Glossar

Übersicht

Projekt: Projekt Episko

Inkrement: 0
Autor: Simon Blum
Datum: 11.11.2024
Zuletzt geändert:
von: Max Rodler
am: 02.04.2025
Version: 2
Prüfer: Ben Oeckl
Letzte Freigabe:
durch: Ben Oeckl
am: 02.04.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
11.11.2024	Simon Blum	Initiales Erstellen und Verfassen
02.04.2025	Max Rodler	Hinzufügen weiterer Begriffe

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Begriff	Erklärung
Metadaten	Repräsentierung von Stammdaten eines Projektes im Zuge der Software. Eine genaue Definierung welche Daten hier beinhaltet sind erfolgt im Laufe des ersten Inkrements.
Projekt	Softwareprojekt des Nutzers welches durch Metadaten beschrieben werden kann.
Manifest (Datei)	Repräsentierung von Metadaten in einer Datei. Die genaue Struktur wird im Laufe des ersten Inkrements festgelegt.
Inkrement	Zeitraum in dem festgelegte Arbeitspakete bearbeitet werden. Abschluss dieser kann auch als Meilenstein bezeichnet werden.

Begriff	Erklärung
Meilenstein	Abschluss eines Inkrementes.
Kritikalität	Im Rahmen des Projektes kann die Kritikalität verschiedener ???? folgendermaßen angegeben werden: 0 - Absolut unabdingbar, 1 - Sehr wichtig, 2 - Normal 3 - Unwichtig
Serialisieren	Umwandlung der menschenlesbaren Manifestdatei (toml) in speicherbare / nutzbare Daten.
Deserialisieren	Umwandlung gespeicherter Daten in menschenlesbare Manifestdatei (toml).
Gecachete Date	Metadaten welche in einer Datenbank zwischengespeichert sind um das wiederholte deserialiseren von Manifest-Dateien zu vermeiden.
Lokale Daten	Metadaten welche sich in Manifest-Dateien befinden und nicht gecached sind.
Cache	Der Begriff Cache deutet im Rahmen des Projektes auf einen persistenten Cache in Form einer Datenbank hin. Diese ist nur ein “Cache” da die Manifeste die “single source of thruth” sind.
CI/CD Pipeline	Prozess zum automatischen Testen beim Deployment.
Use-Cases	Anwendungsfälle die durch die Software abgedeckt werden sollen.
Requirements	Funktionale und Nichtfunktionale Anforderungen, die die Software erfüllen muss.
Designpaper	Dokument zur Beschreibung bestimmter Elemente der Anwendung.
Config	Dateien welche, für den Programmablauf notwendige, Configurationen beinhalten.
Linting	Statische Codeanalyse, die Programmierfehler aufzeigt.
Rust-Crates	Eine, für den Rust-Compiler zusammengefasste Kompilierungseinheit, verwaltet durch den Cargo-Paketmanager.

Projekt Episko

Eine besser strukturierte Version dieses Dokumentes kann auf dieser Seite gefunden werden.

Die Markdown Version aller Dokumente kann in diesem Github Repository gefunden werden.

Initialisierung Inkrement 4

Übersicht

Projekt: Projekt Episko
Inkrement: 4
Autor: Max Rodler
Datum: 22.11.2024
Zuletzt geändert:
von: Max Rodler
am: 06.12.2024
Version: 2
Prüfer: Ben Oeckl
Letzte Freigabe:
durch: Ben Oeckl
am: 06.12.2024

Changelog

Datum	Verfasser	Kurzbeschreibung
22.11.2024	Max Rodler	Initiales Erstellen und Verfassen
06.12.2024	Max Rodler	Ergänzen des Reviews

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Zeitplan

- Beginn: 22.11.2024
- Ende: 06.12.2024

Arbeitspakete

Die folgenden Arbeitspakete werden als Teil des Inkrements bearbeitet: ### A4.1 Klassendiagramm erstellen - Klassendiagramm zur Strukturierung des Programms planen. (Dient als Vorbereitung zur späteren Umsetzung)

Verantwortlich: Simon Blum
Beauftragte: Simon Blum, Ben Oeckl, Paul Stöckle, Max Rodler

A3.2 Sequenzdiagramm erstellen

- Sequenzdiagramme zur Veranschaulichung von Abläufen.

Verantwortlich: Simon Blum
Beauftragte: Simon Blum

A3.3 Tracing erstellen

- Anforderungstracing ausarbeiten.

Verantwortlich: Ben Oeckl
Beauftragte: Ben Oeckl, Paul Stöckle

4.4 Bessere Strukturierung von Inkrementen

- Strukturierung der Inkremente überarbeiten (Entwicklungsplan beachten)

Verantwortlich: Max Rodler
Beauftragte: Max Rodler

Besonderheiten

- Da dieses Inkrement noch vorbereitende Arbeitspakete enthält, existieren hierzu noch nicht alle notwendigen Dokumente. (Der Anforderungskatalog, das Designpaper, die Entwicklerdokumentation und ein ausführlicher Abschlussreport werden aufgrund mangelnder Notwendigkeit / Sinnhaftigkeit weggelassen.)
- Das Review erfolgt in diesem Inkrement im Rahmen der Vorstellung des Grobdesigns.

Review-Ergebnis

- Das Grobdesign ist abgenommen. Somit sind alle vorbereitenden Maßnahmen abgeschlossenen und es kann mit der Implementierung der Vorbereitungen begonnen werden.

Grobdesign

Übersicht

Projekt: Projekt Episko
Inkrement: 4
Autor: Simon Blum, Ben Oeckl, Paul Stöckle
Datum: 05.12.2024
Zuletzt geändert:
von: Simon Blum
am: 24.01.2025
Version: 2
Prüfer: Max Rodler
Letzte Freigabe:
durch: Max Rodler
am: 24.01.2025

Changelog

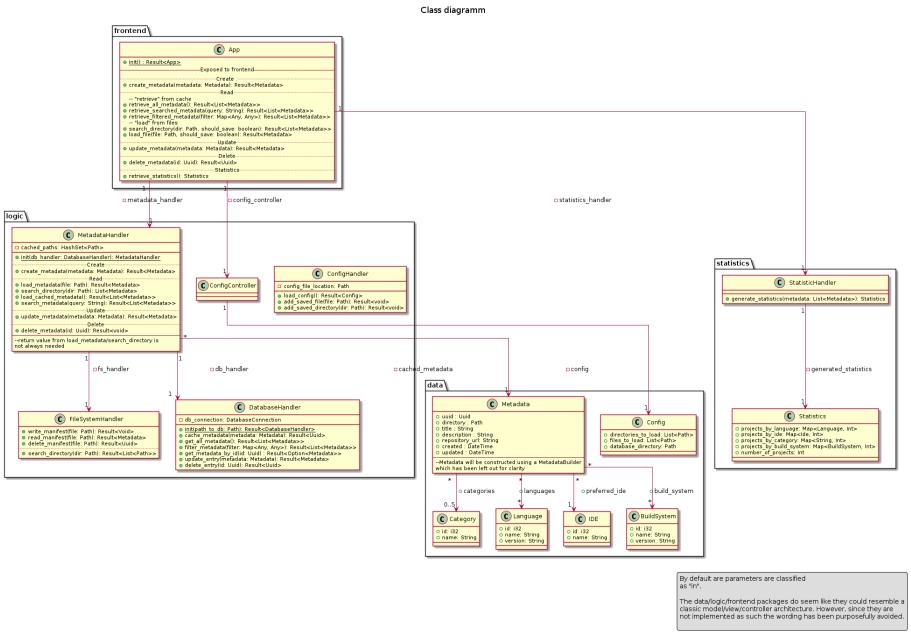
Datum	Verfasser	Kurzbeschreibung
05.12.2024	Simon Blum	Initiales Erstellen und Verfassen
24.01.2025	Simon Blum	Hinzufügen von Paketen/Modulen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Klassendiagramm

Anmerkung:
Während das Klassendiagramm als Grundlage zur Implementierung sicherlich eine Stütze stellen kann, sollte trotzdem beachtet werden, dass Rust als Sprache sich nur begrenzt zur Objektorientierung anbietet und einiges durch leichte Veränderung idiomatischer, sauberer und effizienter implementiert werden kann und sollte. Hierfür sollen bei auftretenden Fällen, entsprechende Anmerkungen im Klassendiagramm angefügt werden.



Sequenzdiagramme

Die Sequenzdiagramme basieren auf den Use-Cases und sind dementsprechend aufgeteilt.

U1.1

U1.2

U1.3

U1.4

U2.1

U2.2

U3.1

U3.2

U3.3

Anforderungstracing

Struktur

Die Anforderungsverfolgung ist aktuell nach folgender Struktur aufgebaut:

[Use Case] -> [Anforderung] -> [Klassenattribut]

In Zukunft soll diese noch in ein passendes Diagramm überführt werden.

Verfolgung

UC1.1 -> FA1.1.1 -> App.retrieve_all_metadata()

UC1.1 -> FA1.1.2 -> FileSystemHandler.read_manifest()

UC1.2 -> FA1.2.1 -> App.create_metadata(), MetadataHandler.create_metadata(), DatabaseHandler.update_entry() FileSystemHandler.write_manifest()

UC1.2 -> FA1.2.2 -> FileSystemHandler.write_manifest()

UC1.2 -> FA1.2.3 -> DatabaseHandler.cache_metadata()

UC1.3 -> FA1.3.1 -> App.update_metadata()

UC1.3 -> FA1.3.2 -> MetadataHandler.update_metadata(), DatabaseHandler.update_entry()

UC1.3 -> FA1.3.3 -> MetadataHandler.load_metadata(), MetadataHandler.search_directory(), FileSystemHandler.read_manifest()

UC1.4 -> FA1.4.1 -> App.delete_metadata, MetadataHandler.delete_metadata, FileSystemHandler.delete_manifest()

UC1.4 -> FA1.4.2 -> DatabaseHandler.delete_entry(),

UC1.4 -> FA1.4.3 -> metadata_handler.load_metadata(), MetadataHandler.search_directory(), MetadataHandler.delete_metadata, DatabaseHandler.delete_entry()

UC2.1 -> FA2.1.1 -> App.loadFile, ConfigController.add_saved_file()

UC2.1 -> FA2.1.2 -> Siehe FA1.1.2

UC2.1 -> FA2.1.3 -> siehe FA1.2.3

UC2.1 -> FA2.1.4 -> App.search_directory(), ConfigController.add_saved_directory()

UC2.1 -> FA2.1.5 -> MetadataHandler.search_directory(), FileSystemHandler.search_directory()

UC2.2 -> FA2.2.1 -> Siehe FA2.1.4

UC2.2 -> FA2.2.2 -> Siehe FA2.1.5

UC2.2 -> FA2.2.3 -> MetadataHandler.load_metadata(), MetadataHandler.update_metadata(), DatabaseHandler.update_entry()

UC2.2 -> FA2.2.4 -> Siehe FA2.1.5

UC2.2 -> FA2.2.5 -> Siehe FA2.1.5

UC3.1 -> FA3.1.1 -> MetadataHandler.search_metadata()

UC3.1 -> FA3.1.2 -> App.retrieve_searched_metadata() - Input der Methode stellt Nutzereingabe da

UC3.1 -> FA3.1.3 -> App.retrieve_searched_metadata() - Output Methode wird nutzer angezeigt

UC3.2 -> FA3.2.1 -> MetadataHandler.filter_metadata()

UC3.2 -> FA3.2.2 -> App.retrieve_filtered_metadata() - Input der Methode stellt Nutzereingabe da

UC3.2 -> FA3.2.3 -> App.retrieve_filtered_metadata() - Output Methode wird nutzer angezeigt

UC3.3 -> FA3.3.1 -> StatisticsController.generate_statistics()

UC3.3 -> FA3.3.2 -> StatisticsController.retrieve_statistics()

Aufwandsschätzung

Übersicht

Projekt: Projekt Episko
Inkrement: 5
Autor: Max Rodler
Datum: 10.01.2025
Zuletzt geändert:
von: Paul Stöckle
am: 17.02.2025
Version: 3
Prüfer: Simon Blum
Letzte Freigabe:
durch: Simon Blum
am: 17.02.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
10.01.2025	Max Rodler	Initiales Erstellen und Verfassen
24.01.2025	Paul Stöckle	Faktor und Stunden hinzugefügt
17.02.2025	Paul Stöckle	Function Points angepasst

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Function Point Analyse

Ungewichtete Function Points

Teilaufgabe	Eingabe 3-4-6	Abfrage 3-4-6	Ausgabe 4-5-7	Datenbestand 7-10-15	Referenzdaten 5-7-10	Summe
Anwendung starten		6	5			11
Metadaten anlegen	6			10		16

Teilaufgabe	Eingabe 3-4-6	Abfrage 3-4-6	Ausgabe 4-5-7	Datenbestand 7-10-15	Referenzdaten 5-7-10	Summe
Metadaten bearbeiten	6	4	5	10		25
Metadaten löschen				7		7
Datei angeben	3			5		8
Verzeichnis angeben	3		5		7	15
Datei einlesen				10	7	17
Projekte suchen	3	4				7
Projekte filtern	6	4				10
Ergebnisse anzeigen (Suche+Filter)			5			5
Statistiken		6	7			13

Summe: 134

Gewichtete Function-Points

Summe ungewichteter Function-Points	134	individueller Kommentar
Verflechtung mit anderen Anwendungssystemen (0-5)	1	Dateisystem
Dezentrale Daten, dezentrale Verarbeitung (0-5)	0	Rein lokal
Transaktionsrate (0-5)	1	Reaktion auf Nutzerinteraktion
Rechenoperationen (0-10)	2	Einfache Statistiken
Kokntrollverfahren (0-5)	2	Prüfung von Checksums und Dopplungen
Ausnahmeregelungen (0-10)	3	Formprüfung
Logik (0-5)	1	Suchalgorithmen
Wiederverwendbarkeit (0-5)	3	Interne Wiederverwendung teilweise möglich
Datenbestandskonvertierung (0-5)	2	Manifest <-> Datenbankinhalt
Anpassbarkeit (0-5)	1	Einmalige Entwicklung (wenige optionale modulare Erweiterungen)
Summe der Einflussfaktoren (FE)	16	
Faktor der Einflussbewertung (FE)	0,86	

Summe ungewichteter Function-Points	134	individueller Kommentar
Gewichtete Funktion-Points	115,24	

Stunden pro Function-Point: 2
Gesamtstunden: 230

Initialisierung Inkrement 5

Übersicht

Projekt: Projekt Episko
Inkrement: 5
Autor: Max Rodler
Datum: 10.01.2024
Zuletzt geändert:
von: Max Rodler
am: 10.01.2025
Version: 1
Prüfer: Ben Oeckl
Letzte Freigabe:
durch: Ben Oeckl
am: 10.01.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
10.01.2025	Max Rodler	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Zeitplan

- Beginn: 10.01.2025
- Ende: 20.01.2025

Arbeitspakete

Die folgenden Arbeitspakete werden als Teil des Inkrements bearbeitet: ### A5.1 Aufwandsschätzung Erstellung der Aufwandsschätzung zur Zeiteinteilung

Verantwortlich: Paul Stöckle
Beauftragte: Simon Blum, Ben Oeckl, Paul Stöckle, Max Rodler

Besonderheiten

- Da dieses Inkrement noch vorbereitende Arbeitspakete enthält, existieren hierzu noch nicht alle notwendigen Dokumente. (Der Anforderungskatalog, das Designpaper, die Entwicklerdokumentation und ein ausführlicher Abschlussreport werden aufgrund mangelnder Notwendigkeit / Sinnhaftigkeit weggelassen.)
- Das Review erfolgt in diesem Inkrement im Rahmen der Vorstellung des Grobdesigns.

Review-Ergebnis

- Die Aufwandsschätzung ist abgenommen. Sie muss vermutlich im Laufe der Zeit angepasst werden.

Review Inkrement 7

Übersicht

Projekt: Projekt Episko
Inkrement: 7
Autor: Maximilian Rodler
Datum: 13.02.2025
Zuletzt geändert:
von: Maximilian Rodler
am: 03.03.2025
Version: 2
Prüfer: Paul Stöckle
Letzte Freigabe:
durch: Paul Stöckle
am: 06.03.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
13.02.2025	Maximilian Rodler	Initiales Erstellen und Verfassen
03.03.2025	Maximilian Rodler	Formatierungsanpassungen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Bewertung Arbeitspakete

Bewertung Arbeitspaket 6.3

Abgeschlossen

Bewertung Arbeitspaket 6.2

Abgeschlossen

Bewertung Arbeitspaket 7.1

Abgeschlossen. Tests stehen aus.

Ergebnis

Alle Arbeitspakete aus Inkrement 6 und 7 sind jetzt abgeschlossen.

Initialisierung Inkrement 7

Übersicht

Projekt: Projekt Episko
Inkrement: 7
Autor: Max Rodler
Datum: 27.01.2025
Zuletzt geändert:
von: Maximilian Rodler
am: 29.01.2025
Version: 1
Prüfer: Ben Oeckl
Letzte Freigabe:
durch: Ben Oeckl
am: 30.01.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
29.01.2025	Max Rodler	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Zeitplan

- Beginn: 27.01.2025
- Ende: 13.02.2025

Arbeitspakete

Die folgenden Arbeitspakete werden als Teil des Inkrements bearbeitet:

> Arbeitspaket 6.2 und 6.3 werden weiterhin bearbeitet

A7.1 Datenbank Modul für Library

Verantwortlich: Simon Blum
Beauftragte: Simon Blum, Paul Stöckle

Relevante UseCases/Requirements FA1.2.3, FA1.3.2, FA1.3.3, FA1.4.2, FA1.4.3, FA2.1.3, FA2.2.3

Anforderungsbewertung 7.1

Übersicht

Projekt: Projekt Episko
Inkrement: 7
Arbeitspaket: 1
Autor: Simon Blum
Datum: 04.02.2025
Zuletzt geändert:
von: Simon Blum
am: 07.02.2025
Version: 1
Prüfer: Max Rodler
Letzte Freigabe:
durch: Max Rodler
am: 07.02.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
07.02.2025	Simon Blum	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Bewertung relevanter Requirements und UseCases

Die in der Initialisierung des Projektes genannten Anforderungen sind ungeändert relevant in diesem Arbeitspaket.

Hierbei muss angemerkt werden, dass diese nicht zu 100% erfüllt sein werden, vielmehr soll durch das Arbeitspaket eine Basis geschaffen werden, um in Zukunft eine simple Integration der Datenbank inform eines Caches zu ermöglichenm.

Designpaper 7.1

Übersicht

Projekt: Projekt Episko
Inkrement: 7
Arbeitspaket: 1
Autor: Simon Blum
Datum: 04.02.2025
Zuletzt geändert:
von: Simon Blum
am: 04.02.2025
Version: 1
Prüfer: Max Rodler
Letzte Freigabe:
durch: Max Rodler
am: 07.02.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
04.02.2025	Simon Blum	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Designpaper

Im Rahmen des Arbeitspaket soll ein neues Modul in der `episko_lib` geschaffen werden. Anhand diesem soll ein einfacher Zugriff auf die Datenbank in den Frontend Anwendung ermöglicht werden.

Struktur

In der bestehenden Struktur gibt es insbesondere 2 Fälle zu betrachten:

Advanced Properties Die Properties wie “Category” oder “Language” sollen auch in der Datenbank gespeichert werden. Da alle diese Eigenschaften sehr ähnlich sind und über den `Property` trait auch bereits viel gemeinsames Verhalten teilen, soll auch für Datenbankoperationen eine saubere und geteilte Implementation gefunden werden. Naheliegend wäre hierbei bspw. die Verwendung eines Macros.

Metadata Das Metadaten Objekt selbst muss hierbei gesondert behandelt werden, da die entstehenden Relationen vor Interkation mit der Datenbank entsprechend etabliert/geprüft werden müssen.

Datenbank

Für die Datenbank soll `sqlite` verwendet werden. Hierzu muss ein Schema erarbeitet werden. Alle Informationen hierzu werden im Software Design Paper zu finden sein.

Anforderungsanalyse

Übersicht

Projekt: Projekt Episko
Inkrement: 3
Autor: Simon Blum
Datum: 13.11.2024
Zuletzt geändert:
von: Paul Stöckle
am: 15.11.2024
Version: 7
Prüfer: Paul Stöckle
Letzte Freigabe:
durch: Paul Stöckle
am: 07.12.2024

Changelog

Datum	Verfasser	Kurzbeschreibung
13.11.2024	Simon Blum	Initialer Meeting-Mitschrieb
13.11.2024	Ben Oeckl	Ergänzung von Requirements in Use Cases
13.11.2024	Paul Stöckle	Überarbeitung von Requirements
15.11.2024	Paul Stöckle	Hinzufügen des Headers
15.11.2024	Max Rodler	Fehlerbehebung
21.11.2024	Simon Blum	Aktualisierung von UseCases und Requirements
05.12.2024	Simon Blum	Fehlerbehebung Serialisieren -> Deserialisieren

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

UseCases

Erläuterung Pakete

Die UseCases sind in 3 Pakete aufgeteilt.

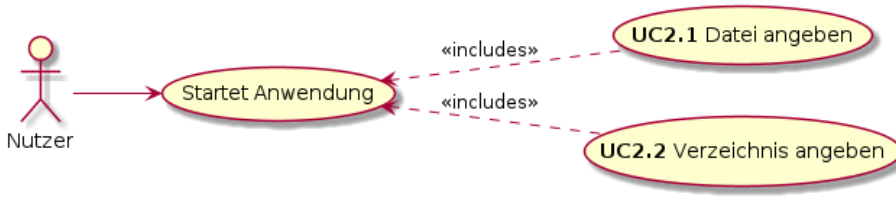
Paket 1 - Create, Read, Update, Delete Bei den UseCases in Paket 1 geht es primäre um die atomare manipulation von Daten.

Paket 2 - Manifest interaktion Bei den UseCases in Paket 2 geht es vor allem um die Interaktionen mit dem lokalen Dateisystem und sich dort befindende Manifestdateien

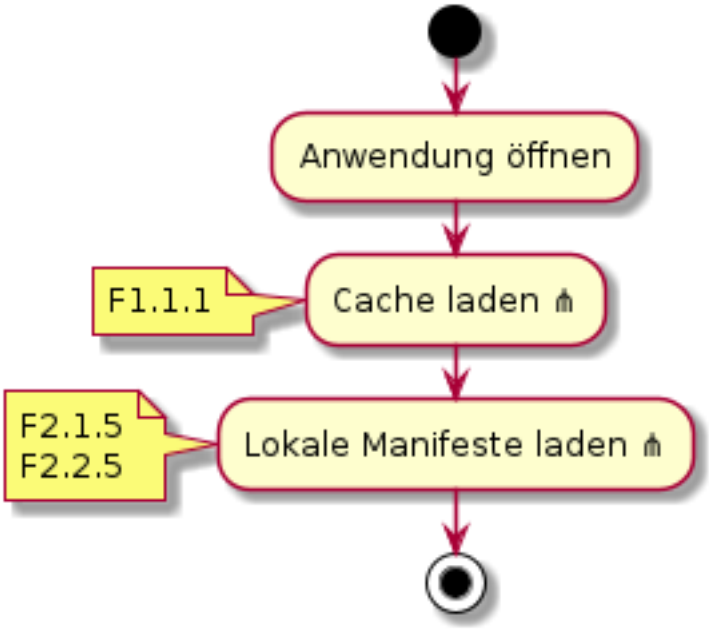
Paket 3 - Funktionalitäten Paket 3 umfasst UseCases welche erweiterte Funktionalitäten des Systems darstellen.

UC1.1 Anwendung starten

	Inkrement	
Id	1	UC1.1
Paket	2	1
Autor	1	
Version	1	4
Kurzbeschreibung	1	Der Nutzer kann die Anwendung starten
Beteiligte Akteure	1	Nutzer
Fachverantwortlicher	1	
Referenzen	2	
Vorbedingungen	2	Die Anwendung ist auf einem kompatiblen System installiert
Nachbedingungen	2	Die Anwendung ist gestartet und nutzungsbereit
Typischer Ablauf	2	s. Ablaufdiagramm
Alternative Abläufe	3	-
Kritikalität	3	0
Verknüpfungen	2	UC2.1, U2.2
Funktionale Anforderungen	4	FA1.1.1, FA1.1.2, FA2.1.5, FA2.2.5
Nicht-funktionale Anforderungen	4	NA2



UC1.1 UseCase Diagramm

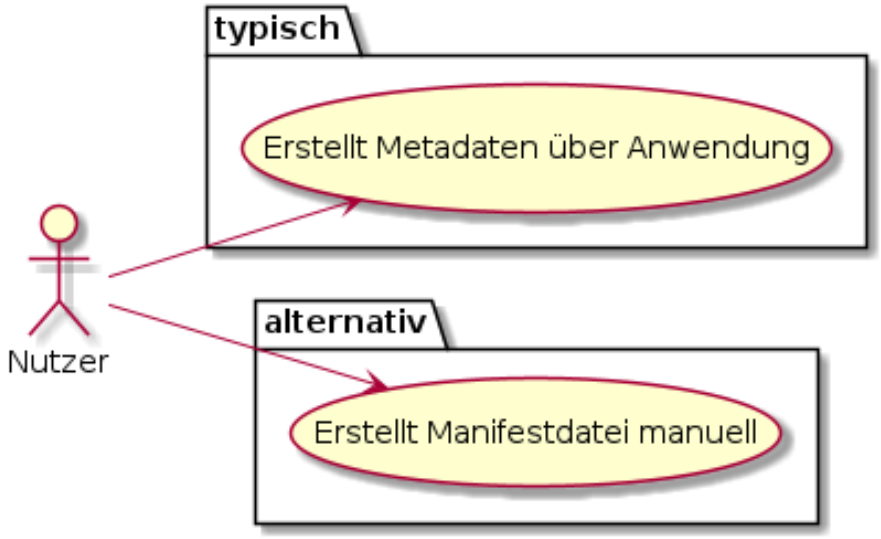


UC1.1 Ablaufdiagramm

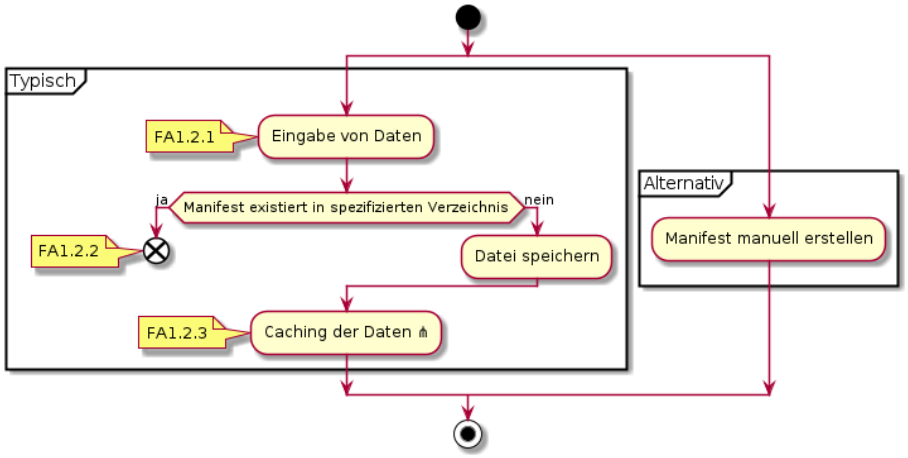
UC1.2 Metadaten anlegen

Metadaten anlegen	Inkrement	
Id	1	UC1.2
Paket	2	P1
Autor	1	
Version	1	5
Kurzbeschreibung	1	Der Nutzer kann mithilfe der Anwendung oder manuelle eine Manifestdatei mit Metadaten erstellen
Beteiligte Akteure	1	Nutzer
Fachverantwortlicher	1	
Referenzen	2	Dateiformat Doku
Vorbedingungen	2	Es muss ein Verzeichnis für das Projekt existieren , in dem sich keine andere Manifestdatei befindet
Nachbedingungen	2	Es existiert eine Manifestdatei in dem gewählten Ordner. Wurde das Projekt über die Anwendung erstellt, wurden die Daten in der Datenbank gecached.
Typischer Ablauf	2	s. Ablaufdiagramm
Alternative Abläufe	3	s. Ablaufdiagramm

Metadaten anlegen	Inkrement	
Kritikalität	3	0
Verknüpfungen	2	
Funktionale Anforderungen	4	FA1.2.1, FA1.2.2, FA1.2.3
Nichtfunktionale Anforderungen	4	



UC1.2 UseCase Diagramm

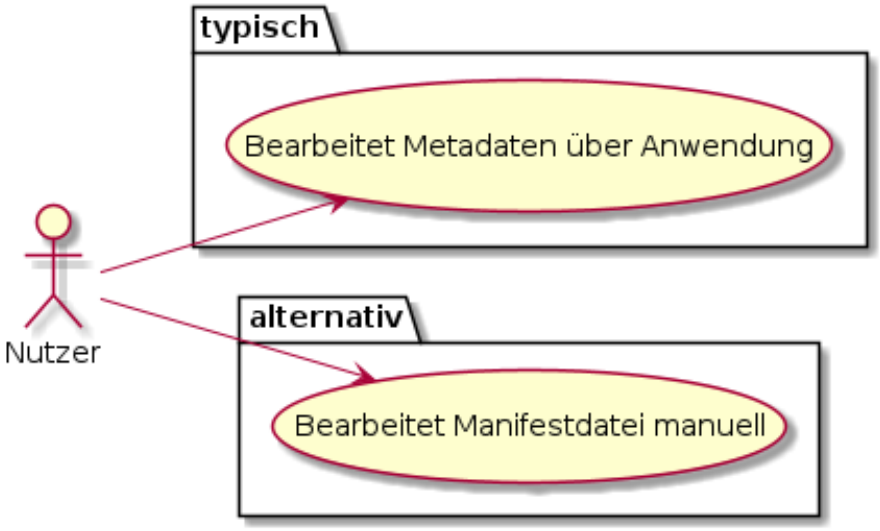


UC1.2 Ablaufdiagramme

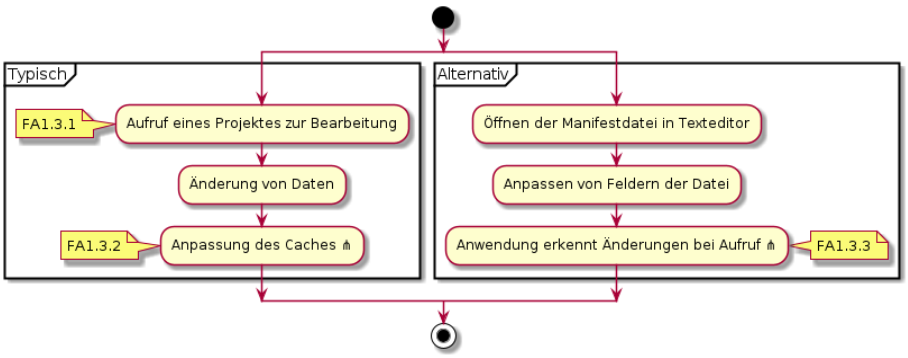
UC1.3 Metadaten bearbeiten

Schritt	Inkrement	
Id	1	UC1.3
Paket	2	P1
Autor	1	
Version	1	5
Kurzbeschreibung	1	Der Nutzer kann die Metadaten eines Projektes über die Anwendung oder manuell in der Datei bearbeiten.
Beteiligte Akteure	1	Nutzer
Fachverantwortlicher	1	
Referenzen	2	Dateiformat Doku
Vorbedingungen	2	Es muss eine Manifestdatei existieren die bearbeitet werden kann.
Nachbedingungen	2	Die angepasste Manifestdatei wird gespeichert. Wurde die Datei über die Anwendung geändert, werden die Änderungen gecached.
Typischer Ablauf	2	s. Ablaufdiagramm
Alternative Abläufe	3	s. Ablaufdiagramm
Kritikalität	3	1
Verknüpfungen	2	
Funktionale Anforderungen	4	FA1.3.1, FA1.3.2, FA1.3.3

Schritt	Inkrement
Nicht-funktionale Anforderungen	4



UC1.3 UseCase Diagramm

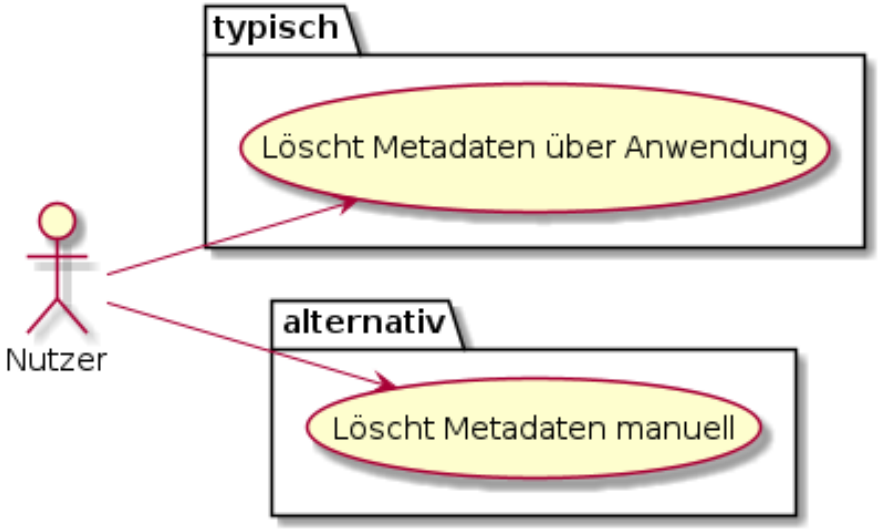


UC1.3 Ablaufdiagramme

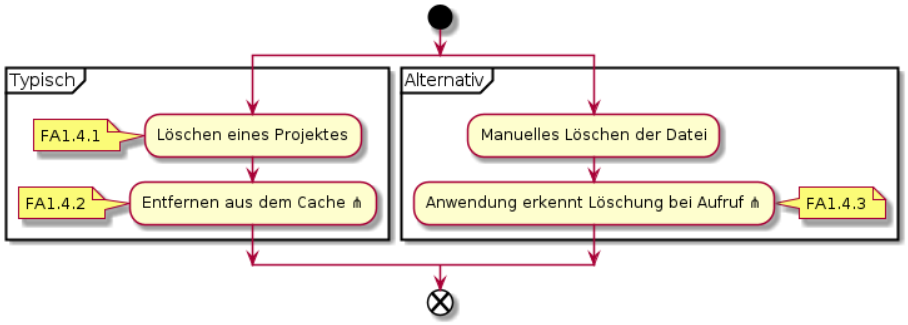
UC1.4 Metadaten löschen

Schritt	Inkrement	
Id	1	UC1.4
Paket	2	P1

Schritt	Inkrement	
Autor	1	
Version	1	5
Kurzbeschreibung	1	Der Nutzer kann die Metadaten für ein Projekt löschen
Beteiligte Akteure	1	Nutzer
Fachverantwortlicher	1	
Referenzen	2	
Vorbedingungen	2	Es existiert eine valide Manifestdatei die gelöscht werden kann
Nachbedingungen	2	Es existiert keine Manifestdatei mehr. Bei manueller Löschung wird der Cache im nachhinein, beim nächsten Starten der Anwendung aktualisiert.
Typischer Ablauf	2	s. Ablaufdiagramm
Alternative Abläufe	3	s. Ablaufdiagramm
Kritikalität	3	1
Verknüpfungen	2	Beim löschen über die Anwendung UseCase 3.1
Funktionale Anforderungen	4	FA1.4.1, FA1.4.2, FA1.4.3
Nichtfunktionale Anforderungen	4	



UC1.4 UseCase Diagramm

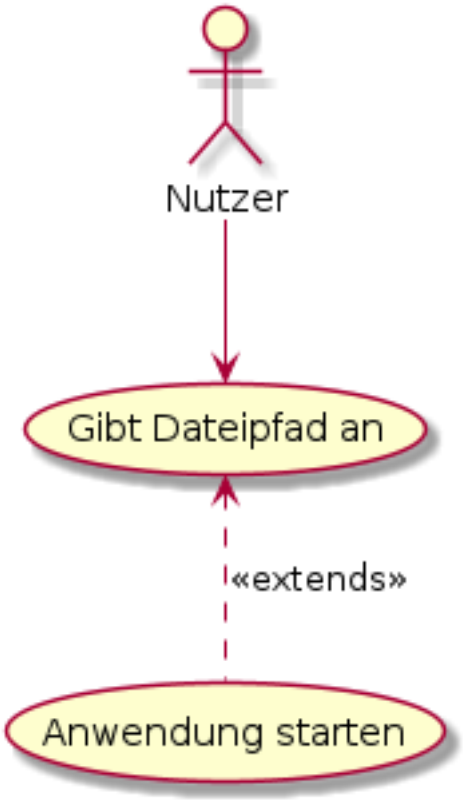


UC1.4 Ablaufdiagramme

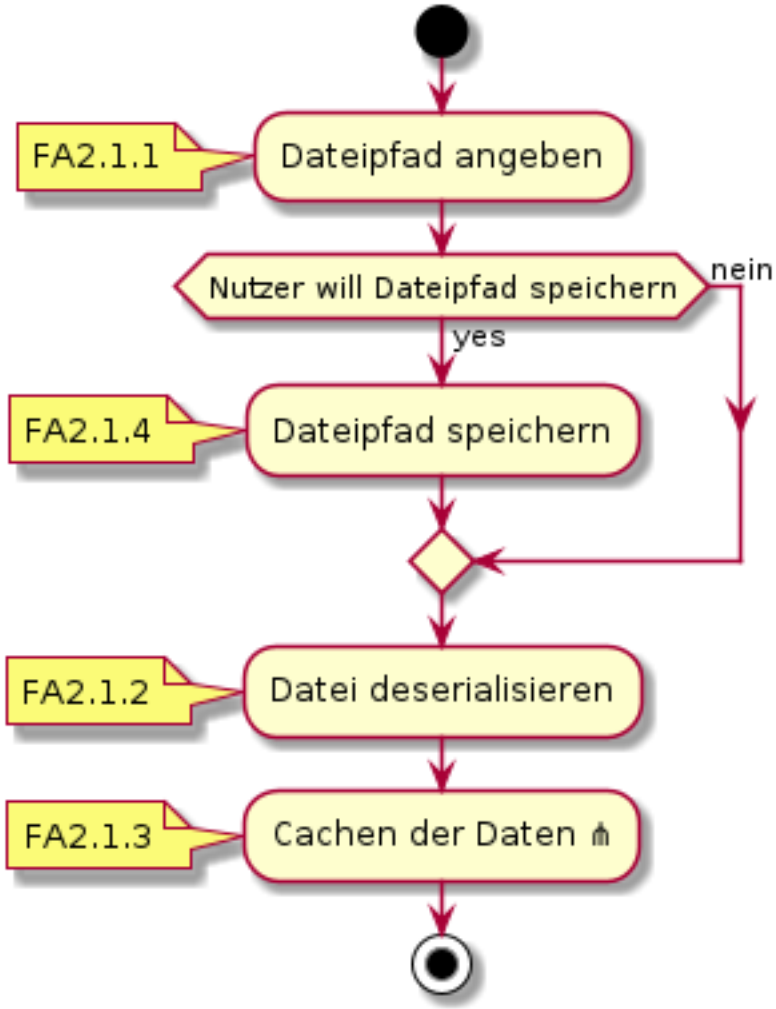
UC2.1 Datei angeben

	Inkrement	
Id	1	UC2.1
Paket	2	2
Autor	1	
Version	1	4
Kurzbeschreibung	1	Der Nutzer kann den Pfad zu einer Manifestdatei angeben, welche dann deserialisiert wird. Der Pfad der Datei kann gespeichert werden und beim nächsten Starten der Anwendung erneut deserialisiert werden.
Beteiligte Akteure	1	Nutzer

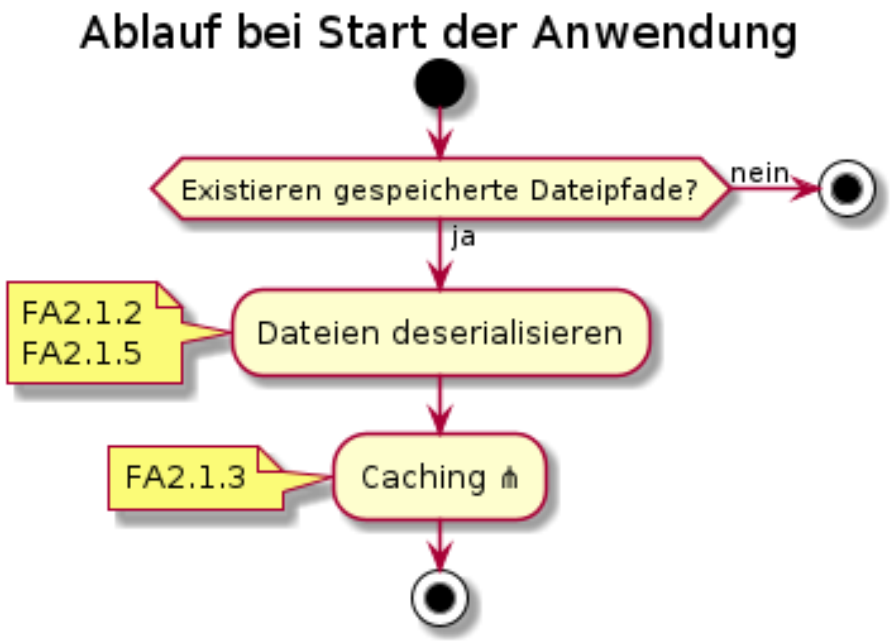
	Inkrement	
Fachverantwortlicher	1	
Referenzen	2	-
Vorbedingungen	2	Es existiert eine valide Manifestdatei welche der Nutzer angeben kann.
Nachbedingungen	2	Die Datei wurde deserialisiert und die Daten können weiterverarbeitet werden.
Typischer Ablauf	2	s. Ablaufdiagramm
Alternative Abläufe	3	-
Kritikalität	3	0
Verknüpfungen	2	U1.1
Funktionale Anforderungen	4	FA2.1.1, FA2.1.2, FA2.1.3, FA2.1.4, FA2.1.5
Nichtfunktionale Anforderungen	4	



UC2.1 UseCase Diagramm



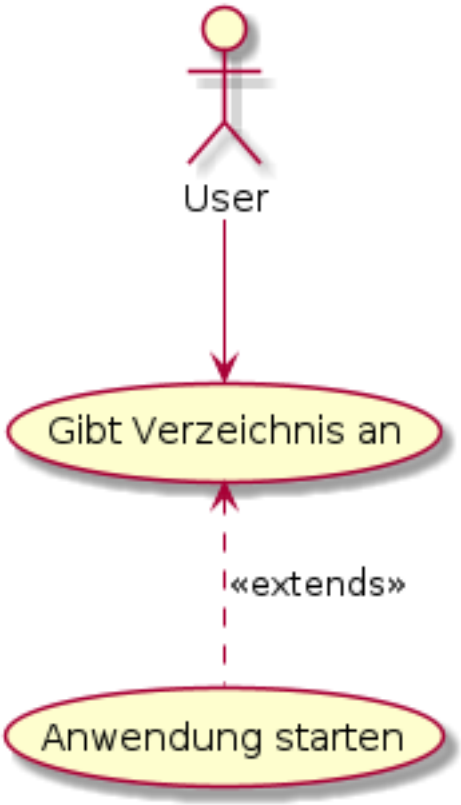
UC2.1 Ablaufdiagramm



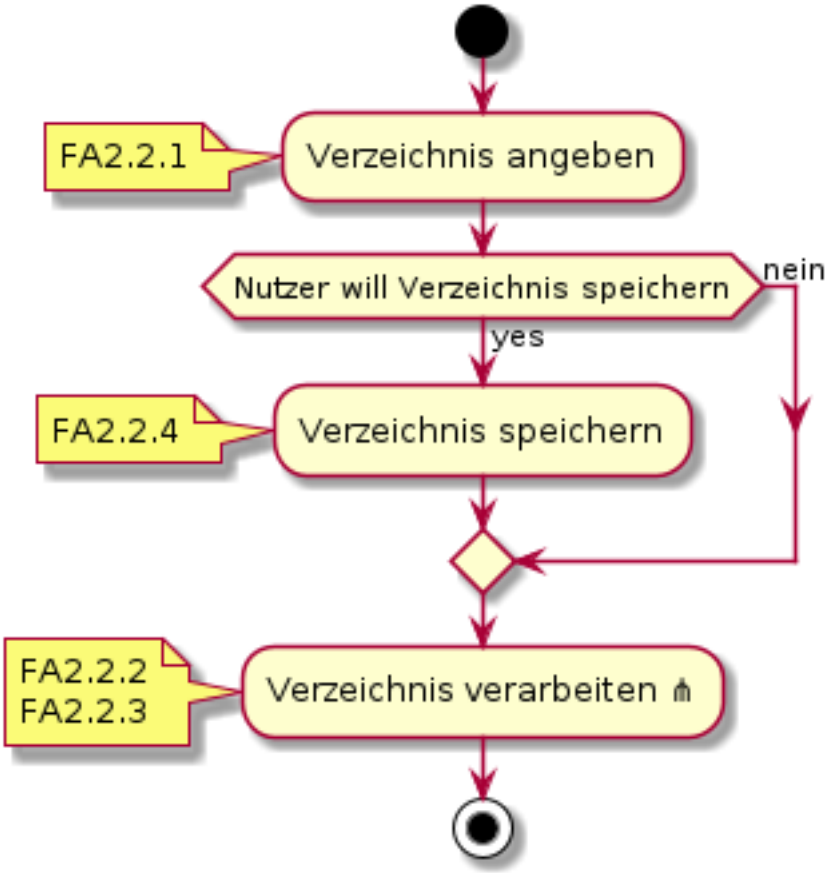
UC2.2 Verzeichnis angeben

	Inkrement	
Id	1	UC2.2
Paket	2	2
Autor	1	
Version	1	4
Kurzbeschreibung	1	Der Nutzer kann einen Pfad angeben, welcher rekursiv nach Manifesten durchsucht wird. Angegebene Pade können gespeichert werden und beim nächsten Ausführen der Anwendung wieder durchsucht werden.
Beteiligte Akteure	1	Nutzer
Fachverantwortlicher	1	
Referenzen	2	
Vorbedingungen	2	Es existiert ein Verzeichnis welches der Nutzer angeben kann.
Nachbedingungen	2	Wenn in dem Verzeichnis Manifeste liegen, wurden diese deserialisiert.
Typischer Ablauf	2	s. Ablaufdiagramm
Alternative Abläufe	3	s. Ablaufdiagramm

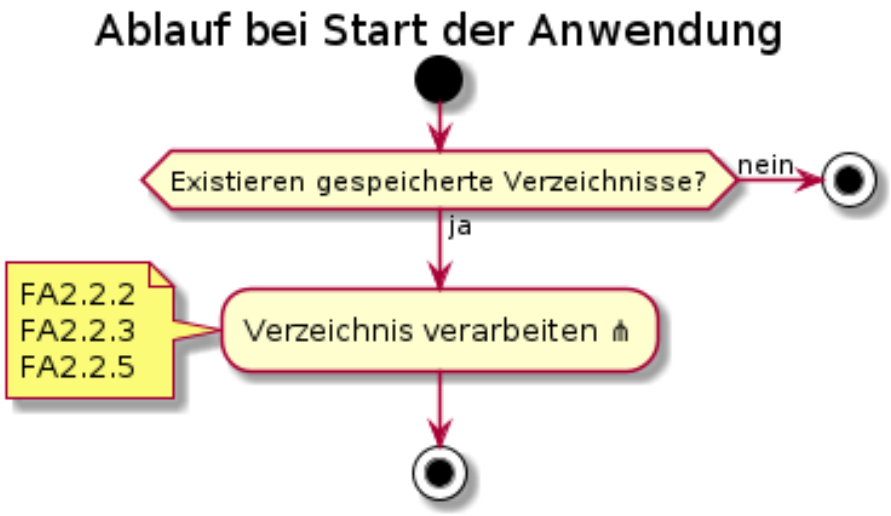
	Inkrement	
Kritikalität	3	
Verknüpfungen	2	U1.1
Funktionale Anforderungen	4	FA2.2.1, FA2.2.2, FA2.2.3, FA2.2.4, FA2.2.5
Nichtfunktionale Anforderungen	4	



UC2.2 UseCase Diagramm

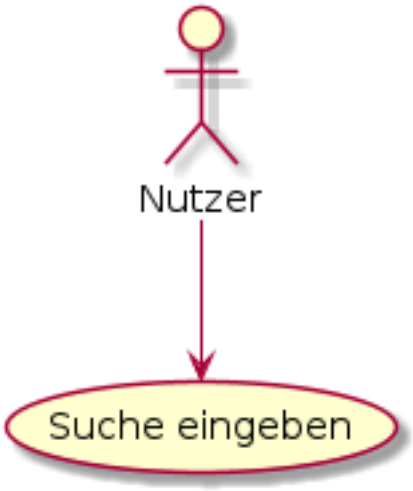


UC2.2 Ablaufdiagramm

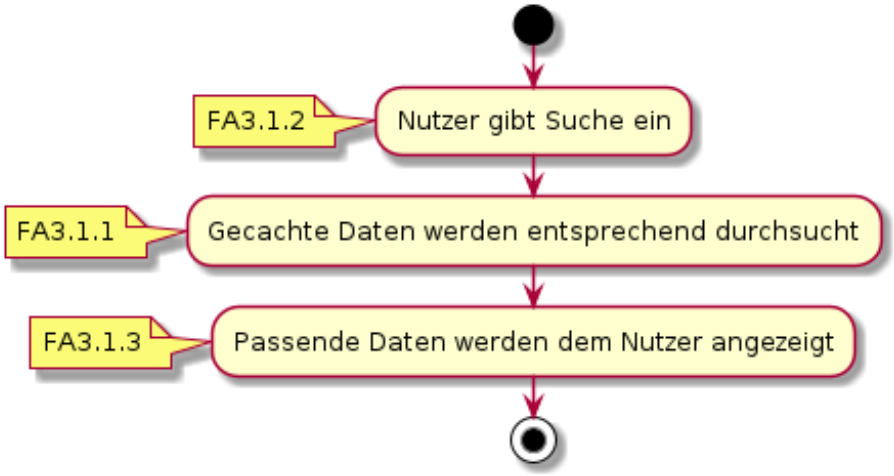


UC3.1 Projekte suchen

Schritt	Inkrement	
Id	1	UC3.1
Paket	2	P3
Autor	1	
Version	1	6
Kurzbeschreibung	1	Der Nutzer kann seine Projekte nach verschiedenen Eigenschaften durchsuchen
Beteiligte Akteure	1	Nutzer
Fachverantwortlicher	1	
Referenzen	2	
Vorbedingungen	2	Die Anwendung ist gestartet und gecachete und lokale Daten wurden geladen.
Nachbedingungen	2	Dem Nutzer werden die Projekte angezeigt, die den gegebenen Eigenschaften entsprechen
Typischer Ablauf	2	s. Ablaufdiagramm
Alternative Abläufe	3	-
Kritikalität	3	3
Verknüpfungen	2	-
Funktionale Anforderungen	4	FA3.1.1, FA3.1.2, FA3.1.3
Nicht-funktionale Anforderungen	4	



UC3.1 UseCase Diagramm

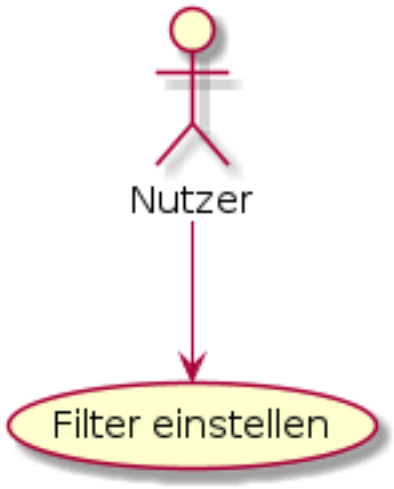


UC3.1 Ablaufdiagramm

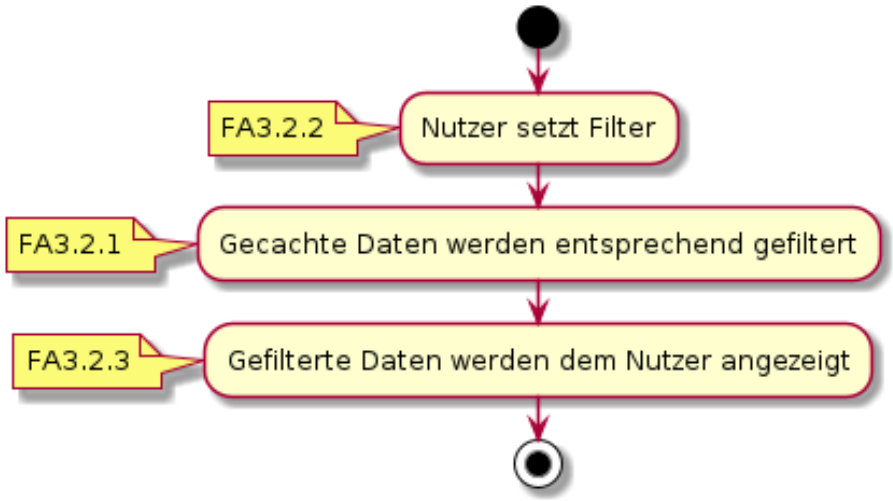
UC3.2 Projekte filtern

Schritt	Inkrement	
Id	1	UC3.2
Paket	2	P3
Autor	1	
Version	1	6
Kurzbeschreibung	1	Der Nutzer kann seine Projekte nach verschiedenen Kriterien filtern
Beteiligte Akteure	1	Nutzer

Schritt	Inkrement	
Fachverantwortlicher	1	
Referenzen	2	
Vorbedingungen	2	Die Anwendung ist gestartet und gecachete und lokale Daten wurden geladen.
Nachbedingungen	2	Dem Nutzer werden die Projekte angezeigt, die den gegebenen Kriterien entsprechen
Typischer Ablauf	2	s. Ablaufdiagramm
Alternative Abläufe	3	-
Kritikalität	3	3
Verknüpfungen	2	-
Funktionale Anforderungen	4	FA3.2.1, FA3.2.2, FA3.2.3
Nicht-funktionale Anforderungen	4	



UC3.2 UseCase Diagramm

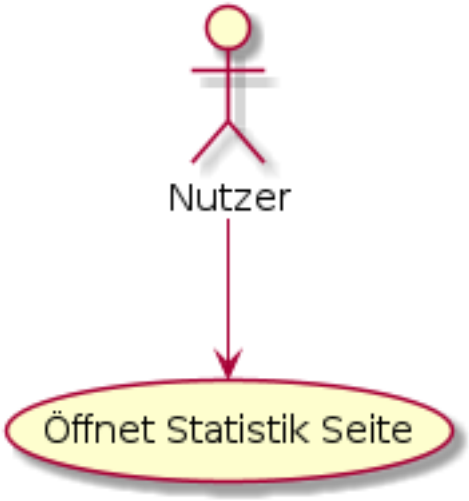


UC3.2 Ablaufdiagramm

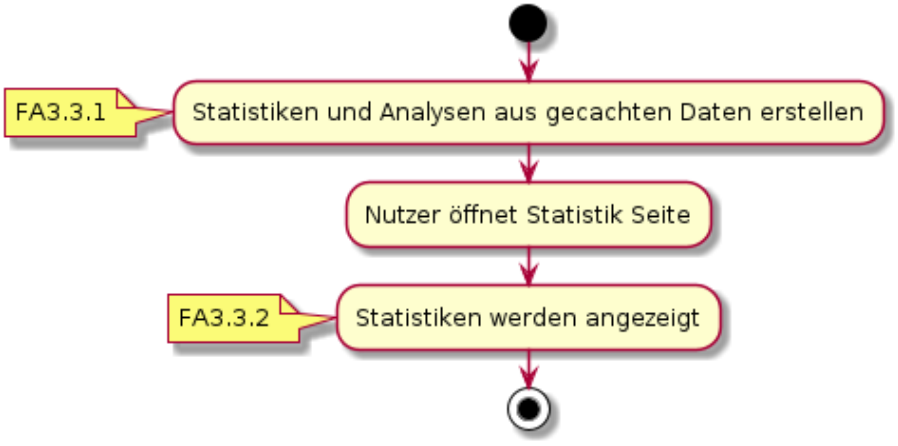
UC3.3 Statistiken

Schritt	Inkrement	
Id	1	UC3.3
Paket	2	P3
Autor	1	
Version	1	5
Kurzbeschreibung	1	Dem Nutzer werden in der graphischen Anwendung verschiedene Statistiken und Analysen zu seinen Projekten bereitgestellt
Beteiligte Akteure	1	Nutzer
Fachverantwortlicher	1	
Referenzen	2	
Vorbedingungen	2	Die Anwendung ist gestartet und gecachete und lokale Daten wurden geladen.
Nachbedingungen	2	Der Nutzer kann die bereitgestellten Statistiken in der graphischen Anwendung einsehen
Typischer Ablauf	2	s. Ablaufdiagramm
Alternative Abläufe	3	-
Kritikalität	3	3
Verknüpfungen	2	
Funktionale Anforderungen	4	FA3.3.1, FA3.3.2

Schritt	Inkrement
Nicht-funktionale Anforderungen	4



UC3.3 UseCase Diagramm



UC3.3 Ablaufdiagramm

Anforderungen

Funktionale Anforderungen

FA1.1.1 Beim Starten soll das System gecachte Daten laden.

FA1.1.2: Das System muss Manifestdateien aus dem lokalen Datensystem deserialiseren.

FA1.2.1 Das System muss dem Nutzer die Möglichkeit bieten eine Manifestdatei mit Metadaten zu erstellen.

FA1.2.2 Beim Erstellen muss das System prüfen, ob in dem relevanten Verzeichnis bereits eine Manifestdatei existiert.

FA1.2.3 Nach Erstellen soll das System die Metadaten im Cache speichern.

FA1.3.1 Das System soll dem Nutzer die Möglichkeit bieten Metadaten anzupassen.

FA1.3.2 Bei Änderungen soll das System relevante Metadaten automatisch im Cache aktualisieren.

FA1.3.3 Bei manuellen Änderungen an der Datei muss das System diese erkennen und dementsprechende Anpassungen im Cache vornehmen.

FA1.4.1 Das System muss dem Nutzer die Möglichkeit bieten Metadaten zu löschen.

FA1.4.2 Bei Löschung soll das System die relevanten Daten automatisch aus dem Cache entfernen.

FA1.4.3 Bei manueller Löschung muss das System dies erkennen und den relevanten Eintrag aus dem Cache entfernen.

FA2.1.1 Das System muss dem Nutzer die Möglichkeit bieten den Pfad zu einer einzelnen Manifestdatei anzugeben.

FA2.1.2 Gibt der Nutzer den Pfad zu einer valide Datei ein, muss das System in der Lage sein diese zu deserialisieren.

FA2.1.3 Nach der Deserialisierung soll das System die Daten im Cache speichern.

FA2.1.4 Das System soll dem Nutzer die Möglichkeit bieten Dateipfade für zukünftiges deserialisieren zu speichern.

FA2.1.5 Wenn gespeicherte Dateipfade existieren soll, das System beim Starten diese automatisch deserialisieren.

FA2.2.1 Das System soll dem Nutzer die Möglichkeit bieten ein Verzeichnis anzugeben, welches rekursiv nach Manifestdateien durchsucht wird.

FA2.2.2 Wenn in diesem Verzeichnis Manifeste existieren soll das System diese deserialisieren.

FA2.2.3 Wenn das System ein Manifest aus einem Verzeichnis deserialisiert hat, soll es bei Abweichungen den Cache aktualisieren.

FA2.2.4 Das System soll dem Nutzer die Möglichkeit bieten Verzeichnisse für zukünftiges durchsuchen zu speichern.

FA2.2.5 Wenn gespeicherte Verzeichnisse existieren, soll das System beim Starten diese automatisch durchsuchen.

FA3.1.1 Das System soll gecachte Metadaten auf verschiedene Eigenschaften durchsuchen können.

FA3.1.2 Das System soll dem Nutzer die Möglichkeit bieten die Suche anzupassen.

FA3.1.3 Das System soll dem Nutzer die Möglichkeit bieten auf Suchergebnisse zugreifen zu können.

FA3.2.1 Das System soll gecachte Metadaten nach verschieden Kriterien filtern können.

FA3.2.2 Das System soll dem Nutzer die Möglichkeit bieten den Filter anzupassen.

FA3.2.3 Das System soll dem Nutzer die Möglichkeit bieten auf den gefilterten Datensatz zugreifen zu können.

FA3.3.1 Das System soll Statistiken aus gecachten Metadaten erstellen können.

FA3.3.2 Das System soll dem Nutzer die Möglichkeit bieten auf diese Statistiken zugreifen zu können.

Nicht funktionale Anforderungen

NA1: Die Manifestdateien müssen von Menschen, als auch von Maschinen lesbar sein.

NA1.1: Die in der Manifestdatei zu findenden Metadaten sollen dem Nutzer nützliche Informationen über das dazugehörige Projekt bieten.

NA2: Die Anwendung soll schnellstmöglich dem Nutzer nach dem Start zur Bedienung bereitstehen.

NA3: Die Anwendung soll möglichst responsiv und nutzerfreundlich sein.

NA4: Die Anwendung muss in den Betriebssystemen Microsoft Windows 10, Microsoft Windows 11, und Linux funktionieren.

NA4.1: Für Linux sollen Pakete in den Formaten für die Distributionen/Paketsystem Debian/Ubuntu (apt), Arch (pacman) und Nix (nixpkgs).

NA5: Die Anwendung soll in Rust und Typescript verfasst sein.

NA5.1: Für die Anwendung sollen die Frameworks “Tauri v2.0+” für das Backend und “SvelteKit v2.8+” für das Frontend genutzt werden.

NA6: Der “Cache” der Anwendung soll als persistenter Cache mithilfe einer SQLite Datenbank implementiert werden.

NA6.1: Bei der Implementierung der Datenbank muss darauf geachtet werden, dass diese vor SQL-Injektionen ausreichend gesichert ist.

NA7: Für relevante Subsysteme müssen Unittests verfasst werden.

NA8: Die Anwendung muss für die Prozessorarchitektur x86_64 ausgelegt sein.

NA9: Die Anwendung muss Barrierefrei konstruiert werden um bspw. die Nutzung von Screenreadern zu erlauben.

NA10: Die Anwendung soll zunächst mit der Oberflächensprache Deutsch oder Englisch gebaut werden.

NA10.1: Texte in der Oberfläche sollen so eingebaut, um zukünftig die Implementierung neuer Sprachen einfach zu gestalten.

NA11: Das Projekt muss bis zum Ende der Theoriephase im Quartal 1 im Jahr 2025 abgeschlossen sein. Ein exaktes Datum hierfür folgt.

NA12: Alle Meetings müssen in Meetingprotokollen festgehalten werden.

NA13: Es muss eine Entwicklerdokumentation angefertigt werden.

Initialisierung Inkrement 3

Übersicht

Projekt: Projekt Episko
Inkrement: 3
Autor: Max Rodler
Datum: 30.10.2024
Zuletzt geändert:
von: Max Rodler
am: 22.11.2024
Version: 3
Prüfer: Paul Stöckle
Letzte Freigabe:
durch: Paul Stöckle
am: 23.11.2024

Changelog

Datum	Verfasser	Kurzbeschreibung
30.10.2024	Max Rodler	Initiales Erstellen und Verfassen
21.11.2024	Max Rodler	Vorläufiges Review formulieren
22.11.2024	Max Rodler	Endgültiges Review ergänzen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Zeitplan

- Beginn: 30.10.2024
- Ende: 22.11.2024

Arbeitspakete

Die folgenden Arbeitspakete werden als Teil des Inkrements bearbeitet: ### A3.1 UseCases anhand Schablone finden - Anhand einer gegebenen Schablone werden UseCases definiert, welche das Ergebnis zu erfüllen hat.

Verantwortlich: Max Rodler
Beauftragte: Ben Oeckl, Paul Stöckle, Max Rodler

A3.2 UseCase- und Ablaufdiagramme erstellen

- Es werden Diagramme zur Veranschaulichung und Erläuterung der UseCases und daraus resultierenden Abläufe arstellt.

Verantwortlich: Simon Blum
Beauftragte: Simon Blum, Ben Oeckl

A3.3 Requirements aus UseCases ableiten

- Zu den definierten UseCases werden nach vorgegebenem Schema Requirements formultiert.

Verantwortlich: Paul Stöckle
Beauftragte: Simon Blum, Paul Stöckle

Besonderheiten

- Da dieses Inkrement noch Teil der Projektinitialisierung ist und somit noch Vorbereitende Arbeitspakete enthält, existieren hierzu noch nicht alle notwendigen Dokumente. (Der Anforderungskatalog, das Designpaper, die Entwicklerdokumentation und ein ausführlicher Abschlussreport werden aufgrund mangelnder Notwendigkeit / Sinnhaftigkeit weggelassen.)
- Das Review erfolgt in diesem Inkrement im Rahmen der Vorstellung der UseCases und der dazugehörigen Diagramme.

Review Ergebnis

- Nach Anpassen einiger Details der UseCases und Umformulieren der nichtfunktionalen Requirements ist das Arbeitspaket abgeschlossen und es kann mit den Ergebnisssen weiter gearbeitet werden.

Review Inkrement 9

Übersicht

Projekt: Projekt Episko
Inkrement: 9
Autor: Max Rodler
Datum: 01.04.2025
Zuletzt geändert:
von: Max Rodler
am: 01.04.2025
Version: 1
Prüfer: Simon Blum
Letzte Freigabe:
durch: Simon Blum
am: 02.04.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
01.04.2025	Max Rodler	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Ergebnis

Im Laufe des Inkrements wurden alle Arbeitspakete erfolgreich bearbeitet. Jedoch wurden, aufgrund des durch die Prüfungsphase ausgelösten Zeitmangels, nicht für alle Arbeitspakete die geforderten Dokumente erstellt.

Aufgrund des zeitnahen Projektabschlusses wird dies auch nicht nachgeholt.

Initialisierung Inkrement 9

Übersicht

Projekt: Projekt Episko
Inkrement: 9
Autor: Max Rodler
Datum: 03.03.2025
Zuletzt geändert:
von: Paul Stöckle
am: 05.03.2025
Version: 2
Prüfer: Ben Oeckl
Letzte Freigabe:
durch: Ben Oeckl
am: 06.03.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
03.03.2025	Max Rodler	Initiales Erstellen und Verfassen
05.03.2025	Paul Stöckle	Anforderung für 9.3 hinzugefügt

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Zeitplan

- Beginn: 03.03.2025
- Ende: 31.03.2025

Arbeitspakete

Die folgenden Arbeitspakete werden als Teil dem Inkrement bearbeitet:

A9.1 Überarbeitung Create/Edit-Pages

Verantwortlich: Ben Oeckl
Beauftragte: Ben Oeckl

A9.2 All-Projects-Darstellung Cards

Verantwortlich: Max Rodler
Beauftragte: Max Rodler

A9.3 Backend-Implementierung Statistiken

Verantwortlich: Paul Stöckle
Beauftragte: Paul Stöckle

Relevante UseCases/Requirements

- FA3.3.1

A9.4 Fundament für Commands

Verantwortlich: Simon Blum
Beauftragte: Simon Blum

Anforderungsbewertung 9.4

Übersicht

Projekt: Projekt Episko
Inkrement: 9
Arbeitspaket: 4
Autor: Paul Stöckle
Datum: 05.03.2025
Zuletzt geändert:
von: Paul Stöckle
am: 05.03.2025
Version: 1
Prüfer: Simon Blum
Letzte Freigabe:
durch: Simon Blum
am: 06.03.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
05.03.2025	Paul Stöckle	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Bewertung relevanter Requirements und UseCases

FA3.3.1 Das System soll Statistiken aus gecachten Metadaten erstellen können.

Es wird eine neue Komponente benötigt die mit Hilfe der gecachten Daten aus der Datenbank vorgefertigte Metriken (Statistiken) berechnet und diese somit anschließend dem Frontend zur Anzeige zur Verfügung gestellt werden können.

Designpaper 9.4

Übersicht

Projekt: Projekt Episko
Inkrement: 9
Arbeitspaket: 4
Autor: Paul Stöckle
Datum: 05.03.2025
Zuletzt geändert:
von: Paul Stöckle
am: 05.03.2025
Version: 1
Prüfer: Simon Blum
Letzte Freigabe:
durch:
am:

Changelog

Datum	Verfasser	Kurzbeschreibung
05.03.2025	Paul Stöckle	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Designpaper

In der episko_lib wird eine neue Komponente mit einem Statistiken-Struct, der die Daten strukturiert hält, benötigt. Die Ausgabedaten werden mit Hilfe von Eingabedaten aus der Datenbank erstellt. Die SQL-Statements werden in die Datenbankkomponente ausgelagert.

Initiale Statistiken

Beschreibung	Interne Darstellung
Anzahl Projekte nach Sprache	Map <Language, Int>
Anzahl Projekte nach IDE	Map <Ide, Int>
Anzahl Projekte nach Kategorie	Map <String, Int>
Anzahl Projekte nach Bau-System	Map <BuildSystem, Int>
Anzahl aller Projekte	Int

Initialisierung Inkrement 2

Übersicht

Projekt: Projekt Episko
Inkrement: 2
Autor: Max Rodler
Datum: 18.10.2024
Zuletzt geändert:
von: Max Rodler
am: 30.10.2024
Version: 2
Prüfer: Paul Stöckle
Letzte Freigabe:
durch: Paul Stöckle
am: 30.10.2024

Changelog

Datum	Verfasser	Kurzbeschreibung
29.10.2024	Max Rodler	Initiales Erstellen und Verfassen
30.10.2024	Max Rodler	Update nach Review

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Zeitplan

- Beginn: 18.10.2024
- Ende: 30.10.2024

Arbeitspakete

Die folgenden Arbeitspakete werden als Teil des Inkrements bearbeitet: ### A2.1 Erstellen des Projektentwicklungsplans Im Laufe des Inkrements soll der Projektentwicklungsplan entstehen.

Verantwortlich: Ben Oeckl
Beauftragte: Ben Oeckl, Paul Stöckle, Max Rodler

A2.2 Erstellen von Diagrammen

Für den erstellten Entwicklungsplan sollen passende Diagramme erstellt werden.

Verantwortlich: Simon Blum
Beauftragte: Simon Blum

Besonderheiten

- Da die Arbeitsweise für das Projekt in diesem Inkrement erst festgelegt wird existieren hier nicht alle notwendigen Dokumente.
- Das Review erfolgte in diesem Inkrement im Rahmen der Vorstellung des Projektentwicklungsplans.

Review-Ergebnis

- Der Projektentwicklungsplan ist abgenommen.

Entwicklungsplan

Übersicht

Projekt: Projekt Episko
Inkrement: 2
Autor: Maximilian Rodler
Datum: 11.10.2024
Zuletzt geändert:
von: Maximilian Rodler
am: 29.01.2024
Version: 3
Letzte Freigabe:
durch: Simon Blum
am: 04.02.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
01.10.2024	Maximilian Rodler	Vorgehensmodell entwickeln uns ausarbeiten
12.10.2024	Simon Blum	Diagramme hinzufügen
29.01.2024	Maximilian Rodler	Dokument Entwicklerdoku durch SDD ersetzt

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Vorgehensmodell

Gearbeitet wird primär Iterativ-Inkrementell:

- Zu Beginn des Projektes werden Inkremente (Meilensteine) geplant.
- Dieses Inkrement kann ein oder mehrere Arbeitspakete beinhalten.

Diese Iterativ-Inkrementelle Arbeitsweise beinhaltet agile Elemente:

- Es gibt wöchentliche Meetings zur Absprache.

- Hier können die Ziele/der Zeitraum eines Inkrements angepasst werden, falls nötig.
- Es können parallel mehrere Arbeitspakete in einem Inkrement durchgeführt werden.

Eine genauere Übersicht kann unter “Diagramme” gefunden werden.

Teamzusammensetzung

- Simon Blum, TIT23
- Paul Stöckle, TIT23
- Maximilian Rodler, TIT23
- Ben Oeckl, TIT23

Rollen

Projektmanager - Maximilian Rodler
Head of Development - Paul Stöckle
Head of Quality & Operations - Simon Blum
Head of Testing & Integrations - Ben Oeckl

Verantwortlichkeiten und Aufgaben

Aufgabe	Verantwortliche
Protokollierung	Maximilian Rodler
Erstellung und Verwaltung von Arbeitspaketen/Meilensteine	Maximilian Rodler, Paul Stöckle
Einhaltung und Planung von Deadlines	Maximilian Rodler
Sicherung von Codequalität	Simon Blum
Sicherung von Dokumentenqualität	Simon Blum
Erstellung und Verwaltung von CI/CD Pipelines	Ben Oeckl, Simon Blum
Übersichtlichkeit und Struktur der Organisation	Ben Oeckl, Simon Blum
Testen der Funktionalität	Ben Oeckl
Koordinierung und Verantwortlichkeit für Arbeiten am Source Code	Paul Stöckle

Dokumente

Im Laufe des Prozesses werden diverse Dokumente erstellt.

Meetings

Zu jedem Meeting wird ein Dokument erstellt welches die folgenden Informationen beinhaltet:

- Datum, Ort

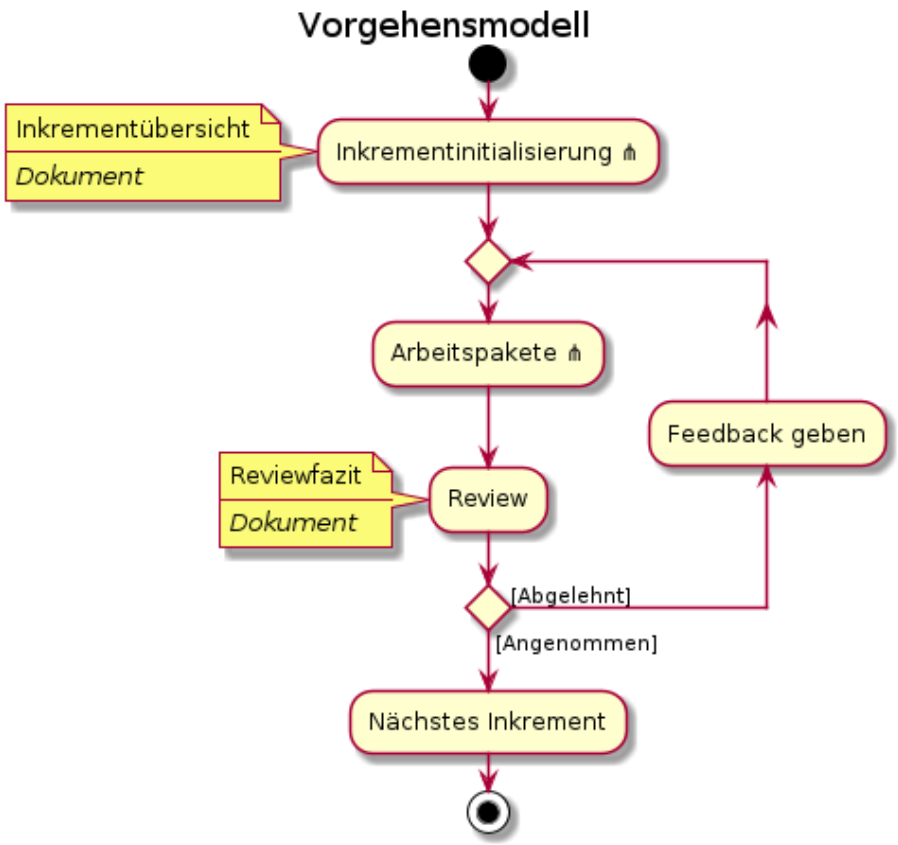
- Teilnehmer
- Moderation
- Diskussionspunkte
- Ergebnisse
- Folgeaktionen
- *Optional:* Hinweis auf relevante Dokumente
- *Optional:* Notizen

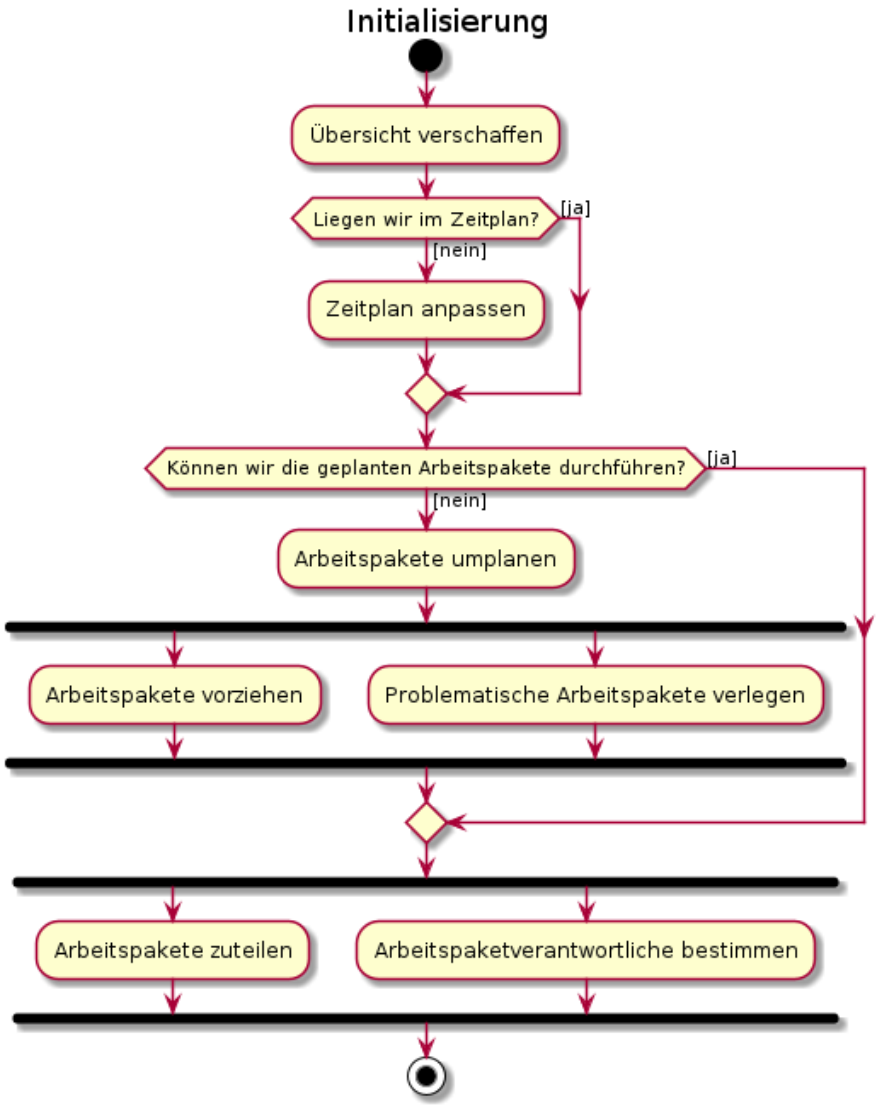
Inkremente

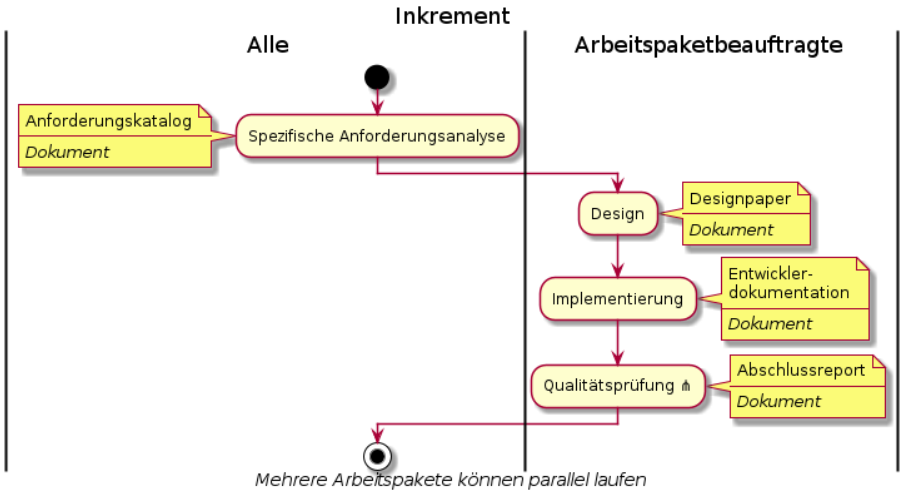
Im Rahmen eines Inkrementes werden folgende Dokumente erstellt:

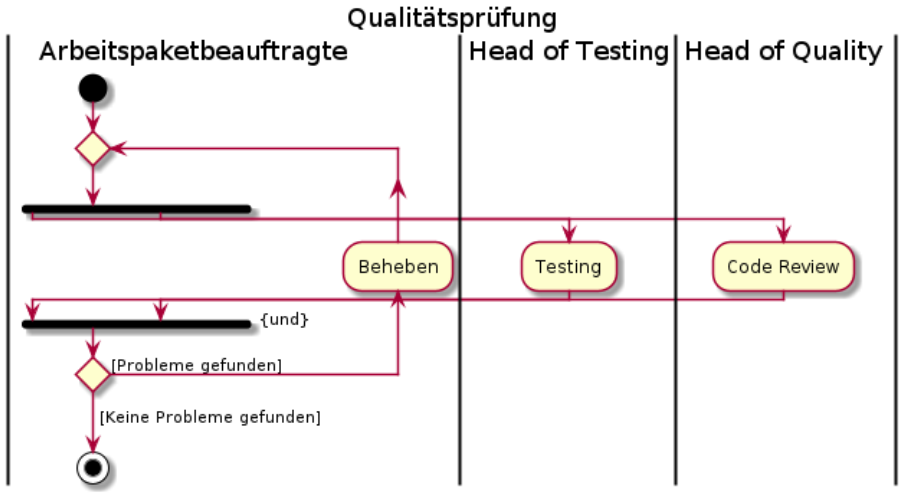
- Inkrementübersicht
  - Beinhaltet Arbeitspakete und Verantwortliche des Projektes
- Reviewfazit
  - Dieses wird bei mehreren Reviews erweitert
- Dokumente der/des Arbeitspaket/es:
  - Anforderungskatalog
  - Designpaper
  - Anpassen des jeweiligen SDD-Moduls
  - Abschlussreport
  - *Wird das Arbeitspaket in einer Iteration überarbeitet, werden diese Dokumente ergänzt!*

Diagramme


Vorgehensmodell
Inkrementübersicht
Dokument
Inkrementinitialisierung
Arbeitspakete
Feedback geben
Reviewfazit
Dokument
Review
[Abgelehnt]
[Angenommen]
Nächstes Inkrement


Initialisierung
Übersicht verschaffen
Liegen wir im Zeitplan?
[ja]
[nein]
Zeitplan anpassen
Können wir die geplanten Arbeitspakete durchführen?
[ja]
[nein]
Arbeitspakete umplanen
Arbeitspakete vorziehen
Problematische Arbeitspakete verlegen
Arbeitspakete zuteilen
Arbeitspaketverantwortliche bestimmen


Inkrement
Alle
Arbeitspaketbeauftragte
Anforderungskatalog
Dokument
Spezifische Anforderungsanalyse
Design
Designpaper
Dokument
Implementierung
Entwickler-dokumentation
Dokument
Qualitätsprüfung
Abschlussreport
Dokument
Mehrere Arbeitspakete können parallel laufen


Qualitätsprüfung
Arbeitspaketbeauftragte
Head of Testing
Head of Quality
Beheben
Testing
Code Review
{und}
[Probleme gefunden]
[Keine Probleme gefunden]

Projektskizze

Übersicht

Projekt: Projekt Episko
Inkrement: 0
Autor: Simon Blum
Datum: 01.10.2024
Zuletzt geändert:
von: Ben Oeckl
am: 11.10.2024
Version: 5
Prüfer: Simon Blum
Letzte Freigabe:
durch: Simon Blum
am: 11.10.2024

Changelog

Datum	Verfasser	Kurzbeschreibung
01.10.2024	Simon Blum	Initiales Erstellen und Verfassen
01.10.2024	Paul Stöckle	Hinzufügen von Formalitäten
01.10.2024	Maximilian Rodler	Anpassung an Feedback nach "Go/No-Go" Meeting
07.10.2024	Simon Blum	Aktualisierung von Formalitäten
11.10.2024	Ben Oeckl	Finalisierung zur Abgabe

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Scope of Work / Projektziele

- Verwaltung und Übersicht von Programmierprojekten
- Erstellung eines "Manifeststandards" für Metadaten eines Projektes
- Erstellung einer Kommandozeilenanwendung zum initialisieren/betrachten dieser Metadaten
- Erstellung einer Graphischen Anwendung zur Verwaltung und Übersicht

Systemgrenzen

- Interaktion mit Metadaten der Projekte
- Keine Interaktion mit Projekten selbst (Paketmanagement, Deployment, etc.)

  [!Note] Das Design der Anwendung soll flexibel genug sein um diesen Grenzen in zukünftigen Aufwänden erweitern zu können und so mehr Funktionalität einzubinden.

Risiken

- Vorerst keine identifiziert

Stakeholder

- Projektinterne Entwickler
- Auftraggeber (Dozent)
- Kunden:
  - Professionelle Entwickler
  - Hobbyentwickler
  - Studentische Entwickler

Randbedingungen

- Zeitraum 6 Monate
- Vorgaben zur Projektorganisation

Initialisierung Inkrement 1

Übersicht

Projekt: Projekt Episko
Inkrement: 1
Autor: Max Rodler
Datum: 01.10.2024
Zuletzt geändert:
von: Max Rodler
am: 29.10.2024
Version: 1
Prüfer: Simon Blum
Letzte Freigabe:
durch: Simon Blum
am: 31.10.2024

Changelog

Datum	Verfasser	Kurzbeschreibung
29.10.2024	Max Rodler	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Zeitplan

- Beginn: 01.10.2024
- Ende: 18.10.2024

Arbeitspakete

Die folgenden Arbeitspakete werden als Teil des Inkrement bearbeitet: ### A1.1 Erstellen einer Projektskizze Im Laufe des Inkrements soll eine Projektskizze entstehen.

Verantwortlich: Paul Stöckle
Beauftragte: Simon Blum, Ben Oeckl, Paul Stöckle, Max Rodler

Besonderheiten

- Da dieses Inkrement lediglich die Initialisierung des Projekts lostritt und ein Entwicklungsplan erst zu einem späteren Zeitpunkt ausgearbeitet wird, existieren zu diesem Inkrement nicht alle notwendigen Dokumente.
- Das Review erfolgt in diesem Inkrement im Rahmen der Vorstellung der Projektskizze.

Review-Ergebnis

- Das Projekt kann gestartet werden.

Review Inkrement 8

Übersicht

Projekt: Projekt Episko
Inkrement: 8
Autor: Max Rodler
Datum: 13.02.2025
Zuletzt geändert:
von: Max Rodler
am: 13.02.2025
Version: 1
Prüfer: Paul Stöckle
Letzte Freigabe:
durch: Paul Stöckle
am: 15.02.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
13.02.2025	Maximilian Rodler	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Bewertung Arbeitspakete

Bewertung Arbeitspaket 8.1

- Metadata-Interface nochmal überarbeiten (Unterinterfaces) (eigenes Arbeitspaket)
- Mehrfachauswahl bei Kategorien, Sprachen, Build-Systems ermöglichen

Abgeschlossen

Bewertung Arbeitspaket 8.2

- Alternativdarstellung über Karten versuchen

Abgeschlossen

Bewertung Arbeitspaket 8.3

Abgeschlossen

Bewertung Arbeitspaket 8.4

- Nicht möglicher Teil wird in neuem Arbeitspaket bearbeitet

Abgeschlossen

Ergebnis

- Die Reste aus Arbeitspaket 1 werden als neue Arbeitspakete weiter bearbeitet
- Alternativdarstellung für Arbeitspaket 2 als neues Arbeitspaket
- Nicht möglicher Teil von Arbeitspaket 4 wird in neuem Arbeitspaket bearbeitet

Designpaper 8.2

Übersicht

Projekt: Projekt Episko
Inkrement: 8
Arbeitspaket: 2
Autor: Max Rodler
Datum: 02.03.2025
Zuletzt geändert:
von: Max Rodler
am: 02.03.2025
Version: 1
Prüfer: Ben Oeckl
Letzte Freigabe:
durch: Ben Oeckl
am: 02.03.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
02.03.2025	Max Rodler	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Designpaper

Auf der Seite der Projektübersicht ("All Projects") sollen alle, der Anwendung hinzugefügten, Projekte übersichtlich angezeigt werden. Der Nutzer soll die Möglichkeit haben hier nach bestimmten Projekten zu suchen, oder aber Filter auf bestimmte Eigenschaften anwenden zu können. Des Weiteren sollte man von hier auf die Projektübersichtsseite gelangen, bzw. Projekte bearbeiten können.

Die Übersichtlichkeit der Projekte wird mithilfe einer Tabelle sichergestellt. Diese beinhaltet als Spalten einige ausgewählte Attribute des Projekts, und lässt sich mithilfe eines Suchfeldes nach dem Titel durchsuchen.

Befüllt wird die Tabelle mit allen Datensätzen der Datenbank, welche an anderer Stelle importiert und als Type definiert werden.

Anforderungsbewertung 8.2

Übersicht

Projekt: Projekt Episko
Inkrement: 8
Arbeitspaket: 2
Autor: Max Rodler
Datum: 02.03.2025
Zuletzt geändert:
von: Max Rodler
am: 02.03.2025
Version: 1
Prüfer: Ben Oeckl
Letzte Freigabe:
durch: Ben Oeckl
am: 02.03.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
02.03.2025	Max Rodler	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Bewertung relevanter Requirements und UseCases

FA3.1.2

Es muss die Möglichkeit geben in der Projektübersicht per Texteingabe nach Projekt-Titeln zu suchen.

FA3.1.3

Die Eingabe der Suche soll lediglich die Liste der angezeigten Projekte anpassen und nichts an der Funktonalität der Übersicht ändern. Mit den angezeigten Projekten soll nach wie vor auf die gleiche Weise interagiert werden können.

FA3.2.2

Es soll nach bestimmten Eigenschaften in dieser Projektübersicht gefiltert werden können.

FA3.2.3

Die Anwendung eines Filters soll lediglich die Liste der angezeigten Projekte anpassen und nichts an der Funktonalität der Übersicht ändern. Mit den angezeigten Projekten soll nach wie vor auf die gleiche Weise interagiert werden können.

Anforderungsbewertung 8.1

Übersicht

Projekt: Projekt Episko
Inkrement: 8
Arbeitspaket: 1
Autor: Ben Oeckl
Datum: 27.02.2025
Zuletzt geändert:
von: Ben Oeckl
am: 27.02.2025
Version: 1
Prüfer: Simon Blum
Letzte Freigabe:
durch: Simon Blum
am: 01.03.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
27.02.2025	Ben Oeckl	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Bewertung relevanter Requirements und UseCases

FA1.2.1 Das System muss dem Nutzer die Möglichkeit bieten eine Manifestdatei mit Metadaten zu erstellen.

Die Anforderung soll wie beschrieben erfüllt werden. ### FA1.3.1 Das System soll dem Nutzer die Möglichkeit bieten Metadaten anzupassen. Die Anforderung soll wie beschrieben erfüllt werden. ### FA1.4.1 Das System muss dem Nutzer die Möglichkeit bieten Metadaten zu löschen. Die Anforderung soll wie beschrieben erfüllt werden. ### FA2.1.1 Das System muss dem Nutzer die Möglichkeit bieten den Pfad zu einer einzelnen Manifestdatei anzugeben. Die Anforderung soll wie beschrieben erfüllt werden. ### NA von Arbeitspaket-6_2 sollten beachtet werden

Designpaper 8.1

Übersicht

Projekt: Projekt Episko
Inkrement: 8
Arbeitspaket: 1
Autor: Ben Oeckl
Datum: 02.03.2025
Zuletzt geändert:
von: Ben Oeckl
am: 02.03.2025
Version: 1
Prüfer: Simon Blum
Letzte Freigabe:
durch: Simon Blum
am: 02.03.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
02.03.2025	Ben Oeckl	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Designpaper

Ziel

Das Arbeitspaket hat das Ziel, eine moderne und modulare Frontend-Lösung zur Erstellung, Bearbeitung und Löschung von Manifestdateien mit Metadaten zu realisieren. Durch die Implementierung der Create- und Edit-Seiten soll der Nutzer die Möglichkeit haben, Metadaten zu verwalten und den Pfad zu einer Manifestdatei anzugeben. Die Lösung wird unter Einsatz von TypeScript zur Definition der Metadatenmodelle sowie moderner UI-Komponenten (aus der shadcn-svelte Bibliothek) entwickelt.

Umsetzung

- **Gemeinsame Komponente:**
  Eine zentrale Komponente wird erstellt, die sämtliche Eingabefelder für die Manifestdatei umfasst. Diese Komponente arbeitet in zwei Modi, die mittels Tabs gesteuert werden:
  - **Manual Entry:** Der Nutzer gibt alle erforderlichen Daten direkt ein.
  - **From File:** Der Nutzer kann einen Dateipfad entweder manuell eingeben oder über einen Dateiauswahldialog setzen.
- **Seiten für Create und Edit:**
  - **Create Project:** Übergibt leere Standardwerte an die gemeinsame Komponente.
  - **Edit Project:** Lädt existierende Projektdaten und übergibt diese als Initialwerte an die gemeinsame Komponente.
- **TypeScript-Datenmodell:**
  Die Metadaten werden über ein TypeScript-Interface definiert.
- **Die Logik wird in einen der nächsten Arbeitspaketen umgesetzt**

Anforderungsbewertung 8.3

Übersicht

Projekt: Projekt Episko
Inkrement: 8
Arbeitspaket: 3
Autor: Simon Blum
Datum: 17.02.2025
Zuletzt geändert:
von: Simon Blum
am: 17.02.2025
Version: 1
Prüfer: Max Rodler
Letzte Freigabe:
durch: Max Rodler
am: 20.02.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
17.02.2025	Simon Blum	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Bewertung relevanter Requirements und UseCases

FA2.1.4

Dateipfade welche der Nutzer speichern möchte, müssen in die Config geschrieben werden und von dort wieder gelesen werden können.

FA2.2.4

Für FA2.2.4 gilt das selbe wie für FA2.1.4.

Designpaper 8.3

Übersicht

Projekt: Projekt Episko
Inkrement: 8
Arbeitspaket: 3
Autor: Simon Blum
Datum: 17.02.2025
Zuletzt geändert:
von: Simon Blum
am: 17.02.2025
Version: 1
Prüfer: Max Rodler
Letzte Freigabe:
durch: Max Rodler
am: 20.02.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
17.02.2025	Simon Blum	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Designpaper

Für die Config soll ein seperates Modul erstellt werden. Entlang des Klassendiagram wird eine Struktur für die Config selbst und eine für den ConfigHandler erstellt.

Die Config selbst speichert den Pfad der Datenbank, Pfade von Dateien und Pfade von Verzeichnissen. Der Pfad der Config wird im ConfigHandler gespeichert.

Pfade

Für die Pfade sollen systemspezifische defaults verwendet werden, welche gängigen Konventionen folgen. #### Config Unix-like (falls eine Umgebungsvariable nicht vorhanden, das nächste): 1. $XDG_CONFIG_HOME/episko/config.toml 2. $HOME/.config/episko/config.toml

Windows: - %APPDATA%/episko/config.toml #### Datenbank Unix-like: 1. $XDG_CACHE_HOME/episko/cache.db 2. $HOME/.cache/cache.db

Windows: - %LOCALAPPDATA%/episko/cache.db

Serialisierung/Deserialisierung

Für das serialisieren und deserialisieren der Config Datei wird das bereits vorhandene `files` Modul und der darin vertretene `FILE` trait verwendet.

Initialisierung Inkrement 8

Übersicht

Projekt: Projekt Episko
Inkrement: 8
Autor: Max Rodler
Datum: 13.02.2025
Zuletzt geändert:
von: Max Rodler
am: 23.03.2025
Version: 2
Prüfer: Paul Stöckle
Letzte Freigabe:
durch: Paul Stöckle
am: 25.03.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
13.02.2025	Maximilian Rodler	Initiales Erstellen und Verfassen
23.03.2025	Maximilian Rodler	Anpassen des Bearbeitungszeitraums

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Zeitplan

- Beginn: 13.02.2025
- Ende: 03.03.2025

Arbeitspakete

Die folgenden Arbeitspakete werden als Teil der Inkremente bearbeitet:

A8.1 Frontend: Create & Edit Page realisieren

- TypeScript Klasse für Metadaten erstellen

- Löschen Button

Verantwortlich: Ben Oeckl
Beauftragte: Ben Oeckl

Relevante UseCases/Requirements

- FA1.2.1, FA1.3.1, FA1.4.1
- FA2.1.1

A8.2 Frontend: View Project / View all Projects

- Projekte nach Eigenschaften filtern
- Projekte nach Namen Durchsuchen
- Bearbeiten Button

Verantwortlich: Maximilian Rodler
Beauftragte: Maximilian Rodler

Relevante UseCases/Requirements

- FA3.1.2, FA3.1.3, FA3.2.2, FA3.2.3

A8.3 Backend: Config-Module

Verantwortlich: Simon Blum
Beauftragte: Simon Blum, Paul Stöckle

Relevante UseCases/Requirements

- FA2.1.4, FA2.2.4

A8.4 Backend: Load saved files/directories

Verantwortlich: Simon Blum
Beauftragte: Simon Blum, Paul Stöckle

Relevante UseCases/Requirements

- FA2.1.5, FA2.2.1, FA2.2.2, FA2.2.3, FA2.2.5

Anforderungsbewertung 8.4

Übersicht

Projekt: Projekt Episko
Inkrement: 8
Arbeitspaket: 4
Autor: Simon Blum
Datum: 18.02.2025
Zuletzt geändert:
von: Simon Blum
am: 18.02.2025
Version: 1
Prüfer: Ben Oeckl
Letzte Freigabe:
durch: Ben Oeckl
am: 20.02.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
18.02.2025	Simon Blum	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Bewertung relevanter Requirements und UseCases

FA2.1.5

Das System muss in der Lage sein, alle in der Config gespeicherten Dateipfade automatisch zu serialisieren.

FA2.2.1

Die Anforderung soll wie beschrieben erfüllt werden.

FA2.2.2

Die Anforderung soll wie beschrieben erfüllt werden.

FA2.2.3

Die Anforderung soll wie beschrieben erfüllt werden.

FA2.2.5

Das System muss in der Lage sein, alle in der Config gespeicherten Verzeichnisse automatisch zu durchsuchen.

Designpaper 8.4

Übersicht

Projekt: Projekt Episko
Inkrement: 8
Arbeitspaket: 4
Autor: Simon Blum
Datum: 18.02.2025
Zuletzt geändert:
von: Simon Blum
am: 18.02.2025
Version: 1
Prüfer: Ben Oeckl
Letzte Freigabe:
durch: Ben Oeckl
am: 20.02.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
18.02.2025	Simon Blum	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Designpaper

Zur Bearbeitung dieses Arbeitspaketes wird der im Klassendiagramm zu findende MetadataHandler erstellt.

Über diesen werden die nötigen Schnittstellen bereitgestellt um das Deserialisieren von Dateien und Pfaden zu vereinfachen.

Um wiederholtes laden zu speichern werden bereits geladene Instanzen in einer HashMap zwichengespeichert. Dieser Schritt führt eine weitere Caching Ebene ein. Gegebenenfalls sollte dies aber wieder entfernt werden um ein doppeltes zwichenspeichern im Front- und Backend zu vermeiden. Bei Implementierung der Tauri-Commands sollte sich hiermit auseinandergesetzt werden.

Designpaper 6.1

Übersicht

Projekt: Projekt Episko
Inkrement: 6
Arbeitspaket: 1
Autor: Simon Blum
Datum: 21.01.2025
Zuletzt geändert:
von: Simon Blum
am: 21.01.2025
Version: 1
Prüfer: Max Rodler
Letzte Freigabe:
durch: Max Rodler
am: 27.01.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
21.01.2025	Simon Blum	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Designpaper

Im Rahmen des Arbeitspakets soll eine Schnittstelle entstehen welche es erlaubt Manifestdateien zu serialisieren und zu deserialisieren.

API Design

Serialisieren

- Input: Datenstruktur mit Metadaten
- Output: Erfolgreich?
- SideEffects: Datei wird geschrieben/erstellt #### Deserialisieren

- Input: Pfad
- Output: Datenstruktur mit Metadaten
- SideEffects: Datei wird gelesen

Code Aufbau

Für das Erstellen der Schnittstelle wird ein Modul in der “episko_lib” erstellt, welches sowohl für die Cli als auch für das Gui Backend zur Verfügung steht.

Referenz Klassendiagramm

Im Bezug auf das Klassendiagramm werden folgende Klassen als “structs” entstehen: - Metadata - Language - IDE - BuildSystem - MetadataController (mit begrenzter Funktionalität) - FileSystemHandler (mit begrenzter Funktionalität)

Referenz Sequenzdiagramme

In diesem Arbeitspaket sollen Abschnitte aus folgenden den Sequenzdiagrammen behandelt werden: - zu U1.2 (Sequenz zwischen MetadataController und FileSystemHandler, mit Schnittstelle zu App)

Anforderungsbewertung 6.1

Übersicht

Projekt: Projekt Episko
Inkrement: 6
Arbeitspaket: 1
Autor: Simon Blum
Datum: 20.01.2025
Zuletzt geändert:
von: Simon Blum
am: 20.01.2025
Version: 1
Prüfer: Max Rodler
Letzte Freigabe:
durch: Max Rodler
am: 27.01.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
20.01.2025	Simon Blum	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Bewertung relevanter Requirements und UseCases

FA1.1.2 - Das System muss Manifestdateien aus dem lokalen Datensystem deserialiseren

Die Anforderung FA1.1.2 soll im Rahmen des Arbeitspakets wie angegeben erfüllt werden. Nach Angabe eines Pfades soll, falls vorhanden eine Datenstruktur zurückgegeben werden, welche alle Metadaten der Manifestdatei enthält. ### FA1.2.2 - Beim Erstellen muss das System prüfen, ob in dem relevanten Verzeichnis bereits eine Manifestdatei existiert Die Anforderung FA1.2.2 soll im Rahmen des Arbeitspakets wie angegeben erfüllt werden. Soll eine Manifestdatei geschrieben werden, muss das System prüfen, ob in dem gegebenen Verzeichnis

bereits ein Manifest existiert. ### NA1 - Die Manifestdateien müssen von Menschen, als auch von Maschinen lesbar sein Die Anforderung NA1 soll im Rahmen des Arbeitspakets wie angegeben erfüllt werden. Durch die Verwendung des Dateiformats “toml” soll dies garantiert werden.

Designpaper 6.2

Übersicht

Projekt: Projekt Episko
Inkrement: 6
Arbeitspaket: 2
Autor: Ben Oeckl
Datum: 12.02.2025
Zuletzt geändert:
von: Ben Oeckl
am: 12.02.2025
Version: 1
Prüfer: Paul Stöckle
Letzte Freigabe:
durch: Paul Stöckle
am: 19.02.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
12.02.2025	Ben Oeckl	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Designpaper

Im Rahmen des Arbeitspaketes soll die grundlegende Struktur der GUI-Komponente entwickelt werden, inklusive Sidebar und welche Seiten es geben soll. Hierzu wurde zunächst ein grobes Design für die Webseite erstellt: Episko Design

Anforderungsbewertung 6.2

Übersicht

Projekt: Projekt Episko
Inkrement: 6
Arbeitspaket: 2
Autor: Ben Oeckl
Datum: 12.02.2025
Zuletzt geändert:
von: Ben Oeckl
am: 12.02.2025
Version: 1
Prüfer: Paul Stöckle
Letzte Freigabe:
durch: Paul Stöckle
am: 19.02.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
12.02.2025	Ben Oeckl	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Bewertung relevanter Requirements und UseCases

FA1.2.1 Das System muss dem Nutzer die Möglichkeit bieten eine Manifestdatei mit Metadaten zu erstellen.

Die GUI muss eine benutzerfreundliche Oberfläche bereitstellen, die es ermöglicht, eine Manifestdatei mit den erforderlichen Metadaten zu erstellen. Hierzu soll aber zunächst nur das Design erstellt werden und die dazugehörige Seite vorhanden sein. Erst in einem spätereren Arbeitspaket soll die Seite vollständig umgesetzt werden. ### FA1.3.1 Das System soll dem Nutzer die Möglichkeit bieten Metadaten anzupassen. Hier gilt das gleiche wie bei der ersten Anforderungen. ### FA1.4.1 Das System muss dem Nutzer die

Möglichkeit bieten Metadaten zu löschen. Auch hier gilt das gleiche wie bei der ersten Anforderungen. ### FA2.1.1 Das System soll dem Nutzer die Möglichkeit bieten ein Verzeichnis anzugeben, welches rekursiv nach Manifestdateien durchsucht wird. Auch hier gilt das gleiche wie bei der ersten Anforderungen. ### NA3 Die Anwendung soll möglichst responsiv und nutzerfreundlich sein. Auf die nichtfunktionale Anforderung sollte geachtet werden, insbesondere bei diesem Arbeitspaket sollte beim Design auf die Nutzerfreundlichkeit geachtet werden. ### NA9 Die Anwendung muss Barrierefrei konstruiert werden um bspw. die Nutzung von Screenreadern zu erlauben. Bei diesem Arbeitspaket sollte deswegen auf einfache und leicht verständliche Sprache geachetet werden. ### NA10 Die Anwendung soll zunächst mit der Oberflächensprache Deutsch oder Englisch gebaut werden. Die Anwendung soll Englisch als Oberflächensprache verwenden. ### NA10.1 Texte in der Oberfläche sollen so eingebaut, um zukünftig die Implementierung neuer Sprachen einfach zu gestalten. Diese nichtfunktionale Anforderung soll in einem späteren Arbeitspaket behandelt werden.

Review Inkrement 6

Übersicht

Projekt: Projekt Episko
Inkrement: 6
Autor: Maximilian Rodler
Datum: 27.01.2025
Zuletzt geändert:
von: Maximilian Rodler
am: 27.01.2025
Version: 1
Prüfer: Ben Oeckl
Letzte Freigabe:
durch: Ben Oeckl
am: 27.01.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
27.01.2025	Maximilian Rodler	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Bewertung Arbeitspaket 6.1

- FA1.1.2: Implementiert und Funktionsfähig
- FA1.2.2: Implementiert und Funktionsfähig
- NA1: Umgesetzt

Arbeitspaket abgeschlossen.

Bewertung Arbeitspaket 6.2

Anmerkungen: - Create und Import zusammen legen - Löschen Button in “bearbeiten” - Alle Basic Seiten vorbereiten

Wird um ein Inkrement verlängert.

Bewertung Arbeitspaket 6.3

- FA1.2.1: Implementiert und Funktionsfähig
- FA1.4.1: Implementiert und Funktionsfähig
- FA2.1.1: Implementiert und Funktionsfähig

Implementierung abgeschlossen. Testing + Doku fehlt. Wird im nächsten Inkrement wieder aufgenommen.

Ergebnis

- Arbeitspaket 1: Abgeschlossen
- Arbeitspaket 2: Verlängert
- Arbeitspaket 3: Teilweise verlängert

Designpaper 6.3

Übersicht

Projekt: Projekt Episko
Inkrement: 6
Arbeitspaket: 3
Autor: Paul Stöckle
Datum: 20.01.2025
Zuletzt geändert: 20.01.2025
von: Paul Stöckle
am: 25.01.2025
Version: 2
Prüfer: Max Rodler
Letzte Freigabe:
durch: Max Rodler
am: 27.01.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
20.01.2025	Paul Stöckle	Initiales Erstellen und Verfassen
25.01.2025	Paul Stöckle	Designänderungen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Designpaper

Im Rahmen des Arbeitspaketes soll die grundlegende Struktur der CLI-Komponente entwickelt werden. ### CLI Es wird Clap zur Argumentübergabe verwendet. ### Commands, Argumente und Funktionalitäten | Command | Funktion | |———————-|———————————————————————————————————-| | create [OPTIONS] | Erstellen einer neuen Manifestdatei. Daten können durch Flags mitgegeben werden. Alle anderen werden durch interaktiv eingegeben Daten ergänzt. | | create [OPTIONS mit -n] | Erstellen einer neuen Manifestdatei. Daten werden nur den Flags entnommen. Bei relevanten Daten, die

fehlen wird das Programm abgebrochen. | | remove <FILE> | Löschen der angegeben Datei und Entfernung der Informationen aus dem System. | | add <FILE> | Hinzufügen der angegeben Datei zum System. | | validate <FILE> | Validerung auf Gültigkeit der angegeben Datei. |

Code-Isolation

Die CLI wird in einem eigenen Paket entwickelt: “episko_cli”. Die verschiedenen Komponenten werden auf mehrere Dateien aufgeteilt. Zur Argumentübergabe wird clap genutzt. Zur interaktiven Eingabe wird dialoguer benutzt.

Anforderungsbewertung 6.3

Übersicht

Projekt: Projekt Episko
Inkrement: 6
Arbeitspaket: 3
Autor: Paul Stöckle
Datum: 20.01.2025
Zuletzt geändert:
von: Paul Stöckle
am: 25.01.2025
Version: 2
Prüfer: Max Rodler
Letzte Freigabe:
durch: Max Rodler
am: 27.01.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
20.01.2025	Paul Stöckle	Initiales Erstellen und Verfassen
25.01.2025	Paul Stöckle	Hinzufügen neuer Punkte

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Bewertung relevanter Requirements und UseCases

FA1.2.1 Das System muss dem Nutzer die Möglichkeit bieten eine Manifestdatei mit Metadaten zu erstellen.

Es wird ein CLI-Programm benötigt mit welchem der Nutzer interaktiv und nicht interaktiv seine Daten angeben kann und anschließend eine Manifestdatei mit ihnen erstellt wird. ### FA1.4.1 Das System muss dem Nutzer die Möglichkeit bieten Metadaten zu löschen. Das CLI-Programm kann mit einer Dateiangabe diese löschen und die zugehörigen Informationen aus dem System entfernen. ### FA2.1.1 Das System muss dem Nutzer die Möglichkeit

bieten den Pfad zu einer einzelnen Manifestdatei anzugeben. Beim Erstellen einer neuen Manifestdatei soll der Nutzer angeben, ob diese auch im Programm aufgenommen wird. Zusätzlich muss es auch die Möglichkeit geben manuell neue Dateien im System zu registrieren. ### Zusatz Das CLI-Programm soll dem Nutzer auch die Möglichkeit zur Datei validierung bei händisch veränderten Manifestdateien geben.

Initialisierung Inkrement 6

Übersicht

Projekt: Projekt Episko
Inkrement: 6
Autor: Simon Blum
Datum: 20.01.2025
Zuletzt geändert:
von: Simon Blum
am: 20.01.2025
Version: 1
Prüfer: Ben Oeckl
Letzte Freigabe:
durch: Ben Oeckl
am: 27.01.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
21.01.2025	Simon Blum	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Zeitplan

- Beginn: 20.01.2025
- Ende: 27.01.2025

Arbeitspakete

Die folgenden Arbeitspakete werden als Teil des Inkrements bearbeitet:

A6.1 Backend - Manifestdatei einlesen und generieren

Verantwortlich: Simon Blum
Beauftragte: Simon Blum

6.1 - Relevante UseCases/Requirements

- FA1.1.2, FA1.2.2, NA1

A6.2 Frontend - Basic GUI Interface

Verantwortlich: Ben Oeckl
Beauftragte: Max Rodler, Ben Oeckl

6.2 - Relevante UseCases/Requirements

- FA1.2.1, FA1.3.1, FA1.4.1, FA2.1.1, NA3, NA9, NA10, NA10.1

A6.3 Frontend - Basic CLI Interface

Verantwortlich: Paul Stöckle
Beauftragte: Paul Stöckle

6.3 - Relevante UseCases/Requirements

- FA1.2.1, FA1.4.1, FA2.1.1

Inkrement Übersicht

Nr.	Titel	Begin	Ende	Arbeitspakete	Verantwortlicher	Initialisierung
1	Initialisierung	01.10.2024	18.10.2024	4	Max Rodler	(Link)[01/Initialisierung-1.md]
2	Entwicklungsplan	18.10.2024	30.10.2024	4	Max Rodler	(Link)[02/Initialisierung-2.md]
3	Anforderungsanalyse	30.10.2024	22.11.2024	4	Max Rodler	(Link)[03/Initialisierung-3.md]
4	Grobdesign	22.11.2024	06.12.2024	4	Max Rodler	(Link)[04/Initialisierung-4.md]
5	Aufwandsschätzung	10.01.2025	20.01.2025	5	Max Rodler	(Link)[05/Initialisierung-5.md]
6	Fundamentale Struktur	20.01.2025	27.01.2025	5	Max Rodler	(Link)[06/Initialisierung-6.md]
7	Datenbank Grundlagen	27.01.2025	13.02.2025	5 (2 übernommen)	Max Rodler	(Link)[07/Initialisierung-7.md]
8	Frontend + Configs	13.02.2025	24.02.2025	5	Max Rodler	(Link)[08/Initialisierung-8.md]
9	Zusammenführung	03.03.2025	31.03.2025	5	Max Rodler	(Link)[09/Initialisierung-9.md]

Software Detailed Design

Übersicht

Projekt: Projekt Episko
Autor: Simon Blum
Datum: 27.01.2025
Zuletzt geändert:
von: Paul Stöckle
am: 07.02.2025
Version: 5
Prüfer:
Letzte Freigabe:
durch: Ben Oeckl
am: 08.02.2025

Changelog

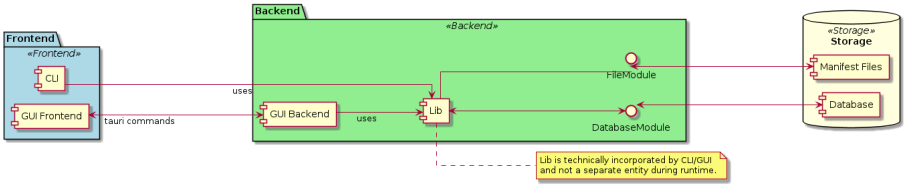
Datum	Verfasser	Kurzbeschreibung
27.01.2025	Simon Blum	Initiales Erstellen und Verfassen
01.02.2025	Paul Stöckle	Hinzufügen von CLI Modul
03.02.2025	Simon Blum	Hinzufügen von Datenbank Modul
07.02.2025	Simon Blum	Hinzufügen von Links fürs Wiki
07.02.2025	Paul Stöckle	Aktualisierung der CLI Library-Abhängigkeiten

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Übersicht Systemarchitektur

- Aufgeteilt in Lib, Cli, Gui
- Klassendiagramme/Sequenzdiagramme in Grobdesign
- Responsibility, wer macht was, was ist wo?
- Noch ein kleines Diagramm zu Komponentenübersicht?



Module

Backend - Datenbank

- Mehr ...

Backend - Lib

- Mehr ...

Backend - Derive Macro

- Mehr ...

Backend - Gui

- Mehr ...

Frontend- Gui

- Mehr ...

Frontend - Cli

- Mehr ...

SDD - GUI Frontend

Das GUI Frontend gibt dem Nutzer die Möglichkeit, die durch die Metadateien angelegten Projekte, übersichtlich und intuitiv anzeigen und verwalten zu können.

Verwendete Sprachen und Frameworks

Das GUI der Anwendung entspricht einer Web-Applikation, welche HTML und CSS nutzt und in Java-/ Typescript programmiert ist. Genutzt wird das UI-Framework "Svelte" unter der Verwendung von "shadcn", welches Grundlagen für die in der Anwendung verwendeten Komponenten bereitstellt.

Das Frontend greift auf die Logik des Backends (Rust) zurück um die notwendigen Daten zu erhalten.

Aufbau

Das implementierte Frontend ist in 2 Teile aufgeteilt: - Unter "routes" befinden sich die verschiedenen "Seiten" der Anwendung. Hier wird der allgemeine Aufbau aller Seiten definiert, als auch der spezifische Aufbau einzelner Seiten angepasst. Diese Seiten werden größtenteils aus den Komponenten des UI-Frameworks aufgebaut. - Alle verwendeten Komponenten sind unter "lib" zu finden. Hier werden die Inhalte des Frameworks abgelegt und können, bei Bedarf, für die eigene Verwendung angepasst werden.

SDD - Command Line Interface (CLI)

In dieser Dokumentation wird ausschließlich die Struktur und der Aufbau des Moduls beschrieben. Eine nähere Beschreibung der Komponenten befindet sich in Doc-Kommentare im Code selber.

- Rust Crate: episko_cli
- Dokumentation: docs.rs

Verwendete Sprachen

Die CLI-Komponente ist ausschließlich in der Programmiersprache Rust geschrieben.

Verwendete Librarys

Library	Zweck	Version
camino	UTF8-Pfade zur Verwendung bei CLI-Argumenten	1.1.9
clap	Übergabe und Verwaltung von CLI-Argumenten. Automatisch generierte Hilfe-Beschreibung	4.5.26
color_eyre	Optische Verbesserungen der interaktiven Nutzereingabe	0.6.3
dialoguer	Interaktive Nutzereingaben	0.11.0
tokio	Ermöglicht asynchrone Funktionsaufrufe für die Datenbankanbindung	1.43.0

Modulbeschreibung

Modul	Beschreibung
cli	Beinhaltet die clap-Einstellungen für die CLI (Argumente und Flags)
creation	Beinhaltet die Sammlung von Eingabedaten des Nutzers (interaktiv und nicht-interaktiv), Verarbeitung und anschließender Weitergabe dieser an die entsprechende interne Library-Funktion zum Erstellen und Aufnehmen einer neuen Manifestdatei
removal	Beinhaltet den Aufruf zur Löschung einer Manifestdatei und ihrer internen Speicherung
validation	Beinhaltet den Aufruf zur Validierung und zum Caching einer Manifestdatei

Modul creation

Da das creation-Modul umfangreich ist, wurden die Nutzereingaben, die Prompts, in das Modul cli/prompts ausgelagert.

CLI-Argumente

Das Programm wird wie folgt aufgerufen: `episko <COMMAND>`
Die Angabe eines Commands ist verpflichtend.
Auf die Angabe von clap standardmäßig bereitgestellten Commands wird folgend verzichtet! ### Liste aller Commands | Command | Nächstes Argument | Beschreibung | |————|————-|———————————————————| | `create` | `[OPTIONS]` | Erzeugen einer neuen Manifestdatei | | `remove` | `<FILE>` | Entfernen einer gegeben Datei aus dem Dateisystem und dem Programm | | `cache` | `<FILE>` | Hinzufügen einer gegeben Datei in das Programm | | `validate` | `<FILE>` | Validierung einer gegeben Datei | Die Angabe einer Datei ist verpflichtend. Die Angabe von Optionen allerdings ist optional und wird im folgenden genauer beschrieben. ### Optionen des `create`-Commands | Short-Flag | Long-Flag | Argument | Bedeutung | Syntax | |————|————-|——————|——————|———————————| | `n` | `non-interactive` | - | Anschalten des nicht-interaktiven Modus | | `d` | `directory` | Verzeichnis des Projekts | Datenangabe | 1 x Pfad | | `t` | `title` | Titel des Projekts | Datenangabe | 1 x Text | | `c` | `categories` | Kategorien | Datenangabe | n x Text (Leerzeichen-separiert) | | `l` | `languages` | Sprachen | Datenangabe | n x Text:Version (Leerzeichen-separiert) | | `p` | `preferred_ide` | Entwicklungsumgebung | Datenangabe | 1 x Text:Version | | `b` | `build-systems` | Build-System | Datenangabe | n x Text:Version (Leerzeichen-separiert) | | `D` | `description` | Beschreibung | Datenangabe | 1 x Text | | `r` | `repository-url` | Repository-Link | Datenangabe | 1 x Text | Wenn der Nutzer Daten per Flag eingibt und er sich im interaktiven Modus befindet (`n`-Flag nicht gesetzt), werden alle Daten abgefragt, bis auf die gegeben. Diese werden ohne weitere Eingabe verwendet. Im nicht-interaktiven Modus (`n`-Flag gesetzt) wird der Nutzer nie nach Daten gefragt. Diese muss er alle per Flag mitgeben. Falls hierbei notwendige Daten fehlen (Verzeichnis und Titel) schlägt das Programm fehl.

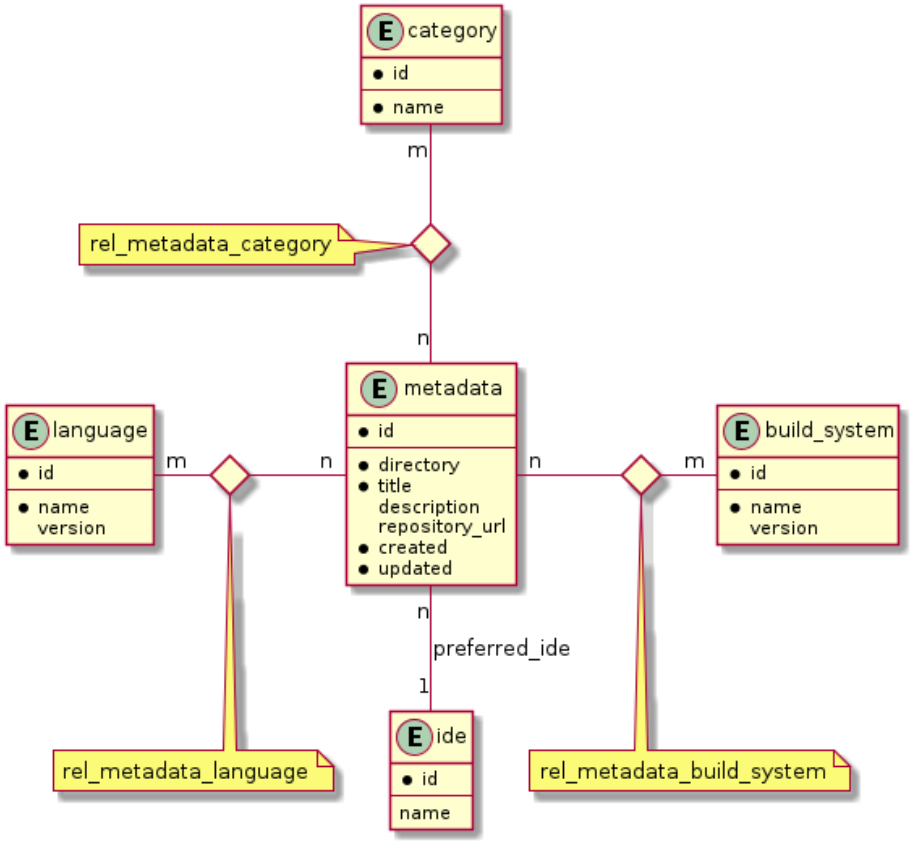
SDD - Datenbank

Die Datenbank dient im Kontext des Projektes als persistenter Cache. Hier werden Metadaten und alle dazugehörigen Relationen zwischengespeichert um ein performantes und responsives nutzen der Gui Anwendung zu ermöglichen.

Datenbanktechnologie

Als Datenbank wird sqlite als lokale Lösung genutzt. Diese ist simpel performant und erfüllt somit alle unsere Anforderungen an eine Speicherlösung.

ERM Diagramm



Datadictionary

metadata

Attribut	Datentyp	Länge	Nullable	Default	Key	Beschreibung
id	BLOB	128bit	Nein	-	Primary	UUIDv4
directory	TEXT	variabel	Nein	-	-	Lokales Verzeichnis des Manifestes
title	TEXT	variabel	Nein	-	-	Titel des Projektes
description	TEXT	variabel	Ja	NULL	-	Beschreibung
preferred_ide	BLOB	variabel	Ja	NULL	Foreign(Ide)	Id des bevorzugten Code Editor/Ide
repository_url	TEXT	variabel	Ja	NULL	-	URL des remote git repositorys
created	TEXT	variabel	Nein	-	-	Erstellzeipunkt in ISO 8601 Format
updated	TEXT	variabel	Nein	-	-	Aktualisierungszeipunkt in ISO 8601 Format

category

Attribut	Datentyp	Länge	Nullable	Default	Key	Beschreibung
id	BLOB	variabel	Nein	-	Primary	Id basierend auf Hash
name	TEXT	variabel	Nein	-	-	Name der Kategorie

language

Attribut	Datentyp	Länge	Nullable	Default	Key	Beschreibung
id	BLOB	variabel	Nein	-	Primary	Id basierend auf Hash
name	TEXT	variabel	Nein	-	-	Name der Programmiersprache
version	TEXT	variabel	Nein	-	-	Version der Programmiersprache

build_system

Attribut	Datentyp	Länge	Nullable	Default	Key	Beschreibung
id	BLOB	variabel	Nein	-	Primary	Id basierend auf Hash
name	TEXT	variabel	Nein	-	-	Name des Build Systems

Attribut	Datentyp	Länge	Nullable	Default	Key	Beschreibung
version	TEXT	variabel	Nein	-	-	Version des Build Systems

ide

Attribut	Datentyp	Länge	Nullable	Default	Key	Beschreibung
id	BLOB	variabel	Nein	-	Primary	Id basierend auf Hash
name	TEXT	variabel	Nein	-	-	Name des Code Editors/Ide

rel_metadata_category

Attribut	Datentyp	Länge	Nullable	Default	Key	Beschreibung
metadata_id	BLOB	variabel	Nein	-	Primary + Foreign(metadata)	Metadata Id
category_id	BLOB	variabel	Nein	-	Primary + Foreign(category)	Category Id

rel_metadata_language

Attribut	Datentyp	Länge	Nullable	Default	Key	Beschreibung
metadata_id	BLOB	variabel	Nein	-	Primary + Foreign(metadata)	Metadata Id
language_id	BLOB	variabel	Nein	-	Primary + Foreign(language)	Language Id

rel_metadata_build_system

Attribut	Datentyp	Länge	Nullable	Default	Key	Beschreibung
metadata_id	BLOB	variabel	Nein	-	Primary + Foreign(metadata)	Metadata Id
build_system_id	BLOB	variabel	Nein	-	Primary + Foreign(build_system)	Build System Id

SDD - Episko derive

Die `episko_derive` crate ist ein proc-macro welches die Implementierung des DatabaseObject traits für einfache Strukturen vereinfacht.

- Rust Crate: episko_derive
- Dokumentation: docs.rs

Verwendete Sprachen

Der derive macro ist ausschließlich in der Programmiersprache Rust geschrieben.

Verwendete Libraries

Library	Zweck	Version
deluxe	Parsen von derive Attributen wie `#[db(table = "category")]`	0.5.0
proc-macro2	Wird im zu Zusammenhang mit `deluxe` verwendet	1.0.93
quote	Umwandlung in TokenStreams	1.0.38
syn	Parsing von TokenStreams	2.0.98

Verwendung

Für die Verwendung des Macros sind sowohl in der oben genannten Dokumentation der Crate als auch in Dokumentation der Library Beispiele zu finden.

SDD - Gui Backend

- Rust Crate episko
- Dokumentation docs.rs

Verwendete Libraries

Library	Zweck	Version
tauri	GUI Backend	2.4.1
chrono	Datums- und Zeitmanipulation	0.4.40
tauri-plugin-shell	Shell Befehle ausführen	2.2.1
serde_json	JSON Serialisierung	1.0.140
serde	JSON Serialisierung	1.0.219
uuid	UUIDs	1.16.0
tauri-plugin-dialog	Dialoge erstellen	2.2.1
thiserror	Fehlerbehandlung	2.0.12
env_logger	Logging	0.11.7
tokio	Asynchrone Programmierung	1.43.0
episko_lib	Backend Library	1.3.1

SDD - Episko Lib

Die Library ist der zentrale Schlüsselpunktes des Projektes. Über sie wird mit allen weiteren Modulen des Projektes kommuniziert.

- Rust Crate: episko_lib
- Dokumentation: docs.rs

Verwendete Sprachen

Der Bibliothek ist ausschließlich in der Programmiersprache Rust geschrieben.

Verwendete Libraries

Library	Zweck	Version
-	-	

Aufbau

Die Library ist in die folgenden Module aufgeteilt, welche auch durch feature flags identifiziert werden können, aufgeteilt: - core - files - database - statistics - config

Im Kontext der Library stellt die Datenbank den persistenten Cache der Anwendung dar

Dokumentation der einzelnen Module kann in der oben genannten Dokumentation gefunden werden.

Software Quality Assurance - Plan

Übersicht

Projekt: Projekt Episko
Inkrement: 8
Autor: Simon Blum
Datum: 28.02.2025
Zuletzt geändert:
von: Max Rodler
am: 28.02.2025
Version: 1
Prüfer: Ben Oeckl
Letzte Freigabe:
durch: Ben Oeckl
am: 03.03.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
28.02.2025	Max Rodler	Initiales Erstellen und Verfassen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Einleitung und Anwendungsbereich

Beschreibung des Projekts und der betroffenen Softwareprodukte:

- Verwaltungssoftware für Software-Projekte, anhand eigens dafür eingeführter Manifest-Dateien.

Zielsetzung und Geltungsbereich des Plans

- Sicherstellen der Codequalität durch das ganze Projekt. Setzt klare Vorgaben für Entwickler. Sorgt für eine konstante und durchgänge Qualitätssicherung.
- Soll Fehler präventiv ausschließen, um eine spätere Korrektur von Fehlern zu vermeiden.

Normative Referenzen

Rust:

- Code-Formatting nach Rust-Style-Guide (https://doc.rust-lang.org/nightly/style-guide/)
- Linting in CI-Pipeline via Clippy (nach "Clippy 1.0 RFC")

Svelte:

- Code-Formatting nach Svelte-Guide-Lines via Prettier (https://github.com/sveltejs/prettier-plugin-svelte)
- Linting in CI-Pipeline via SvelteCheck (https://svelte.dev/docs/cli/sv-check)

Audit- und Bewertungskriterien

Audit

- Regelmäßige Reviews in jedem Inkrement
- Abschlussmeeting nach jedem Inkrement
- Beiträge zum Projekt nur via Pull-Requests

Metriken

- CI-Pipeline-Report

Dokumentation und Berichterstattung

- Freigaben von Reviews erfolgen via GitHub
- Dokumentation findet in Inkrement-Dokumenten statt.
- Verweis auf SQAReport

Software Quality Assurance - Report

Übersicht

Projekt: Projekt Episko
Autor: Simon Blum
Datum: 02.04.2025
Zuletzt geändert:
von: Max Rodler
am: 03.04.2025
Version: 2
Prüfer: Paul Stöckle
Letzte Freigabe:
durch: Paul Stöckle
am: 03.04.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
02.04.2025	Simon Blum	Initiales Erstellen und Verfassen
03.04.2025	Max Roder	Formatierung und Korrekturen

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Zusammenfassung der Qualitätssicherungstätigkeiten

Überblick Reviews

Inkrement	Ergebnis	Kommentar
1	Abgenommen	Abweichendes Verfahren
2	Abgenommen	Abweichendes Verfahren
3	Abgenommen	Abweichendes Verfahren
4	Abgenommen	Abweichendes Verfahren
5	Abgenommen	Abweichendes Verfahren
6	Teilweise Abgenommen	2 Arbeitspakete verlängert

Inkrement	Ergebnis	Kommentar
7	Abgenommen	-
8	Abgenommen	-
9	Teilweise Abgenommen	Teils fehlende Dokumentation

Nach der Durchführung der Inkrement Reviews wurde festgestellt, dass in der Dokumentation noch mehr Informationen enthalten sein sollten. Es wird zudem darauf hingewiesen, dass bei jedem Inkrement Review manuelle Tests durchgeführt werden, um die Funktionalität und Stabilität des Systems zu gewährleisten. Zusätzlich werden vor jedem Merge CodeReviews vorgenommen, um den Abschluss eines jeden Inkrements abzusichern.

Konformität mit Software Quality Assurance Plan

Im Rahmen des Projekts wird die Codequalität automatisch mithilfe von Clippy und SvelteCheck in der CI-Pipeline geprüft. Zudem wird der Code in der Pipeline automatisch mit Prettier und rustfmt formatiert, um einen einheitlichen Stil sicherzustellen. Des Weiteren erfolgt die Testdurchführung automatisch in der Pipeline. Somit wurden alle im QA-Plan vorgesehenen Maßnahmen vollständig umgesetzt.

Ergebnisse der Verifikations- und Validierungsmaßnahmen

Beim manuellen Lasttest wurde zunächst festgestellt, dass die Nutzererfahrung bei einer hohen Anzahl von Manifesten unbefriedigend war. Diese Situation konnte jedoch durch die Einführung einer Pagination optimiert werden, sodass die Leistung und Benutzerfreundlichkeit verbessert wurden.

Audit- und Review-Ergebnisse

Festgestellte Abweichungen und nicht erfüllte Anforderungen.

Requirement	Status	Kommentar
FA1.2.2	Teilweise erfüllt	Es wird geprüft, Ergebnisse der Prüfung jedoch ignoriert
FA1.4.3	Nicht erfüllt	Keine Überprüfung bei Laden aus Konfiguration

Requirement	Status	Kommentar
FA2.2.3	Teilweise erfüllt	Gelöschte Dateien werden nicht als solche erkannt
FA3.1.1	Teilweise erfüllt	Daten können nur nach Titel durchsucht werden
NA4.1	Nicht erfüllt	Es wird nur ein ".deb" Artifact erstellt
NA10.1	Nicht erfüllt	Auf zukünftige Internationalisierung wurde keine Rücksicht genommen

Maßnahmen zur Behebung identifizierter Probleme.

Dank der modularen Architektur können die identifizierten Probleme relativ leicht behoben werden. Es wurden keine kritischen Probleme festgestellt, sodass anstehende Korrekturen in zukünftigen Releases fix umgesetzt werden können.

Qualitätsmetriken und Leistungsbewertung

Analyse der erfassten Qualitätsmetriken

Die Analyse der Qualitätsmetriken ergab, dass die C0 Coverage bei 48 % liegt, was unter dem angestrebten Zielwert liegt. Ebenso beträgt die C1 Coverage 56 %, was ebenfalls unter dem Zielwert liegt. Positiv hervorzuheben ist, dass sowohl durch Clippy als auch durch SvelteCheck keine Probleme festgestellt wurden.

Lessons Learned und Empfehlungen

Identifikation von Verbesserungspotenzialen für zukünftige Projekte.

Die retrospektive Analyse hat gezeigt, dass bestehende Prozesse stellenweise nicht ausreichend dokumentiert waren. Zukünftige Projekte sollten stärker auf frühzeitige Prozessanalyse und kontinuierliches Feedback setzen. Einzelne Schwachstellen im QA-Prozess, wie z. B. die verspätete Einführung strukturierter Tests, weisen auf Verbesserungspotenziale hin. Eine genauere Auswertung dieser Punkte ist für künftige Projekte sinnvoll.

Vorschläge zur Optimierung von Qualitätssicherungsprozessen.

Es wird empfohlen, bereits in früheren Projektphasen eine strukturierte Pipeline zu implementieren, um die Qualitätssicherungsprozesse weiter zu optimieren und zu beschleunigen.

Schlussfolgerungen und Freigabeempfehlungen

Gesamtbewertung der Softwarequalität.

Die Gesamtbewertung der Softwarequalität zeigt, dass die Codequalität als gut einzustufen ist und auch die zugrundeliegende Architektur den Anforderungen entspricht. Allerdings weist die Testabdeckung noch deutlichen Verbesserungsbedarf auf.

Empfehlung für die Freigabe oder notwendige Nachbesserungen.

Aufgrund der Tatsache, dass keine kritischen Probleme vorliegen und es sich um eine nicht kritische Anwendung handelt, kann die Freigabe erteilt werden. Zwar sind stellenweise Verbesserungen möglich, diese stellen jedoch keine Blockade für die Freigabe dar.

Initialisierung Inkrement 7 / Review Inkrement 6

Übersicht

Projekt: Projekt Episko
Inkrement: 6+7
Datum, Ort: 27.01.2025
Teilnehmer: Simon Blum, Ben Oeckl, Paul Stoeckle, Max Rodler
Moderation: Maximilian Rodler
Diskussionspunkte:

- Review Inkrement 6
- Übernahmen von Inkrement 6 klären.
- Arbeitspakete Inkrement 7 festlegen:
  - Arbeitspaket 1: Datenbank Modul für Library
- Zeitraum festlegen:

Ergebnisse:

- Review Inkrement 6 abgeschlossen
- 6.2 wird in diesem Inkrement weiter bearbeitet
- Testing und Doku von 6.3 wird in diesem Inkrement weiter bearbeitet
- Beauftragte und Verantwortlichen für jedes Arbeitspaket festgelegt
- Requirements den Arbeitspaketen zugeordnet
- Zeitraum festgelegt: 27.01. - 03.02.

Aktionen:

Aktion	Verantwortlich	Deadline
Arbeitspaket bearbeiten	Alle	03.02.2025

Relevante Dokumente

- Initialisierung-7.md

Abnahme Entwicklungsplan

Übersicht

Projekt: Projekt Episko
Iteration: 0
Arbeitspaket: Projektinitialisierung
Datum, Ort: 13.11.2024, DHBW-Friedrichshafen
Teilnehmer: Simon Blum, Ben Oeckl, Paul Stoeckle, Max Rodler, Sascha Wanninger
Moderation: Sascha Wanninger
Diskussionspunkte:

- Vorstellung Engtwicklungsplan
- Abnahme Entwicklungsplan

Ergebnisse:

- Begriff Anforderungsanalyse ändern
- Enwicklungsplan abgenommen

Aktionen:

Aktion	Verantwortlich	Deadline
Begriff Anforderungsanalyse im Inkrement ändern	Simon Blum	15.11.2024
Link zu Dokument reparieren	Simon Blum	21.11.2024

Relevante Dokumente

-

Initialisierung Inkrement 9 / Review Inkrement 8

Übersicht

Projekt: Projekt Episko
Inkrement: 8+9
Datum, Ort: 03.03.2025, DHBW-Friedrichshafen
Teilnehmer: Simon Blum, Ben Oeckl, Paul Stoeckle, Max Rodler
Moderation: Max Rodler
Diskussionspunkte:

- Review Inkrement 8
- Arbeitspakete Inkrement 9 festlegen:
  - Arbeitspaket 1: Typescript Interfaces
  - Arbeitspaket 2: Überarbeitung Create/Edit-Pages
  - Arbeitspaket 3: All-Projects-Darstellung Cards
  - Arbeitspaket 4: Backend-Implementierung Statistiken
  - Arbeitspaket 5: Fundament für Commands
- Zeitraum festlegen

Ergebnisse:

- Review Inkrement 8 abgeschlossen
- Beauftragte und Verantwortlichen für jedes Arbeitspaket festgelegt
- Zeitraum festgelegt: 03.02. - 24.03.

Aktionen:

Aktion	Verantwortlich	Deadline
—		

Relevante Dokumente

- Review-8.md
- Initialisierung-9.md

Initialisierung Inkrement 8 / Review Inkrement 7

Übersicht

Projekt: Projekt Episko
Inkrement: 7+8
Datum, Ort: 13.02.2025, DHBW-Friedrichshafen
Teilnehmer: Simon Blum, Ben Oeckl, Paul Stoeckle, Max Rodler
Moderation: Maximilian Rodler
Diskussionspunkte:

- Review Inkrement 7
- Arbeitspakete Inkrement 8 festlegen:
  - Arbeitspaket 1: Frontend Create / Edit Page
  - Arbeitspaket 2: Frontend View Project / View all Projects
  - Arbeitspaket 3: Backend Configs
- Zeitraum festlegen

Ergebnisse:

- Review Inkrement 7 abgeschlossen
- Beauftragte und Verantwortlichen für jedes Arbeitspaket festgelegt
- Requirements den Arbeitspaketen zugeordnet
- Zeitraum festgelegt: 13.02. - 24.02.

Aktionen:

Aktion	Verantwortlich	Deadline
Arbeitspaket bearbeiten	Alle	03.02.2025

Relevante Dokumente

- Initialisierung-7.md

Abnahme SDD

Übersicht

Projekt: Projekt Episko
Inkrement: 8
Datum, Ort: 14.02.2025, DHBW-Firedrichshafen
Teilnehmer: Simon Blum, Ben Oeckl, Paul Stoeckle, Max Rodler, Sascha Wanninger
Moderation: Sascha Wanninger
Diskussionspunkte:

- Abnahme SDD

Ergebnisse:

- SDD abgenommen

Aktionen:

Aktion	Verantwortlich	Deadline
—		

Relevante Dokumente

- Alles Dokumente des Software Detailed Designs

Definition Use-Cases

Übersicht

Projekt: Projekt Episko
Inkrement: 0
Arbeitspaket: Projektvorbereitung
Datum, Ort: 13.11.2024, DHBW-Friedrichshafen
Teilnehmer: Simon Blum, Ben Oeckl, Paul Stoeckle, Max Rodler
Moderation: Simon Blum
Diskussionspunkte:

- Gedanken zu Use-Cases machen
- Use-Case Schablone ausfüllen
- zugehörige funktionale Anforderungen finden
- nichtfunktionale Anfforderungen definieren

Ergebnisse:

- Use-Cases 1.1 - 3.4 definiert
- zugehörige funktionale Anforderungen formuliert
- nichtfunktionale Anforderungen formuliert

Aktionen:

Aktion	Verantwortlich	Deadline
Diagramme erstellen	Simon Blum	15.11.24
Anforderungen umformulieren nach Rupp	Paul Stöckle	15.11.24
Zuordnung zu den use-Cases	Ben Oeckl	15.11.24

Relevante Dokumente

Abnahme SQAP

Übersicht

Projekt: Projekt Episko
Inkrement: 8
Datum, Ort: 28.02.2025, DHBW-Friedrichshafen
Teilnehmer: Ben Oeckl, Paul Stoeckle, Max Rodler, Sascha Wanninger
Moderation: Sascha Wanninger
Diskussionspunkte:

- Vorstellung SQAP

Ergebnisse:

- SQAP grundsätzlich in Ordnung
- Relevante Metriken aus Pipeline-Report noch definieren

Aktionen:

Aktion	Verantwortlich	Deadline
Limits für CI Metriken festlegen	Ben Oeckl	07.03.2025

Relevante Dokumente

- SQAP

Erstellung Software Quality Assurance Plan

Übersicht

Projekt: Projekt Episko
Inkrement: 8
Datum, Ort: 28.02.2025, DHBW-Friedrichshafen
Teilnehmer: Ben Oeckl, Paul Stoeckle, Max Rodler
Moderation: Max Rodler
Diskussionspunkte:

- Erstellung SQA-Plan

Ergebnisse:

- SQAP erstellt
- Enthält unter anderem:
- – Verweis auf Formatting- und Linting Referenzen
- – Beschreibung der Audits und Dokumentation

Aktionen:

Aktion	Verantwortlich	Deadline
Platzhalter		

Relevante Dokumente

Überarbeitung Aufwandsschätzung

Übersicht

Projekt: Projekt Episko
Inkrement: 8
Datum, Ort: 17.02.2025
Teilnehmer: Simon Blum, Ben Oeckl, Paul Stoeckle, Max Rodler
Moderation: Paul Stoeckle
Diskussionspunkte:

- Überarbeitung der Kostenschätzung in der Mitte der Implementierungszeit

Ergebnisse:

- Reduktion der antizipierten Gesamtstunden bum 31 bei gleichem Function Point-Multiplikator
- Gegen Ende des Projekts soll eine erneute Schätzung durchgeführt werden.

Aktionen:

Aktion	Verantwortlich	Deadline
Erneute Schätzung gegen Ende des Projekts	Maximilian Rodler	TBD

Relevante Dokumente

- Aufwandsschätzung.md

Planning Meeting

Übersicht

Projekt: Projekt Episko
Inkrement: 0
Arbeitspaket: Projektinitialisierung
Datum, Ort: 01.10.2024, DHBW Friedrichshafen
Teilnehmer: Simon Blum, Ben Oeckl, Paul Stoeckle, Max Rodler
Moderation: Max Rodler
Diskussionspunkte:

- Was ist unser Ziel?
- Wie erreichen wir dieses?

Ergebnisse:

- Erstellung der Projektskizze

Aktionen:

Aktion	Verantwortlich	Deadline
Meeting mit Auftraggeber	Alle	01.10.2024

Projekt Ziele

Ziel	Was soll erreicht werden?
Stakeholder	Zielgruppe, Entwickler, Abnehmer
Auswirkungen auf Stakeholder	Einfachere Verwaltung von Projekten
Randbedingungen	Zeitrahmen (6 Monate), Vorgaben für das Projekt
Abhängigkeiten	Hauptziel - Keine Abhängigkeiten
Sonstiges	Klare Struktur und Dokumentation

Rahmenbedingungen

Risiken

Go - Checklist

- ☐ Sind die Ziele klar und eindeutig?
  - Ja
- ☐ Sind die Ziele messbar?
  - Messbar auf Basis von Feedback
  - Aufwand manuell vs mit Andwendung
- ☐ Bedeuten die Ziele einen klaren Vorteil für den Kunden/Anwender?
  - Ja, Ziel ist es das Verwalten und die Übersicht von Projekten signifikant zu vereinfachen
- ☐ Kann man die Ziele in der gegebenen Zeit und mit dem gegebenen Budget erreichen?
  - Ja
- ☐ Gibt es Risiken mit hoher Wahrscheinlichkeit, die es unmöglich machen das Projekt erfolgreich durchzuführen?
  - Nein, wir sind flexibel und zuversichtlich alles überwinden zu können
- ☐ Sind alle Stakeholder bereit mitzuarbeiten?
  - ?
- ☐ Gibt es weitere Untersuchungen, die vor dem Start durchgeführt werden müssen?
  - Findung von Technologien etc.
  - Marktanalyse - gibt es schon ähnliche Produkte?

Goal concept

- Project mangement system

Das Ziel des Projektes ist es eine Anwendung zu erstellen, welche genutzt werden kann um Programmierprojekte zu verwalten. Hierfür soll eine konsolenbasierte und eine graphische Anwendung existieren. Das System soll über eine standartisierte Manifestdatei ermöglicht werden. Folgende Funktionen sollen ermöglicht werden:

- Übersicht über vorhandene Projekte
  - Name, Pfad…
- Sortierung durch Kategorieren/Labels
- Kreation und verwaltung von Projekten

Zukunft

Zusätzlich kann hierbei erweitert werden mit:

- Integration Git/Github
  - Status

- Statisktiken
- Öffnen in favorisierter IDE
- Möglichkeiten der Fernverwaltung

Systemgrenzen

- Interaktion mit Metadaten der Projekte
- Keine Interaktion mit Projekten selbst (paketmanagement, deployment, etc)

  [!Note] Das Design der Anwendung soll flexibel genug sein um diesen Grenzen in zukünftigen Aufwänden erweitern zu können und so mehr Funktionalität einzubinden.

Todo:

- ☐ Team orga
- ☐ Projektname
- ☐ Technologien

Abnahme Aufwandsschätzung

Übersicht

Projekt: Projekt Episko
Inkrement: 5
Arbeitspaket:
Datum, Ort: 10.01.2025, DHBW Friedrichshafen
Teilnehmer: Ben Oeckl, Paul Stöckle, Max Rodler, Sascha Wanninger
Moderation: Sascha Wanninger
Diskussionspunkte:

- Absprache Stand Projekt
- Vorstellung der Aufwndsschätzung und unserer Bedenken bezüglich dem Ergebnis

Ergebnisse:

- Bedenken berechtigt, aber nicht zu vermeiden, da Methode für Vollzeitteams gedacht
- Zeitplanung im Auge behalten und während der Arbeit nachjustieren (Sowohl Gesamtgewichtung (100:400) als auch einzelne Werte)

Aktionen:

Aktion	Verantwortlich	Deadline

Relevante Dokumente

- Aufwandsschätzung.md

Software Verifikation Plan

Übersicht

Projekt: Projekt Episko
Inkrement: 8
Datum, Ort: 14.02.2025, DHBW Friedrichshafen
Teilnehmer: Ben Oeckl, Paul Stoeckle, Max Rodler
Moderation: Ben Oeckl
Diskussionspunkte:

- Definition von:
  - Verifikationsstrategien
  - Verifikationsumgebung
  - Testfällen
  - Verantwortlichkeiten
  - Zeitlicher Einordnung

Ergebnisse:

- Grundsätzlichen SVP definiert.

Aktionen:

Aktion	Verantwortlich	Deadline
Umsetzung der Definierten Tests	Alle Entwickler	TBD

Relevante Dokumente

- SVP.md

Meeting Grobdesign

Übersicht

Projekt: Projekt Episko
Inkrement: 0
Arbeitspaket: Projektinitialisierung
Datum, Ort: 19.11.2024, bei S. Blum, Friedrichshafen
Teilnehmer: Simon Blum, Ben Oeckl, Paul Stoeckle, Max Rodler
Moderation: Ben Oeckl
Diskussionspunkte:

- Frameworks
- Benutzeroberfläche
- Objekte und Klassen
- Klassendiagramm

Ergebnisse:

- Liste an Klassen
- Klassendiagramm

Aktionen:

Aktion	Verantwortlich	Deadline
Multiplizitäten überlegen	Max Rodler	—
Klassendiagramm fortführen	Simon Blum	—
Eigenschaften überlegen	Paul Stöckle & Ben Oeckl	—

Relevante Dokumente

#
Zwis-
chen-
stand
Ab-
sprache

Übersicht

Projekt: Projekt Episko
Inkrement: 9
Datum, Ort:
Teilnehmer: Simon Blum, Ben Oeckl, Paul Stoeckle, Max Rodler
Moderation: Simon Blum
Diskussionspunkte:

- Was gibt es noch zu tun?
- Wie machen wir weiter

Ergebnisse:

- Features-ToDo:
  - Formular richtig
    - UI: Ben
    - Data-Validation: Simon
  - Datei/Ordner auswählen
    - Simon
  - Einzelansicht
    - Max
  - Statistiken
    - Backend: Paul
    - Commands: Simon
    - UI: Max/Ben
  - (Settings)
  - Harmonisierung
    - Simon
- Drumherum:
  - Dokumentation
  - Software Verification Plan
  - Software Quality Assurance Plan
    - CI-Metriken
  - SQAR
  - Glossar
    - Alle Dokumente durchgehen, “unklar” rausschreiben
  - Alles überprüfen

    - SDD
- GitHub-Issues mit genauer Aufgabenverteilung erstellt

Aktionen:

Aktion	Verantwortlich	Deadline
Issues bearbeiten	Alle	02.04.2025

Kickoff Meeting

Übersicht

Projekt: Projekt Episko
Inkrement: 0
Arbeitspaket: Projektinitialisierung
Datum, Ort: 01.10.2024, DHBW Friedrichshafen
Teilnehmer: Simon Blum, Ben Oeckl, Paul Stoeckle, Max Rodler, Sascha Wanninger
Moderation: Sascha Wanninger
Diskussionspunkte:

- Vorstellung der Projektskizze
- Feedback des Auftraggebers

Ergebnisse:

- "Go" für das Projekt wurde gegeben
- Feedback zur spezifizierung der Stakeholder

Aktionen:

Aktion	Verantwortlich	Deadline
Feedback umsetzen	Max Roder	07.10.2024
Abgabe der Projektskizze	Ben Oeckl	11.10.2024
Link zu Dokument reparieren	Simon Blum	21.11.2024

Relevante Dokumente

-

SDD-Struktur

Übersicht

Projekt: Projekt Episko
Inkrement: 6
Datum, Ort: 24.01.2025, DHBW-Friedrichshafen
Teilnehmer: Simon Blum, Ben Oeckl, Paul Stoeckle, Max Rodler
Moderation: Max Rodler
Diskussionspunkte:

- Struktur SDD überlegen

Ergebnisse:

- SDD Struktur festgelegt
- SDP angepasst

Aktionen:

Aktion	Verantwortlich	Deadline
—		

Relevante Dokumente

Ordner SoftwareDetailedDesign

Abnahme Anforderungsanalyse

Übersicht

Projekt: Projekt Episko
Inkrement: 0
Arbeitspaket: Projektinitialisierung
Datum, Ort: 15.11.2024, DHBW-Friedrichshafen
Teilnehmer: Simon Blum, Ben Oeckl, Paul Stoeckle, Max Rodler, Sascha Wanninger
Moderation: Sascha Wanninger
Diskussionspunkte:

- Vorstellung & Abnahme Anforderungsanalyse

Ergebnisse:

- Schritte abstrahieren (loslösen Von GUI/CLI (teilweise gleiche Schritte))
- Nicht der USER serialisiert (Soll kein Use-Case sein)
- Checksum erstellen ist auch kein Use-Case
- U3.4 Titel fehlt
- Metadaten laden auch kein Use-Case
- F0.1.1 nicht funktional
- Funktionale Requirements umformulieren (Nutzer weniger im Fokus)

Aktionen:

Aktion	Verantwortlich	Deadline
Funktionale Requirements umformulieren	Paul Stöckle	19.11.2024
Use-Cases überarbeiten (abstrahieren)	Simon Blum	19.11.2024
Link zu Dokument reparieren	Simon Blum	21.11.2024

Relevante Dokumente

-

Initialisierung Inkrement 6

Übersicht

Projekt: Projekt Episko
Inkrement: 6
Datum, Ort: 20.01.2025, DHBW-Friedrichshafen
Teilnehmer: Simon Blum, Ben Oeckl, Paul Stoeckle, Max Rodler
Moderation: Max Rodler
Diskussionspunkte:

- Arbeitspakete definieren:
  - Arbeitspaket 1: Backend - Manifestdatei einlesen / generieren
  - Arbeitspaket 2: Frontend - Basic GUI Interface
  - Arbeitspaket 3: Frontend - Basic CLI Interface
- Zeitraum festlegen

Ergebnisse:

- Beauftragte und Verantwortlichen für jedes Arbeitspaket festgelegt
- Requirements den Arbeitspaketen zugeordnet
- Zeitraum: 20.01. - 27.01.

Aktionen:

Aktion	Verantwortlich	Deadline
Arbeitspaket starten	Alle	27.01.2025

Relevante Dokumente

- Initialisierung-6.md

Meetingtitel

08-11-2024-Use-Cases

Übersicht

Projekt: Projekt Episko
Inkrement: 0
Arbeitspaket: Anforderungsanalyse
Datum, Ort: 08.11.2024 , DHBW Friedrichshafen
Teilnehmer: Simon Blum, Ben Oeckl, Paul Stoeckle, Max Rodler
Moderation: Max Rodler
Diskussionspunkte:

- Anforderungen und Use-Cases definieren

Ergebnisse:

- Basisinformationen
- Prozessdaten
- Funktionalitäten
- Angrenzende Systeme

Aktionen:

Aktion	Verantwortlich	Deadline
Anforderungsanalyse erstellen	Alle	15.11.2024

Basisinformationen

- Metadatenverwaltung
  - Neues Projekt anlegen
  - Recursives auffinden von Manifestdateien
  - Einlesen von Manifestdateien
  - Cachen von Manifesten
  - Metadaten bearbeiten
  - Projekt löschen

Prozessdaten

- Kein vorhanden

Funktionalität

- Detailierte Übersicht, Analysen
- Suchen, Filtern

Angrenzende Systeme

- Keine, ggf. in Zukunft git

Sonstiges

- Auf Folien auch algemeinerer Ablauf zusammengefasst

Relevante Dokumente

Abnahme SVP

Übersicht

Projekt: Projekt Episko
Inkrement: 8
Datum, Ort: 17.02.2025, Zoom-Meeting
Teilnehmer: Simon Blum, Ben Oeckl, Paul Stoeckle, Max Rodler, Sascha Wanninger
Moderation: Sascha Wanninger
Diskussionspunkte:

- Vorstellung und Abnahme des SVP

Ergebnisse:

- SVP abgenommen

Aktionen:

Aktion	Verantwortlich	Deadline
—		

Relevante Dokumente

- SoftwareVerificationPlan.md

Abnahme Grobdesign

Übersicht

Projekt: Projekt Episko
Inkrement: 4
Arbeitspaket: Grobdesign
Datum, Ort: 06.12.2024, DHBW-Friedrichshafen
Teilnehmer: Simon Blum, Ben Oeckl, Paul Stoeckle, Max Rodler
Moderation: Sascha Wanninger
Diskussionspunkte:

- Vorstellung und Abnahme Gorbdesign
- Vorstellung Änderungen Anforderungsaanalyse

Ergebnisse:

- Neue Anforderungsanalyse Abgenommen
- Actor im Sequenzdiagramm "runter zeihen" an die Stelle wo er erzeugt wird
- Diagramm U2.2 berichtigen
- Grobdesign abgenommen

Aktionen:

Aktion	Verantwortlich	Deadline
—		

Relevante Dokumente

-

Abnahme finaler Zwischenstand

Übersicht

Projekt: Projekt Episko
Inkrement: 9
Datum, Ort:
Teilnehmer: Simon Blum, Ben Oeckl, Paul Stoeckle, Max Rodler, Sascha Wanninger
Moderation: Sascha Wanninger
Diskussionspunkte:

- Besprechen des aktuellen Standes
- Plan bis zur Abgabe
- Aufbau des Inhaltsverzeichnis

Ergebnisse:

- Keine Probleme bis zur Abgabe zu erwarten
- Abschlussbericht & Reflektion des ganzen Projekts in SQAR
- Inhaltsverzeichnis für Abgabe nach folgendem Muster:
  - 00-Inhaltsverzeichnis
  - 01-Glossar
  - 02-Projektskizze
  - 03-Entwicklungsplan
  - 04-Anforderungsanalyse
  - 05-Grobdesign
  - 06-Aufwandsschätzung
  - 07-Software-Detailed-Design
  - 08-Software-Verification-Plan
  - 09-Software-Quality-Assurance
    - Plan
    - Report (Abschlussbericht+Reflektion)
  - 10-Inkrement-Dokumente
    - Initialisierung
    - Arbeitspakete
      - Anforderungsbewertung
      - Designpaper
    - Review
  - 11-Meeting-Dokumente

Aktionen:

Aktion	Verantwortlich	Deadline
Dokumente vollenden & aufbereiten	Alle	03.04.2025

Entwicklungsplan Meeting

Übersicht

Projekt: Projekt Episko
Inkrement: 0
Arbeitspaket: Projektinitialisierung
Datum, Ort: 11.10.2024, DHBW Friedrichshafen
Teilnehmer: Simon Blum, Ben Oeckl, Paul Stoeckle, Max Rodler
Moderation: Max Rodler
Diskussionspunkte:

- Genauer Aufbau der Vorgehensweise
- Aufgabenverteilung
- Struktur UML-Diagramms/Vorgehensmodell

Ergebnisse:

- Inhaltliche Fertigstellung des Entwicklunsplans

Aktionen:

Aktion	Verantwortlich	Deadline
Diagramme formalisieren	Simon Blum	11.10.2024
Formales dokument erstellen	Max Rodler	18.10.2024
Link zu Dokument reparieren	Simon Blum	21.11.2024

Relevante Dokumente

-

Meeting Aufwandsschätzung

Übersicht

Projekt: Projekt Episko
Inkrement: 5
Arbeitspaket:
Datum, Ort: 10.01.2025, DHBW Friedrichshafen
Teilnehmer: Ben Oeckl, Paul Stöckle, Max Rodler
Moderation: Paul Stöckle
Diskussionspunkte:

- Aufwand des Projektes für Q1 2025 schätzen
- Function-Point Analyse durchführen und bewerten

Ergebnisse:

- Funktion-Point Analyse Abgeschlossen
- Ergebnis (benötigte Stunden nach Studie) in Frage gestellt

Aktionen:

Aktion	Verantwortlich	Deadline
—		

Relevante Dokumente

Aufwandsschätzung.md

Software Verification Plan

Übersicht

Projekt: Projekt Episko
Inkrement: 8
Autor: Ben Oeckl
Datum: 14.02.2025
Zuletzt geändert:
von: Simon Blum
am: 30.03.2025
Version: 1
Prüfer: Paul Stöckle
Letzte Freigabe:
durch: Paul Stöckle
am: 02.04.2025

Changelog

Datum	Verfasser	Kurzbeschreibung
14.02.2025	Ben Oeckl	Initiales Erstellen und Verfassen
30.03.2025	Simon Blum	Anpassung an Projekt

Distribution List

- Simon Blum simon21.blum@gmail.com
- Ben Oeckl ben@oeckl.com
- Maximilian Rodler maximilianreinerrodler@gmail.com
- Paul Stöckle paul.stoeckle@t-online.de

Software Verification Plan

Einleitung und Zielsetzung

Die Verifikation der Software ist notwendig, um eine zuverlässige Funktion in allen Anforderungsbereichen Sicherzustellen und Fehler zu minimieren um Kosten (in Form von Aufwand) gering zu halten.

Im Kontext des Projektes existieren keine kritischen sicherheitsbedingten Anforderungen. Primär geht es beim Testen und Verifizieren dabei, eine reibungslose und zuverlässige Benutzerfahrung zu garantieren.

Anwendungsbereich

Getestet werden sollen alle Funktionen der Anwendung. Darunter fällt die Erstellung, Betrachtung und Verwaltung von Softwareprojekten anhand eigens dafür eingeführten Manifest-Dateien.

Verifikationsstrategie

Zur Verifikation der Software wird auf verschiedenen Ansätzen gebaut.

Unittests Um das Verhalten einzelner Funktionen im Code zu überprüfen werden Unittests verfasst. Vorhandene Unittests werden im Rahmen unserer CI/CD Pipeline automatisch ausgeführt und verhindern somit das Hinzufügen von fehlerhaftem Code auf dem `main` Branch der Anwendung.

Test Umgebungen: - *(automatisiert)* GitHub Actions Umgebung | `Ubuntu 24.04` Betriebssystem - *(manuell)* x86_64 Rechner | NixOS / Debian / Mint

Ziel Metriken: - **C0** (Region) Coverage: **60%** - **C1** (Branch) Coverage: **60%**

Die genannten Prozentwerte wurden so gewählt, dass eine flächendeckende Verifikation der Anwendung sichergestellt werden kann, sie aber im Kontext des Zeitrahmens trotzdem erreichbar sind.

Hierbei ist zudem anzumerken, dass Metriken für Unittests sich primär auf das Backend, somit den Rust Code beziehen, da sich hier die gesamte Logik der Anwendung befindet.

Automatisierte Last- und Integrationstests Um das Gesamtverhalten der Anwendung, beziehungsweise der einzelnen Module zu verifizieren werden Last- und Integrationstests erstellt.

Testfall-Definition: Im Rahmen des Testes sollen 2000 Metadaten Objekte erstellt, ihre Existenz verifiziert und diese abschließend wieder gelesen werden.

Umgebungen: Durch die modulare Architektur, welche das Backends in mehrere Crates aufteilt gibt es 3 verschiedene Umgebungen wo dieser Test durchgeführt wird.

1. Als Integrationstests im Bezug auf `episko_lib`

- Hierbei wird in dem Test direkt die Bibliothek des Projektes verwendet, jedoch wird dieser als seperate Crate kompiliert und ausgeführt

2. Als Integrationstest im Bezug auf `episko_gui_backend`

- Hierbei werden in dem Test die Schnittstellen für das Frontend verwendet. Somit stellt dieser einen Platzhalter für jenes dar.

3. Seperat von jeglichem Rust Code durch Aufruf der `episko_cli`

- Hierbei wird durch ein seperates Programm das Command Line Interface wiederholt aufgerufen um geforderten Projekte zu erstellen.

Auch diese Tests werden im Zuge der CI/CD Pipeline automatisert ausgeführt.

Manuelle Last- und Integrationstests Um das Verhalten der GUI Anwendung im oben gennanten Testfall zu prüfen, wird die geforderte Anzahl an Projekten generiert.

Folgend wird geprüft, wie die Anwendung sich in diversen Situation verhält. Diese beinhalten:

- Start der Anwendung
- Ansicht Gesamtprojektübersicht
- Ansicht Projektdetails
- Ansicht Statistiken
- Erstellen neuer Projekte
- Bearbeiten existierender Projekte

Hierbei ist jedoch zu beachten, dass aufgrund systemspezifischer Leistungsunterschiede nur subjektive Urteile getroffen werden können und dass zu ziehende Folgen auch aufgrund der anomal hohen Anzahl vorhandener Projekte jeweils seperat abgewogen werden müssen.

Rollen & Verantwortlichkeiten

- Unit-Tests werden von den Entwicklern des betroffenen Codes erstellt.
- Automatiserte Last- und Integrationstests werden von der zu Beginn hierfür bestimmtem Person erstellt.
- Manuelle Tests werden vor dem `Mergen` eines Branches von für diesen verantwortlichen Person durchgeführt.

Zeitplan & Meilensteine

- Unit-Test werden während den Entwicklungsphasen erstellt.
- Unit-Tests werden beim Bauen automatisch ausgeführt (CI-Pipeline)
- Oberflächentests werden während der Entwicklung und an der fertigen Anwendung durchgeführt.
- Lasttests werden durchgeführt wenn alle, zu einem Ablauf gehörenden, Funktionen implementiert sind.

Dokumentation & Nachverfolgbarkeit

- Auffälligkeiten bei Oberflächentests werden bei Reviews dokumentiert.
- Unit-Tests werden durch die Implementierung selbst Dokumentiert.
- Die Ergebnisse von Lasttests werden Dokumentiert.

Testdurchführung

Anbei finden sich Anleitungen zum ausführen einzelner Testgruppen.

Unittests und automatisierte Last-/Integrationstests Umgebung 1+2
Sowohl Unittests als auch die automatiserten Lasttests definiert für Umgebung 1 und 2 können gemeinsam über Rust/Cargo ausgeführt werden.

Hierbei wird die Verwendung von `cargo-nextest` empfohlen, jedoch kann auch `cargo test` verwendet werden. Hierbei werden einige Tests der `episko_cli` hier übersprungen. Mehr Infos diesbezüglich sind an den relevanten Stellen im Quellcode dokumentiert.

```
# Im root Verzeichnis des Projektes
cargo nextest run --workspace
# Alternativ
cargo test --workspace
```

Generierung Coverage Report Zur Einsicht von C0 und C1 Coverage kann mithilfe von `cargo-llvm-cov` ein Bericht erstellt werden.

```
# Generierung eines Berichts mit Konsolenausgabe
cargo +nightly llvm-cov --all-features --workspace --branch

# (Alternativ) Generierung eines html basierten Berichts
cargo +nightly llvm-cov --all-features --workspace --branch --html

# Öffnen des Berichts (beispielhaft mit Firefox)
firefox target/llvm-cov/html/index.html
```

Automatisierter Last-/Integrationstest Umgebung 3 Der dritte automatisierte Test der Kategorie verwendet die Programmiersprache `go` zur Generation zufälliger Daten und Aufrufen der `episko_cli` Anwendung.

```
go run util/data-generation/main.go -test -count 2000 #INOP
```

Manuelle Last-/Integrationstest Zur Durchführung der manuellen Tests kann die gewünchte Anzahl der Daten auch mithilfe des zuvor genannten `go` Programms generiert werden. Diese werden automatich zum Cache hinzugefügt und mittels der Config bei Start automatisch geladen.

```
go run util/data-generation/main.go -base util/test-data -count 2000
```