Zeitplanung**](#_Anhang) auf Seite i.

|  |  |
| --- | --- |
| Projektphase | Geplante Zeit |
| Analysephase | 8 h |
| Entwurfsphase | 12 h |
| Implementierung inkl. Tests | 44 h |
| Abnahme und Einführung | 5 h |
| Dokumentation | 8 h |
| Gesamt | 77 h |

Tabelle 1: Zeitplanung

## Ressourcenplanung

Im Anschluss an die Zeitplanung wurden alle für die Umsetzung des Projekts benötigten Ressourcen im [**Anhang A.2: Ressourcenplanung**](#_Anhang) aufgelistet. Dabei wurden sowohl die eingesetzten Software- und Infrastrukturressourcen als auch das benötigte Personal berücksichtigt.

Für die Entwicklung kamen ausschließlich bereits lizenzierte oder intern bereitgestellte Werkzeuge und Umgebungen zum Einsatz. Ziel war es, die Projektkosten so gering wie möglich zu halten und keine zusätzlichen Lizenzen zu beschaffen.

Im Vorfeld wurden bestehende Module mit ähnlicher Funktionalität geprüft. Da diese jedoch nicht alle Anforderungen erfüllten und zusätzliche Lizenzkosten verursacht hätten, wurde die Entwicklung eines eigenen Moduls beschlossen. Die vollständige Bewertung erfolgt in Abschnitt [**3.2.1 Make-or-Buy-Entscheidung**](#_„Make_or_Buy“-Entscheidung).

## Entwicklungsprozess

Bei der Firma Sirum wird ein Kanban-basierter Entwicklungsprozess verwendet, der über das GitLab-Issue-Board organisiert ist. Das Board ist in die typischen Statusspalten **To Do**, **In Progress**, **Review** und **Done** unterteilt. Jedes neue Issue wird zunächst im Board erfasst und priorisiert.

Für jedes angelegte Issue wird ein eigener Git-Branch erstellt, auf dem die Umsetzung erfolgt. Nach der Entwicklung des jeweiligen Features werden Unit-Tests geschrieben, um die Funktionsweise des Moduls automatisiert zu überprüfen.

Anschließend erfolgt ein Code-Review durch einen Entwickler, der den Branch gemeinsam mit dem zugehörigen Issue prüft. Wenn keine Änderungen mehr notwendig sind, wird der Branch durch den unternehmensinternen Merge-Bot verarbeitet. Die dabei durchlaufenen Prüfschritte und Qualitätssicherungsmaßnahmen werden in Abschnitt [**4: Maßnahmen zur Qualitätssicherung**](#_Maßnahmen_zur_Qualitätssicherung) näher beschrieben.

Durch dieses strukturierte Vorgehen werden Fehler frühzeitig erkannt und können bereits während der Implementierungsphase behoben werden. Gleichzeitig wird die Qualität des Quellcodes sichergestellt und der Entwicklungsprozess bleibt jederzeit transparent und nachvollziehbar.

## SMART-Ziele

Um zu überprüfen, ob die Entwicklung des Moduls den Anforderungen an ein Abschlussprojekt im Rahmen der Umschulung zum Fachinformatiker für Anwendungsentwicklung gerecht wird, wurden die Projektziele anhand der SMART-Kriterien definiert

|  |  |
| --- | --- |
| Kriterium | Zielbeschreibung |
| **S**pezifisch | Es soll ein Odoo-Modul entwickelt werden, mit dem Benutzer Dokumente im DMS gezielt nach einem Datumsbereich und einer Kategorie filtern, in einem Record speichern und als ZIP-Datei herunterladen können. |
| **M**essbar | |  | | --- | | Die Umsetzung gilt als erfolgreich, wenn Benutzer mindestens 3 verschiedene Kategorien auswählen, einen Zeitraum eingrenzen, Dokumente in einem Record speichern und diese als ZIP-Datei herunterladen können. | |
| **E**rreichbar | Das Ziel ist mit den vorhandenen Ressourcen (Odoo-Instanz, Entwicklungsumgebung, interner Reviewprozess) innerhalb der verfügbaren 80 Projektstunden umsetzbar. |
| **R**elevant | Das Projekt löst ein reales Problem im Arbeitsalltag: Es reduziert den manuellen Aufwand beim Dokumentendownload und erhöht die Benutzerfreundlichkeit des DMS. |
| **Z**eitgebunden | |  | | --- | | Das Modul wird innerhalb des vorgegebenen Projektzeitraums von 70–80 Stunden fertiggestellt und bis zum Projektende vollständig funktionsfähig in das bestehende System integriert. | |

# Analysephase

Um zu ermitteln, was sich die Fachabteilung wünscht und wie der derzeitige Stand ist, wurde eine Analyse durchgeführt. Neben der Ermittlung der Anwendungsfälle wurde außerdem eine wirtschaftliche Betrachtung des Projektes durchgeführt.

## Ist-Analyse

Im aktuellen Zustand werden im DMS sämtliche Dateien zentral im Modell ir.attachment gespeichert. Dieses Modell dient als generische Ablage für alle Arten von Dokumenten, beispielsweise Rechnungen, Lieferscheine oder Excel-Dateien. Eine gezielte Filterung nach Kriterien wie Datum oder Kategorie ist im Standard nicht möglich. Die Suche und Auswahl von Dokumenten muss manuell über die Benutzeroberfläche erfolgen.

Ein weiterer Nachteil besteht darin, dass mehrere Dateien nur einzeln heruntergeladen werden können. Eine Funktion zum Sammel-Download mehrerer Dokumente, etwa durch eine ZIP-Komprimierung, ist nicht vorhanden. Gerade bei größeren Datenmengen stellt dies für den Benutzer einen hohen manuellen Aufwand dar. Zusätzlich limitieren moderne Webbrowser aus Sicherheitsgründen die Anzahl gleichzeitiger Downloads, was den Prozess weiter verlangsamt.

Die abgelegten Dokumente enthalten teilweise geschäftsrelevante und buchhaltungsbezogene Informationen. Eine strukturierte Verwaltung oder Trennung nach Dokumententypen ist aktuell nicht gegeben. Auch eine Möglichkeit, ausgewählte Dokumente temporär zu speichern, um sie später erneut herunterzuladen, existiert nicht.

Insgesamt ist die aktuelle Lösung funktional zwar ausreichend für die einfache Dokumentenablage, bietet jedoch keine effizienten Werkzeuge zur gezielten Auswahl, Verwaltung und strukturierten Bereitstellung von Dokumenten.

## Wirtschaftlichkeitsanalyse

Aufgrund des geschilderten Sachverhalts der in Abschnitt [**1.4 Projektbegründung**](#_Projektbegründung) und in Abschnitt [**3.1 Ist-Analyse**](#_Ist-Analyse) beschrieben wurde, ist die Umsetzung des Projekts erforderlich. Ob die Realisierung und die damit verbundenen wirtschaftlichen Aufwendungen gerechtfertigt sind, soll in den folgenden Abschnitten betrachtet werden.

### „Make or Buy“-Entscheidung

Im Zuge der Projektplanung wurde geprüft, ob eine bestehende Lösung zur Erweiterung des DMS-Moduls verfügbar ist. Dabei wurde ein Modul im offiziellen Odoo-Appstore identifiziert, das den Download mehrerer Dokumente ermöglicht und käuflich erworben werden kann.

Nach genauer Analyse wurde jedoch festgestellt, dass das Modul die spezifischen Anforderungen des Kunden nicht vollständig erfüllt.

Darüber hinaus ist der Einsatz des Drittanbieter-Moduls mit zusätzlichen Lizenzkosten verbunden, ohne dass eine vollständige Anpassung an das bestehende System garantiert werden kann.

Aus diesen Gründen wurde entschieden, das benötigte Modul eigenständig zu entwickeln, um eine maßgeschneiderte Lösung bereitzustellen, die exakt auf die Anforderungen und Prozesse abgestimmt ist.

### Projektkosten

Im Folgenden werden die während der Projektlaufzeit entstehenden Kosten kalkuliert. Grundlage der Kalkulation sind interne Stundensätze, die pauschal pro Rolle angesetzt wurden. Neben den Personalkosten für Entwicklung und Code-Review wurden keine zusätzlichen Kosten für Ressourcen wie Software oder Hardware angesetzt, da ausschließlich firmeneigene Infrastruktur und bereits lizenzierte Softwarekomponenten verwendet wurden (siehe Abschnitt [**2.2 Ressourcenplanung**](#_Ressourcenplanung)).

Das Modul wurde von zwei Mitarbeitern umgesetzt, wobei einer der beiden zusätzlich die Qualitätssicherung durch Code-Reviews übernahm. Für die Kalkulation wurde ein Stundensatz von 10 € für den Auszubildenden (Entwicklung) und 75 € für einen festangestellten Mitarbeiter (Code-Review) angesetzt.

Es ergeben sich aus den Projektvorgängen und dem Zeitrahmen von 80h für die Entwicklung die in Tabelle 2 dargestellten Projektkosten von 1.150€.

|  |  |  |  |
| --- | --- | --- | --- |
| **Vorgang** | **Zeit** | **Kosten pro Stunde** | **Kosten** |
| Entwicklungskosten | 80 h | 10 € | 800 € |
| Code Review | 10 h | 75 € | 750 € |
| **Gesamt** | 90 h | // | 1.150 € |

Tabelle 2: Kostenaufstellung

### Amortisationsdauer

**Berechnung der Amortisationsdauer**

Die Gesamtkosten für die Entwicklung und das Code-Review des Moduls belaufen sich auf 1.150 €, wie in Tabelle 2 dargestellt. Da der Kunde, die Wolfgang Matthiessen GmbH & Co. KG, für die Nutzung des Systems monatlich 4.500 € zahlt, ergibt sich eine Amortisationsdauer von ca. 0,26 Monaten. Das bedeutet, dass sich die Investitionskosten für das Modul bereits innerhalb von etwa 8 Tagen vollständig amortisiert haben.

Aufgrund dieser sehr kurzen Amortisationszeit kann das Projekt aus wirtschaftlicher Sicht als äußerst effizient bewertet werden. Die schnelle Kostendeckung sorgt dafür, dass das Unternehmen bereits kurz nach Abschluss der Entwicklung einen nachhaltigen Mehrwert generiert.

## Nicht-monetäre Aspekte

Durch die gezielte Filterung von Dokumenten nach Datum und Kategorie wird der bisher manuelle und zeitaufwändige Suchprozess deutlich vereinfacht. Benutzer können relevante Dokumente nun schneller auffinden und strukturierter verarbeiten.

Die Möglichkeit, gefilterte Dokumente in einem temporären Record zu speichern und als ZIP-Datei herunterzuladen, erhöht zusätzlich die Effizienz und reduziert das Fehlerrisiko bei der Dateiauswahl. Dadurch wird die tägliche Arbeit nicht nur beschleunigt, sondern auch benutzerfreundlicher gestaltet.

Zudem trägt das Modul zu einer höheren Systemintegration bei, da es sich nahtlos in das bestehende Odoo-System einfügt und auf bereits vorhandene Strukturen wie das ir.attachment-Modell aufbaut. Eine zusätzliche Einarbeitung oder externe Tools sind nicht erforderlich.

Nicht zuletzt steigert das Modul auch die Zufriedenheit der Anwender, da es deren Bedürfnisse gezielt adressiert und die tägliche Arbeit erleichtert. Dies kann langfristig zu einer höheren Akzeptanz und effizienteren Nutzung des DMS-Systems führen.

## Anwendungsfälle

Zur Erfassung der fachlichen Anwendungsfälle und zur Veranschaulichung des Nutzerablaufs wurde im Rahmen der Analyse ein Ablaufdiagramm erstellt. Es zeigt die einzelnen Schritte vom Start des Prozesses bis zum finalen Download der gefilterten Dokumente. Das vollständige Diagramm befindet sich im Anhang [A.3: Ablaufdiagramm](#_Anhang).

## Lastenheft

// Kubillus fragen, ob externer Person für das Lastenheft

Das Lastenheft wurde nicht in direkter Abstimmung mit dem Kunden erstellt, sondern durch einen internen Wirtschaftsmitarbeiter, der als Schnittstelle zum Kunden fungierte. Dieser war für die Kommunikation der Anforderungen mit der Wolfgang Matthiessen GmbH & Co. KG verantwortlich und dokumentierte die gewünschten Funktionalitäten in Form von Issues innerhalb des GitLab-Systems.

Die dort festgehaltenen Anforderungen bildeten die Grundlage für die weitere Planung und Umsetzung des Projekts. Sie umfassten unter anderem die Möglichkeit, Dokumente im DMS nach Datum und Kategorie zu filtern, gefilterte Ergebnisse als Record zu speichern sowie mehrere Dateien als ZIP-Archiv herunterzuladen.

Die technische und fachliche Ausarbeitung erfolgte anschließend auf Basis dieser Informationen im Rahmen des 0.0: Pflichtenhefts !LINK HINZUFÜGEN!.

# Entwurfsphase

## Maßnahmen zur Qualitätssicherung

Zur Sicherstellung der Qualität des entwickelten Moduls wurden verschiedene Maßnahmen und Prüfprozesse durchgeführt.

Bereits während der Entwicklung wurden für zentrale Funktionen Unit-Tests erstellt. Diese Tests wurden lokal ausgeführt, um die Funktionsfähigkeit einzelner Komponenten frühzeitig zu validieren. Nach Abschluss der Entwicklung wurden die Tests erneut ausgeführt, um sicherzustellen, dass alle Funktionen wie erwartet arbeiten.

Bevor ein Feature in den Hauptzweig (Master) übernommen werden konnte, wurde für jedes GitLab-Issue ein Code-Review durch einen Entwickler durchgeführt. Dabei wurde der Quellcode geprüft und ggf. mit Kommentaren versehen, falls Verbesserungsbedarf bestand. Wenn dies erfolgreich geprüft wurde, dann musste es einen mehrstufigen Qualitätssicherungsprozess auf GitLab durchlaufen. Dieser bestand aus folgenden automatisierten Tests:

* **Deploy Test:** Prüfung, ob das Modul erfolgreich gebaut und gestartet werden kann
* **Translate Test:** Kontrolle auf korrekte Übersetzungsdateien und -strukturen
* **Lint Test:** Prüfung des Quellcodes auf Einhaltung von Syntax- und Stilvorgaben
* **Manual Action Test:** Ausführung definierter manueller Testschritte

Erst nach erfolgreicher Prüfung und Bestehen aller Tests konnte der Merge über den unternehmensinternen Merge-Bot erfolgen.

Dieser strukturierte Ablauf ermöglichte es, Fehler frühzeitig zu erkennen, die Codequalität konstant hochzuhalten und gleichzeitig einen kontrollierten Übergang der Features in das produktive System sicherzustellen.

# Kosten/Nutzen-Analyse

# Risiko-Berechnung

# SMART-Ziele

# SWOT-Analyse

# Durchführung

# Qualitätssicherung

# Soll-Ist-Vergleich

# Fazit

# Glossar

# Anhang