Technische Dokumentation des Projekts „Taskitory“

Programmentwurf für die Vorlesung Software Engineering II

für die Prüfung zum

Bachelor of Science

des Studiengangs Informatik

Studienrichtung Angewandte Informatik

an der

Dualen Hochschule Baden-Württemberg Karlsruhe

von

Fabian Schwickert

16. Mai 2022

Matrikelnummer 4439027

Kurs TINF19 B4

Ausbildungsfirma Siemens AG, Karlsruhe

Gutachter der Studienakademie Mirko Dostmann

Inhaltsverzeichnis

[Inhaltsverzeichnis II](#_Toc97828054)

[Abbildungsverzeichnis III](#_Toc97828055)

[Tabellenverzeichnis IV](#_Toc97828056)

[Abkürzungsverzeichnis V](#_Toc97828057)

[1 Problemstellung, Ziel und Aufbau 1](#_Toc97828058)

[2 Anforderungsanalyse 1](#_Toc97828059)

[3 Domain Driven Design 8](#_Toc97828060)

[3.1 Ubiquitous Language 8](#_Toc97828061)

[3.2 Taktische Muster des Domain Driven Design 9](#_Toc97828062)

[4 Clean Architecture 9](#_Toc97828063)

[5 Entwurfsmuster 9](#_Toc97828064)

[6 Programming Principles 9](#_Toc97828065)

[7 Testing 9](#_Toc97828066)

[7.1 Unit Testing 9](#_Toc97828067)

[7.2 Mocks 9](#_Toc97828068)

[8 Refactoring 9](#_Toc97828069)

[Literaturverzeichnis XI](#_Toc97828070)

Abbildungsverzeichnis

Es konnten keine Einträge für ein Abbildungsverzeichnis gefunden werden.

Tabellenverzeichnis

Es konnten keine Einträge für ein Abbildungsverzeichnis gefunden werden.

Abkürzungsverzeichnis

DTO Datentransferobjekt

# Problemstellung, Ziel und Aufbau

Es gibt eine Reihe von Aufgabenverwaltungs-Systemen wie z. B. Jira, Youtrack oder Mi­cro­soft Planner, die mit einem Kanban Board arbeiten. Die hier dokumentierte Klau­sur­er­­­satz­leis­tung besteht in der Entwicklung einer App­li­ka­tion, die eine solche Auf­ga­ben­ver­wal­­­tung mo­delliert. Es wird kei­ne graphische Benutzeroberfläche (GUI) entwickelt und le­­diglich die not­wendige An­wen­dungs­logik pro­grammiert und über einen REST-Service (Re­­pre­sentational State Transfer) verfügbar ge­macht. Für die Implementierung des Rest-Ser­vices wird die Pro­gram­miersprache Java mit dem Package Manager „Ma­­ven“ und dem „Spring-Boot-Framework“ verwendet.

Mit der Anwendung soll es möglich sein, die Aufgaben in einem Projekt gemeinsam in ei­nem Projekt-Team zu verwalten. Dazu müssen Benutzer Accounts anlegen, die in den Kon­­­text gemeinsamer Projekte gebracht werden. Das Identity- und Accessmanagement (IAM) soll dabei durch einen Keycloak umgesetzt werden. Ei­ne persistente Da­ten­spei­che­rung wird mit einer Postgresql-Datenbank (DB) um­ge­setzt. In der DB werden die Daten des Keycloaks und ein eigenes DB-Modell gehalten. Die benannten Komponenten (Key­cloak, DB und REST-Service) der App­likation sollen do­cke­risiert werden und mit „Docker-Com­pose“ konfiguriert und be­trie­ben werden.

Um die Bedienung des REST-Services zu vereinfachen, wird eine Sammlung von Anfragen und Konfigurationen für einen HTTP-Client zur Verfügung gestellt. Dazu wird eine „Collection“ für Anwendung Postman zur Verfügung gestellt.

Alle Inhalte der hier dokumentierten Klausurersatzleistung sind auf GitHub unter dem Link „https://github.com/Krayaty/Taskitory“ erreichbar.

# Anforderungsanalyse

Die Anforderungen an die Applikation, die im Rahmen dieser Klausurersatzleistung ent­wi­ckelt wird, sind weniger umfangreich als bei Referenz-Applikationen wie z. B. Youtrack oder Jira. Allerdings orientiert sich die die eigene Applikation an solchen Referenz-App­li­ka­­tionen. Nachfolgend werden allgemeine funktionale Anforderungen definiert und zen­tra­le Use-Cases festgehalten, die implementiert werden sollen. Nicht-Funktionale An­for­de­rungen werden nur dann betrachtet, wenn entsprechende Probleme auftreten.

**Projekte und Benutzer**

Die Projekt-Entitäten sind der Haupt-Bezugspunkt für die anderen Entitäten. Benutzer sind ei­nem oder mehreren Projekten zugeordnet und können über eine Projekt-Entität auf die Res­sourcen eines Projekts („Kanban Boards“, „Aufgaben“ und „Tags“) zugreifen. Dabei kön­nen User die Rollen eines einfachen „Users“ oder eines Projekt-Administrators („Ad­min“) innehaben. Es muss zu jeder Zeit mindestens einen Projekt-Administrator ge­ben. Es gibt keine Obergrenze für die Anzahl von Projekt-Administratoren. Die Rollen be­schrän­­ken den Zu­griff auf die Ressourcen eines Projekts. x Zudem sollen einfache User keine be­ste­hen­den User entfernen oder neue hinzufügen können. Einfache User können Datensätzen der an­deren Entitäten hinzufügen, verändern und löschen. Projekt-Administratoren sollen al­le Zu­griffs-Rechte auf Ressourcen und Funktionen eines Projekts haben. Das schließt das Hin­zufügen, Befördern und Entfernen von Benutzern ein.

Über die Zeit sollen Benutzer ein Projekt verlassen und wieder beitreten können. Es ist nicht mög­lich einem Projekt, ohne die Zustimmung des Projekt-Teams beizutreten. Dafür wird ein geheimer Projekt-Schlüs­­sel verwendet, der nur von Projekt-Administratoren einsehbar ist.

1. Benutzer-Konto registrieren

Benutzer müssen ein Konto für die Applikation besitzen und einem Projekt zu­ge­ordnet sein, um die zugehörigen Informationen einsehen zu können. Ein Be­nut­zer-Konto soll durch einen Username identifizierbar sein und mit einem Passwort au­thentifiziert wer­den. Weitere Benutzerdaten sind für diese Applikation nicht relevant. Damit ein User sich bei der Applikation registrieren kann muss er per HTTP mit dem Keycloak kom­mu­ni­zie­ren. Mit einer HTTP-Anfrage sollen Username und Passwort übermittelt werden. Diese An­frage steht in der Postman-Collection zur Verfügung. Bei erfolgreicher Registrierung soll der Keycloak den Benutzer direkt anmelden und einen Access-Token zurückgeben. Bei einem Fehler wird ein passender Fehlercode zurückgegeben.

Die­se Funktion wird bereits von dem Keycloak-Server zur Verfügung gestellt und kann da­her nicht vollständig konfiguriert werden.

1. Benutzer-Konto anmelden

Benutzer müssen sich gegenüber dem Keycloak identifizieren und authentifizieren, um Zu­griff auf den REST-Service der Applikation zu erhalten. Der Anmelde-Vorgang wird über HTTP-kommunikation mit dem Keycloak abgewickelt. Diese Funktion wird bereits von dem Keycloak-Server zur Verfügung gestellt und kann daher nicht vollständig kon­fi­gu­riert werden.

Die Anmeldung mit einem bestimmten Benutzer-Konto wird durch die Postman-Col­lec­tion zur Verfügung gestellt werden. Es soll mit jeder Anfrage an den REST-Service der Ap­plikation geprüft werden, ob ein gültiger Access-Token vorhanden ist. Wenn das nicht der Fall ist, soll ein neuer Access-Token für das gerade verwendete Konto angefordert wer­den ohne, dass der Benutzer etwas tun muss. Die einzige Aktion des Benutzers ist die Wahl eines Benutzer-Accounts.

1. Benutzer-Konto löschen

Wenn ein Benutzer sich entscheidet, sein Konto zu löschen, werden alle damit verbundenen Daten unwiederbringlich gelöscht. Dazu muss per HTTP mit dem Keycloak kommuniziert werden. Das Ergebnis des Löschvorgangs wird durch einen entsprechenden Status-Code quittiert. Diese Funktion wird bereits von dem Keycloak-Ser­ver zur Verfügung gestellt und kann daher nicht vollständig kon­fi­gu­riert werden.

1. Projekt anlegen

Wenn ein Benutzer ein Konto registriert hat, ist er noch keinem Projekt zugeordnet. Um Funktionen der Applikation nutzen zu können, muss ein User mindestens einem Projekt zugeordnet sein. Denn ein Projekt ist der gemeinsame Kontext für alle anderen Entitäten. Aufgaben, Kanban Boards oder Tags können nicht ohne Projekt existieren.

Ein Projekt wird von einem Benutzer mit einer Be­zeichnung und Beschreibung angelegt. Der geheime Gruppen-Schlüssel wird automatisch generiert. Optional kann der Benutzer direkt ein Projekt-Team aus Benutzern zusammenstellen. Dabei kann er entscheiden, wel­che Rolle ein Benutzer-Konto erhält. Der Ersteller eines Projekts wird automatisch als Ad­mi­nistrator dem Projekt zugeordnet. Ein Projekt hat nach der Initialisierung ein leeres Back­log, keine Tags und keine Kanban Boards.

1. Benutzer-Konto zu einem Projekt hinzufügen

Das Projekt-Team kann auch nach der Initialisierung erweitert werden. Dazu muss ein Benutzer den geheimen Schlüssel eines Projekts eingeben. Diesen geheimen Schlüssel können nur Projekt-Administratoren einsehen. Ein Administrator kann einem Benutzer eine Nachricht schicken, die diesen Projekt-Schlüssel enthält oder den Projekt-Schlüssel auf andere Weise übermitteln. Wenn ein Benutzer über den Schlüssel einem Projekt beigetreten ist, werden alle Projekt-Mitglieder und auch der neu hinzugefügt Benutzer über eine Nachricht informiert.

1. Ein Benutzer-Konto befördern

Projekt-Administratoren sollen einfache User zu Projekt-Administratoren befördern können. Der beförderte Benutzer und alle Mitglieder eines Projekt-Teams sollen darüber informiert werden.

1. Benutzer-Konto aus Projekt-Team entfernen

Ein Benutzer kann aus eigenem Antrieb aus einem Projekt austreten. Ein Benutzer mit der Rolle „User“ kann auch gegen seinen Willen von einem Projekt-Administrator aus einem Projekt entfernt werden. Dadurch werden die Referenzen auf den Benutzer aus dem Projekt entfernt. Das betrifft nur die Aufgaben in einem Projekt, die dem Benutzer zugeteilt waren. Das entsprechende Attribut eines Aufgabe-Datensatzes wird dann auf „NULL“ gesetzt. Der entfernte Benutzer soll per Nachricht informiert werden, wenn ein Administrator ihn aus einem Projekt entfernt hat.

1. Projekt-Attribute anpassen

Die Bezeichnung und die Beschreibung eines Projekts sollen von einem Projekt-Ad­mi­nis­tra­tor geändert werden können. Benutzer mit der Rolle „User“ können diese At­tri­bute nicht ändern. Der geheime Schlüssel eines Projekts soll nicht geändert werden kön­nen.

**Aufgaben**

Eine zentrale Funktion der Applikation ist die Erstellung und Verwaltung von Aufgaben. Die Aufgaben-En­­tität ist dem entsprechend wichtig. Die Aufgaben werden neben Entitäs-At­­­tri­bu­ten auch mit Meta-Informationen in „Tags“ be­schrie­ben. Aufgaben haben die At­tri­bute „Be­zeich­nung“, „Beschreibung“, „Erstellungs-“ und „Fertigstellungs-Zeit­punkt“, „Kom­­ple­xi­tät“, „Bearbeitungs-Status“ und „zuständige Person“ besitzen. Die Bezeichnung einer Aufgabe ist in einem Projekt eindeutig. Es sollen Aufgaben-Kompositionen und Abhängigkeiten zwischen Aufgaben mög­lich sein. Auf­gaben sind nach der Erzeugung automatisch in dem „Backlog“ eines Pro­jekts ein­sehbar, von wo sie einem „Kanban Board“ hinzugefügt werden können.

1. Aufgaben anlegen

Aufgaben sind immer einem Projekt zugeordnet. Aufgaben können daher nur angelegt werden, wenn ein Benutzer mindestens einem Projekt angehört. Im Stan­dard­fall wird eine neue Aufgabe zu dem Backlog eines Pro­jekts hinzugefügt. Wenn bereits ein Kanban Board erstellt wurde, kann eine Aufgabe auch direkt dem Kanban Board hinzugefügt wer­den. Dort wird die Aufgabe in der Spalte „Todo“ abgelegt. In diesem Fall wird die Auf­ga­be nicht mehr Im Backlog angezeigt.

1. Benutzern Aufgaben zuweisen

Eine bestehende Aufgabe kann einem Mitglied des Projekt-Teams zur Bearbeitung zu­ge­wie­sen werden. Dabei ist es egal, ob die Aufgabe vorher bereits einer Person zugewiesen war oder nicht. Es ist auch möglich die Zuweisung aufzuheben und keine neue Zuweisung vor­zunehmen („NULL“). Diese Funktion ist für jeden Benutzer in einem Projekt zu­gäng­lich. Der zu­gewiesene Benutzer wird über eine Nachricht informiert.

1. Aufgaben zu einem Kanban Board hinzufügen

Aufgaben können aus dem Backlog zu einem (1) Kanban Board hinzugefügt werden. Dazu muss eines aus vielen möglichen Kanban Boards eines Projekts gewählt werden. Außerdem muss festgelegt werden, in welcher Spalte die Aufgabe abgelegt werden soll. Der Standardfall ist, dass die Aufgabe in der Spalte „Todo“ abgelegt wird.

1. Aufgaben bearbeiten

Die Attribute „Be­zeich­nung“, „Beschreibung“, „Komplexität“ und „zuständige Person“ (siehe oben) kön­nen nach der Erstellung einer Aufgabe jederzeit durch einen Benutzer verändert werden. Es bleibt zu beachten, dass die Bezeichnung der Aufgaben in einem Pro­jekt eindeutig ist.

1. Aufgaben löschen

Aufgaben können von allen Benutzern gelöscht werden. Dabei spielt es keine Rolle, ob die Aufgabe auf einem Kanban Board oder dem Backlog liegt.

1. Unter-Aufgaben erstellen

Aufgaben können aus einer Sammlung von Unteraufgaben bestehen. Um diesen Zusammenhang mit der Applikation „Taskitory“ abzubilden, ist es jederzeit möglich einer bereits bestehenden Aufgabe bereits bestehende Unter-Aufgaben zuzuweisen. Außerdem können neue Aufgaben bei der Erzeugung als Unter-Aufgabe einer bereits bestehenden Aufgabe definiert werden.

1. Unter-Aufgaben löschen

Die Unter-Aufgaben einer übergeordneten Aufgabe können aus diesem Verhältnis ent­fernt werden. Die Unter-Aufgaben sollen dabei nicht automatisch gelöscht werden, son­dern werden in dem gleichen Kontext (Kanbanboard 🡪 Spalte, Backlog) der über­ge­ord­ne­ten Aufgabe abgelegt.

1. Abhängigkeiten von Aufgaben erstellen

Aufgaben können von anderen Aufgaben abhängen. Das meint, dass eine Bearbeitung ei­ner Aufgabe nicht möglich ist, bis die Abhängigkeit erledigt wurde. Übergeordnete Auf­ga­ben sind automatisch abhängig von ihren Unter-Aufgaben und können nicht erneut mit ihren Unter-Aufgaben in Beziehung gesetzt werden. Alle anderen Abhängigkeiten zwi­schen bereits bestehenden Aufgaben können festgelegt werden.

1. Abhängigkeiten von Aufgaben löschen

Die Abhängigkeiten zwischen Aufgaben können wieder gelöscht werden. Die Ab­hän­gig­keit (Aufgabe) soll dabei nicht automatisch gelöscht werden, son­dern wird in dem glei­chen Kontext (Kanbanboard 🡪 Spalte, Backlog) der abhängigen Aufgabe ab­ge­legt.

**Tags**

1. Tag erstellen

Tags können mit einer Bezeichnung und einer Beschreibung erzeugt werden und sind einem Projekt zugeordnet. Die Bezeichnung ist in einem Projekt eindeutig.

1. Tag bearbeiten

Die Bezeichnung und die Beschreibung eines Tags können im Nachhinein geändert wer­den. Dabei bleibt zu beachten, dass die Bezeichnung in einem Projekt eindeutig sein muss.

1. Tag löschen

Ein Tag kann gelöscht werden. Dadurch werden automatisch alle Referenzen von Auf­ga­ben entfernt.

1. Tags zuweisen

Aufgaben können über Tags mit Metadaten wie z. B. Versionsnummern uvm. versehen werden. Die Tags und Aufgaben müssen dabei aus dem gleichen Projekt stammen.

**Kanban Board**

Ein Kanban Board be­trifft einen be­stimm­ten Zeitraum („Sprint“). Nach einem Sprint besteht das Kan­ban Board weiter, kann aber nicht mehr bearbeitet werden. Ein altes Kan­ban Board kann in ein neues Kanban Board überführt werden.

Ein Kanban Board kann im Rah­men dieser Klausurersatzleistung aus drei bis zu fünf Spalten bestehen, die den Status ei­ner Auf­gabe be­schreiben. Die Spalten „ToDo“, „In Progress“, und „Done“ sind ob­li­ga­to­risch. Die Spalten „Review“ und „Testing“ können bei Bedarf hinzugefügt und wieder ent­fernt werden. Dabei sollen dort eingeordnete Aufgaben ihren Bearbeitungs-Status nicht ver­lieren. Die Spalte und die darin enthaltenen Aufgaben werden lediglich ausgeblendet. Auf­gaben können einem Kanban Board hin­zu­­gefügt und auf dem Kanban Board in einen an­deren Bearbeitungs-Status ver­scho­­ben wer­­den.

Über einen Sprint können Statistiken an­­ge­legt werden. Dafür sollen ent­spre­chen­de Kanban Boards als Repräsentation eines Sprints genutzt werden. Es soll z. B. aus­ge­wer­tet werden, welcher Anteil der Aufgaben auf ei­nem Kanban Board in einem be­stimm­ten Bearbeitungs-Status ist.

1. Kanban Board anlegen

Kanban Boards können mit einer für ein Projekt eindeutigen Bezeichnung, einer Beschreibung und einem festgelegten Sprint-Zeitraum erzeugt werden. Optional können noch die zwei Spalten „Review“ und „Testing“ aktiviert werden.

Ein Kanban Board ist bei Er­zeu­gung leer und besitzt die oben benannten Spalten. Es sollen mehrere Kanban Boards für ein Projekt gleichzeitig bestehen können.

1. Attribute eines Kanban Boards bearbeiten

Die Bezeichnung und die Beschreibung eines Kanban Boards können durch die Benutzer des entsprechenden Projekts angepasst werden. Dabei bleibt zu beachten, dass die Bezeichnung in einem Projekt eindeutig sein muss.

Außerdem ist es möglich die Spalten „Review“ und „Testing“ auszublenden. Alle in diesen Spalten befindlichen Aufgaben werden daraufhin ausgeblendet. Allerdings behalten sie den Status und können anschließend wieder eingeblendet werden.

1. Kanban Board löschen

Kanban Boards können durch Benutzer des entsprechenden Projekts gelöscht werden. Die darauf befindlichen Aufgaben werden daraufhin in das Backlog gelegt. Die Aufgaben behalten ihren Bearbeitungs-Status.

1. Aufgabe auf einem Kanban Board verschieben

Die zentrale Funktion eines Kanban Boards ist, Aufgaben über ihren Bearbeitungs-Sta­­tus zu verwalten. Be­nut­zer können die Aufgaben auf einem Kanban Board in einen anderen Be­arbeitungs-Status verschieben. Der Bearbeitungs-Status einer Aufgabe im Backlog ei­nes Projekts kann nicht geändert werden.

1. Kanban Board migrieren

Das Kanban Board eines vergangenen Sprints kann in ein neues Kanban Board überführt wer­den. Das ist dann interessant, wenn sich auf dem alten Kanban Board noch nicht ab­­ge­schlos­se­ne Aufgaben befinden. Diese Aufgaben sollen in den gleichen Status des neu­en Kanban Boards über­nommen werden.

1. Statistiken erstellen

Es soll möglich sein, Statistiken über ein bestehendes Kanban Board anzulegen. Mögliche Anwendungsfälle sind z. B. die Verteilung von Aufgaben nach Bearbeitungs-Status oder die Berechnung einer durchschnittlichen Bearbeitungsdauer.

**Nachrichten**

Ein Benutzer erhält Nachrichten über bestimmte Änderungen in den Projekten, denen er an­gehört. Diese Nachrichten haben einen Absender, einen Adressaten und einen Inhalt. Zu­dem wird über ein Feld festgehalten, ob die Nachricht vom Benutzer abgerufen wur­de. Nachrichten sind nicht dazu gedacht, um Konversation abzuhalten.

1. Nachrichten empfangen

Benutzer können die erhaltenen Nachrichten empfangen und lesen. Dabei ist es möglich Nachrichten einzeln und in größeren Gruppen zu empfangen. Wurde eine Nachricht einmal empfangen, wird sie unwiederbringlich gelöscht.

# Clean Architecture

## Grundlagen

Unter „Clean Architecture“ versteht man einen Architektur-Ansatz für Software-Applikationen. Die Clean Architecture fußt auf der Idee, dass Applikationen in mehrere Schichten aufgeteilt werden können, die eigene Aufgabenbereiche abdecken. Wie bei einer Zwiebel umschließt eine Schicht die andere und nur die äußerste Schicht ist für Benutzer der Applikation zugänglich. Die Funktionen der inneren Schichten sind über definierte Schnittstellen für die jeweils folgende Schicht zugänglich. Die äußeren sind dadurch von den inneren Schichten abhängig; umgekehrt sind die inneren jedoch nicht von den äußeren Schichten abhängig. Daher wird es schwerer den Code auf einer Schicht zu ändern, je weiter innen sie liegt. Die Clean Architecture ermöglicht durch den be­schrie­be­nen modularen Aufbau, dass die für eine Domäne und Applikation spezifische Ge­schäfts­logik nicht geändert werden muss, wenn konkrete technische Implementierungen ei­ner Applikation wie geändert werden müssen. Auf diese Weise kann ein Applikation lang­fristig mit wenig Aufwand gewartet und angepasst werden.

Im Allgemeinen werden die folgenden fünf Schichten der Clean Architecture definiert (von innen nach außen):

1. Abstraktions-Schicht:

Die Abstraction-Schicht ist der Kern einer Applikation. In dieser Schicht wird sog. do­mä­nen­­übergreifendes Wissen implementiert. Das meint Algorithmen und Konzepte, die nicht an die Domäne einer Applikation gebunden sind. Zudem werden hier Software-Werk­­zeuge und -Bibliotheken ein­gebunden, die auf allen folgenden Schichten verwendet wer­­den und technologische Standards umsetzen. Eine Applikation muss keine Ab­strak­tions-Schicht besitzen.

1. Domänen-Schicht:

Die Domänen-Schicht umschließt die Abstraktions-Schicht und hängt daher von der ihr ab. In der Domänen-Schicht wird die Fachlichkeit einer Applikation implementiert. Hier wird die übergreifende Geschäftslogik implementiert, die für die Abbildung einer Domäne in vielen verschiedenen App­li­kation benötigt wird. Die Domänen-Schicht wird durch das Domain Driven Desing (siehe Ka­pitel 4) modelliert.

1. Applikations-Schicht:

Die Applikations-Schicht folgt direkt auf die Domänen-Schicht und hängt daher von ihr ab. In der Applikations-Schicht werden die Use Cases einer Applikation abgebildet. In diesem Teil einer Applikation werden die für den Benutzer erfahrbaren Funktionen implementiert. Das umfasst auch die Implementierung der Applikations-spezifischen Geschäftslogik einer Domäne.

1. Adapter-Schicht:

Die Adapter-Schicht baut auf der Applikations-Schicht auf und hängt daher von ihr ab. Der Zweck der Adapter-Schicht ist es, Benutzerinteraktionen an die Applikations-Schicht wie­ter­zugeben. Dabei dient die Adapter-Schicht als „Anti Corruption Layer“, der Domänen- und Applikations-spezifischen Quellcode von festen technischen Implementierungen ab­kop­pelt. Das meint auch die Konvertierung der Daten aus eingehenden Anfragen in ein für die darunter liegenden Schichten verständliches Format.

1. Plugin-Schicht:

Die Plugin-Schicht liegt ganz außen und hängt von der darunter liegenden Adapter-Schicht ab. Auf der Plugin-Schicht werden technische Implementierungen wie z. B. das Ein­binden einer Datenbank oder eines IAM-Services vorgenommen. Außerdem werden auf dieser Schicht die Funktionen einer Applikation den Benutzern zugänglich gemacht.

## Implementierung in der Applikation „Taskitory“

Die Applikation Taskitory verwendet das Konzept der Clean Architecture mit allen oben be­schriebenen Schichten. Um die Eigenschaft der nach innen gerichteten Abhängigkeiten in der Applikation Taskitory sicherzustellen, wird für jede Schicht ein eigenes Maven- Mo­dul angelegt. Die Maven-Module werden wie oben beschrieben in Form von Ab­hän­gig­kei­ten miteinander verbunden. Nachfolgend soll auf alle Schichten und deren Aufgabenfelder ein­gegangen werden. Dadurch wird die Existenz der jeweiligen Schicht gerechtfertigt.

1. Abstraktions-Schicht:

Die Abstraction-Schicht der Applikation Taskitory enthält keinen eigenen Quellcode. Lediglich die folgenden Abhängigkeiten werden auf dieser Schicht eingebunden:

* Java Persistence API (JPA)
* Spring-Context
* Lombok

Die „Java Persistence API“ (JPA) ist ein de fakto Standard für die Integration einer Da­ten­bank in eine Java-Applikation. Da die JPA eine weit verbreitete, beständige Software-Bib­lio­thek ist, ist es nicht notwendig sie als Teil der Plugin-Schicht einzubinden. Denn das wür­­de umfassende Implementierungen auf der Adapter-Schicht nach sich ziehen. Die JPA hat keinen Bezug zu der Domäne der Applikation Taskitory und kann als Standard-tech­nik angesehen werden. Daher gehört diese Abhängigkeit auf die Abstraktions-Schicht.

Die Abhängigkeiten „Spring-Context“ und „Lombok“ sind zwei Software-Bibliotheken, de­ren Funktionen allen anderen Schichten zur Verfügung stehen soll. Mit Spring-Context wer­den Funktionen zur Verfügung gestellt, die das Betreiben einer eigenen Applikation mit dem Spring-Framework ermöglichen. Hingegen Lombok ist eine Software-Bibliothek mit der Quellcode vereinfacht werden kann. Lombok ermöglicht es, umfangreiche Stan­dard-Konstrukte in Java wie z. B. Getter und Setter durch eine Annotation zu ersetzen. Die Funk­tionen der beiden Software-Bibliotheken Spring-Context und Lombok beziehen sich auf allgemeine Programmier-Konzepte und Konstrukte der Programmiersprache Java. Da­her gehören beide Abhängigkeiten auf die Abstraktions-Schicht.

1. Domänen-Schicht:

Die Domänen-Schicht ist das Herzstück der Applikation Taskitory. Hier wird die grund­le­gen­de Geschäftslogik der Domäne definiert. Es wird ein statisches Modell fachlicher Ak­teu­re aufgestellt. Darin werden die fachlichen Akteure definiert und wie diese in­ter­agie­ren. Das umfasst besonders die Entitäten „User“, „Projekt“, „Projekt-Mitgliedschaft“, „Auf­ga­be“, „Kanbanboard“, „Tag“, Nachricht und noch weitere fachlich relevanten Elemente. In Kapitel 4 wird die Domänen-Schicht näher beschrieben.

1. Applikations-Schicht:

Auf Grundlage des statischen Modells der Akteure aus der Domänen-Schicht wird in der App­likations-Schicht ein dynamisches Modell der Abläufe aufgestellt. Hier werden die Use Ca­ses umgesetzt. Dabei wird der Informationsfluss von Akteur zu Akteur unter Be­rück­sich­tigung der Geschäftslogik aus der Domänen-Schicht abgebildet. Die Use Cases werden selbst durch Applikations-spezifische Geschäftslogik gelenkt. Die wichtigsten Use Cases der Applikation Taskitory sind in Kapitel 2 beschrieben.

1. Adapter-Schicht:

Die Applikation Taskitory benutzt für Benutzerinteraktionen eine REST-Schnittstelle. Es gibt kein Frontend, das die Validierung von Anfrage-Parametern übernehmen könnte. Da­her müssen die eingehenden HTTP-Anfragen in dem Backend validiert und unter Um­stän­den umgewandelt werden, bevor sie an die inneren Schichten der Applikation weiter­ge­ge­ben werden. Für diesen Zweck ist die Adapter-Schicht gedacht. Hier werden Da­ten­trans­ferobjekte (DTO) und die nötige Validierungs-Logik definiert.

1. Plugin-Schicht:

Die Plugin-Schicht der Applikation Taskitory enthält die technischen Implementierungen für die Kommunikation mit einer Postgresql Datenbank, einem Keycloak IAM-Server. Zu­dem werden die Funktionen der Applikation Taskitory auf dieser Schicht den Benutzern über eine REST-Schnittstelle verfügbar gemacht.

# Domain Driven Design

## Problemdomäne

Die Problemdomäne einer Applikation setzt sich aus einer Kerndomäne, mehreren un­ter­stüt­zenden und mehreren ge­nerischen Domänen zusammen. Die Kerndomäne bestimmt die zentralen Funk­tio­na­li­tä­ten und den Zweck einer Applikation. Unterstützenden Do­mä­nen beziehen sich auf Aufgaben-Felder, die nicht direkt Teil der Kerndomäne sind, aber die Kerndomäne mit ihren Funktionen unterstützen. Die Kerndomäne verwendet die un­ter­stützenden Domänen, um die zentralen Funktionen einer Applikation umzusetzen. Die Ge­nerischen Domänen beziehen sich auf Aufgabenfelder, die nicht direkt mit dem Zweck ei­ner Applikation in Verbindung stehen, jedoch für das Betreiben einer Applikation not­wen­dig sind. Die Generischen Domänen liefern Basis-Funktionalitäten, ohne die eine App­li­kation nicht lauffähig ist.

Die Problemdomäne der Applikation „Taskitory“ ist die Aufgaben­ver­waltung in Projekten mit mehreren Projekt-Teilnehmern. Dabei ist die Kerndomäne das Verwalten von Auf­ga­ben. Das Umfasst nicht nur das Erstellen, Verändern und Löschen von Aufgaben, sondern auch die Funktionen eines Kanban Boards.

Die über die Kerndomäne zur Verfügung gestellten Daten und Funktionen werden Be­nut­zern der Applikation über die Domäne IAM zur Verfügung gestellt. Ohne eine Be­nut­zer­ver­waltung könnte das Konzept von Projekt-Teams und damit die ganze Applikation nicht um­gesetzt werden. IAM hat keine inhaltliche Verbindung zu der Kerndomäne und wird daher als generische Domäne eingestuft.

Die Verwaltung von Pro­jekten ist kein zwingend notwendiger Aufgabenbereich für eine Auf­­­gabenverwalten. Allerdings ist dieser Aufgabenbereich bei der Applikation „Tas­ki­tory“ relevant für die Funktionen der Kerndomäne und inhaltlich mit dieser verbunden. Denn alle Objekte, die in der Kerndomäne verwendet, erzeugt oder gelöscht werden, sind ei­­­nem Projekt zugeordnet. Auch die Rolle eines Benutzers in einem Projekt kann in diese Do­­­mäne gezählt werden. Denn der Zugriff auf die Funktionen der Kerndomäne wird durch die Rolle eines Be­­nutzers in einem Projekt eingeschränkt. Auf Grund der in­halt­li­chen Verbindung zur Kerndomäne wird die Verwaltung von Projekten und der damit as­so­ziierten Projekt-Teams als un­ter­stüt­zen­­de Domäne eingestuft.

Das Erstellen von Statistiken ist keine für eine Aufgabenverwaltung notwendige Funktion. Al­­lerdings werden mit Statistiken die Daten von Kanban Boards ver­ar­bei­tet, die aus der Kern­domäne stammen. Es besteht damit eine klare inhaltliche Verbindung zur Kern­do­mä­ne. Das Erstellen von Statistiken ist damit eine unterstützende Domäne.

Die Applikation „Taskitory“ informiert Benutzer mit Nachrichten über bestimmte Er­eig­nis­­se, die für den Benutzer relevant sind. Diese Ereignisse betreffen i. d. R. nicht die Kern­do­­mäne, sondern unterstützende Domänen. Es besteht daher keine inhaltliche Ver­bin­dung zur Kerndomäne. Das Messaging ist eher eine Basisfunktionalität und wird daher als ge­nerische Domäne eingestuft. Abbildung 1 zeigt noch einmal die Problemdomäne der App­likation „Taskitory“ mit allen Bestandteilen.

**Problemdomäne**

IAM

**Generische Domänen**

**Unterstützende Domänen**

Projekt-Verwaltung

**Kerndomäne**

Aufgabenverwaltung

Statistiken

Messaging

Abb. 1: Die Problemdomäne der Applikation „Taskitory“

Um ein Modell wie die Problemdomäne verstehen zu können, muss dieses Mo­dell in ei­nem fachlichen Kontext betrachtet werden. Erst durch den Kontext wird konkretisiert, wie die Applikation benutzt wird und welche An­for­derungen an sie gestellt werden. Im Do­main Driven Design wird der „Bounded Kontext“ als fachlicher Kontext für eine Do­mä­ne definiert.

Die Kerndomäne der Applikation Taskitory muss immer im Kontext eines Projekts ge­se­hen werden. Die Kerndomäne ist hauptsächlich für die Benutzer relevant, die Aufgaben be­arbeiten. Das sind i. d. R. die Benutzer mit der Rolle „User“.

Die Benutzer mit der Rolle „Admin“ können als eine Art Projektleiter angesehen werden. Die­­ser Projektleiter ist unter Umständen weniger an den konkreten Aufgaben und eher an den Statistiken interessiert. Natürlich können Aufgaben auch im Kontext von Pro­jekt­lei­­tern betrachtet werden. Dadurch entsteht allerdings kein Un­­terschied zu der Be­trach­tungs­­weise einfacher User. Projektleiter sind auch für die Ver­wal­tung von Projekten in­klu­­sive Projekt-Team zuständig.

Das Verwalten der eigenen Kontoinformationen betrifft nur den Eigentümer des Kontos selbst. Die generische Domäne IAM steht daher nur im Kontext von Benutzer-Konten. Auch die generische Domäne Messaging betrifft den Benutzer. Das Messaging muss da­her im Kontext eines Benutzerkontos gesehen werden. Allerdings beziehen sich Nach­rich­ten auch immer auf ein Projekt und sind daher von dem Projekt-Kontext betroffen.

Die einzelnen Teildomänen werden in verschiedenen Kontexten betrachtet. Allerdings müs­­sen fast alle Teildomänen im Kontext eines Projekts betrachtet werden. Nur die un­ter­stützende Domäne IAM wird nicht im Kontext eines Projekts betrachtet. Das lässt da­rauf schließen, dass die Applikation Takitory in zwei übergeordneten Bounded Kontexts be­trachtet werden kann. Das IAM muss im Kontext der Benutzerverwaltung und der Rest der Applikation im Bounded Kontext einer Projektarbeit betrachtet werden.

Eine Komposition mehrere fachliche Kontexte mit einem ge­mein­sa­men Be­zugs­punkt wird durch das Konzept vom „Shared Kernel“ umgesetzt. Hier wird eine ge­mein­sa­me Ba­sis für alle Kontexte geschaffen und für jeden Kontext speziell erweitert.

## Ubiquitous Language

Der Begriff „Ubiquitous Language“ bezeichnet eine Sammlung von Begriffen, die ver­wen­det wird, um sich über eine Applikation zu unterhalten. Das umfasst Fachbegriffe der Pro­blem­domäne und der technischen Implementierung. Es müssen alle nicht trivialen Objekte, Konzepte und Regeln eindeutig definiert werden.

Die Ubiquitous Language der Applikation Taskitory umfasst die folgenden Be­griffe für das Beschreiben problemdomänenspezifischer Objekte, Konzepte und Regeln:

1. Projekt

Ein Projekt ist ein zielgerichtetes Vorhaben, das aus einer Menge von Aufgaben besteht, die von einer Gruppe von Benutzern verwaltet und bearbeitet werden. Für die Erfüllung dieser Aufgabenstellen Projekte Kanbanboards und Tags zur Verfügung. Projekte sind der zentrale Kontext, in dem ein Benutzer die Applikation Taskitory verwendet.

Über die Zeit können Benutzer ein Projekt verlassen und wieder beitreten. Für das Ver­las­sen eines Projekts wird keinerlei Zustimmung von Seiten des Projekt-Team benötigt. Al­lerdings kann ein Benutzer einem Projekt nur dann beitreten, wenn er den geheimen Pro­jekt-Schlüs­­sel kennt. Dieser Projekt-Schlüssel kann nur von Projekt-Administratoren ein­­gesehen werden. Der Projekt-Schlüssel kann ausschließlich von Projekt-Ad­mi­nis­tra­to­ren nur auf einen zufällig generierten Wert geändert werden.

Ein Projekt hat eine eindeutige Bezeichnung und eine Beschreibung, die von Pro­jekt-Ad­mi­nistratoren festgelegt und verändert werden kann.

1. Benutzer

Ein Benutzer wird durch ein Benutzer-Konto repräsentiert. Dieses Benutzer-Konto be­steht aus einem Benutzernamen und einem Passwort. Beide Werte können nach der Ini­tia­lisierung durch den Benutzer verändert werden.

Benutzer müssen keinem Projekt angehören, können aber beliebig vielen Projekten bei­tre­­ten. Benutzer können die Applikation Taskitory ohne eine Projektmitgliedschaft nicht sinn­voll nutzen. Daher können Benutzer neue Projekte erstellen, in denen sie automatisch als Projekt-Administrator eingetragen werden.

Es werden der Beginn von Projektmitgliedschaften und die eingenommene Rolle in einem Pro­jekt festgehalten. Benutzer können die Rollen „User“ und „Admin“ (Projekt-Ad­mi­nis­tra­tor) einnehmen.

1. Projekt-Administrator

Projekt-Administratoren sind Benutzer mit der Admin-Rolle. Wie der Name schon sagt, ist diese Eigenschaft projektspezifisch. Benutzer mit der Admin-Rolle haben alle Rechte der User-Rolle und zu­sätzliche Rechte für die Projekt-Verwaltung.

Es muss zu jeder Zeit einen Ad­mi­nis­tra­tor in einem Projekt ge­ben. Wenn der letz­te Ad­mi­nis­trator ein Projekt verlässt, muss er einen neuen Projekt-Administrator aus dem Pro­jekt-Team bestimmen. Wenn es keine anderen Team-Mitglieder gibt, wird das Pro­jekt ge­löscht. Es gibt keine Ober­gren­­ze für die Anzahl von Projekt-Administratoren. Nur Projekt-Ad­­ministratoren können Benutzer ein­la­den, entfernen und deren Rolle än­dern.

1. Einfacher User

Einfache User sind Benutzer mit der User-Rolle. Diese Eigenschaft ist projektspezifisch. Die User-Rolle ermächtigt einen Benutzer Aufgaben, Kanbanboards und Tags in einem Pro­jekt zu verwalten.

1. Aufgabe

a

1. Backlog

a

1. Kanban Board

a

1. Todo (Kanban Board Spalte)

a

1. In Progress (Kanban Board Spalte)

a

1. Review (Kanban Board Spalte)

a

1. Testing (Kanban Board Spalte)

a

1. Done (Kanban Board Spalte)

a

1. Tag

a

1. Unteraufgabe

a

1. Abhängigkeit (Aufgabe)

a

1. Nachricht

a

1. Statistik

A

Die Ubiquitous Language der Applikation Taskitory umfasst die folgenden Be­griffe für das Be­schreiben technischer Objekte, Konzepte und Regeln für die Implementierung der App­li­kation:

1. REST Service (Backend)

a

1. JPA

a

1. Keycloak-Server (IAM)

a

1. Postgresql (Datenbank)

a

1. Docker-Container

a

1. Docker Compose

a

1. Postman Collection

a

## Taktische Muster des Domain Driven Design

Entities

Aggregates

Repositories

Domain Services

Value Objects

# Entwurfsmuster

A

# Programming Principles

A

# Testing

## Unit Testing

A

## Mocks

A

# Refactoring

A

Literaturverzeichnis

1. a