LOCAL ADMIN

Machbarkeitsstudie

VERSION	AUTOR	QS	DATUM	STATUS	KOMMENTAR
1.0	TOBIAS SCHROTTWIESER	NILS BRUGGER	14.10.2019	DRAFT	ERSTER ENTWURF
1.1	TOBIAS SCHROTTWIESER	TOBIAS WEISS	17.10.2019	DRAFT	KAPITEL 2-4
1.2	TOBIAS SCHROTTWIESER	KALIAN DANZER	21.10.2019	DRAFT	VERVOLLSTÄNDIGUNG
2.0	TOBIAS SCHROTTWIESER	NILS BRUGGER	24.10.2019	RELEASE	KORREKTUR UND ERWEITERUNG

Inhaltsverzeichnis

1 Einführung

Die MitarbeiterInnen in den Burger King Filialen brauchen ein Finanzverwaltungssystem, welches elektronisch, die anfallenden Daten verwaltet. Es ist bereits ein elektronisches Hilfsmittel in Betrieb, jedoch hat dieses einige Designfehler, die zu Performanceproblemen führen. Die eingetragenen Daten werden noch nicht in einer Datenbank gespeichert was zu einigen Fehlern, langen Ladezeiten, sowie langsamer Fehlerbehebung führt. Um das generelle Reporting und Verwalten dieser Reporte der Filialen effizienter und übersichtlicher zu gestalten, möchten wir „Local Admin“ entwickeln. Wir möchten mithilfe von „Local Admin“ das einfach und schnelle Eintragen und Auslesen von Betriebs-relevanten Daten ermöglichen. Dadurch soll es ermöglicht werden mit wenigen Mausklicks, ausführliche Berichte zu erstellen.

Das möchten wir mit Local Admin bieten:

- Eine Weboberfläche, welche von technisch nicht versiertem Personal bedient werden kann.
- Eine Plattform, welche durch nachkommende Entwickler, nach den Bedürfnissen des Arbeitgebers verändert und erweitert werden kann.
- Ein Tool, mit welchem sich anhand bereits eingegebener Daten, zukünftige Ereignisse planen bzw. prognostizieren lassen.
- Ein Tool, welches das Arbeiten mit Daten, durch übersichtliche und schnelle Suche, effizienter gestaltet.
- Ein Tool, welches den Papierverbrauch stark vermindern soll.

Local Admin soll es den Mitarbeitern ermöglichen schnell und effizient Daten, welche vom Benutzer festgelegt wurden, einzutragen. All diese Daten werden dazu in einer Datenbank gespeichert, welche das Verarbeiten und das Veranschaulichen von Daten vereinfacht und vielseitig einsetzbar macht.

2 Ist-Zustand

Derzeit verwenden die Burger King Filialen ein analoges Finanzverwaltungssystem aber auch zum Teil noch eine elektronische Version. Jedoch ist die analoge Version nicht mehr zeitgerecht und die digitale Version ist umständlich in der Anwendung und beinhaltet Designfehler.

2.1 Analoge Verwaltung

Vorteile

- Sicher

Nachteile

- Platzverschwendung
  - Papier
  - Akten
- Zeitaufwändig
  - Eintragen
  - Einsehen
- Fehlerfindung sehr schwer

2.2 Digitale Verwaltung

Vorteile

- Platzeffizient
- Schnelle Anwendung

Nachteile

- Unsicher
- Schwierige Handhabung

2.3 Lösung

In Local Admin möchten wir die Vorteile von dem analogen und dem digitalen Verwaltungssystem vereinen. Zum einen die Sicherheit des analogen Systems und zum anderen die Platzeffizienz des Systems und die Usability, sprich die einfache Anwendung des Produkts. Hinzufügen möchten wir auch noch, dass die Fehlerfindung vereinfacht werden soll.

3 Soll-Zustand

Wir möchten, dass Local Admin das bisherige Management der Daten verbessert. Um das zu erreichen, sind folgende Ziele wichtig:

3.1 Muss-Ziele

3.1.1 Akzeptanz der Mitarbeiter und Mitarbeiterinnen erreichen

Wir wollen bis 9. Dezember 2019 mit dem Großteil der Arbeit fertig sein und das Produkt unserem Projektauftragsgeber in einer Burger King Filiale überreichen, um Feedback zu eventuellen Verbesserungen sowie allgemeine Rückmeldung einzuholen.

- Es sollen keine interaktionsstörenden Fehler vorhanden sein (Fehler, die die Eingabe oder das Auslesen von Daten stören)
- Es sollen keine Dateninkonsistenzen vorhanden sein -> Alle gespeicherten Daten sollen den „Regeln“ entsprechen

Sollten diese Punkte erfüllt sein wird die effektive Zeit für das Ein- bzw. Nachtragen von Daten sowie das Abrufen von verschiedensten Berichten um bis 30% gesteigert.

3.1.2 Local Admin Online mit einem Account erreichbar zu machen

Jeder/Jede Mitarbeiter/in kann, sofern Sie einen Account von einem Admin erhalten hat und über einen gültigen Internetanschluss verfügt, von jedem Ort aus auf Local Admin zugreifen. Dies hat den Vorteil, dass keine analogen Hilfsmittel, wie z.B. Ordner mit Unterlagen verwendet werden müssen, um die Arbeitszeiten und Kosten zu planen. Durch die Möglichkeit über das Internet auf Local Admin zuzugreifen, erhält der Nutzer ebenfalls die Möglichkeit jederzeit sich über vergangene Aktivitäten zu informieren. Dies sollte das Arbeiten um ca. 10% effizienter gestalten, da man überall arbeiten kann und nicht erst in die jeweilige Filiale gehen muss, um die Daten einzusehen.

3.1.3 Einen Überblick für Mitarbeiter & Mitarbeiterinnen über die eingetragenen Daten geben

Local Admin soll den Mitarbeitern und den Mitarbeiterinnen die Möglichkeit geben schnell und einfach einen Überblick über bereits getätigte Aktivitäten zu erhalten. Dieser Vorgang soll ca. nur 30-50 % der Zeit beanspruchen, die es auf dem analogen Weg braucht.

3.2 Soll-Ziele

3.2.1 Optimierung der verbrauchten Zeit für das Management

Durch die Online Version der administrativen Arbeiten wird das Suchen in Ordnern überflüssig. Damit soll sich eine Zeitersparnis von bis zu 30% ausgehen. Das Ändern von Daten ist ebenfalls um ca. das dreifache effizienter, da das Eintragen mit wenig Aufwand verbunden ist.

3.3 Kann-Ziele

3.3.1 Prognosen Tool

Durch die, von den Mitarbeitern, eingetragenen bzw. vervollständigten Datensätzen können wir zu Marktforschungszwecken eine Ansicht anbieten, welche die Datenin einem Graph, über eine bestimmte Zeit, geordnet anzeigt. Dies würde das Auswerten der Daten bei eventuellen Meetings um ein Vielfaches schneller gestalten und außerdem sind dadurch Fehler in der Eingabe bzw. Unregelmäßigkeiten leichter zu erkennen.

3.3.2 Durch Plug-In erweiterbare Applikation

Local Admin kann, durch selbstständig geschriebene Plug-Ins, um Aktionen und Endpunkte erweitert werden.

3.4 Nicht-Ziele

3.4.1 Das Produkt an andere Restaurants weiterzugeben

Local Admin ist vor erst nur für 9 Burger King Filialen vorgesehen. Es ist nicht geplant diese großflächig in Österreich anzuwenden. Ebenfalls ist es nicht vorgesehen diese Software an weitere Restaurantketten weiterzugeben.

3.4.2 Ein universell anwendbares Webinterface (Vorlage) zu besitzen

Das Layout, wie es bei Local Admin verwendet wird, ist ausschließlich für Local Admin zu verwenden, und in keinem Fall in einem anderen Projekt.

3.4.3 Plug-Ins für die Applikation schreiben

Plug-Ins zu entwickeln, welche den Umfang der Website erweitern, um weitere Funktionen zu erstellen, welche je nach Anwendung und Effizienz die Arbeitszeit um bis zu 30% senken könnten.

3.4.4 Local Admin als Desktop Applikation für die Mitarbeiter bereit zu stellen

Local Admin als Desktop Applikation zu entwickeln würde den Mitarbeitern das Suchen in einem Browser abnehmen und Ladezeiten im Web verhindern. Dies würde eine Zeitersparnis von 5-10 Minuten pro Anwendung sparen.

4 Produktauswahl

4.1 Trendanalyse

Der Bedarf an Finanzverwaltungssystemen ist stets präsent, da in der heutigen, meist sehr schnellen Zeit, Lokale einen Überblick über Finanzen und Zeiten benötigen. Local Admin soll den MitarbeiterInnen das Verwalten der Daten so schnell und einfach wie möglich machen.

Vergleich zur Entwicklung der Konkurrenten per Gegenüberstellung. Dies geschieht mit einem 1-5 Punkte System.

4.1.1 Kosten

Die Kosten, die im Vergleich zu den anderen Anbietern, in Bezug auf Installation und Betrieb des Produktes anfallen.

4.1.2 Dauer der Entwicklung

Die Dauer der Entwicklung des Produkts.

4.1.3 Nutzen

Die Anzahl an Funktionen und deren Wert für die Firmen.

4.1.4 Zukünftige Relevanz

Die Relevanz des Produkts auf lange Sicht.

4.1.5 Konkurrentzrepräsentation

	Toast	Mirus	Peachworks Analytics	Restaurant 365
Kosten	2	3	4	5
Dauer der Entwicklung	2	3	5	4
Nutzen	4	2	3	1
Zukünftige Relevanz	4	2	3	2
Gesamt	12	10	15	12

1…Sehr Gut

5…Sehr Schlecht

4.2 Marktanalyse

Da heutzutage die Digitalisierung voll im Gange ist, gibt es klarerweise schon ähnliche Lösungen die sich mit demselben Problem(en) wie Local Admin beschäftigen. Es gibt zwar etliche Projekte, die unserem stark ähnlich sind, jedoch ist Local Admin in seiner Gesamtheit einzigartig. Leider sind alle Lösungen ziemlich unklar über genaue Funktionen und alles andere als informativ. Die ähnlichsten Softwarelösungen sind Folgende:

4.2.1 Toast

Ist die wahrscheinlich bekannteste, allumfassende, Restaurant-Management-Software. Allumfassend bedeutet, dass die Software darauf ausgelegt ist alle Prozesse in dem jeweiligen Restaurant zu übernehmen und nicht nur das Reporten. Es ist eine sehr facettenreiche Lösung, die jedoch nur als großes ganzes erwerbbar und verwendbar ist. Die Informationen über das sogenannte „back-of-house“ (die gängigste Bezeichnung für Software wie Local Admin) Modul ist jedoch sehr spärlich, bis kaum, vorhanden. Allerdings ist es nicht allein; also ohne die anderen Komponenten, verwendbar, wodurch es als Lösung für Burger King nicht infrage kommt.

Es kann (beschränkt auf die für uns relevanten Funktionen):

- Analyse der anfallenden Daten
- Graphen anzeigen
- Ist mobil kompatibel (nicht benötigt)
- Sendet Emails

Kann nicht:

- Nur als Reporting-Software verwendet werden

Nicht auffindbar:

- Eigene Spalten definierbar
- Manuelle Daten Eintragung
- Spezielle Reports erstellen
- Daten In/Export

Zudem werden die Daten alle zentral gespeichert und verwaltet was in unseren Augen (und in denen von Burger King) eine untragbare Datenunsicherheit mit sich bringt, da alle Daten streng vertraulich sind.

Der Preis dieser Software wird pro Terminal verrechnet: 79 $/Terminal

https://pos.toasttab.com/solutions/back-of-house

4.2.2 Mirus

Mirus ist das uns ähnlichste Projekt, jedoch auch hier ist der Funktionsumfang mithilfe der Webseite nicht ganz zu klären. Diese Software fokussiert sich komplett auf das Sammeln und Verwerten von Daten, womit es unserem Projekt am nächsten kommt. Es hat einige Features die unsere bei weitem überragen, jedoch sind sie entweder nicht notwendig oder überladen.

Es fehlen viele Informationen:

- Können die Daten manuell eingetragen werden?
- Können die von Burger King benötigten Reports von dieser Software erstellt werden.
- Ob die Spalten manuell angelegt werden können oder nur Formeln (ist wahrscheinlich aber nicht geklärt)

Darin ist Mirus besser:

- Formelerstellung
- Visualisierung
- Ansichtsoptionen

Klingt okay, jedoch werden die Daten auch hier zentral gespeichert – damit ist auch hier eine gewisse Datenunsicherheit gegeben, wobei Mirus angibt die Daten zu verschlüsseln. Der Preis ist jedoch das größte Abschreckargument. Der Preis für diese Software beträgt 75€ pro Restaurant und Monat, jedoch mindestens 1500 €/Monat (also 1500 € pro Monat bis man 20 Restaurants verwalten will). Dazu kommt eine Aufsetzungs-Gebühr von 6500 €. Das ergibt 12 * 1500 € (=18000 €) pro Jahr.

https://www.mirus.com/

4.2.3 Peachworks Analytics

Peachworks Analytics ist beinahe so nahe an unserer Software dran wie Mirus. Die größten Unterschiede sind (soweit man aus den online auffindbaren Screenshots sagen kann), dass es einige Funktionen gibt, die Mirus nicht anbietet:

- Selbstdefinierbare Spalten
- Manuelle Dateneintragung (stake Vermutung)

Jedoch gibt es einige Punkte die stark von dieser Lösung abweichen:

- Es steht nichts darüber wie/was für Reports man machen kann
- Die Firma hat Standorte in Amerika, was einerseits zu Datenschutzproblemen führt und zusätzlich Probleme bei technischen Problemen/Support verursachen könnte.
- Die Webseite ist sehr intransparent, was Informationen angeht. Es ist kaum möglich, nähere Information zu finden.
- Auch hier werden die Daten zentral gespeichert
- Der Preis beträgt laut einer Quelle ca. 195€/Restaurant und Monat

1 755€ pro Monat für 9 Restaurants oder ~21.000€ pro Jahr

https://apps.peachworks.com/app/analytics/

4.2.4 Restaurant 365

Bietet ähnliche Features an wie wir, jedoch ist auch hier nicht klar, wo die Daten herkommen, die in das System eingespielt werden. Auch Berichte werden kaum erwähnt, was ein wichtiger Bestandteil unserer Software ist. Beinhaltet jedoch die wichtige „Email sending“ Funktion. Leider findet man auch hier sehr wenig Spezifisches über die Funktionalität.

46,332€ pro Jahr

https://www.restaurant365.com/

5 Produktfunktionen

Hier werden alle Funktionen der verschiedenen Benutzergruppen definiert.

5.1 Administrator Funktionen

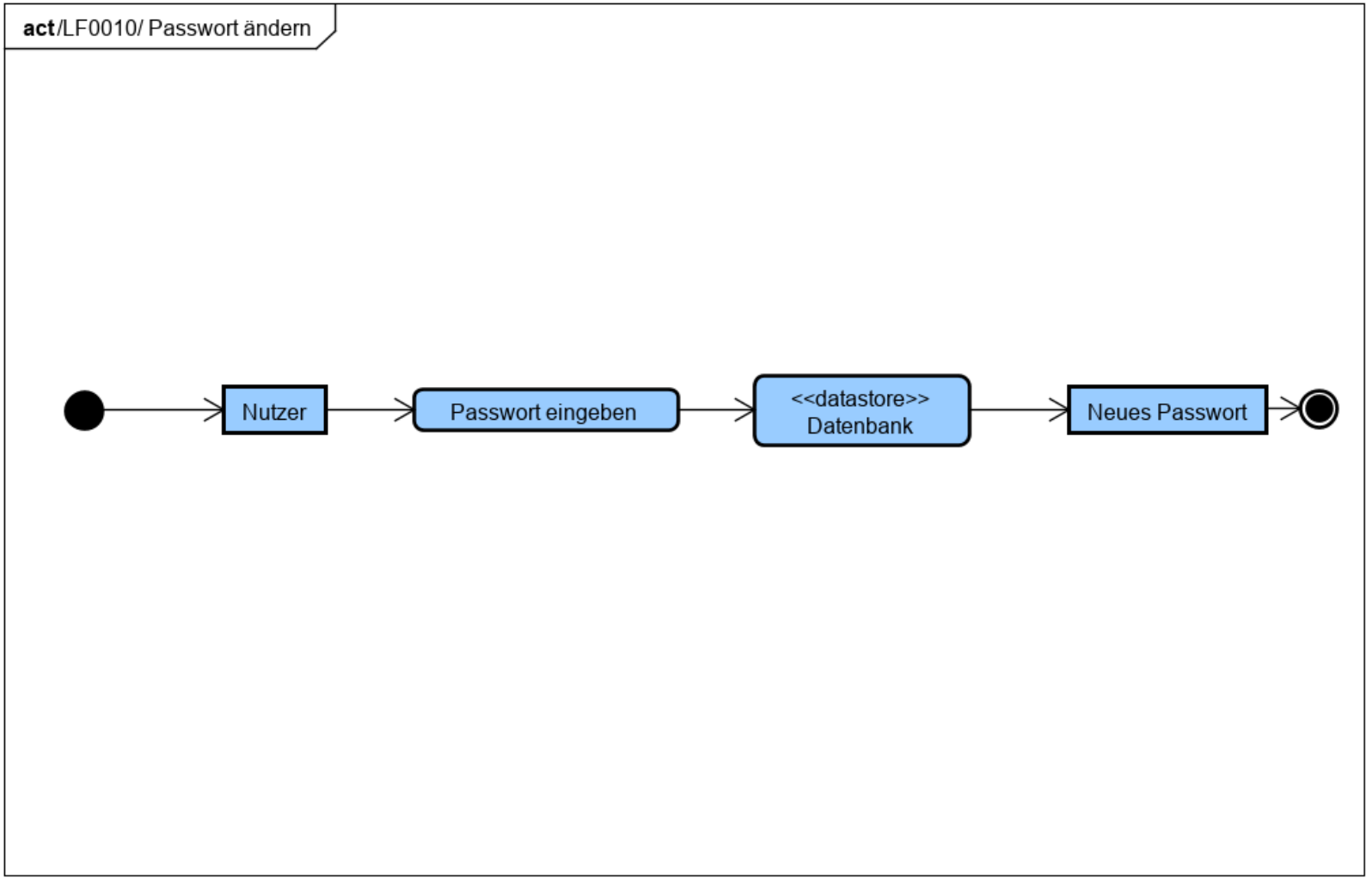
Die Administratoren Funktionen beschreib en alle Aktionen, die ein Administrator im Rahmen dieses Produktes ausführen kann.

5.1.1 /LF0010/ Passwort ändern

5.1.1.1 Funktionsdefinition

<table>
<tr><th colspan="2">Funktion</th><th>Nutzen</th><th>Aufwand</th><th>Must Have/ Should Have/ Nice to Have</th></tr>
<tr><td colspan="2">
<table>
<tr><th>Use Case</th><th>Beschreibung</th></tr>
<tr><td>Name</td><td>Passwort ändern</td></tr>
<tr><td>Art</td><td>Anwendungsfall</td></tr>
<tr><td>Kurzbeschreibung</td><td>Der Administrator kann sein altes Passwort durch ein neues Passwort ersetzen. Dies kann er machen sofern er angemeldet ist.</td></tr>
<tr><td>Auslöser</td><td>Der Administrator möchte sein Passwort ändern.</td></tr>
<tr><td>Ergebnis</td><td>Der Administrator hat ein neues Passwort.</td></tr>
<tr><td>Akteur</td><td>Administrator</td></tr>
<tr><td>Daten</td><td>• Passwort
• Neues Passwort</td></tr>
<tr><td>Vorbedingung</td><td>Der Administrator muss sein altes Passwort wissen und ein neues Überlegen.</td></tr>
<tr><td>Nachbereitung</td><td>Das neue Passwort wird in der Datenbank gespeichert. Der Administrator kann sein Passwort jederzeit ändern.</td></tr>
<tr><td>Vorgang</td><td>Der Administrator, der sich auf der Website eingeloggt hat, verifiziert auf der Startseite sein neues Passwort, dazu gibt er es zweimal ein.</td></tr>
</table>
</td><td>Mittel</td><td>Mittel</td><td>Should Have</td></tr>
</table>

5.1.1.2 Aktivitätsdiagramm

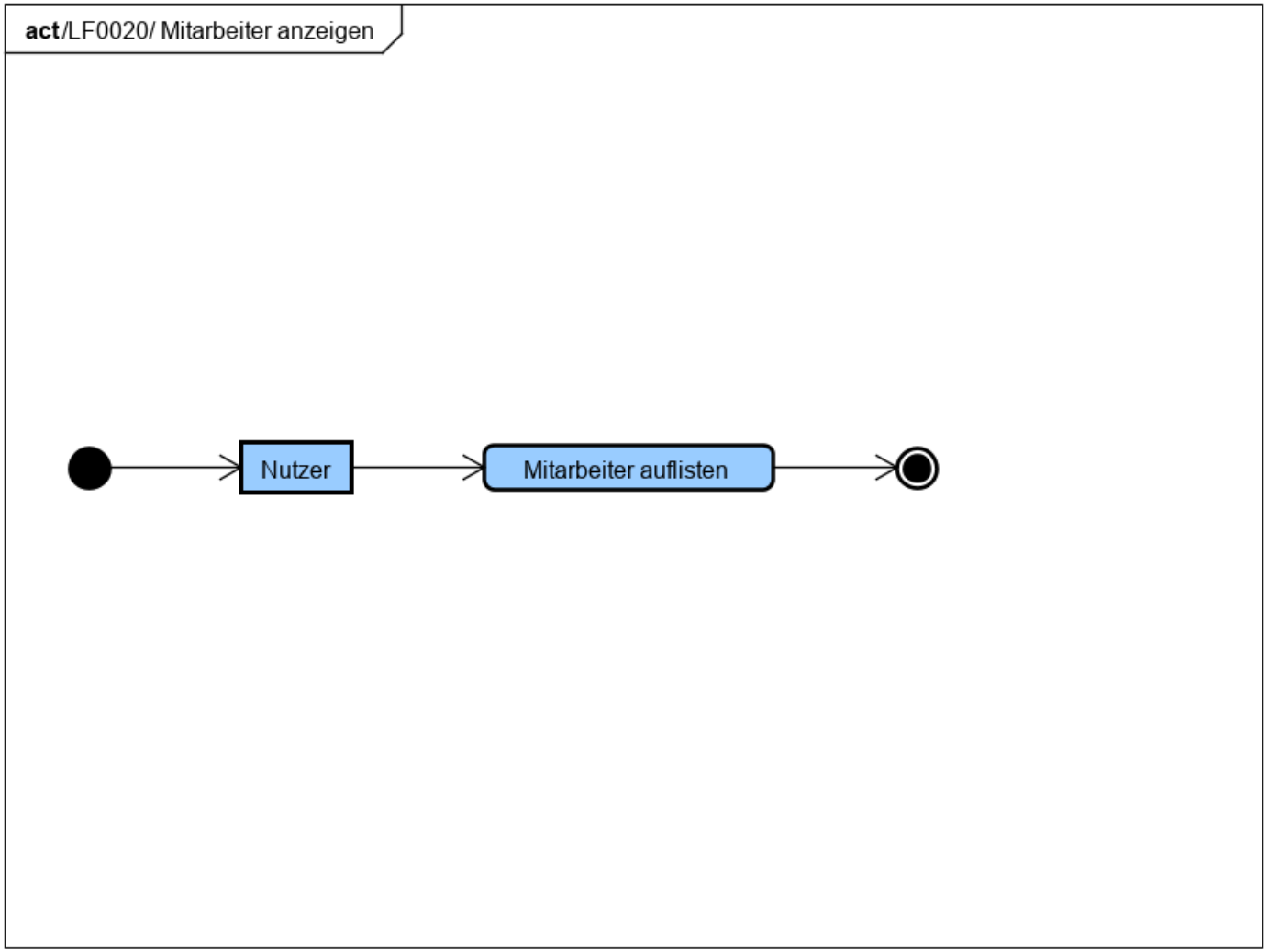


5.1.2 /LF0020/ Mitarbeiter anzeigen

5.1.2.1 Funktionsdefinition

<table>
<tr><th colspan="2">Funktion</th><th>Nutzen</th><th>Aufwand</th><th>Must Have/ Should Have/ Nice to Have</th></tr>
<tr><td colspan="2">
<table>
<tr><th>Use Case</th><th>Beschreibung</th></tr>
<tr><td>Name</td><td>Mitarbeiter anzeigen</td></tr>
<tr><td>Art</td><td>Anwendungsfall</td></tr>
<tr><td>Kurzbeschreibung</td><td>Der Administrator kann alle Nutzer innerhalb seiner Filialen einsehen. Das bedeutet er sieht seine E-Mail-Adresse seine Berechtigungen und seine Berichte.</td></tr>
<tr><td>Auslöser</td><td>Der Administrator möchte sich die Mitarbeiter einer Filiale anzeigen.</td></tr>
<tr><td>Ergebnis</td><td>Der Administrator bekommt eine Liste mit allen seinen Mitarbeitern angezeigt.</td></tr>
<tr><td>Akteur</td><td>Administrator</td></tr>
<tr><td>Daten</td><td>• Filiale
• Mitarbeiter</td></tr>
<tr><td>Vorbedingung</td><td>Der Administrator muss den Namen der Filiale wissen.</td></tr>
<tr><td>Nachbereitung</td><td>Es wird nichts im System geändert.</td></tr>
<tr><td>Vorgang</td><td>Der Administrator drückt, sofern er angemeldete ist, auf der Navigationsliste auf den Button „Benutzer“ daraufhin wird er eine Liste aller seiner Mitarbeiter angezeigt.</td></tr>
</table>
</td><td>Hoch</td><td>Gering</td><td>Must Have</td></tr>
</table>

5.1.2.2 Aktivitätsdiagramm

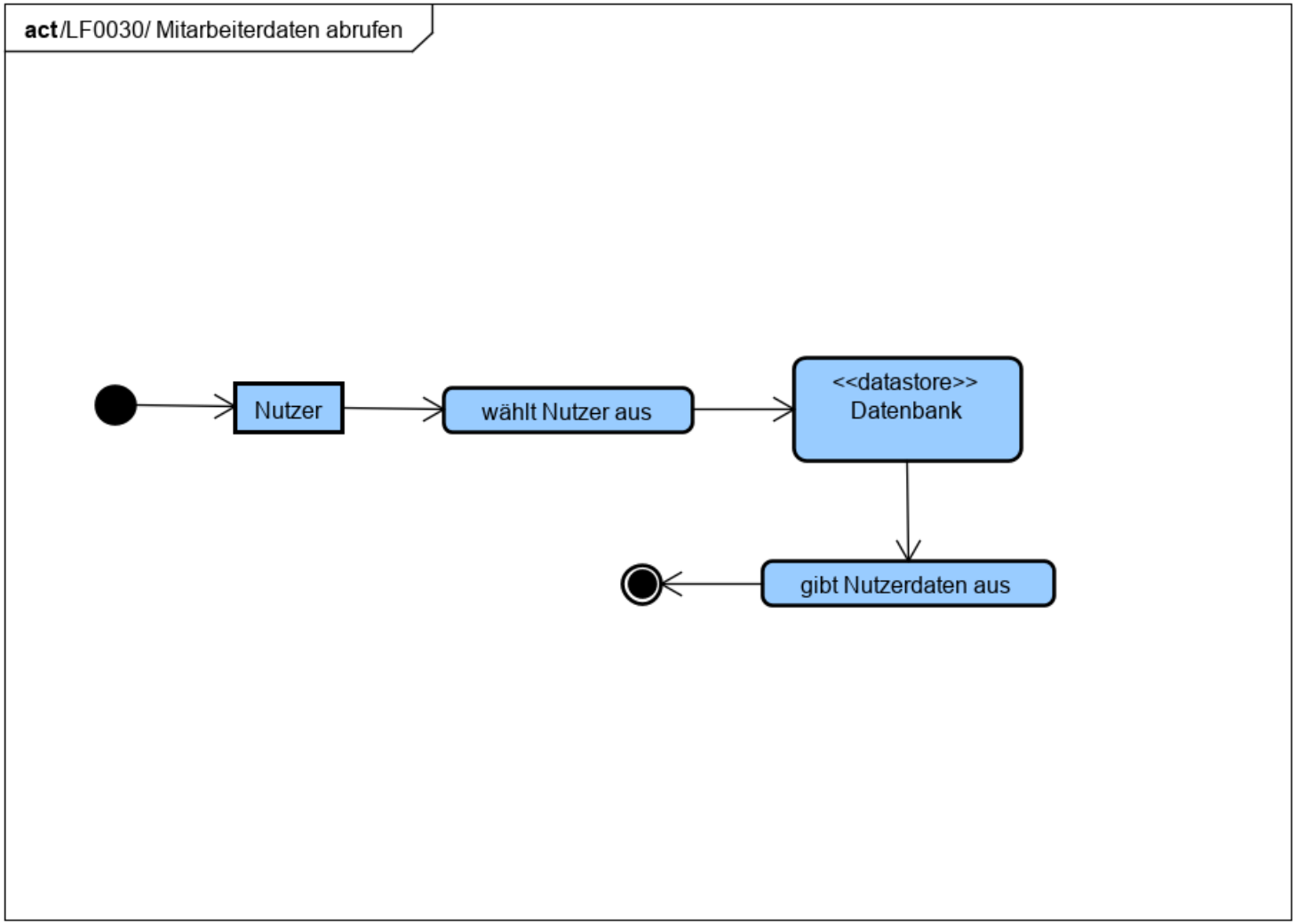


5.1.3 /LF0030/ Mitarbeiterdaten abrufen

5.1.3.1 Funktionsdefinition

<table>
<tr><th colspan="2">Funktion</th><th>Nutzen</th><th>Aufwand</th><th>Must Have/ Should Have/ Nice to Have</th></tr>
<tr><td colspan="2">
<table>
<tr><th>Use Case</th><th>Beschreibung</th></tr>
<tr><td>Name</td><td>Mitarbeiterdaten abrufen</td></tr>
<tr><td>Art</td><td>Anwendungsfall</td></tr>
<tr><td>Kurzbeschreibung</td><td>Der Administrator erhält genauere Informationen über einen einzelnen Mitarbeiter.</td></tr>
<tr><td>Auslöser</td><td>Einzelne Daten der Mitarbeiter abzurufen und einzusehen.</td></tr>
<tr><td>Ergebnis</td><td>Der Administrator erhält die Daten seiner Mitarbeiter.</td></tr>
<tr><td>Akteur</td><td>Der Administrator</td></tr>
<tr><td>Daten</td><td>• Daten des Mitarbeiters</td></tr>
<tr><td>Vorbedingung</td><td>Der Administrator muss sich die Liste seiner Mitarbeiter ausgeben lassen.</td></tr>
<tr><td>Nachbereitung</td><td>Der Administrator kann mit diesen Daten weiterarbeiten.</td></tr>
<tr><td>Vorgang</td><td>Sofern der Administrator angemeldet ist drückt er auf den „Benutzer“ Button nun erhält er eine Auflistung aller seiner Mitarbeiter. In dieser Liste wählt er den Mitarbeiter aus und drückt auf ihn. Der Administrator erhält die Daten des Nutzers.</td></tr>
</table>
</td><td>Mittel</td><td>Gering</td><td>Must Have</td></tr>
</table>

5.1.3.2 Aktivitätsdiagramm



5.1.4 /LF0040/ Mitarbeiter löschen

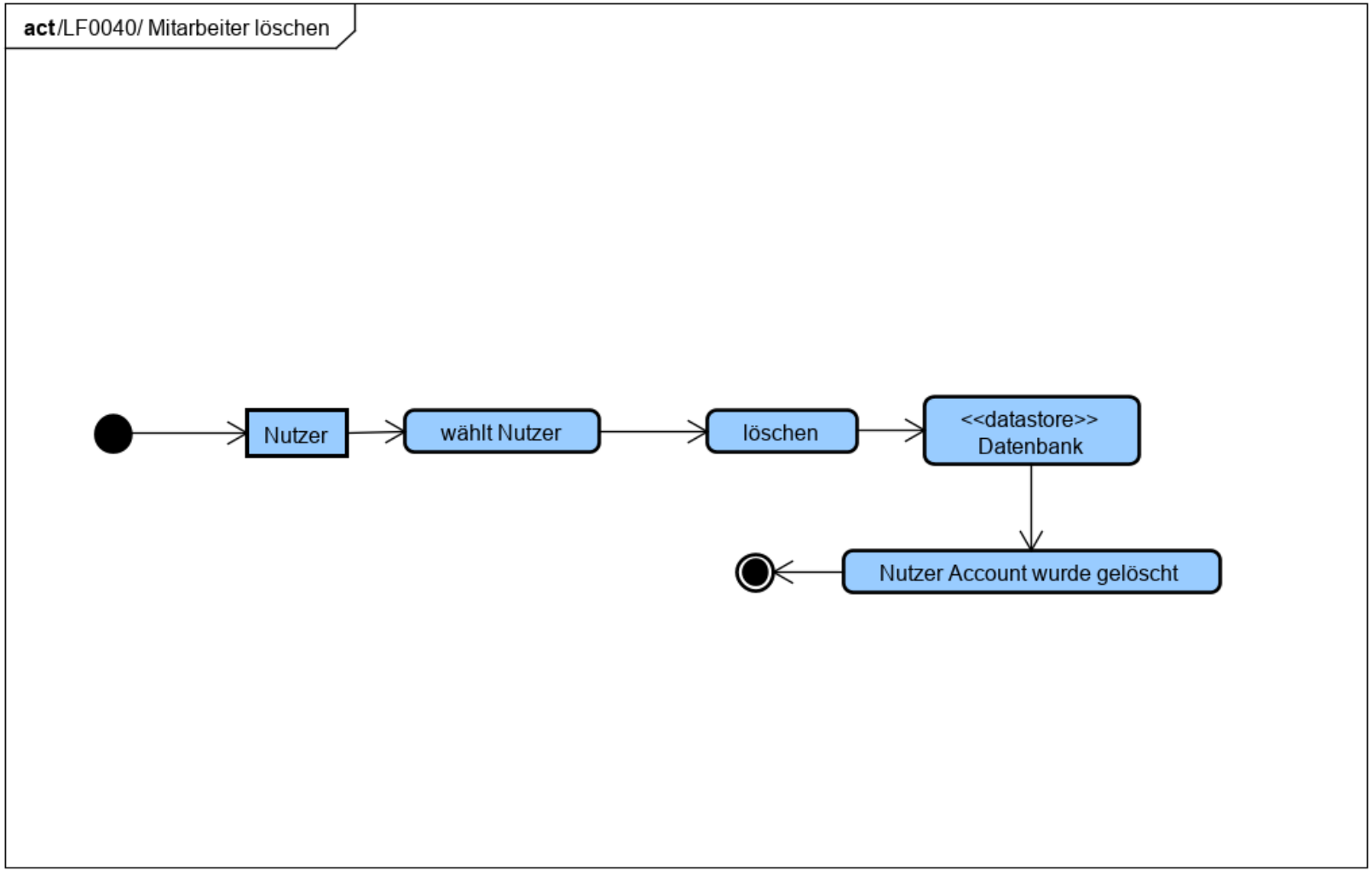
5.1.4.1 Funktionsdefinition

<table>
<tr><th colspan="2">Funktion</th><th>Nutzen</th><th>Aufwand</th><th>Must Have/ Should Have/ Nice to Have</th></tr>
<tr><td colspan="2">

Use Case	Beschreibung
Name	Mitarbeiter löschen
Art	Anwendungsfall
Kurzbeschreibung	Der Administrator kann einen Benutzer aus dem System löschen.
Auslöser	Der Administrator möchte einen Benutzer aus dem System löschen. Zum Beispiel weil ein Mitarbeiter entlassen wurde.
Ergebnis	Ein Benutzer wurde aus dem System gelöscht.
Akteur	Administrator
Daten	• Name des Mitarbeiters
Vorbedingung	Der Administrator muss sich die Liste der Mitarbeiter anzeigen lassen und anschließend den Benutzer auswählen, um ihn zu löschen.
Nachbereitung	Der Administrator kann jederzeit einen neuen Mitarbeiter hinzufügen.
Vorgang	Der Benutzer ist angemeldet und besitzt die Rechte zum Löschen. Der Administrator drückt auf den „Benutzer“ Button. Im nächsten Schritt öffnet sich eine Liste in welcher der Administrator seine Mitarbeiter aufgelistet sieht. Er drückt nun rechts auf ein „Mülleimer“ Symbol, um das Profil des Mitarbeiters zu löschen.

</td><td>Hoch</td><td>Gering</td><td>Must Have</td></tr>
</table>

5.1.4.2 Aktivitätsdiagramm

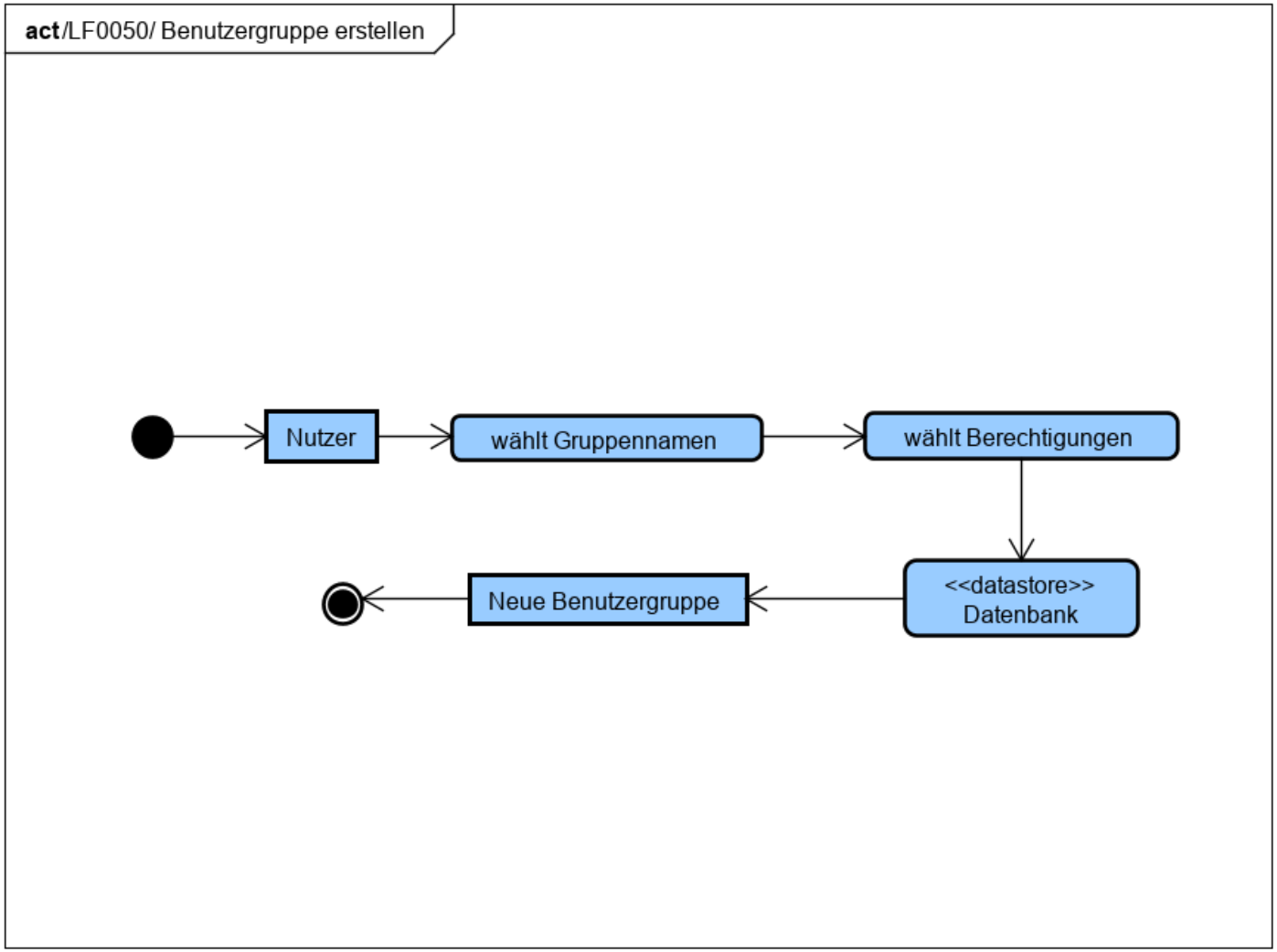


5.1.5 /LF0050/ Benutzergruppe erstellen

5.1.5.1 Funktionsdefinition

Funktion		Nutzen	Aufwand	Must Have/ Should Have/ Nice to Have
Use Case	**Beschreibung**	Hoch	Hoch	Must Have
Name	Benutzergruppe erstellen			
Art	Anwendungsfall			
Kurzbeschreibung	Der angemeldete Administrator kann eine neue Benutzergruppe erstellen mit neu kombinierten Rechten, um diese an einen Mitarbeiter zu vergeben.			
Auslöser	Der Administrator möchte eine neue Benutzergruppe erstellen welche Probleme lösen können, die von anderen Gruppen nicht lösbar sind.			
Ergebnis	Es gibt eine neue Benutzergruppe mit verschiedenen Rechten.			
Akteur	Administrator Mitarbeiter			
Daten	• Name der neuen Benutzergruppe • Rechte der Gruppe			
Vorbedingung	Der Administrator muss wissen welche Rechte er der Gruppe geben möchte und wie er die neue Gruppe nennen möchte.			
Nachbereitung	Der Administrator kann diese Gruppen wieder löschen und Neue erstellen.			
Vorgang	Der Benutzer hat die rechte und ist angemeldet drückt auf den „Gruppen“ Button nun öffnet sich die Liste mit allen seinen Gruppen. Er scrollt nun runter und findet am Ende der List einen Button welcher „Neue Gruppe“ heißt. Der Administrator drück auf „Neue Gruppe“ es öffnet sich ein Fenster in diesem Fenster gibt er eine neue Gruppenbezeichnung ein und legt die Rechte der Gruppe fest.			

5.1.5.2 Aktivitätsdiagramm

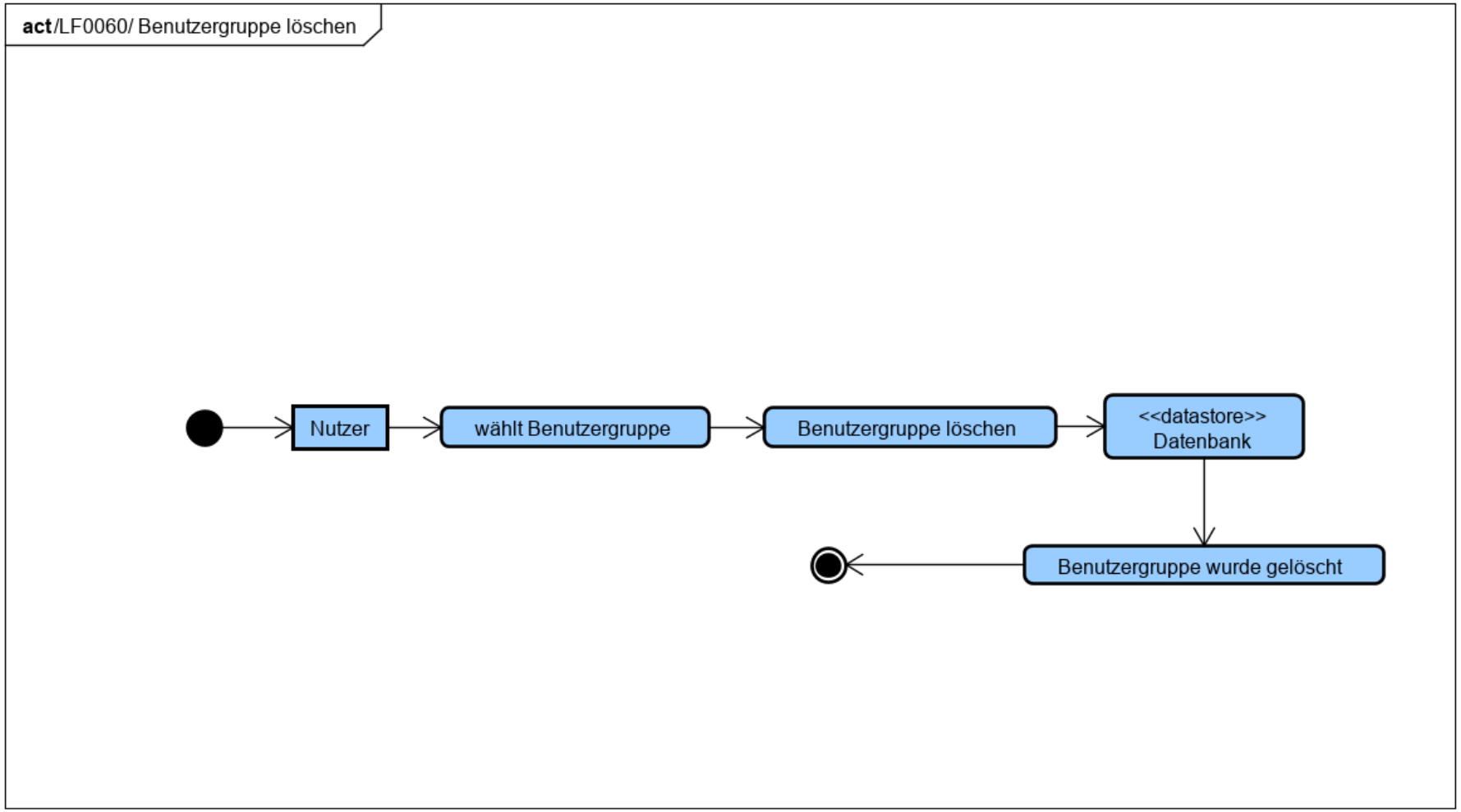


5.1.6 /LF0060/ Benutzergruppe löschen

5.1.6.1 Funktionsdefinition

Funktion		Nutzen	Aufwand	Must Have/ Should Have/ Nice to Have
Use Case	**Beschreibung**	Mittel	Gering	Must Have
Name	Benutzergruppe löschen			
Art	Anwendungsfall			
Kurzbeschreibung	Der Benutzer kann, sofern er angemeldet ist, einen Mitarbeiter auf Wunsch löschen.			
Auslöser	Der Administrator benötigt eine Benutzergruppe nicht mehr.			
Ergebnis	Eine Benutzergruppe wurde aus dem System gelöscht.			
Akteur	Administrator			
Daten	• Name der Benutzergruppe			
Vorbedingung	Es muss eine Benutzergruppe vorhanden sein die gelöscht werden kann.			
Nachbereitung	Der Administrator kann neue Gruppen erstellen und diese wieder löschen.			
Vorgang	Der Administrator drückt auf den „Gruppe“-Button. Im nächsten Schritt öffnet sich eine Liste in welcher er seine Gruppen aufgelistet sieht. Er drückt nun rechts auf ein „Mülleimer“ Symbol, um seine gesuchte Gruppe zu löschen.			

5.1.6.2 Aktivitätsdiagramm

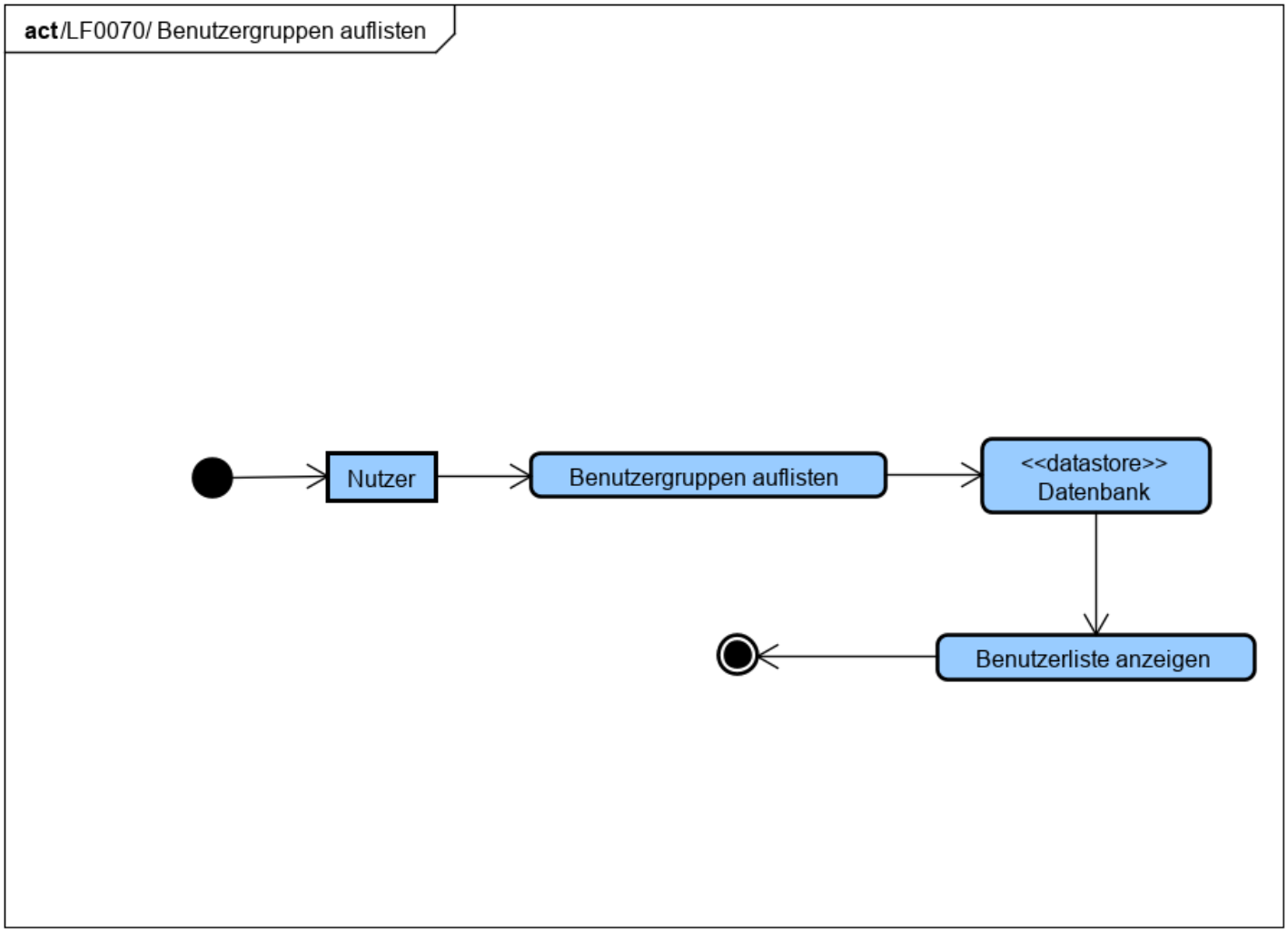


5.1.7 /LF0070/ Benutzergruppen auflisten

5.1.7.1 Funktionsdefinition

<table>
<tr><th colspan="2">Funktion</th><th>Nutzen</th><th>Aufwand</th><th>Must Have/ Should Have/ Nice to Have</th></tr>
<tr><td colspan="2">
<table>
<tr><th>Use Case</th><th>Beschreibung</th></tr>
<tr><td>Name</td><td>Benutzergruppen auflisten</td></tr>
<tr><td>Art</td><td>Anwendungsfall</td></tr>
<tr><td>Kurzbeschreibung</td><td>Der Administrator kann sich die Benutzergruppen ansehen.</td></tr>
<tr><td>Auslöser</td><td>Der Administrator möchte seine Benutzergruppen einsehen.</td></tr>
<tr><td>Ergebnis</td><td>Der Administrator sieht alle erstellten Benutzergruppe.</td></tr>
<tr><td>Akteur</td><td>Administrator</td></tr>
<tr><td>Daten</td><td>• Benutzergruppen</td></tr>
<tr><td>Vorbedingung</td><td>Es müssen Benutzergruppen vorhanden sein.</td></tr>
<tr><td>Nachbereitung</td><td>Der Administrator kann mit den Gruppen weiterarbeiten.</td></tr>
<tr><td>Vorgang</td><td>Der angemeldete Administrator drückt auf den „Gruppen“ Button nun öffnet sich die Liste mit allen seinen Gruppen.</td></tr>
</table>
</td><td>Mittel</td><td>Mittel</td><td>Must Have</td></tr>
</table>

5.1.7.2 Aktivitätsdiagramm

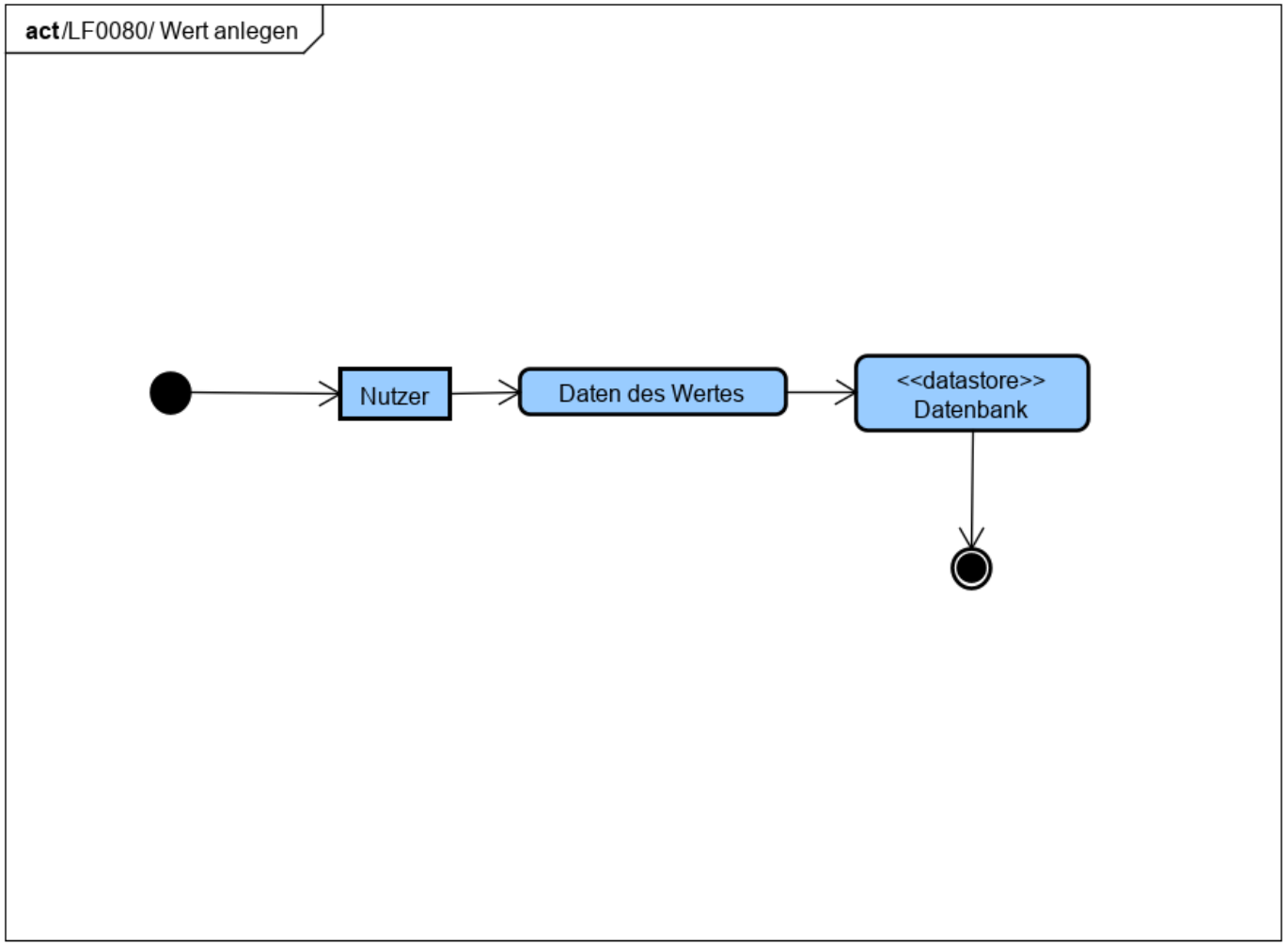


5.1.8 /LF0080/ Wert anlegen

5.1.8.1 Funktionsdefinition

<table>
<tr><th colspan="2">Funktion</th><th>Nutzen</th><th>Aufwand</th><th>Must Have/ Should Have/ Nice to Have</th></tr>
<tr><td colspan="2">
<table>
<tr><th>Use Case</th><th>Beschreibung</th></tr>
<tr><td>Name</td><td>Wert anlegen</td></tr>
<tr><td>Art</td><td>Anwendungsfall</td></tr>
<tr><td>Kurzbeschreibung</td><td>Der Administrator kann neue Werte in seinen Tabellen sehen das können z.B. Das Bruttoeinkommen sein oder die Überstunden von Mitarbeitern.</td></tr>
<tr><td>Auslöser</td><td>Der Administrator möchte neue Werte im System haben, um sich mehr bzw. genauere Datensätze auszugeben.</td></tr>
<tr><td>Ergebnis</td><td>Der Administrator hat mehr Wert welche ausgelesen werden können.</td></tr>
<tr><td>Akteur</td><td>Administrator</td></tr>
<tr><td>Daten</td><td>• Name des neuen Wertes
• Beschreibung des Wertes
• Muss der Wert täglich eingetragen werden</td></tr>
<tr><td>Vorbedingung</td><td>Der Administrator muss einen neuen Wert im System haben.</td></tr>
<tr><td>Nachbereitung</td><td>Der Administrator kann neuen Werte erstellen bzw. alte Werte löschen.</td></tr>
<tr><td>Vorgang</td><td>Der angemeldete Administrator drückt in der Navigationsliste auf den Button „Werte“. Nun ändert sich die Seite und es wird eine Liste mit allen Werten angezeigt. Im nächsten Schritt drückt der Administrator auf den Button „Neuer Wert“ . Es öffnet sich eine neue Seite, auf welcher er den Namen des neuen Wertes eingibt, kann und er eine Beschreibung hinzufügt.</td></tr>
</table>
</td><td>Hoch</td><td>Hoch</td><td>Must Have</td></tr>
</table>

5.1.8.2 Aktivitätsdiagramm

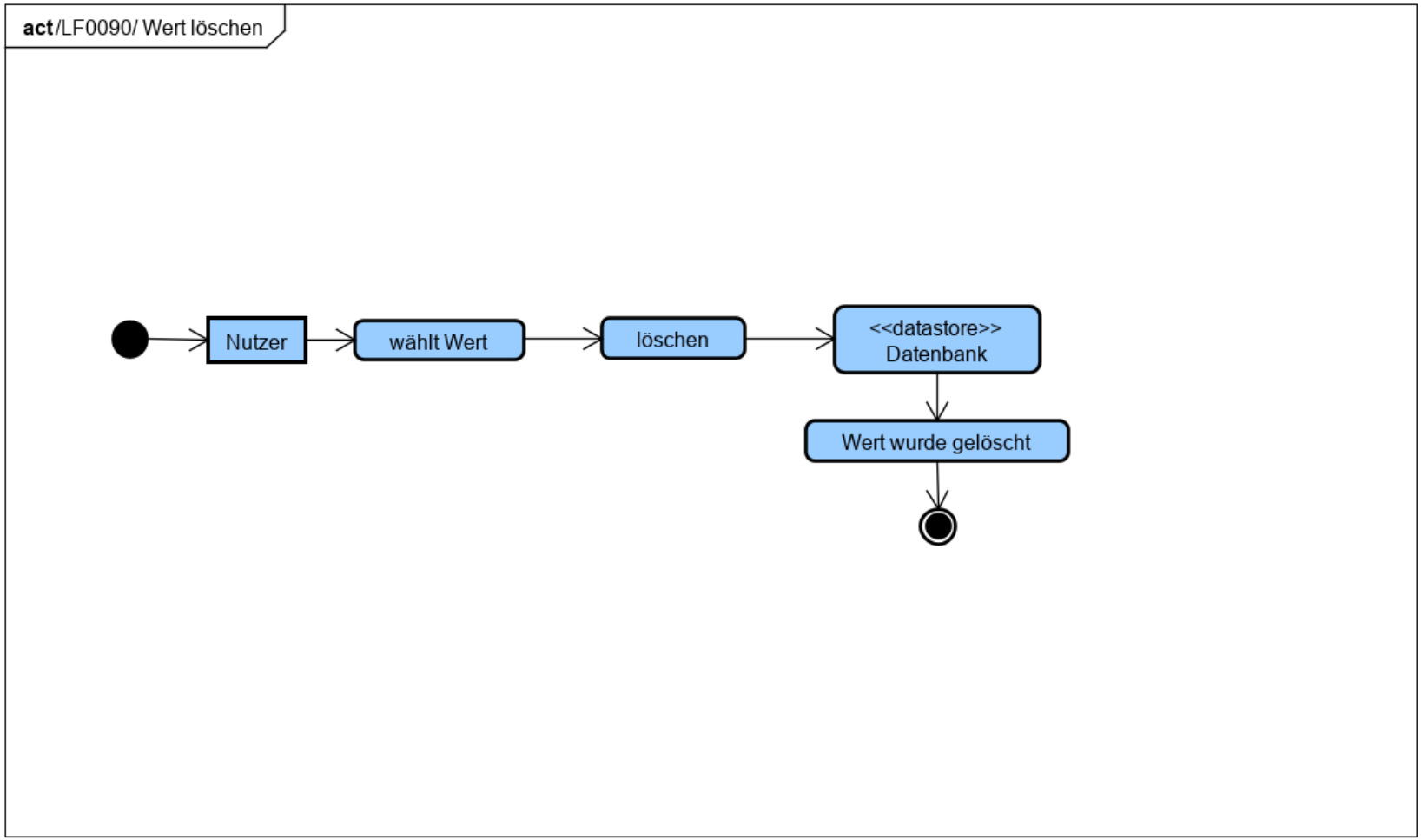


5.1.9 /LF0090/ Wert löschen

5.1.9.1 Funktionsdefinition

<table>
<tr><th colspan="2">Funktion</th><th>Nutzen</th><th>Aufwand</th><th>Must Have/ Should Have/ Nice to Have</th></tr>
<tr><td colspan="2">
<table>
<tr><th>Use Case</th><th>Beschreibung</th></tr>
<tr><td>Name</td><td>Wert löschen</td></tr>
<tr><td>Art</td><td>Anwendungsfall</td></tr>
<tr><td>Kurzbeschreibung</td><td>Der Administrator kann einen gesetzten Wert löschen.</td></tr>
<tr><td>Auslöser</td><td>Der Administrator möchte einen Wert im System löschen.</td></tr>
<tr><td>Ergebnis</td><td>Der Administrator hat einen Wert aus dem System gelöscht.</td></tr>
<tr><td>Akteur</td><td>Administrator</td></tr>
<tr><td>Daten</td><td>• Name des Wertes</td></tr>
<tr><td>Vorbedingung</td><td>Es müssen bereits Werte vorhanden sein.</td></tr>
<tr><td>Nachbereitung</td><td>Der Administrator kann neue Werte erstellen und anschließend wieder löschen.</td></tr>
<tr><td>Vorgang</td><td>Der angemeldete Administrator drückt in der Navigationsliste auf den Button „Werte“. Nun ändert sich die Seite und es wird eine Liste mit allen Werten angezeigt. Der Administrator drückt nun rechts auf den roten „Mülleimer“-Button daraufhin wird der Wert gelöscht.</td></tr>
</table>
</td><td>Hoch</td><td>Gering</td><td>Must Have</td></tr>
</table>

5.1.9.2 Aktivitätsdiagramm

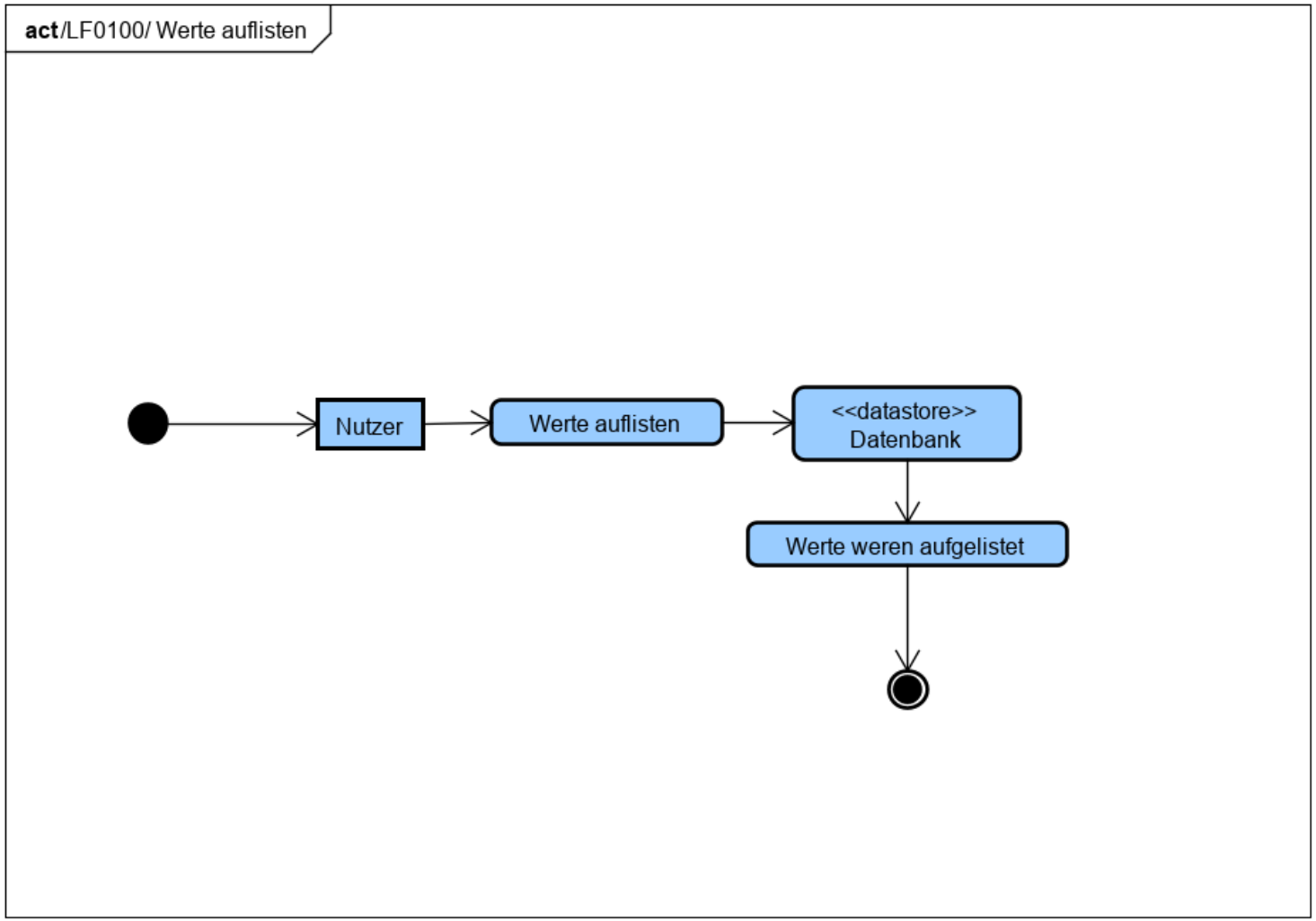


5.1.10 /LF0100/ Wert auflisten

5.1.10.1 Funktionsdefinition

<table>
<tr><th colspan="2">Funktion</th><th>Nutzen</th><th>Aufwand</th><th>Must Have/ Should Have/ Nice to Have</th></tr>
<tr><td colspan="2">
<table>
<tr><th>Use Case</th><th>Beschreibung</th></tr>
<tr><td>Name</td><td>Wert auflisten</td></tr>
<tr><td>Art</td><td>Anwendungsfall</td></tr>
<tr><td>Kurzbeschreibung</td><td>Der Administrator sieht seine Werte ein.</td></tr>
<tr><td>Auslöser</td><td>Der Administrator möchte seine Werte einsehen.</td></tr>
<tr><td>Ergebnis</td><td>Der Administrator hat Kenntnis über seine bestehenden Werte erhalten.</td></tr>
<tr><td>Akteur</td><td>Administrator</td></tr>
<tr><td>Daten</td><td>• Die Werte</td></tr>
<tr><td>Vorbedingung</td><td>Der Administrator muss schon Wert erstellt haben.</td></tr>
<tr><td>Nachbereitung</td><td>Der Administrator kann die Werte löschen.</td></tr>
<tr><td>Vorgang</td><td>Der angemeldete Administrator drückt in der Navigationsliste auf den Button „Werte“. Nun ändert sich die Seite und es wird eine Liste mit allen Werten angezeigt.</td></tr>
</table>
</td><td>Mittel</td><td>Mittel</td><td>Must Have</td></tr>
</table>

5.1.10.2 Aktivitätsdiagramm

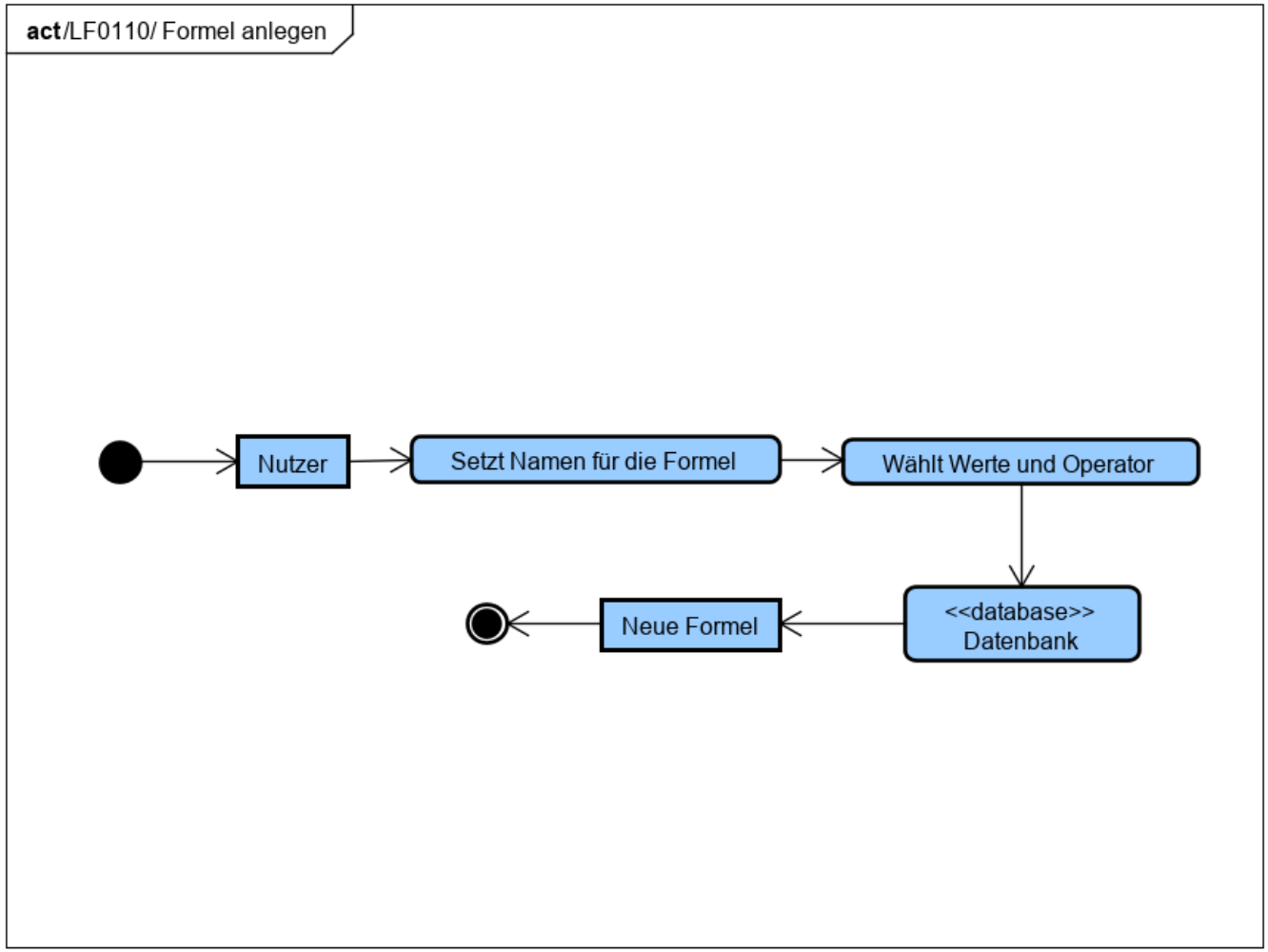


5.1.11 /LF0110/ Formel anlegen

5.1.11.1 Funktionsdefinition

Funktion		Nutzen	Aufwand	Must Have/ Should Have/ Nice to Have
Use Case	**Beschreibung**	Hoch	Hoch	Must Have
Name	Formel anlegen			
Art	Anwendungsfall			
Kurzbeschreibung	Der Administrator kann sich aus Werten neue Werte errechnen lassen, diese Kombination aus Werten wird Formel genannt.			
Auslöser	Der Administrator möchte aus bereits bestehenden Werten eine Formel erstellen, um sich neue Daten ausgeben zu lassen.			
Ergebnis	Der Administrator hat einen neuen Datensatz aus kombinierten Werten, welche er sich ausgeben kann.			
Akteur	Administrator			
Daten	• Kombination der Werte			
Vorbedingung	Es müssen Werte bestehen.			
Nachbereitung	Es können Formeln vom Administrator gelöscht werden.			
Vorgang	Der angemeldete Administrator drückt in der Navigationsliste auf den Button „Werte“. Nun ändert sich die Seite und es wird eine Liste mit allen Werten angezeigt. Nun drückt der Administrator auf den „Neue Formel“ Button, nun wählt der Administrator zwei Werte aus, aus welchen er sich eine Formel zusammenstellt.			

5.1.11.2 Aktivitätsdiagramm

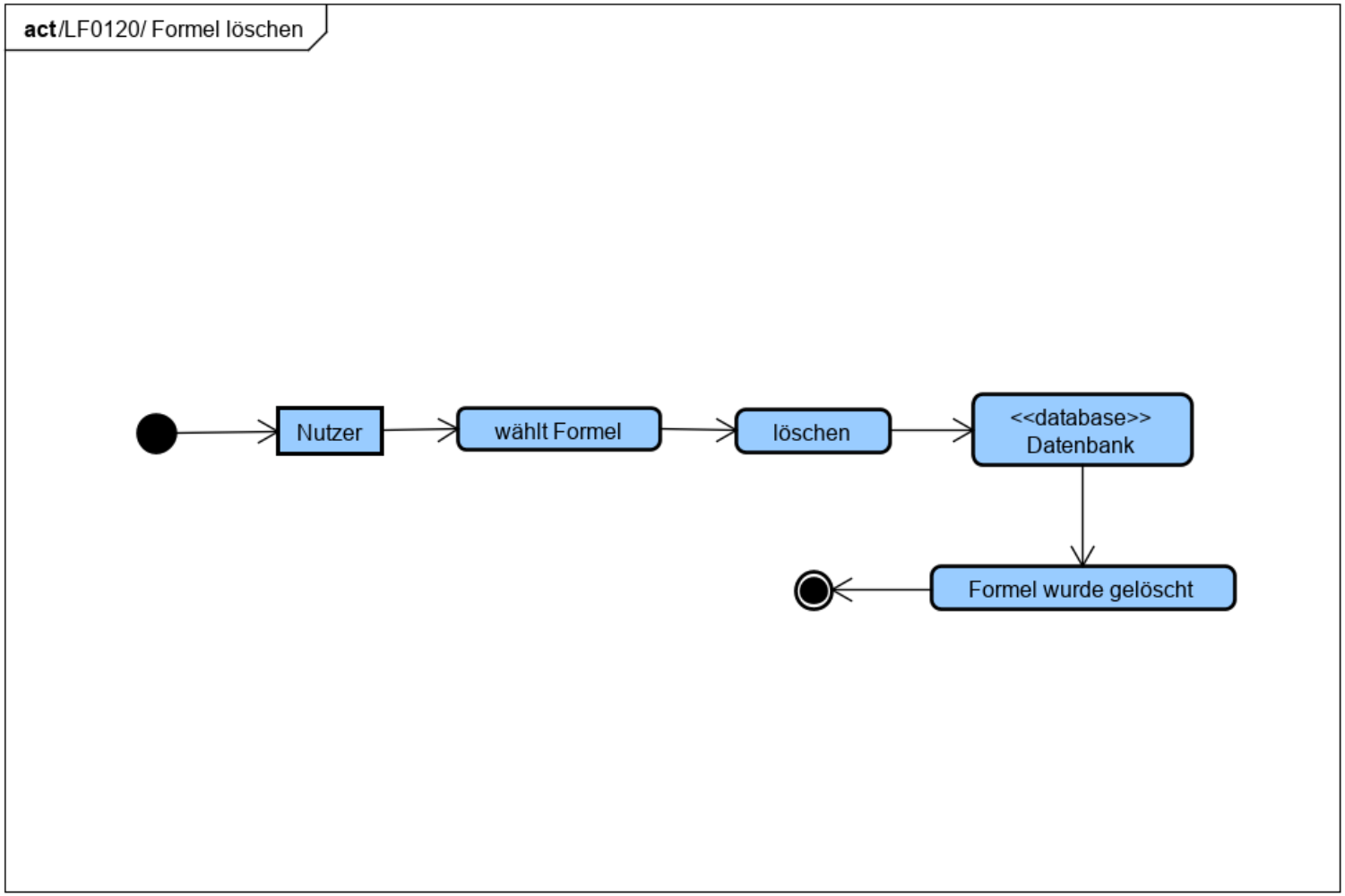


5.1.12 /LF0120/ Formel löschen

5.1.12.1 Funktionsdefinition

<table>
<tr><th colspan="2">Funktion</th><th>Nutzen</th><th>Aufwand</th><th>Must Have/ Should Have/ Nice to Have</th></tr>
<tr><td colspan="2">
<table>
<tr><th>Use Case</th><th>Beschreibung</th></tr>
<tr><td>Name</td><td>Formel löschen</td></tr>
<tr><td>Art</td><td>Anwendungsfall</td></tr>
<tr><td>Kurzbeschreibung</td><td>Der Administrator kann eine vorhandene Formel löschen.</td></tr>
<tr><td>Auslöser</td><td>Der Administrator möchte eine nicht mehr benötigte oder falsche Formel löschen.</td></tr>
<tr><td>Ergebnis</td><td>Es wurde eine Formel aus dem System gelöscht.</td></tr>
<tr><td>Akteur</td><td>Administrator</td></tr>
<tr><td>Daten</td><td>• Die Formel</td></tr>
<tr><td>Vorbedingung</td><td>Es muss eine Formel vorhanden sein.</td></tr>
<tr><td>Nachbereitung</td><td>Es können neue Formeln erstellt werden.</td></tr>
<tr><td>Vorgang</td><td>Der angemeldete Administrator drückt auf den „Werte“ Button, um die Liste der Formeln zu erhalten. Er drückt nun recht auf den „Mülleimer“ Button, um die Formel zu löschen.</td></tr>
</table>
</td><td>Mittel</td><td>Gering</td><td>Must Have</td></tr>
</table>

5.1.12.2 Aktivitätsdiagramm

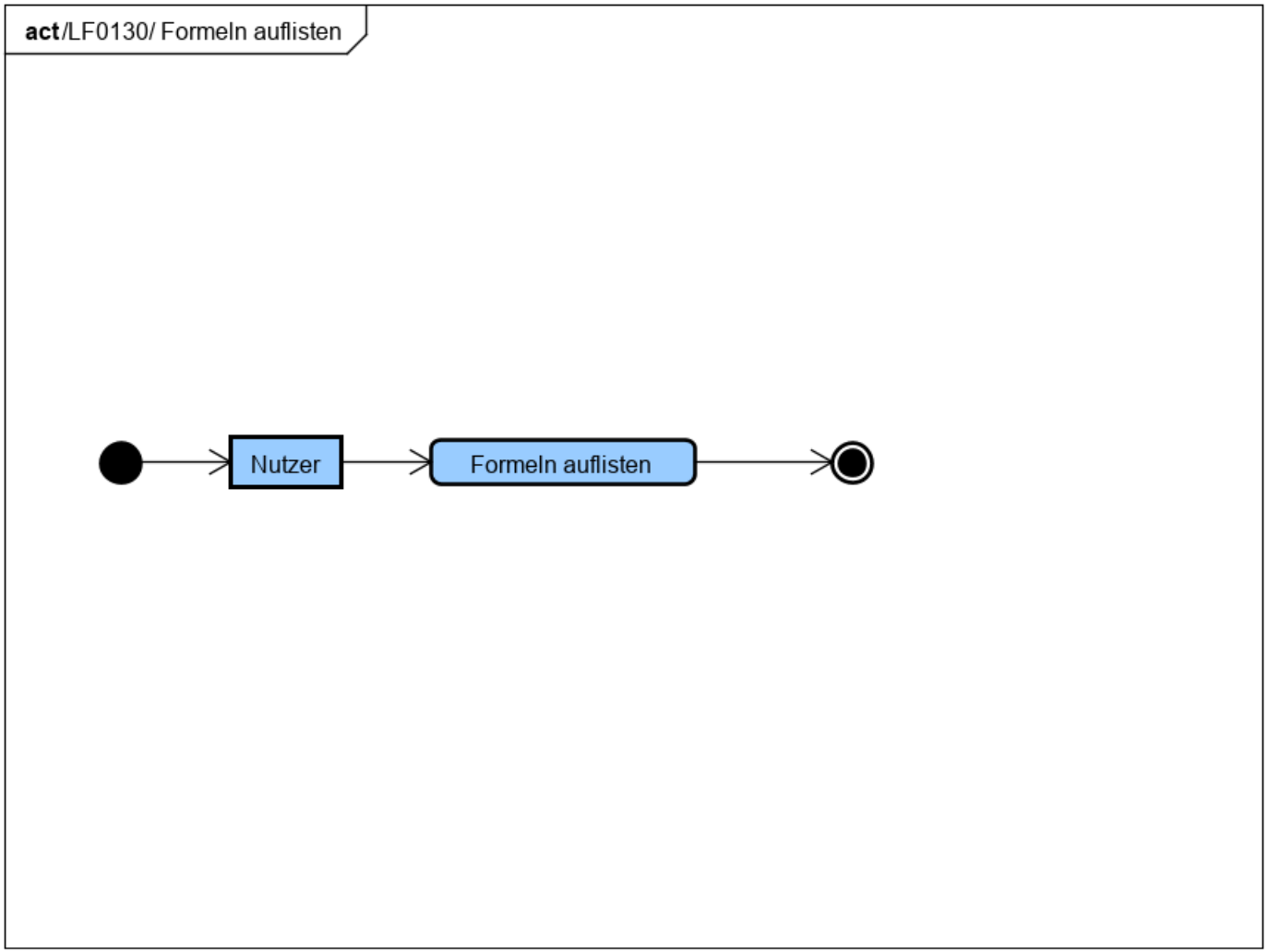


5.1.13 /LF0130/ Formeln auflisten

5.1.13.1 Funktionsdefinition

Funktion		Nutzen	Aufwand	Must Have/ Should Have/ Nice to Have
Use Case	**Beschreibung**	Hoch	Gering	Must Have
Name	Formeln auflisten			
Art	Anwendungsfall			
Kurzbeschreibung	Der Administrator kann alle seine Funktionen sehen.			
Auslöser	Der Administrator möchte seine Formeln sehen.			
Ergebnis	Der Administrator sieht seine Funktionen.			
Akteur	Administrator			
Daten	• Formeln			
Vorbedingung	Es müssen bereits Formeln bestehen.			
Nachbereitung	Es können Formeln ausgewählt werden.			
Vorgang	Der angemeldete Administrator drückt auf den „Werte“-Button. Daraufhin werden ihm alle bestehenden Formeln angezeigt.			

5.1.13.2 Aktivitätsdiagramm

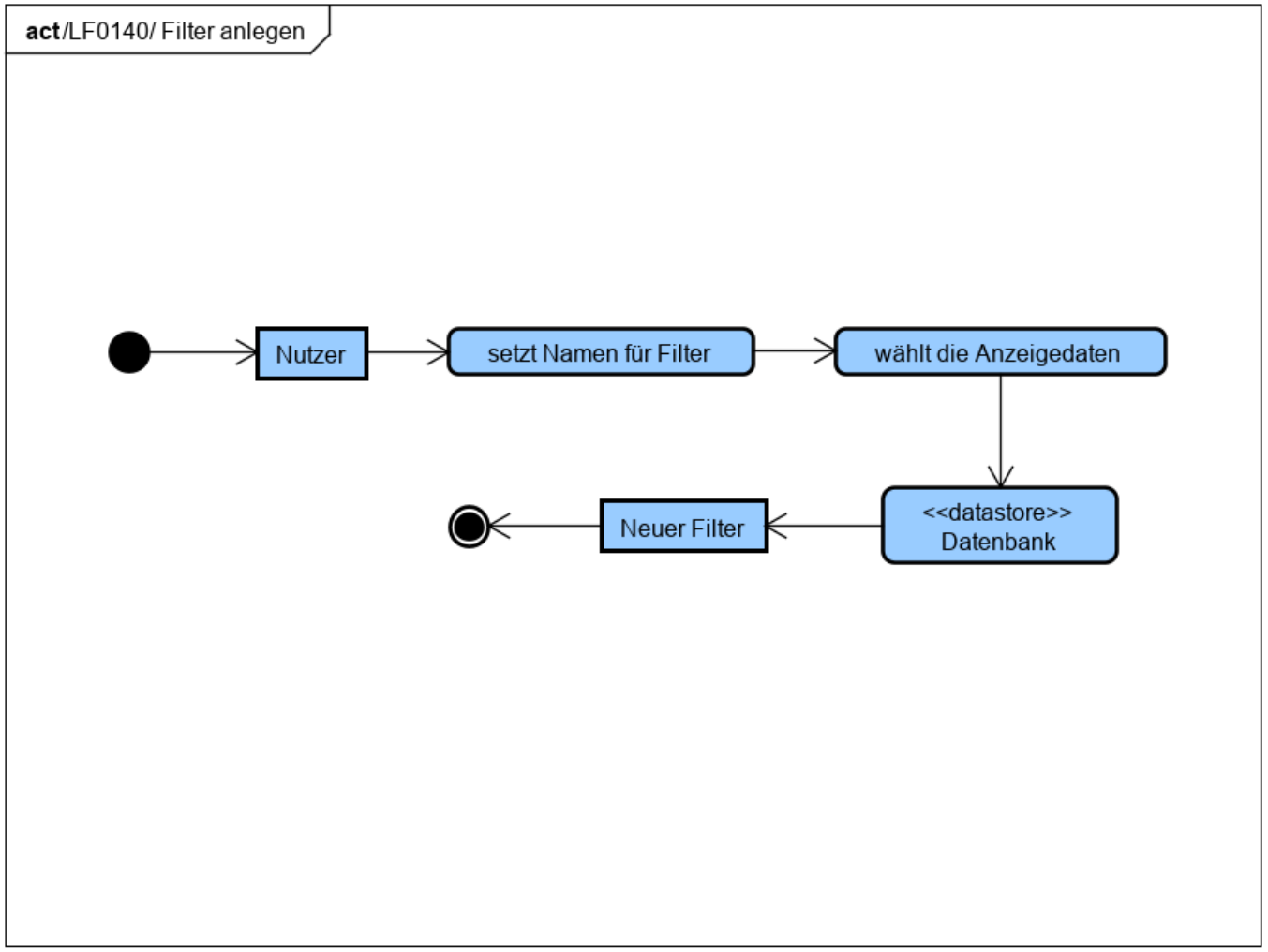


5.1.14 /LF0140/ Filter anlegen

5.1.14.1 Funktionsdefinition

Funktion		Nutzen	Aufwand	Must Have/ Should Have/ Nice to Have
Use Case	**Beschreibung**	Mittel	Hoch	Must Have
Name	Filter anlegen			
Art	Anwendungsfall			
Kurzbeschreibung	Der Administrator kann die Ausgabe Daten ändern, um nur noch die Informationen zu erhalten die er tatsächlich sehen möchte.			
Auslöser	Der Administrator möchte andere Daten angezeigt bekommen dies passiert durch das anpassend er Filter.			
Ergebnis	Der Administrator erhält neue Daten, welche sich aus der Kombination on werten erschließt.			
Akteur	Administrator			
Daten	• Name • Werte bzw. Formeln welche als Filter verwendet werden sollen			
Vorbedingung	Es müssen Werte gegeben sein.			
Nachbereitung	Es können immer neue Formeln erstellt werden bzw. alte gelöscht werden.			
Vorgang	Der angemeldete Administrator drückt nun auf der Website auf den „Übersicht“ Button. Auf dieser Seite legt er nun einen neuen Filter an. Dazu sucht er einen Namen für den neuen Filter aus und drückt anschließend auf „Filter hinzufügen“.			

5.1.14.2 Aktivitätsdiagramm

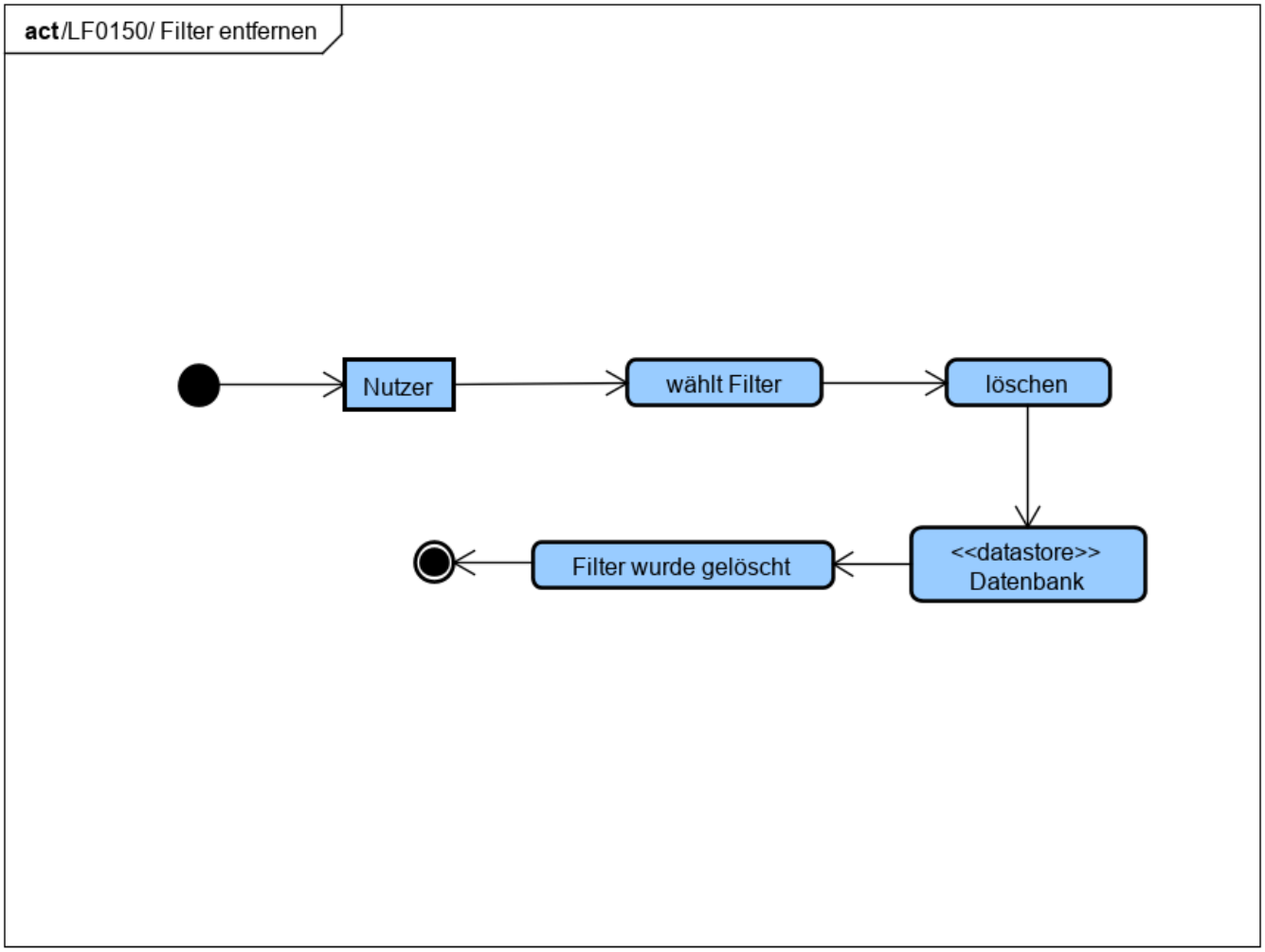


5.1.15 /LF0150/ Filter entfernen

5.1.15.1 Funktionsdefinition

<table>
<tr><th colspan="2">Funktion</th><th>Nutzen</th><th>Aufwand</th><th>Must Have/ Should Have/ Nice to Have</th></tr>
<tr><td colspan="2">
<table>
<tr><th>Use Case</th><th>Beschreibung</th></tr>
<tr><td>Name</td><td>Filter entfernen</td></tr>
<tr><td>Art</td><td>Anwendungsfall</td></tr>
<tr><td>Kurzbeschreibung</td><td>Der Administrator kann einen Filter löschen.</td></tr>
<tr><td>Auslöser</td><td>Der Administrator will einen Filter löschen da er diesen nicht mehr benötigt.</td></tr>
<tr><td>Ergebnis</td><td>Der Administrator hat einen Filter gelöscht und erhält dem entsprechend weniger Information.</td></tr>
<tr><td>Akteur</td><td>Administrator</td></tr>
<tr><td>Daten</td><td>• Der Filter</td></tr>
<tr><td>Vorbedingung</td><td>Es muss ein Filter existieren.</td></tr>
<tr><td>Nachbereitung</td><td>Es können neue Filter erstellt werden.</td></tr>
<tr><td>Vorgang</td><td>Der angemeldete Administrator drückt nun auf der Website auf den „Übersicht“ Button. Auf dieser Seite löscht er nun einen Filter.</td></tr>
</table>
</td><td>Mittel</td><td>gering</td><td>Must Have</td></tr>
</table>

5.1.15.2 Aktivitätsdiagramm

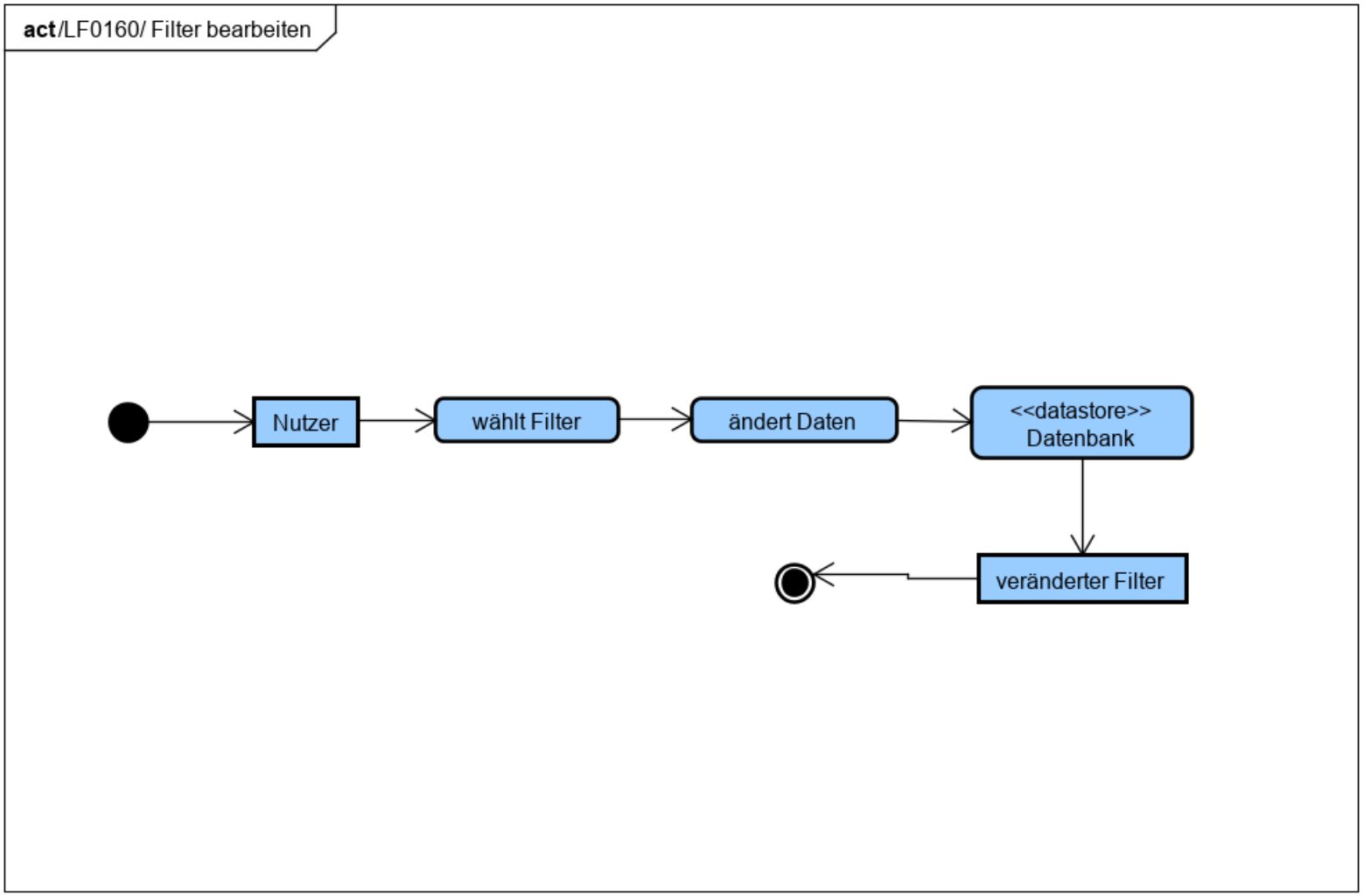


5.1.16 /LF0160/ Filter bearbeiten

5.1.16.1 Funktionsdefinition

Funktion		Nutzen	Aufwand	Must Have/ Should Have/ Nice to Have
Use Case	**Beschreibung**	Hoch	Mittel	Should Have
Name	Filter bearbeiten			
Art	Anwendungsfall			
Kurzbeschreibung	Der Administrator kann den Output an Daten eines Filters ändern.			
Auslöser	Der Administrator möchte einen Filter ändern.			
Ergebnis	Der Filter liefert andere Werte.			
Akteur	Administrator			
Daten	• Neue Filtereigenschaften			
Vorbedingung	Es müssen Werte gegeben sein und ein Filter, der überarbeitet werden soll.			
Nachbereitung	Der Filter kann weitere male geändert werden.			
Vorgang	Der angemeldete Administrator drückt nun auf der Website auf den „Übersicht“ Button. Auf dieser Seite sieht er nun bei einem Filter alle Werte und Formeln, die für den Filter anwählbar sind. Aus diesen Werten und Formeln baut sich der Admin seine Wunsch Kombination aus Werten.			

5.1.16.2 Aktivitätsdiagramm

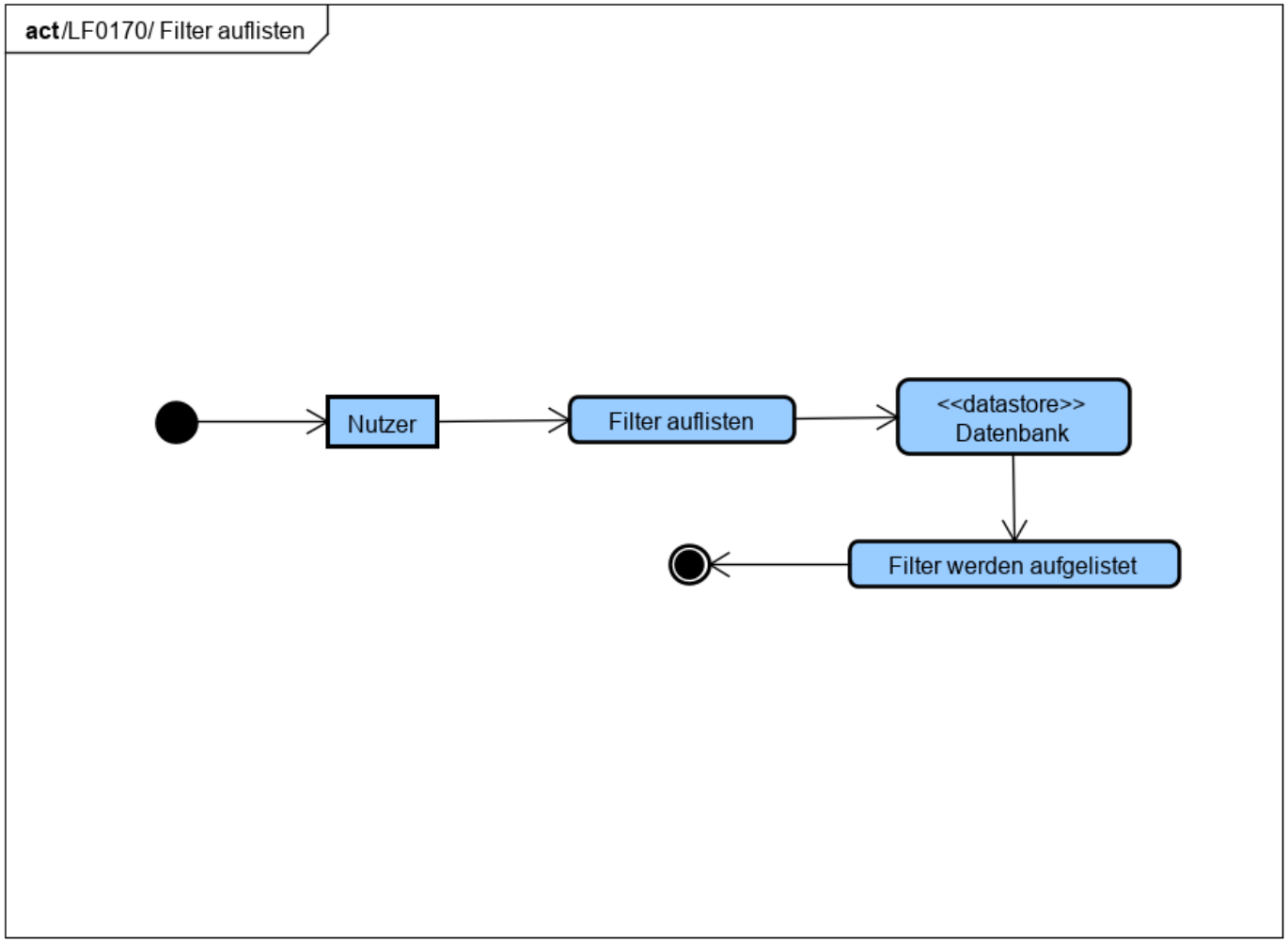


5.1.17 /LF0170/ Filter auflisten

5.1.17.1 Funktionsdefinition

<table>
<tr><th colspan="2">Funktion</th><th>Nutzen</th><th>Aufwand</th><th>Must Have/ Should Have/ Nice to Have</th></tr>
<tr><td colspan="2">
<table>
<tr><th>Use Case</th><th>Beschreibung</th></tr>
<tr><td>Name</td><td>Filter auflisten</td></tr>
<tr><td>Art</td><td>Anwendungsfall</td></tr>
<tr><td>Kurzbeschreibung</td><td>Der Administrator kann alle seine vorhandenen Filter sehen.</td></tr>
<tr><td>Auslöser</td><td>Der Admin möchte wissen welche Filter er besitz.</td></tr>
<tr><td>Ergebnis</td><td>Der Administrator erhält eine Liste mit allen seinen Filtern.</td></tr>
<tr><td>Akteur</td><td>Administrator</td></tr>
<tr><td>Daten</td><td>• Die Filter</td></tr>
<tr><td>Vorbedingung</td><td>Es müssen Filter vorhanden sein.</td></tr>
<tr><td>Nachbereitung</td><td>Er kann mit den gegebenen Filtern weiterarbeiten.</td></tr>
<tr><td>Vorgang</td><td>Der angemeldete Administrator kann nun auf der Website auf den „Übersicht“ Button drücken. Auf dieser Seite sieht er nun alle vorhandenen Filter.</td></tr>
</table>
</td><td>Hoch</td><td>Mittel</td><td>Must Have</td></tr>
</table>

5.1.17.2 Aktivitätsdiagramm

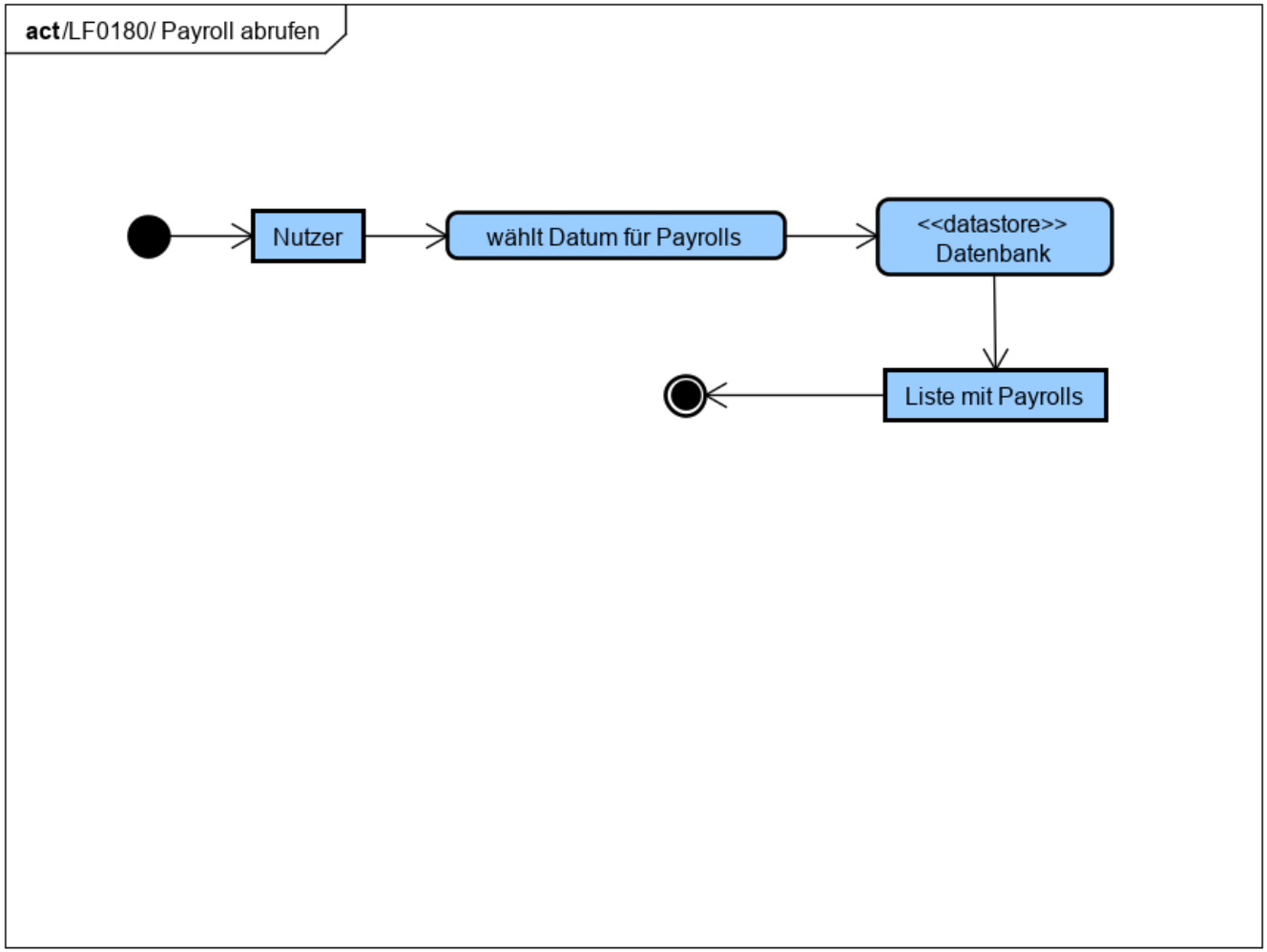


5.1.18 /LF0180/ Payroll abrufen

5.1.18.1 Funktionsdefinition

Funktion		Nutzen	Aufwand	Must Have/ Should Have/ Nice to Have
Use Case	**Beschreibung**	Hoch	Mittel	Must Have
Name	Payroll abrufen			
Art	Anwendungsfall			
Kurzbeschreibung	Der Administrator kann sich vergangene Payrolls anzeigen lassen.			
Auslöser	Der Administrator möchte alle Payrolls seines Restaurants der Vergangenheit sehen.			
Ergebnis	Der Administrator sieht eine Liste mit seinen Payrolls.			
Akteur	Administrator			
Daten	• Liste der Payrolls			
Vorbedingung	Es müssen Payrolls vorhanden sein.			
Nachbereitung	Er kann die Liste an Payrolls auslesen.			
Vorgang	Der angemeldete Administrator drückt auf den „Payroll“ Button. Anschließend erhält er eine Übersicht über alle Payrolls in den verschiedenen Restaurants			

5.1.18.2 Aktivitätsdiagramm

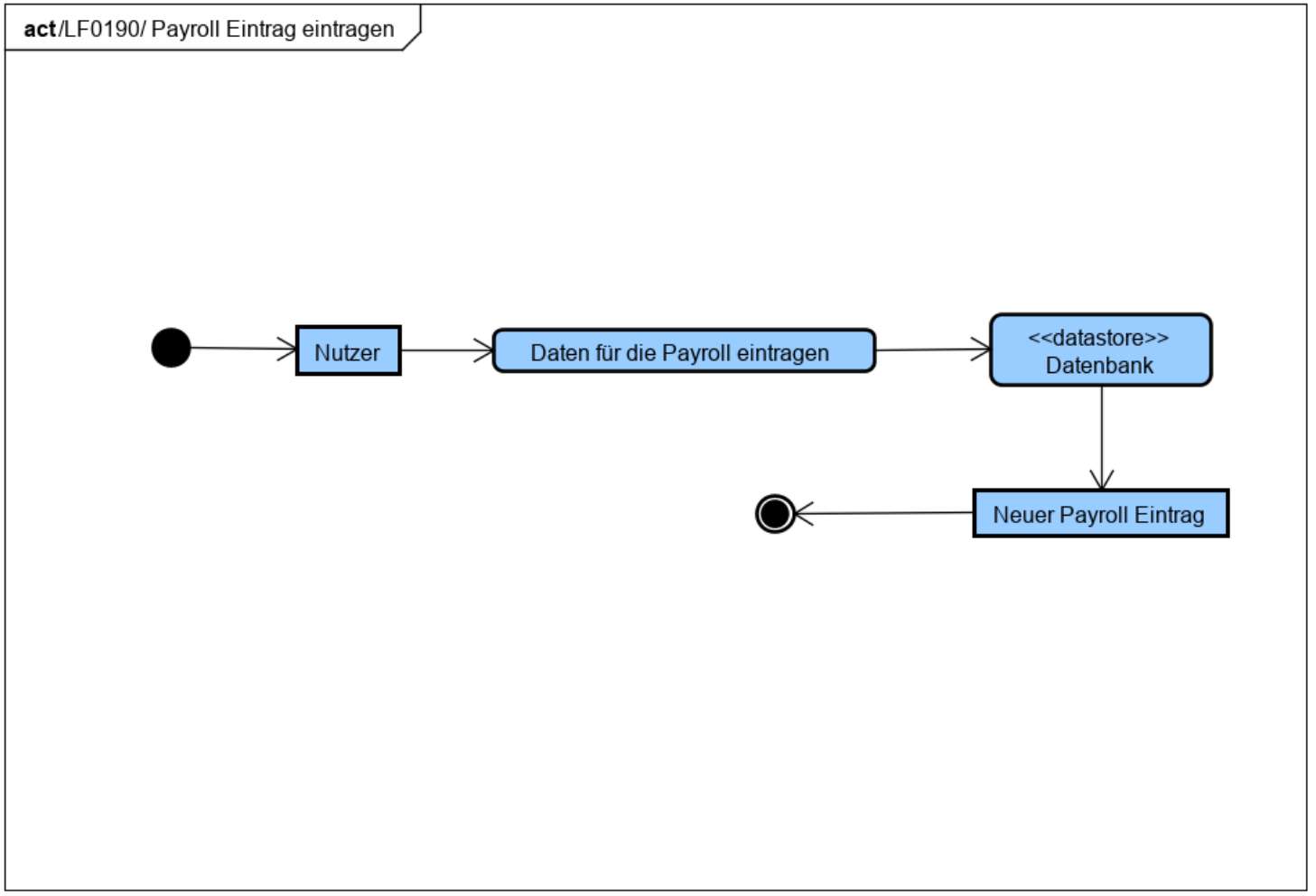


5.1.19 /LF0190/ Payroll Einträge eintragen

5.1.19.1 Funktionsdefinition

<table>
<tr><th colspan="2">Funktion</th><th>Nutzen</th><th>Aufwand</th><th>Must Have/ Should Have/ Nice to Have</th></tr>
<tr><td colspan="2"><table>
<tr><th>Use Case</th><th>Beschreibung</th></tr>
<tr><td>Name</td><td>Payrolls eintragen</td></tr>
<tr><td>Art</td><td>Anwendungsfall</td></tr>
<tr><td>Kurzbeschreibung</td><td>Der Administrator gibt die Werte für eine Payroll ein.</td></tr>
<tr><td>Auslöser</td><td>Der Administrator möchte gerne einen Eintrag in die Payroll tätigen.</td></tr>
<tr><td>Ergebnis</td><td>Es existiert eine neue Payroll im System.</td></tr>
<tr><td>Akteur</td><td>Administrator</td></tr>
<tr><td>Daten</td><td>• Wer
• Wann
• Wie viel Geld</td></tr>
<tr><td>Vorbedingung</td><td>Der Administrator muss die nötigen Daten in Erfahrung bringen.</td></tr>
<tr><td>Nachbereitung</td><td>Payroll Einträge können geändert werden.</td></tr>
<tr><td>Vorgang</td><td>Der angemeldete Administrator geht auf die „Payroll“ Seite, um auf dieser in einem Restaurant an dem richtigen Datum zur richtigen Payroll den richtigen Betrag anzulegen.</td></tr>
</table></td><td>Hoch</td><td>Hoch</td><td>Must Have</td></tr>
</table>

5.1.19.2 Aktivitätsdiagramm

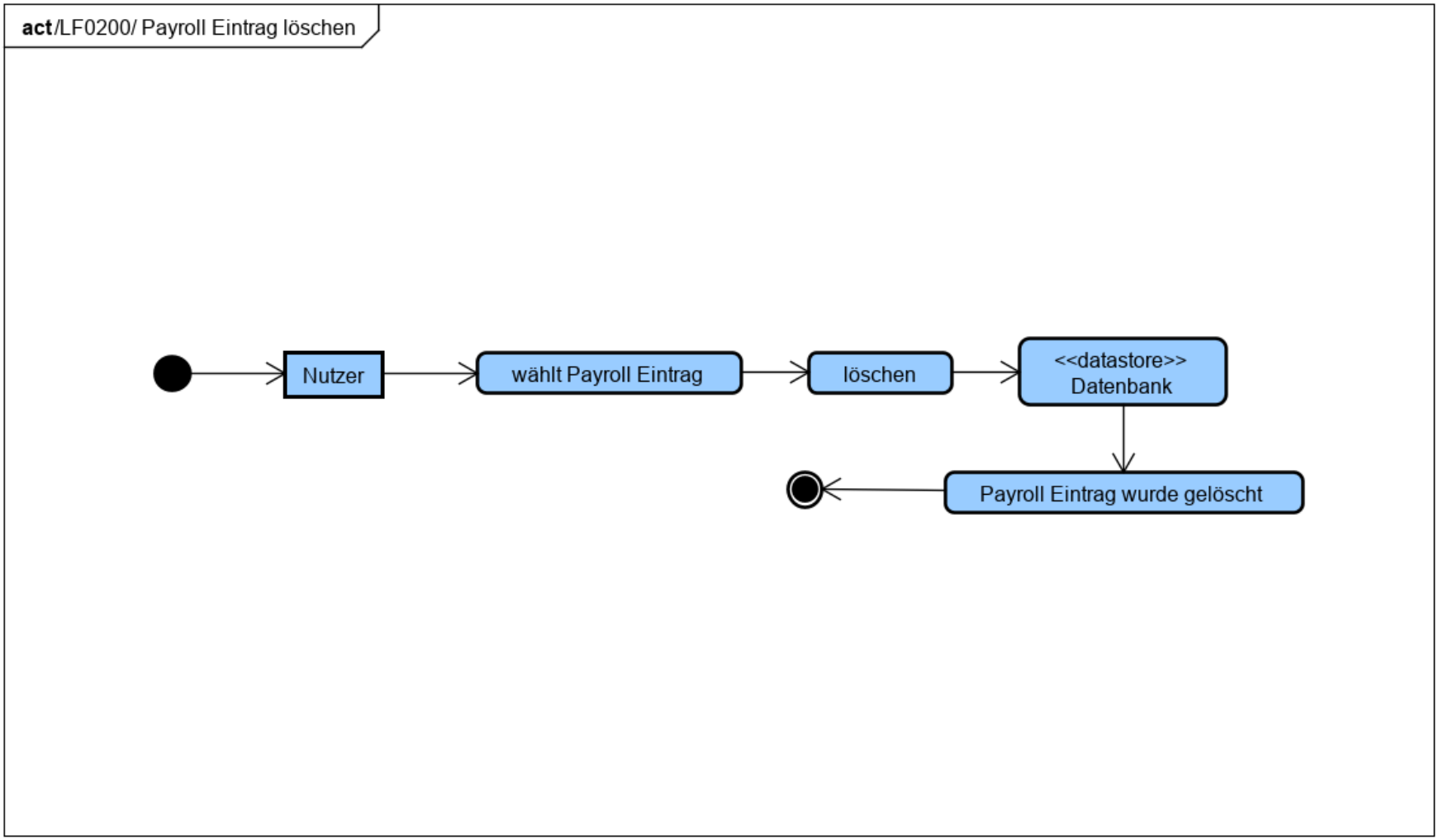


5.1.20 /LF0200/ Payroll Einträge löschen

5.1.20.1 Funktionsdefinition

<table>
<tr><th colspan="2">Funktion</th><th>Nutzen</th><th>Aufwand</th><th>Must Have/ Should Have/ Nice to Have</th></tr>
<tr><td>Use Case</td><td>Beschreibung</td><td rowspan="10">Hoch</td><td rowspan="10">Gering</td><td rowspan="10">Must Have</td></tr>
<tr><td>Name</td><td>Payroll Einträge löschen</td></tr>
<tr><td>Art</td><td>Anwendungsfall</td></tr>
<tr><td>Kurzbeschreibung</td><td>Der Administrator kann einen Payroll Eintrag löschen.</td></tr>
<tr><td>Auslöser</td><td>Der Administrator möchte eine Payroll Eintrag nicht mehr im System haben.</td></tr>
<tr><td>Ergebnis</td><td>Ein Payroll Eintrag wurde aus dem System gelöscht.</td></tr>
<tr><td>Akteur</td><td>Administrator</td></tr>
<tr><td>Daten</td><td>• Name des Payroll Eintrags</td></tr>
<tr><td>Vorbedingung</td><td>Es muss eine Payroll vorhanden sein.</td></tr>
<tr><td>Nachbereitung</td><td>Es können noch weitere Payroll Einträge gelöscht werden.</td></tr>
<tr><td>Vorgang</td><td>Der angemeldete Administrator geht auf die „Payroll“ Seite, auf dieser werden ihm nun die Payrolls angezeigt in dem er rechts auf den „Mülleimer“ Button drückt löscht er nun die einzelnen Payrolls.</td></tr>
</table>

5.1.20.2 Aktivitätsdiagramm

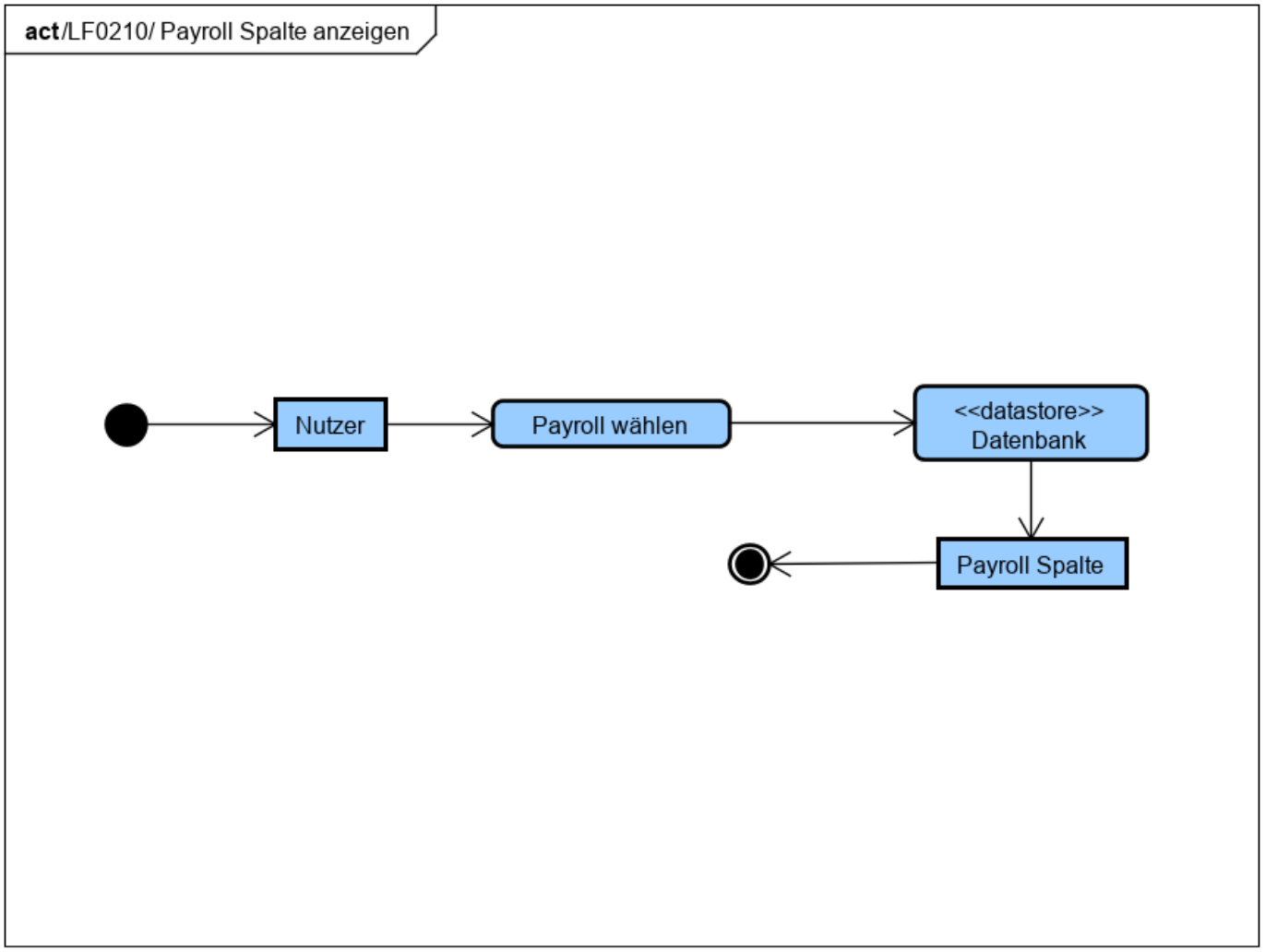


5.1.21 /LF0210/ Payroll Spalte anzeigen

5.1.21.1 Funktionsdefinition

<table>
<tr><th colspan="2">Funktion</th><th>Nutzen</th><th>Aufwand</th><th>Must Have/ Should Have/ Nice to Have</th></tr>
<tr><td colspan="2"><table>
<tr><th>Use Case</th><th>Beschreibung</th></tr>
<tr><td>Name</td><td>Payroll Spalte anzeigen</td></tr>
<tr><td>Art</td><td>Anwendungsfall</td></tr>
<tr><td>Kurzbeschreibung</td><td>Der Administrator kann sich seine Payrolls in Spalten anzeigen lassen.</td></tr>
<tr><td>Auslöser</td><td>Der Administrator möchte seine Payrolls angezeigt bekommen.</td></tr>
<tr><td>Ergebnis</td><td>Der Administrator bekommt eine Liste aus allen seinen Payrolls.</td></tr>
<tr><td>Akteur</td><td>Administrator</td></tr>
<tr><td>Daten</td><td>• Die Payrolls</td></tr>
<tr><td>Vorbedingung</td><td>Es müssen Payrolls vorhanden sein.</td></tr>
<tr><td>Nachbereitung</td><td>Die Payrolls können gelöscht werden.</td></tr>
<tr><td>Vorgang</td><td>Der angemeldete Administrator geht auf die „Payroll“ Seite, auf dieser werden ihm nun die Payrolls angezeigt.</td></tr>
</table></td><td>Hoch</td><td>Mittel</td><td>Must Have</td></tr>
</table>

5.1.21.2 Aktivitätsdiagramm

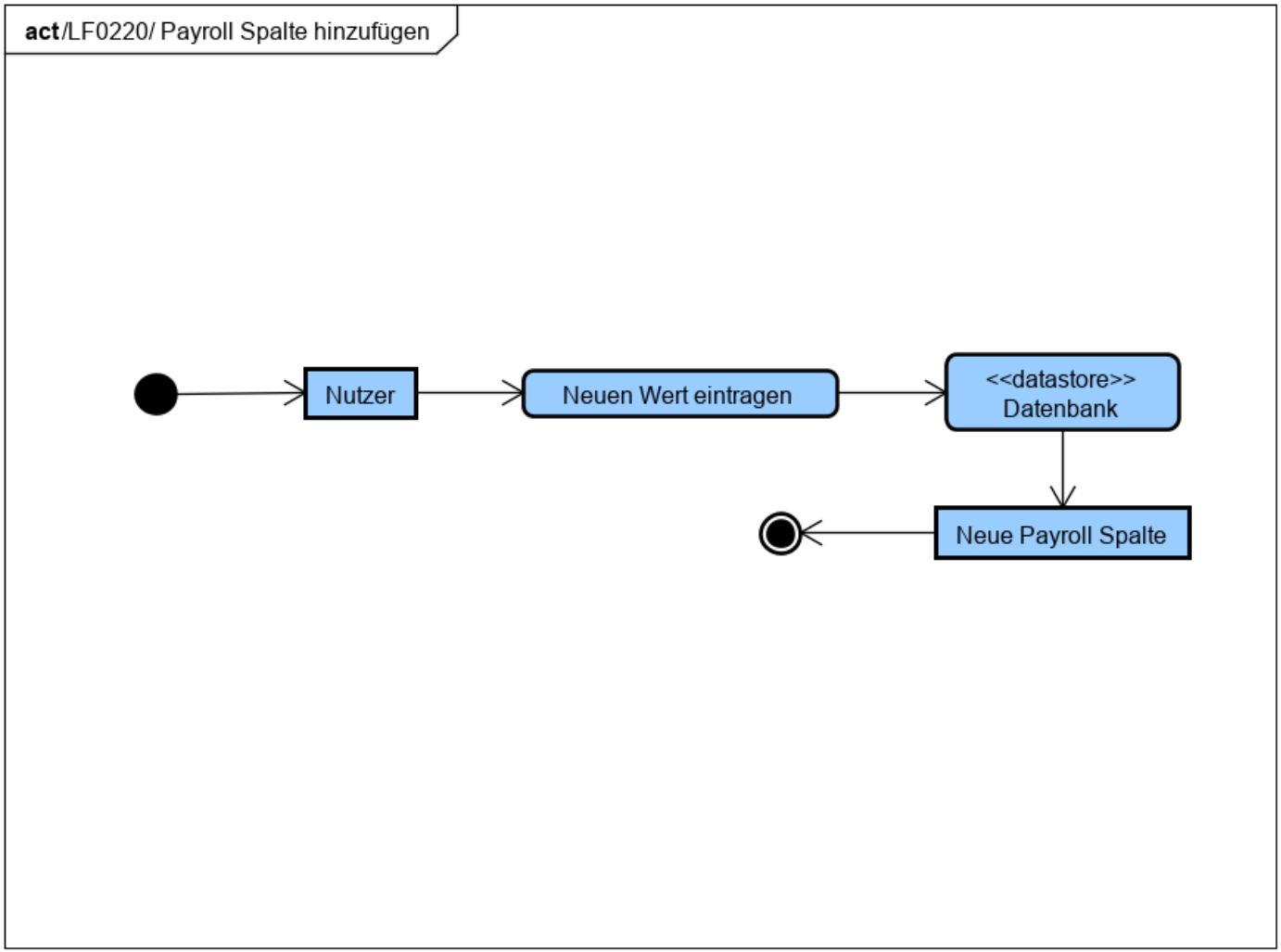


5.1.22 /LF0220/ Payroll Spalte hinzufügen

5.1.22.1 Funktionsdefinition

Funktion		Nutzen	Aufwand	Must Have/ Should Have/ Nice to Have
Use Case	**Beschreibung**	Hoch	Hoch	Must Have
Name	Payroll Spalte hinzufügen			
Art	Anwendungsfall			
Kurzbeschreibung	Der Administrator kann eine neue Payroll Spalte hinzufügen			
Auslöser	Der Administrator hätte gerne eine neue Spalte, um einen neuen Datensatz für seine Löhne zu erhalten.			
Ergebnis	Der Administrator hat eine neue Payroll Spalte.			
Akteur	Administrator			
Daten	• Name der Spalte			
Vorbedingung	Der Administrator muss wissen wie er die Spalte nennen will.			
Nachbereitung	Der Administrator kann diese Spalte wieder löschen.			
Vorgang	Der angemeldete Administrator geht auf die „Payroll“ Seite, auf dieser werden ihm nun die Payrolls angezeigt. In dem er rechts auf den „Wert hinzufügen“-Button drückt während er links, den Wert benannt hat fügt er nun die einzelnen Payroll Spalte hinzu.			

5.1.22.2 Aktivitätsdiagramm

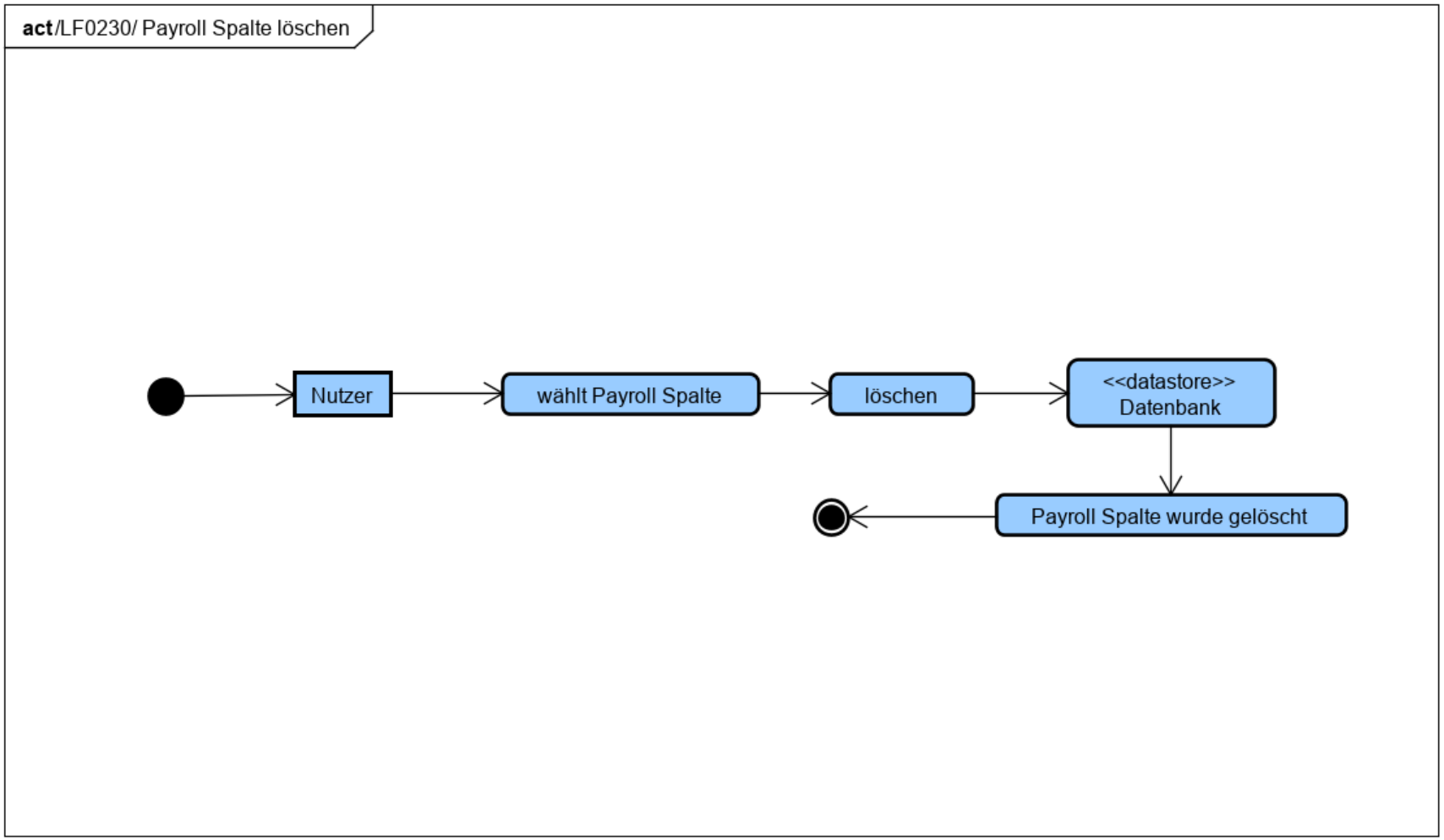


5.1.23 /LF0230/ Payroll Spalte löschen

5.1.23.1 Funktionsdefinition

Funktion		Nutzen	Aufwand	Must Have/ Should Have/ Nice to Have
Use Case	**Beschreibung**	Mittel	Gering	Must Have
Name	Payroll Spalte löschen			
Art	Anwendungsfall			
Kurzbeschreibung	Der Administrator kann eine Payroll Spalte löschen.			
Auslöser	Der Administrator benötigt eine Payroll Spalte nicht mehr.			
Ergebnis	Es ist eine Payroll Spalte weniger im System.			
Akteur	Administrator			
Daten	• Name der Spalte			
Vorbedingung	Es muss eine Payroll Spalte existieren.			
Nachbereitung	Es können neue Spalten erstellt werden.			
Vorgang	Der angemeldete Administrator geht auf die „Payroll“ Seite, auf dieser werden ihm nun die Payrolls angezeigt in dem er rechts auf den „Mülleimer“ Button drückt kann er nun die einzelnen Payrolls löschen.			

5.1.23.2 Aktivitätsdiagramm

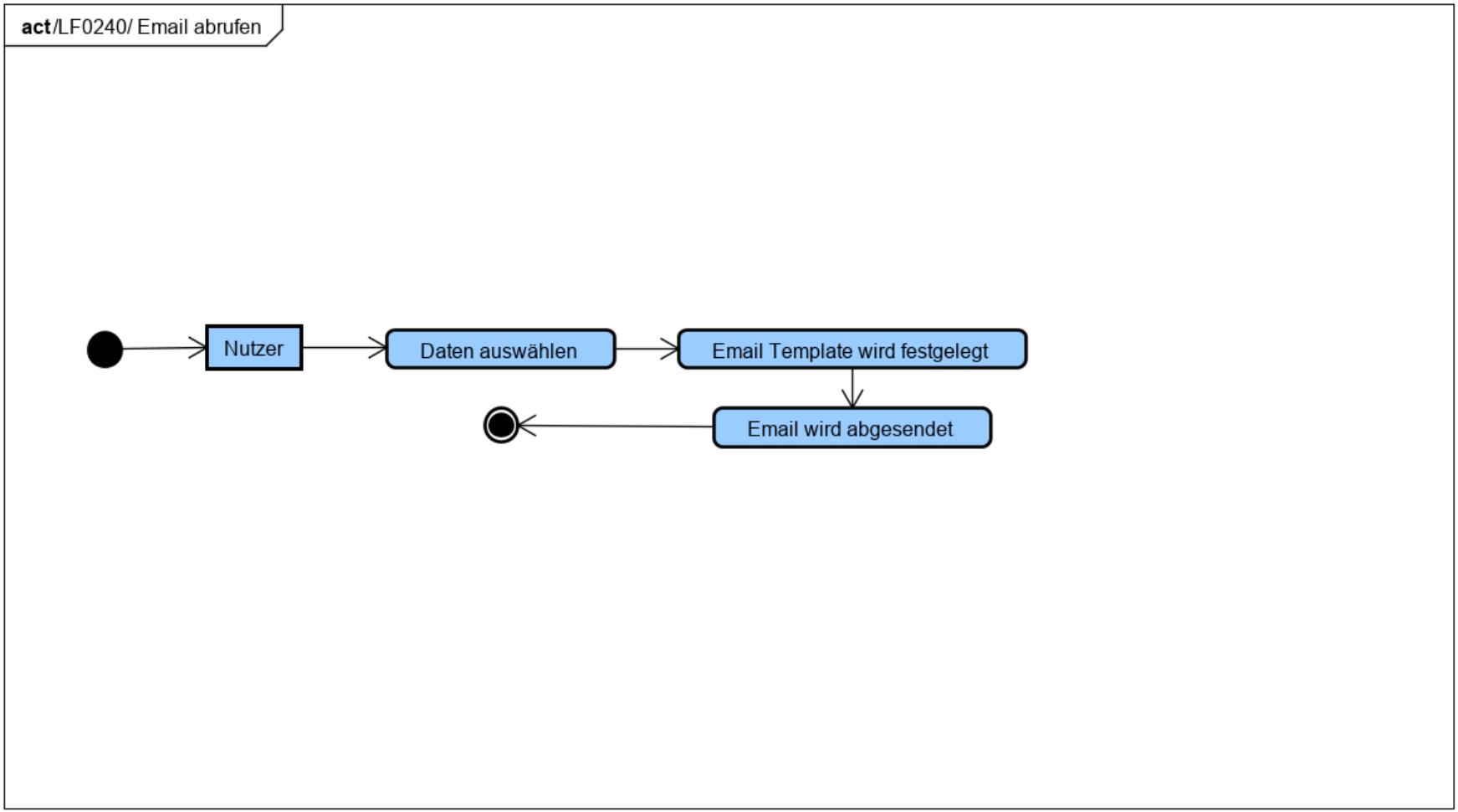


5.1.24 /LF0240/ Email abrufen

5.1.24.1 Funktionsdefinition

<table>
<tr><th colspan="2">Funktion</th><th>Nutzen</th><th>Aufwand</th><th>Must Have/ Should Have/ Nice to Have</th></tr>
<tr><td>Use Case</td><td>Beschreibung</td><td rowspan="11">Hoch</td><td rowspan="11">Mittel</td><td rowspan="11">Must Have</td></tr>
<tr><td>Name</td><td>Email abrufen</td></tr>
<tr><td>Art</td><td>Anwendungsfall</td></tr>
<tr><td>Kurzbeschreibung</td><td>Der Administrator kann sich eine E-Mail von dem Tagesgeschehens schicken lassen.</td></tr>
<tr><td>Auslöser</td><td>Der Administrator will wissen was der Tagesstand ist.</td></tr>
<tr><td>Ergebnis</td><td>Der Administrator erhält eine E-Mail mit den Daten des Tages.</td></tr>
<tr><td>Akteur</td><td>Administrator</td></tr>
<tr><td>Daten</td><td>• Uhrzeit
• Filter & Werten
• Betreff
• Beschreibung
• Empfänger</td></tr>
<tr><td>Vorbedingung</td><td>Es müssen Werte über den Tag eingetragen worden sein.</td></tr>
<tr><td>Nachbereitung</td><td>Der Administrator kann das Inhaltliche Template für weitere E-Mail ändern.</td></tr>
<tr><td>Vorgang</td><td>Der Administrator geht auf die „Email Einstellungen“ Seite auf dieser wählt er dann die Uhrzeit den Betreff und zwei Filter Plus einer Beschreibung aus. Er drückt nun auf „speichern“. Ein darunter gibt der Administrator auch noch die Empfänger angeben.</td></tr>
</table>

5.1.24.2 Aktivitätsdiagramm

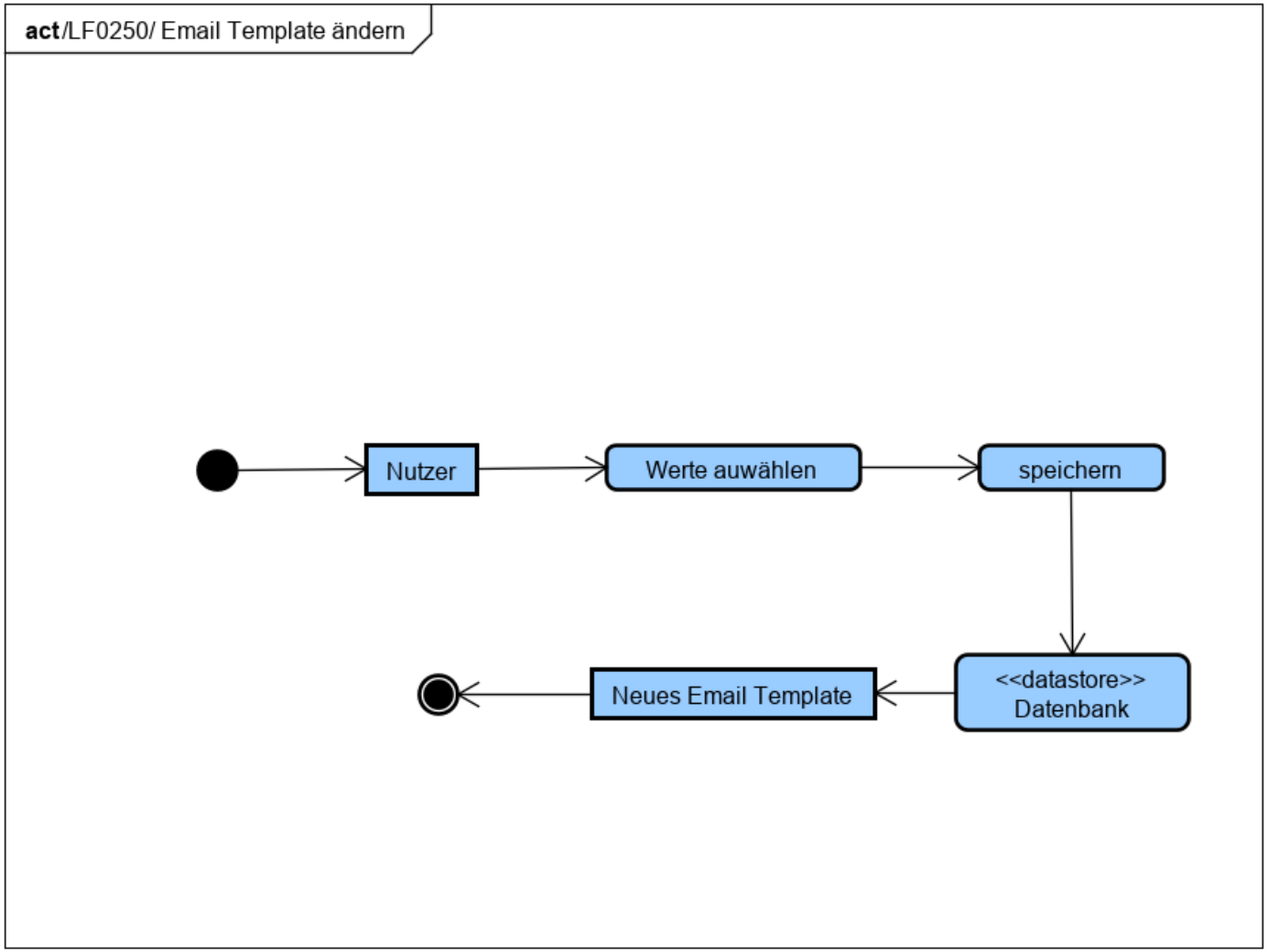


5.1.25 /LF0250/ Email Template ändern

5.1.25.1 Funktionsdefinition

<table>
<tr><th colspan="2">Funktion</th><th>Nutzen</th><th>Aufwand</th><th>Must Have/ Should Have/ Nice to Have</th></tr>
<tr><td>Use Case</td><td>Beschreibung</td><td rowspan="11">Mittel</td><td rowspan="11">Hoch</td><td rowspan="11">Must Have</td></tr>
<tr><td>Name</td><td>Email Template ändern</td></tr>
<tr><td>Art</td><td>Anwendungsfall</td></tr>
<tr><td>Kurzbeschreibung</td><td>Der Administrator kann den Inhaltlichen umfang seiner E-Mail bestimmen.</td></tr>
<tr><td>Auslöser</td><td>Der Administrator möchte andere Werte erhalten.</td></tr>
<tr><td>Ergebnis</td><td>Der Administrator bekommt eine E-Mail mit anderem Inhalt.</td></tr>
<tr><td>Akteur</td><td>Administrator</td></tr>
<tr><td>Daten</td><td>• Uhrzeit
• Filter
• Betreff
• Beschreibung
• Empfänger</td></tr>
<tr><td>Vorbedingung</td><td>Es müssen Filter und ein E-Mail Template vorhanden sein.</td></tr>
<tr><td>Nachbereitung</td><td>Der Administrator kann diese Einstellungen jederzeit ändern.</td></tr>
<tr><td>Vorgang</td><td>Der Administrator geht auf die „Email Einstellungen“-Seite auf dieser wählt er dann die Uhrzeit, den Betreff und zwei Filter Plus einer Beschreibung aus und überschreibt sie. Er drückt nun auf „speichern“.</td></tr>
</table>

5.1.25.2 Aktivitätsdiagramm

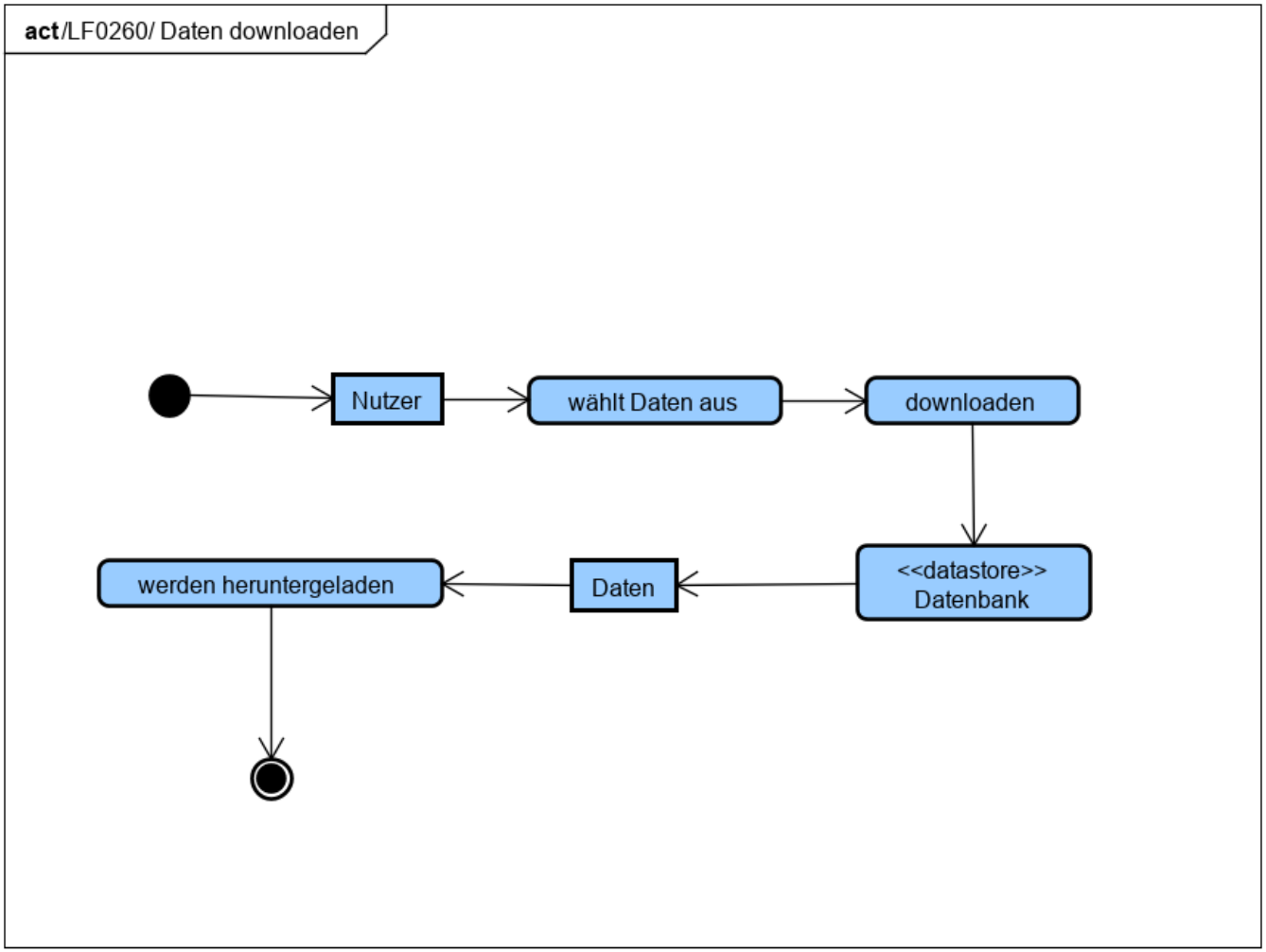


5.1.26 /LF0260/ Daten downloaden

5.1.26.1 Funktionsdefinition

Funktion		Nutzen	Aufwand	Must Have/ Should Have/ Nice to Have
Use Case	**Beschreibung**	Hoch	Hoch	Should Have
Name	Daten downloaden			
Art	Anwendungsfall			
Kurzbeschreibung	Der Administrator kann sich die Daten über einen definierten Zeitraum ausgeben lassen.			
Auslöser	Der Administrator möchte Informationen erhalten darum will er sich Daten aus einem bestimmten Zeitraum holen diese kann er sich herunterladen.			
Ergebnis	Der Administrator erhält eine Datei mit allen eingegebenen Informationen über einen von ihm festgelegten Zeitraum.			
Akteur	Administrator			
Daten	• Welche Art (UTC, Aufgelaufene) • Datum • Restaurants • Filter			
Vorbedingung	Es müssen Daten gesammelt worden sein.			
Nachbereitung	Man kann seine Daten abändern, um andere Zeiträume abzudecken.			
Vorgang	Der Administrator wählt zwischen Aufgelaufenen Daten oder UTC aus. Dementsprechend muss er auf die Seiten gehen, auf dieser wählt er nun nur den Zeitraum und welches Restaurant inkludiert werden sollen aus. Anschließend drückt er nur noch auf den „Download“-Button nun lädt er ein PDF mit seinen Daten herunter.			

5.1.26.2 Aktivitätsdiagramm



5.2 Mitarbeiter Funktionen

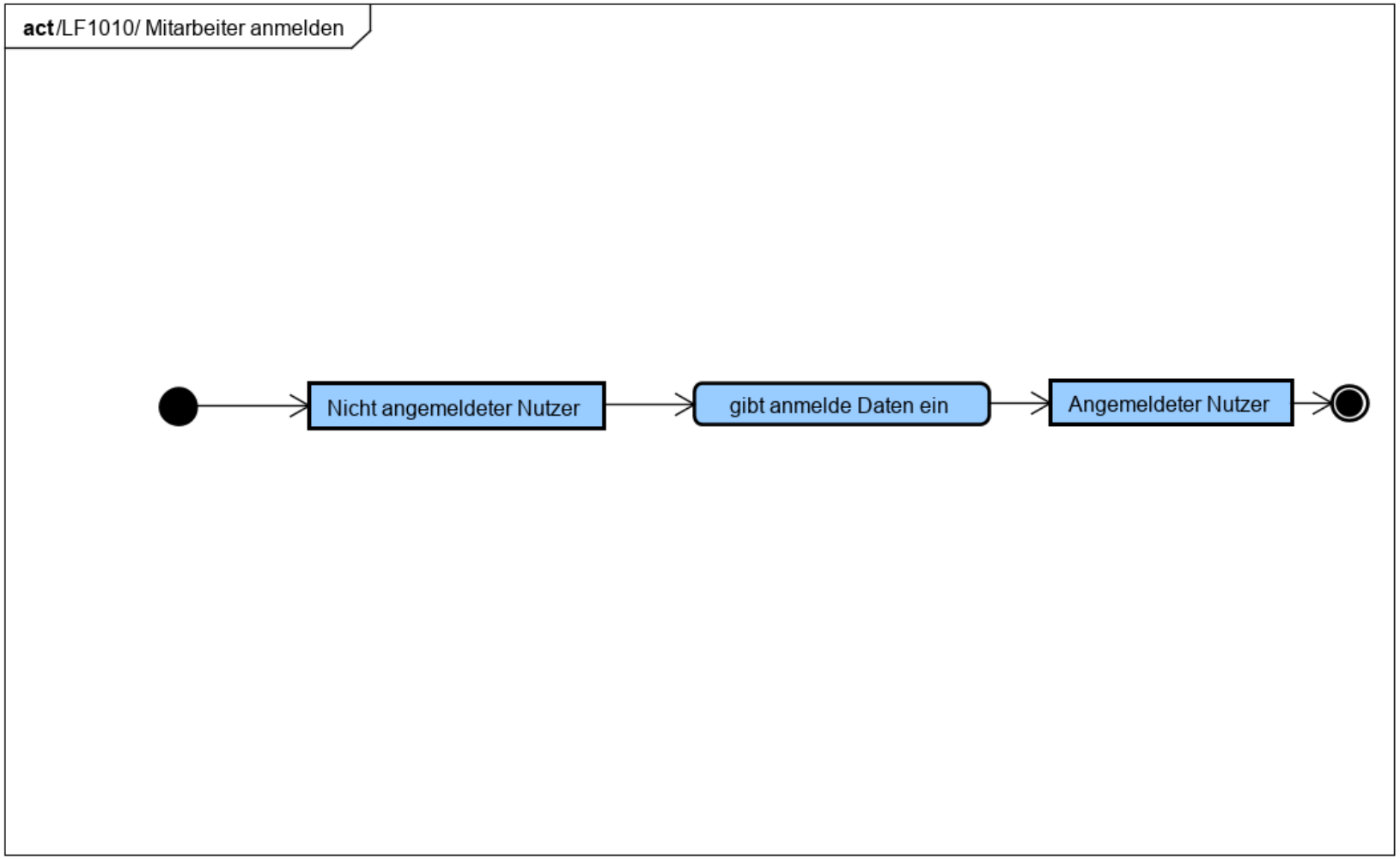
Die Mitarbeiter Funktionen beschreiben jegliche Aktion, welche die Mitarbeiter anhand ihrer verliehenen Rechte durchführen können.

5.2.1 /LF1010/ Mitarbeiter anmelden

5.2.1.1 Funktionsdefinition

Funktion		Nutzen	Aufwand	Must Have/ Should Have/ Nice to Have
Use Case	**Beschreibung**	Hoch	Gering	Must Have
Name	Mitarbeiter anmelden			
Art	Anwendungsfall			
Kurzbeschreibung	Der Mitarbeiter kann sich mit seinen Daten einloggen.			
Auslöser	Der Mitarbeiter möchte sich in das System einloggen.			
Ergebnis	Der Mitarbeiter ist im System angemeldet und kann seinen Rechten entsprechend agieren.			
Akteur	Mitarbeiter			
Daten	• Benutzername oder E-Mail-Adresse • Passwort			
Vorbedingung	Der Mitarbeiter muss seine Log-In Daten wissen.			
Nachbereitung	Der Mitarbeiter kann sich jederzeit abmelden.			
Vorgang	Der Nutzer drückt auf den „Anmelden“-Button und wird anschließend auf das „Anmelden“-Formular weitergeleitet. Das Formular füllt er nun und drückt auf den „anmelden“-Button			

5.2.1.2 Aktivitätsdiagramm

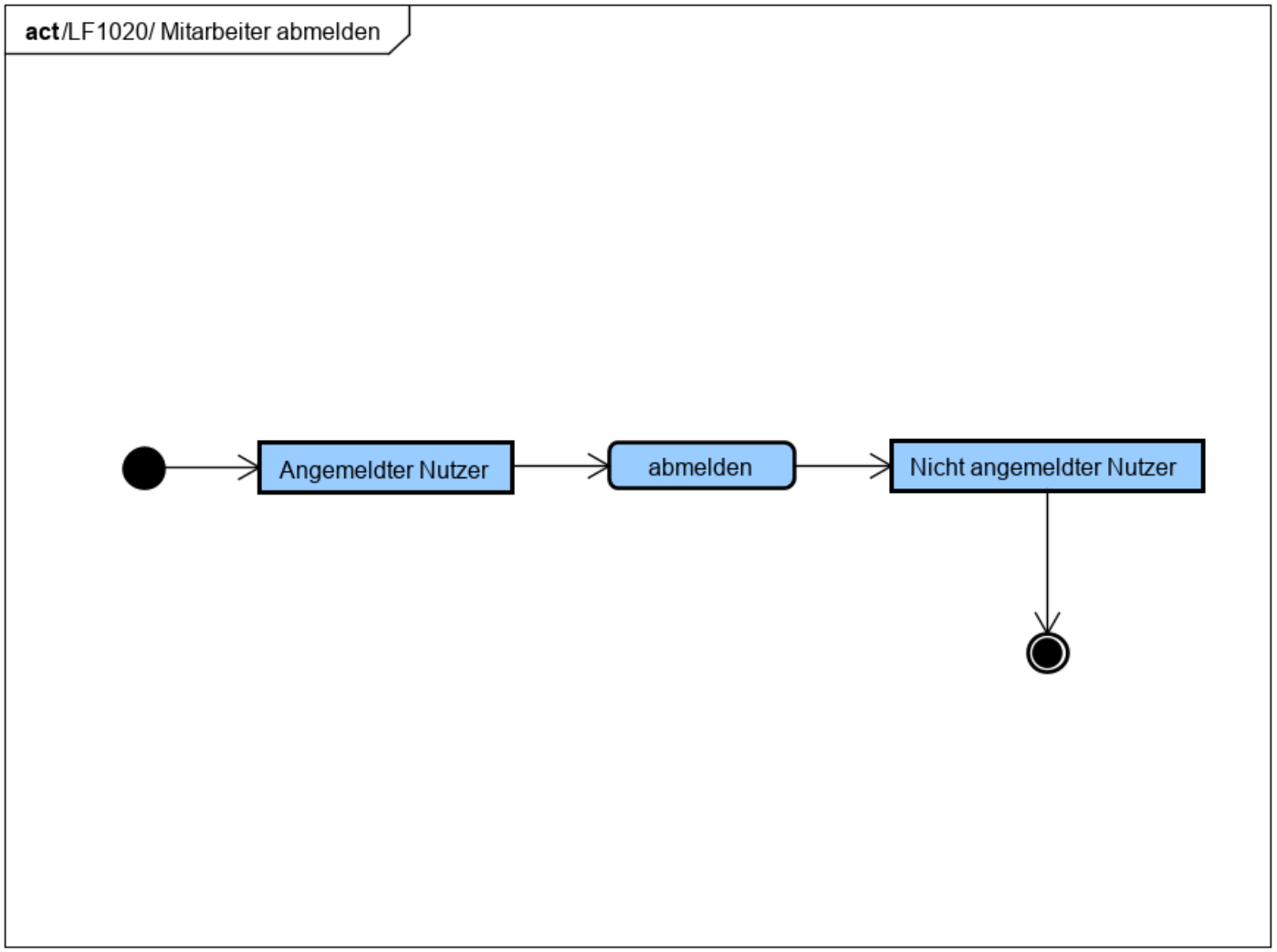


5.2.2 /LF1020/ Mitarbeiter abmelden

5.2.2.1 Funktionsdefinition

<table>
<tr><th colspan="2">Funktion</th><th>Nutzen</th><th>Aufwand</th><th>Must Have/ Should Have/ Nice to Have</th></tr>
<tr><td colspan="2">
<table>
<tr><th>Use Case</th><th>Beschreibung</th></tr>
<tr><td>Name</td><td>Mitarbeiter abmelden</td></tr>
<tr><td>Art</td><td>Anwendungsfall</td></tr>
<tr><td>Kurzbeschreibung</td><td>Der Mitarbeiter kann sich jederzeit abmelden.</td></tr>
<tr><td>Auslöser</td><td>Der Mitarbeiter hat seine Arbeit beendet und möchte sich aus dem System austragen.</td></tr>
<tr><td>Ergebnis</td><td>Der Mitarbeiter ist auf der Anmelde Seite und nicht mehr im System angemeldet.</td></tr>
<tr><td>Akteur</td><td>Mitarbeiter</td></tr>
<tr><td>Daten</td><td>Es werden keine zusätzlichen Daten benötigt.</td></tr>
<tr><td>Vorbedingung</td><td>Der Mitarbeiter muss angemeldet gewesen sein.</td></tr>
<tr><td>Nachbereitung</td><td>Der Mitarbeiter kann sich wieder anmelden.</td></tr>
<tr><td>Vorgang</td><td>Der Nutzer muss auf den „Abmelden“-Button drücken. Abschließend bekomm er wieder die Default-Ansicht der Website.</td></tr>
</table>
</td><td>Hoch</td><td>Gering</td><td>Must Have</td></tr>
</table>

5.2.2.2 Aktivitätsdiagramm

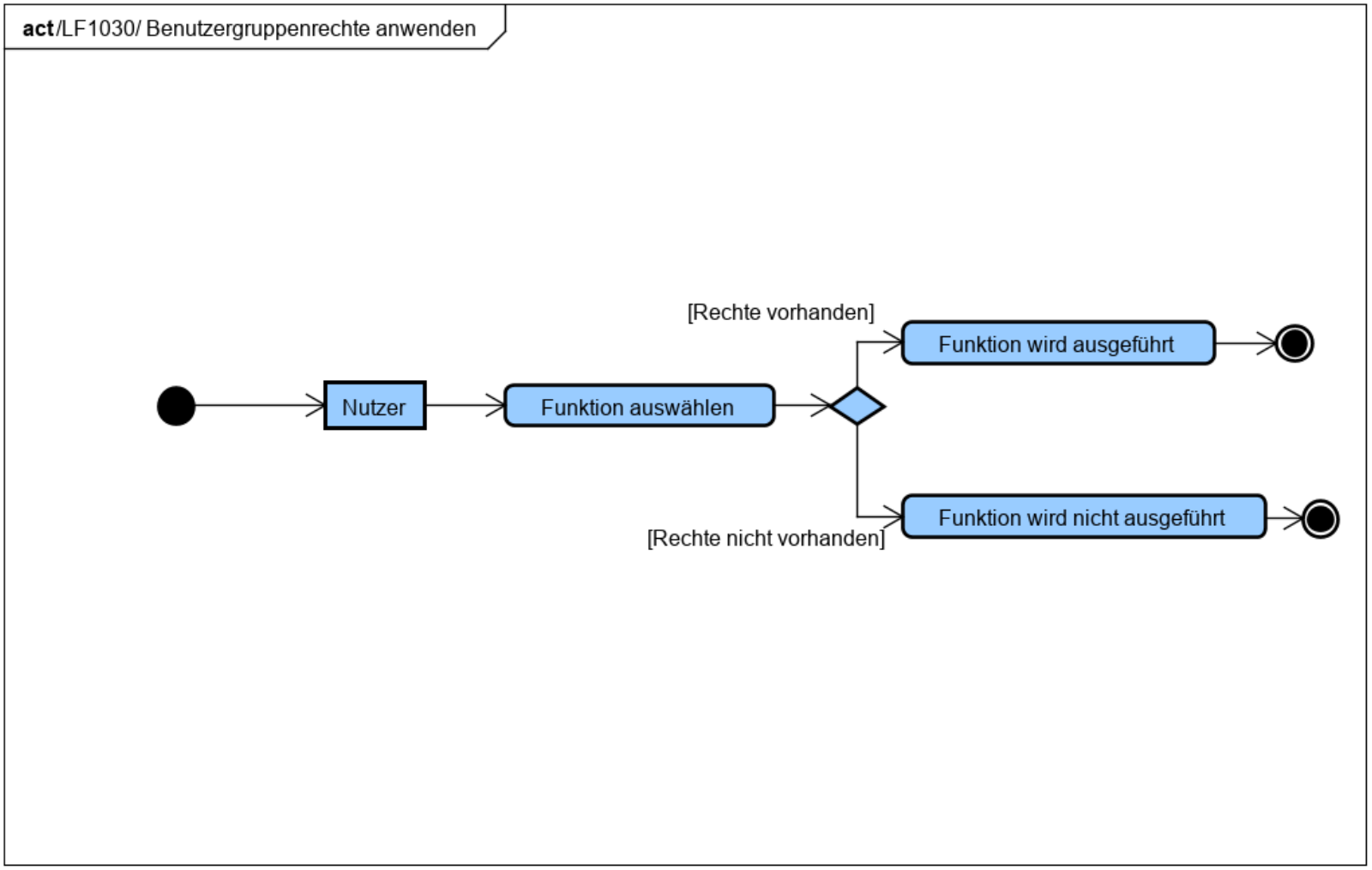


5.2.3 /LF1030/ Benutzergruppenrechte anwenden

5.2.3.1 Funktionsdefinition

<table>
<tr><th colspan="2">Funktion</th><th>Nutzen</th><th>Aufwand</th><th>Must Have/ Should Have/ Nice to Have</th></tr>
<tr><td colspan="2">
<table>
<tr><th>Use Case</th><th>Beschreibung</th></tr>
<tr><td>Name</td><td>Benutzergruppenrechte anwenden</td></tr>
<tr><td>Art</td><td>Anwendungsfall</td></tr>
<tr><td>Kurzbeschreibung</td><td>Jeder Benutzer wird bei der Registrierung, welche von ihrem Admin durchgeführt wird, einer Benutzergruppe zugewiesen. Anhand dieser Benutzergruppe können die Mitarbeiter die Funktionen eines Admins ausführen.</td></tr>
<tr><td>Auslöser</td><td>Die Mitarbeiter müssen arbeiten.</td></tr>
<tr><td>Ergebnis</td><td>Die Mitarbeiter können Local Admin verwenden.</td></tr>
<tr><td>Akteur</td><td>Mitarbeiter
Administrator</td></tr>
<tr><td>Daten</td><td>• Mitarbeiter
• Benutzergruppe
• Filiale</td></tr>
<tr><td>Vorbedingung</td><td>Es muss eine Lizenz von den Sysadmins vergeben worden sein.</td></tr>
<tr><td>Nachbereitung</td><td>Der Admin kann einem Benutzer eine andere Benutzergruppe zuweisen.</td></tr>
<tr><td>Vorgang</td><td>Sofern der Nutzer die nötigen Rechte besitzt wählt er die für ihn freigehaltenen Funktionen aus.</td></tr>
</table>
</td><td>Mittel</td><td>Mittel</td><td>Must Have</td></tr>
</table>

5.2.3.2 Aktivitätsdiagramm



5.3 Sysadmin Funktionen

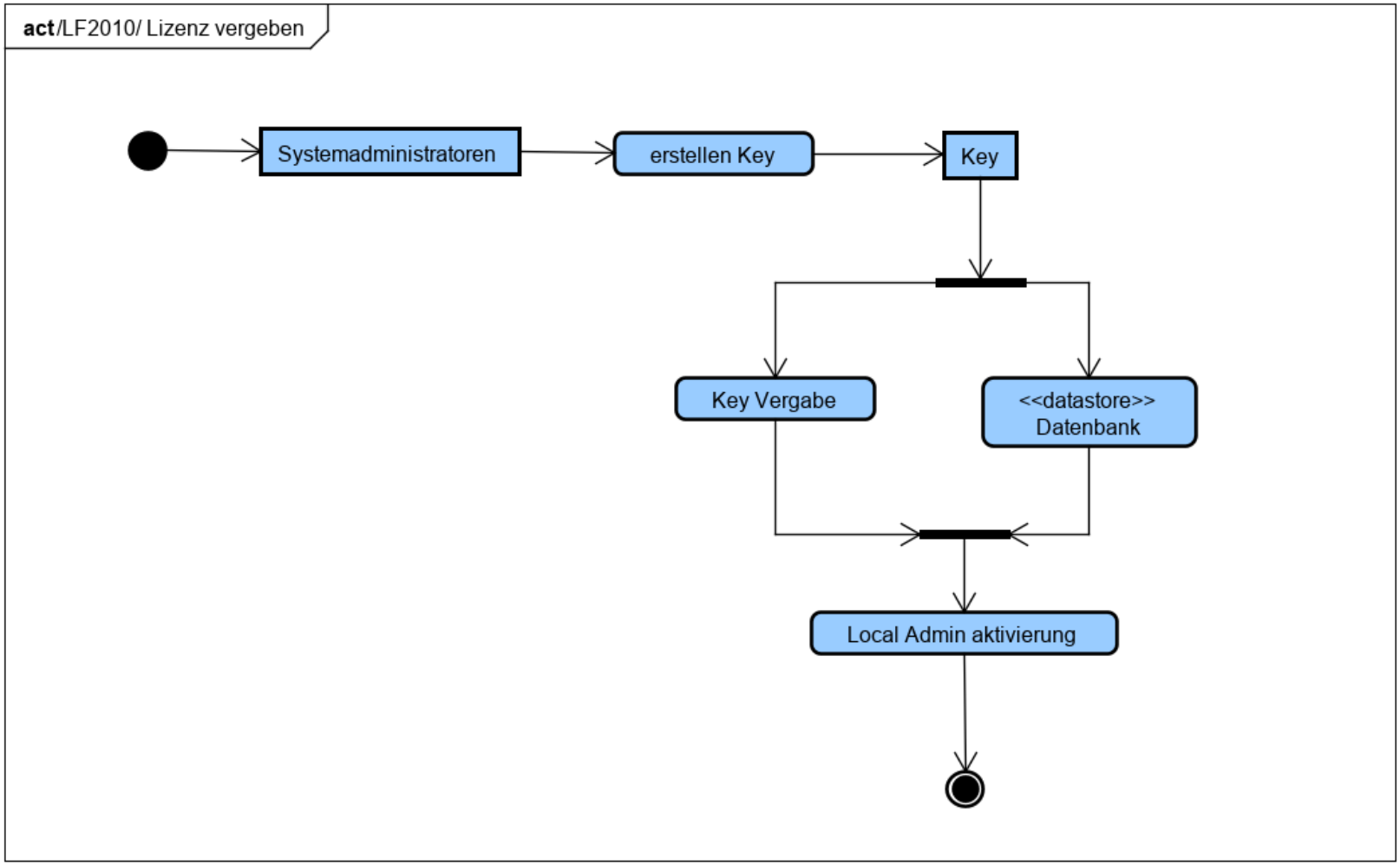
Die Sysadmin Funktionen beschreiben alle Funktionen, die die System Administratoren ausführen können.

5.3.1 /LF2010/ Lizenz verteilen

5.3.1.1 Funktionsdefinition

<table>
<tr><th colspan="2">Funktion</th><th>Nutzen</th><th>Aufwand</th><th>Must Have/ Should Have/ Nice to Have</th></tr>
<tr><td colspan="2"><table>
<tr><th>Use Case</th><th>Beschreibung</th></tr>
<tr><td>Name</td><td>Lizenz verteilen</td></tr>
<tr><td>Art</td><td>Anwendungsfall</td></tr>
<tr><td>Kurzbeschreibung</td><td>Der Sysadmin kann einer Filiale eine Lizenz geben. Mit dieser Lizenz kann die Filiale unsere Software verwenden.</td></tr>
<tr><td>Auslöser</td><td>Die Filiale möchte mit dem System arbeiten.</td></tr>
<tr><td>Ergebnis</td><td>Die Filiale kann mit unserem System arbeiten.</td></tr>
<tr><td>Akteur</td><td>Sysadmins
Administrator</td></tr>
<tr><td>Daten</td><td>• Filiale
• Lizenz-Key
• Admin Daten</td></tr>
<tr><td>Vorbedingung</td><td>Die Lizenzkosten müssen gedeckt sein.</td></tr>
<tr><td>Nachbereitung</td><td>Es gibt keine folgenden Aktionen.</td></tr>
<tr><td>Vorgang</td><td>Die Systemadministratoren geben dem Filialleiter einen Key, welchen er auf der Website eingeben kann um anschließend Einstellungen für den Administrator zu treffen.</td></tr>
</table></td><td>Hoch</td><td>Gering</td><td>Must Have</td></tr>
</table>

5.3.1.2 Aktivitätsdiagramm



6 Technische Machbarkeit

Es sind einige technische Risiken bekannt, die folgendermaßen gelöst werden. Die Daten (Benutzerdaten) werden auf einem Server in einer Datenbank gespeichert und direkt an die betroffenen Endgeräte gesendet, sobald sie vom Nutzer angefordert werden. Dies geschieht nur, wenn der Server und der Computer mit dem Internet verbunden sind.

6.1 Umsetzung

Da es in der Marktwirtschaft nur ein paar entsprechende Lösungsvarianten gibt, ist dieses Produkt mit all seinen Funktionen auf jedenfalls einzigartig. Das System wird zum Großteil in der Programmiersprache Java entwickelt. Der Vorteil liegt darin, dass das Team die meiste Erfahrung in dieser Sprache aufweist und somit wenig Einarbeitungszeit benötigt wird. Dazu kommt, dass Verzögerungen durch Fehler vermieden werden und die Arbeit im Allgemeinen schneller voranschreitet.

6.1.1 Kerntechnologien

6.1.1.1 Server von Burger King

Wir könnten einen internen Server von Burger King verwenden. Dies hätte den Vorteil, dass er sich auf lange Sicht refinanzieren würde und er unabhängig ist und daher nicht wegen externen Problemen abgeschaltet werden muss. Ein Nachteil ist das dieser gewartet werden muss.

6.1.1.2 Server von externem Anbieter

Um die Erreichbarkeit der Website zu ermöglichen, könnten wir auf einen externen Server zurückgreifen, welcher den Vorteil hat, dass keine Wartung betrieben werden muss. Jedoch kostet dieser jeden Monat einen Betrag zwischen 2 - 200 €.

6.1.2 Know-how

Da es in der Marktwirtschaft nur ein paar entsprechende Lösungsvarianten gibt, ist dieses Produkt mit all seinen Funktionen auf jedenfalls einzigartig. Das System wird zum Großteil in der Programmiersprache Java entwickelt. Der Vorteil liegt darin, dass das Team die meiste Erfahrung in dieser Sprache aufweist und somit wenig Einarbeitungszeit benötigt wird. Dazu kommt, dass Verzögerungen durch Fehler vermieden werden und die Arbeit im Allgemeinen schneller voranschreitet.

6.1.3 Risikophasen

Die risikoreichen Phasen sind erstens die Verknüpfung des Backend mit dem Frontend. Zweitens ist die Phase der Erstellung Datenbank eine interessante Phase, da bei falscher Kommunikation, Fehler in den Daten und dementsprechend im Aufbau der Webseite auftauchen könnten.

6.1.4 Variantenwahl

Durch intensive Diskussionen und Breefing innerhalb des Projektteams haben wir uns im Rahmen dieses Projektes für Variante 1 entschieden, da die Vorteile der ersten Variante, die der zweiten Variante übertreffen.

6.1.4.1 Variante 1

In Variante 1 würde das Team, mit den in den Nutzerwert herausgehenden Technologien, an einem von Burger King verwalteten Server arbeiten. Dies hat den Vorteil, dass wir jederzeit Zugriff auf den Server hätten und arbeiten könnten, ohne Einschränkungen oder Regeln.

6.1.4.2 Variante 2

In Variante 2 würde das Projektteam mit einem externen Server Anbieter arbeiten, was den Vorteil hätte, dass Burger King keine Verantwortung übernehmen müsste, was Serverwartung angeht. Weiters könnten sie jederzeit auf einen leistungsstärkeren Server umsteigen.

7 Wirtschaftliche Machbarkeit

7.1 Personenaufwand

Der Umfang des Projekts ist ziemlich groß und ein Server mit Datenbank muss auch konfiguriert werden. Dies wird einiges an Durchführungszeit benötigen. Dazu kommt noch, dass sich die Entwickler bei Problemen möglicherweise das nötige Wissen aneignen müssen, was zu einer zusätzlichen Zeit von etwa 10-20 Stunden führt.
Deshalb arbeiten alle vier Projektteammitglieder an diesem Werk, um eine effiziente Entwicklung zu ermöglichen. Die Gesamtstundenanzahl, die die Entwicklung des Systems und die Konfiguration des Servers fordern, beträgt in etwa 250 Stunden. Diese werden auf alle Mitglieder aufgeteilt, sodass das System innerhalb kürzester Zeit fertiggestellt werden kann.

7.2 Investitionsaufwand

Es gibt einen Investitionsaufwände von etwa 9800€ durch Personenkosten. Das komplette Hardwaresortiment ist vorhanden und gewährleistet erfolgreiches Arbeiten. Die benötigten Softwarepakete sind ebenfalls vorhanden und fordern daher keine zusätzlichen Kosten an.

7.3 Nutzen

Durch das Verwenden von Local Admin hat man, durch die Online Archivierung der Daten, enormen Platz- und Papiereinsparungen. Die Platzeinsparungen sollten sich auf etwa ~$3\mathrm{m}^3$ belaufen. Darüber hinaus soll sich eine Zeitersparnis von bis zu 6 Stunden/Woche herausschlagen lassen, da das Suchen von Informationen erleichtert wird. Auch das Erstellen von Dienstplänen bzw. Einträgen wird einfacher gestaltet.

7.4 Risikoanalyse

Bei diesem Projekt sind einige Risiken vorhanden, wie bei jedem anderen Projekt auch. Einer dieser Faktoren könnte eventuell eine längere, gesundheitlich bedingte, Abwesenheit vom Arbeitsplatz sein. Für diesen Fall werden die übrigen Personen den zu bewältigenden Teil übernehmen. Dieser Fall wird hoffentlich relativ selten eintreten.

Ein kritischer Faktor bei der Umsetzung der Projektziele kann die Teamarbeit werden. Es ist wichtig, dass das Arbeitsklima unter den Mitarbeitern im Einklang ist, da man nur als Team etwas erreichen kann.

Ein Problem, welches auch jederzeit eintreten kann, ist ein Virus auf den Arbeitsgeräten. Das ist leider nicht vorhersehbar und kann verheerende Folgen haben. Man sollte deshalb zwei Back-ups in der Woche machen, um einen zu großen Datenverlust zu verhindern. Diese Back-ups werden extern gesichert, dass diese Daten nicht vernichtet werden können.

Eine weitere Fehlerquelle kann die Kommunikation mit dem Server sein. Es werden möglichst viele Fälle von Datenmanipulation verhindert werden, um ein reibungsloses Arbeiten mit der Software zu gewährleisten.

8 Persönliche Machbarkeit

8.1 Kriterien Begründung

8.1.1 Sprache

8.1.1.1 Kosten

Die Kosten haben für uns eine hohe Priorität da nicht alle Sprachen gratis ihre Lizenzen für gewinnbringende Projekte vergeben. Daher die hohe Gewichtung.

8.1.1.2 Sicherheit

Da es sich bei Local Admin um ein Finanzverwaltungssystem handelt, sind die Daten äußerst vertraulich zu behandeln. Darum sollte die zu wählende Sprache so wenig Sicherheitslücken wie möglich aufweisen.

8.1.1.3 Performance

Die Performance ist ebenfalls ein wichtiger Punkt der Sprache, da der Server permanent im Hintergrund laufen wird. Schlanke Funktion sind zu empfehlen, um das schnelle und effiziente Arbeiten mit dem Produkt zu gewährleisten

8.1.1.4 Datenbankzugriff

Es wäre ebenfalls empfehlenswert, wenn die Sprache das Zugreifen auf eine Datenbank ermöglicht bzw. unterstützt.

8.1.1.5 Entwicklungsgeschwindigkeit

Da ein relativ straffes Zeitfenster bis zur Vervollständigung des Produktes herrscht, sollte die Entwicklungsgeschwindigkeit möglichst gering sein, damit das Projekt ein Erfolg wird. Daher ist hier die Gewichtung am höchsten..

8.1.1.6 Erweiterbarkeit

Als eine Sprache, welche viele verschiedene Möglichkeiten bietet, sollte diese durch APIs und Konsorten erweiterbar sein, damit diese positiv zur Vollendung des Projektes beitragen.

8.1.1.7 Features

Um das Arbeiten mit den Sprachen angenehmer zu bereiten, sollten diese Features beinhalten, wie z.B. OOP.

8.1.1.8 Ausführendes OS

Das ausführende OS spielt eine maßgebliche Rolle, da die Kompatibilität mit Windows und Linux gegeben sein sollte, damit das Backend mit dem Server bzw. dem Frontend kommunizieren kann.

8.1.2 Lizenzserver

8.1.2.1 Performance
Der Lizenzserver sollte eine gute und schnelle Arbeit gewährleisten können.

8.1.2.2 Features
Der Lizenzserver sollte für den Umgang mit einem jenen solchen, Features wie Autocomplete oder Standardbibliotheken enthalten.

8.1.2.3 Datenbankzugriff
Der Lizenzserver benötigt Zugriff auf die Datenbank. Hier spielt vor allem die Zeit eine Rolle. Darum wird hier auf Geschwindigkeit und Menge an Code geschaut, um unsere Mühen so klein wie möglich zu halten, aber dennoch das best mögliche Ergebnis zu erzielen.

8.1.2.4 Sicherheit
Die Sicherheit des Lizenzservers muss auf jeden Fall gegeben sein, da dieser Daten enthält welche Diskret behandelt werden sollen.

8.1.2.5 Komplexität
Der Lizenzserver sollte allerdings auch nicht zu komplex sein, da sonst das Neuerlernen des notwendigen Wissens einen enormen Zeitaufwand darstellen würde, was die pünktliche Abnahme des Projektes verzögern könnte.

8.1.2.6 Skalierbarkeit
Mit der Skalierbarkeit ist der Umfang des Servers gemeint, sprich, Erweiterungen und Modularität. Diese können das Arbeiten erleichtern, da so gewisse Funktionen bereits vorhanden sind, anstatt diese selbst neu entwickeln zu müssen.

8.1.2.7 Aktualisierbarkeit
Mit der Aktualisierbarkeit ist die Neuartigkeit des Severs gemeint. Wie oft wird er gewartet auf, welcher Version läuft er, usw.... Dies ist wichtig, da durchaus neue Hard-/Software, neue Features und Bugfixes mit sich bringen, welche uns die Arbeit erleichtern.

8.1.2.8 Erfahrung
Die Erfahrung beschreibt in diesem Falle unsere praktische und theoretische Erfahrung mit den verschiedenen Lizenzservern.

8.1.3 Framework

8.1.3.1 Hilfestellung
Mit Hilfestellung sind die Dokumentation und Hilfe gemeint, welche uns bei der Entwicklung des Produktes helfen soll.

8.1.3.2 Performance
Das Framework muss performant sein, um Verzögerungen im späteren Betreib zu verhindern.

8.1.3.3 Aktualität
Das Framework sollte möglichst aktuell sein, um Bugs zu vermeiden und mögliche Erweiterungen nützen zu können, welche erst in neueren Versionen hinzugefügt wurden.

8.1.3.4 Addins
Das Framework sollte erweiterbar sein, um das Arbeiten und vor allem die Zeit zu verringern, da wir so schon bereits entwickelten Code verwenden können anstelle der vollständigen Neuentwicklung.

8.1.3.5 Entwicklungsgeschwindigkeit
Die Entwicklungsgeschwindigkeit des Frameworks spielt eine große Rolle, da wir das Framework zum Teil noch erlernen müssen. Deswegen darf die Komplexität und der Umfang nicht allumfassend sein, da ansonsten der Zeitrahmen gesprengt wird.

8.1.3.6 Wissen
Das Wissen beschreibt unsere bisher gesammelten Erfahrungen mit den verschiedenen Frameworks.

8.1.3.7 Features
Die Frameworks sollten Features enthalten, welche uns das Arbeiten erleichtern.

8.1.4 Dokumente

8.1.4.1 Performance
Die Performanz der Dokumente bzw. der Arbeit mit den Dokumenten sollte relativ hoch sein, damit dies schnell von dannen geht und die Programmierer mit dem Code beginnen können.

8.1.4.2 Kompatibilität
Die Kompatibilität sollte das Arbeiten mit anderen Formaten darstellen z.B. Excel und Word.

8.1.4.3 Benutzerfreundlichkeit
Die Dokumenterstellungssoftwares sollten einfach zu erlernen und bedienen sein.

8.1.4.4 Erfahrung
Beschreibt unsere eigenen, bisher gesammelten, Erfahrung mit den verschiedenen Möglichkeiten.

8.1.4.5 Aktualität
Die Aktualität beschreibt die Frequenz der Updates der verschiedenen Möglichkeiten, da neue Versionen meist weniger Bugs beinhalten oder neue Features haben, welche das Arbeiten erleichtern können.

8.1.5 Datenbankzugriff

8.1.5.1 Kompatibilität
Die Datenbankzugriffe sollten mit jeder Art von Datenbank kompatibel sein, ob das nun MySQL oder SQL2O ist.

8.1.5.2 Benutzerfreundlichkeit
Die Menge an Code, die geschrieben werden muss, um Ziele zu erreichen, spielt bei Datenbankabfragen eine große Rolle, da diese sehr zeitaufwändig sein können.

8.1.5.3 Erfahrung
Beschreibt die bisher gesammelten Erfahrungen mit den verschiedenen Möglichkeiten.

8.1.5.4 Aktualität
Je neuer das System der Datenbankabfrage ist, desto höher ist die Wahrscheinlichkeit, dass es verschiedene Typen von Datenbanken unterstützt und möglicherweise fehlerfreier ist.

8.1.6 Datenbank

8.1.6.1 Performance
Die Datenbank sollte eine relativ gute Performance aufweisen da diese in konstanter Benutzung sein wird. Falls diese nicht performant ist könnte dies das Arbeiten ineffizient gestalten.

8.1.6.2 Kompatibilität
Die Datenbank sollte nach Standards konfiguriert worden sein damit auch Teams nach uns diese verwenden können.

8.1.6.3 Softwarefeatures
Die Datenbank sollte Features enthalten, welche das Arbeiten erleichtern.

8.1.6.4 Softwareunterstützung
Die Kompatibilität mit verschieden Sprachen, um das Arbeiten universell angenehmer zu gestalten.

8.1.6.5 Benutzerfreundlichkeit
Die Benutzerfreundlichkeit beschreibt in diesem Fall den Aufwand, der betrieben werden muss, um die Datenbank aufzusetzen und wie schwer es ist, sich einzulesen.

8.1.6.6 Erfahrung
Die Erfahrung beschreibt unsere gesammelten Erfahrungen mit den verschiedenen Varianten.

8.1.6.7 Aktualität
Eine möglichst aktuelle Datenbank wäre zu empfehlen da diese meist Bugfreier sind und mehr Features enthalten.

8.1.7 Datenformat

8.1.7.1 Performance
Das Datenformat sollte möglichst Performant sein da große Dateien möglicherweise zu Verzögerungen im Betrieb führen könnte.

8.1.7.2 Kompatibilität
Die Dateiformate sollten möglichst kompatibel sein sprich eine Standardisierung besitzen.

8.1.7.3 Softwareunterstützung
Wie viele Parser und oder Writer es gibt.

8.1.7.4 Benutzerfreundlichkeit
Wie einfach die Datenformate zu erstellen sind und wie komplex sie zum Nachlernen sind.

8.1.7.5 Erfahrung
Unsere Erfahrungen mit den verschiedenen Dateiformaten.

8.1.7.6 Aktualität
Das Alter der verschiedenen Dateiformate da neuere Dateiformate meist eine einheitliche Syntaxstandardisierung besitzen und daher besser geeignet sind.

8.1.8 Authentifikations Model

8.1.8.1 Sicherheit
Die Sicherheit des Systems ist ein großer Punkt, darum wird beim Authentication Model besonders darauf geachtet das dieses unser System schützt.

8.1.8.2 Performance
Das Authentication Model sollte schnell funktionieren.

8.1.8.3 Erfahrung
Unsere bereits gesammelte Erfahrung mit den verschiedenen Varianten.

8.1.8.4 Aktualität
Die Neuartigkeit des Authentication Models ist meist sehr hilfreich, da dieses Model neue Bedrohungen bereits kennt und dagegen nicht mehr so anfällig ist.

8.1.8.5 Komplexität
Die Komplexität beschreibt wie schwer uns die Arbeit damit fallen wird.

8.1.8.6 Entwicklung
Die Entwicklungs-/Implementierungszeit sollte relativ kurz sein, um den Zeitplan einzuhalten.

8.1.9 Core Testing

8.1.9.1 Hilfestellung
Mit Hilfestellung sind die Dokumentation und Hilfe gemeint, welche uns bei der Entwicklung des Produktes helfen soll-Performance

8.1.9.2 Aktualität
Die Core Testing Software sollte relativ neuartig sein, um auch neue Testmethoden durchführen zu können.

8.1.9.3 Entwicklungsgeschwindigkeit
Die Entwicklungsgeschwindigkeit ist ein wichtiger Punkt da wir für das Testen relativ viel Zeit brauchen und wir sehr genau sein müssen.

8.1.9.4 Wissen
Unsere bereits gesammelte Erfahrung mit den verschiedenen Varianten.

8.1.9.5 Features
Die Core Testing Varianten sollten Features enthalten, welche uns das Testing erleichtern.

8.1.10 API Testing

8.1.10.1 Hilfestellung
Die vorhandene Hilfestellung, welche uns das Debuggen der API vereinfachen sollte.

8.1.10.2 Kompatibilität
Die Kompatibilität aller möglichen zu testenden Komponenten.

8.1.10.3 Aktualität
Die Aktualität beschreibt die Neuartigkeit und die Frequenz der Updates der verwendeten Software.

8.1.10.4 Entwicklungsgeschwindigkeit
Die von uns benötigte Entwicklungsgeschwindigkeit anhand von Code Menge und dem Erlernen von Testing Methoden.

8.1.10.5 Wissen
Das Wissen welches das Projektteam bereits über die verschiedenen Technologien gesammelt hat.

8.1.10.6 Features
Die Testing Varianten sollten Features enthalten, welche uns das Arbeiten erleichtern.

8.1.11 API Design

8.1.11.1 Hilfestellung

Die Community, offizieller Support oder Dokumentation welche uns helfen soll das Testing effizient und genau durchzuführen.

8.1.11.2 Aktualität

Die Aktualität der zu verwendeten Software und deren Kompatibilität mit unserem System.

8.1.11.3 Entwicklungsgeschwindigkeit

Beschreibt die menge an Code die wir benötigen, um unsere Ziele zu erreichen.

8.1.11.4 Wissen

Das bisher vom Projekt Team gesammelten Wissen zu den einzelnen Technologien.

8.1.11.5 Features

Die Testing Software solle Features besitzen, welche uns das Arbeiten erleichtert.

8.2 Nutzwertanalysen

8.2.1 Sprache

Sprache	Gewichtung	Teilkriterien	Gewichtung	Teil Gewichtung	Java		C++		Node JS		ASP.Net (C#)	
					Rang	G*R	Rang	G*R	Rang	G*R	Rang	G*R
Kosten	11	Lizens	1,1	10	5	5,5	5	5,5	4	4,4	4	4,4
		Entwickler	6,05	55	3	18,15	2	12,1	4	24,2	3	18,15
		Zusatzsoftware	3,85	35	5	19,25	3	11,55	4	15,4	4	15,4
		Gesamt	11	100	13	42,9	10	29,15	12	44	11	37,95
Sicherheit	10	Standardbibliotheken	7	70	5	35	4	28	3	21	5	35
		Zusatzbibliotheken	0,5	5	3	1,5	4	2	2	1	3	1,5
		Zusatzsoftware	2,5	25	5	12,5	5	12,5	4	10	5	12,5
		Gesamt	10	100	13	49	13	42,5	9	32	13	49
Performance	13	Networking	5,2	40	4	20,8	5	26	3	15,6	4	20,8
		Verschlüsselungen	2,6	20	3	7,8	5	13	1	2,6	3	7,8
		File I/O	5,2	40	3	15,6	5	26	1	5,2	3	15,6
		Gesamt	13	100	10	44,2	15	65	5	23,4	10	44,2
Datenbankzugriff	17	Geschwindigkeit	10,2	60	4	40,8	4	40,8	3	30,6	4	40,8
		Verwendbare Banken	5,1	30	5	25,5	1	5,1	5	25,5	4	20,4
		Coding dauer	1,7	10	3	5,1	0	0	5	8,5	3	5,1
		Gesamt	17	100	12	71,4	5	45,9	13	64,6	11	66,3
Entwicklungsgeschwindigkeit	21	API	10,5	50	3	31,5	2	21	5	52,5	3	31,5
		Algorithmen	2,1	10	4	8,4	5	10,5	1	2,1	4	8,4
		Datenbankzugriff	8,4	40	3	25,2	1	8,4	5	42	3	25,2
		Gesamt	21	100	10	65,1	8	39,9	11	96,6	10	65,1
Erweiterbarkeit	11	Online APIS	1,43	13	2	2,86	1	1,43	5	7,15	3	4,29
		Offline API	5,5	50	5	27,5	4	22	0	0	5	27,5
		API Qualität	4,07	37	4	16,28	5	20,35	3	12,21	4	16,28
		Gesamt	11	100	11	46,64	10	43,78	8	19,36	12	48,07
Features	12	High Level Features	5,4	45	5	27	1	5,4	4	21,6	5	27
		OOP	5,4	45	5	27	4	21,6	1	5,4	5	27
		Low Level Features	1,2	10	2	2,4	5	6	0	0	3	3,6
		Gesamt	12	100	12	56,4	10	33	5	27	13	57,6
Ausführendes OS	5	Windows	2,5	50	4	10	4	10	4	10	5	12,5
		MAC OS X	0	0	5	0	2	0	3	0	0	0
		Linux	2,5	50	4	10	4	10	5	12,5	1	2,5
		Gesamt	5	100	13	20	10	20	12	22,5	6	15
Summe	100		100		94	395,64	81	319,23	75	329,46	86	383,22
Der Rang ist eine bewertung von 0 bis 5 wobei 5 100% und 0 0% entspricht				REIHUNG	1		4		3		2	

8.2.2 Lizenzserver

Lizenzserver	Gewichtung	Teilkriterien	Gewichtung	Teil Gewichtung	Java Spring		NodeJs ExpressJs		Python Django	
					Rang	G*R	Rang	G*R	Rang	G*R
Performance	10	Fehlertoleranz	3,4	34	4	13,6	2	6,8	3	10,2
		Ressourcenverbrauch	3,3	33	4	13,2	4	13,2	4	13,2
		Geschwindigkeit	3,3	33	3	9,9	4	13,2	3	9,9
		Gesamt	10	100	11	36,7	10	33,2	10	33,3
Features	17	Einfachkeit	5,95	35	2	11,9	4	23,8	3	17,85
		Standardbibliotheken	6,8	40	4	27,2	2	13,6	4	27,2
		Autogenerierter Code	4,25	25	4	17	3	12,75	3	12,75
		Gesamt	17	100	10	56,1	9	50,15	10	57,8
Datenbankzugriffe	11	Geschwindigkeit	2,2	20	4	8,8	3	6,6	4	8,8
		verf. Treiber	2,2	20	5	11	5	11	4	8,8
		Implementierungsdauer	6,6	60	3	19,8	5	33	4	26,4
		Gesamt	11	100	12	39,6	13	50,6	12	44
Sicherheit	12	Bekannte Bugs	3	25	3	9	4	12	4	12
		Standardbibliotheken	5,4	45	4	21,6	3	16,2	3	16,2
		Gesamt	8,4	70	7	30,6	7	28,2	7	28,2
Komplexität	15	verf. Dokumentation	6	40	5	30	3	18	3	18
		Entwicklungsgeschwindigkeit	6	40	4	24	4	24	4	24
		Community-Support	3	20	5	15	4	12	2	6
		Gesamt	15	100	14	69	11	54	9	48
Skalierbarkeit	10	Modularität	4	40	4	16	5	20	3	12
		Design	2	20	5	10	3	6	4	8
		Plugins	4	40	3	12	4	16	3	12
		Gesamt	10	100	12	38	12	42	10	32
Aktualität	10	Regelmäßige Updates	4,5	45	4	18	2	9	3	13,5
		Marktanteil	3,5	35	2	7	4	14	3	10,5
		Anzahl Mitwirkende	2	20	3	6	3	6	4	8
		Gesamt	10	100	9	31	9	29	10	32
Erfahrung	15	Praktische Erfahrung	4,5	45	4	18	3	13,5	0	0
		Theoretisches Wissen	3,5	35	3	10,5	3	10,5	1	3,5
		Gesamt	8	80	7	28,5	6	24	1	3,5
Summe	100		81,4		75	301	71	287,15	68	275
Der Rang ist eine bewertung von 0 bis 5 wobei 5 100% und 0 0% entspricht				REIHUNG	1		2		3	

8.2.3 Frameworks

API-FRAMEWORK	Gewichtung	Teilkriterien	Gewichtung	Teil Gewichtung	Play		Spring Boot		Restlet		Light-rest-4j	
					Rang	G*R	Rang	G*R	Rang	G*R	Rang	G*R
Hilfestellung	21	Dokumentation/Beispiele	8,4	40	5	42	5	42	4	33,6	2	16,8
		Comunity	11,55	55	5	57,75	5	57,75	4	46,2	3	34,65
		Official Support	1,05	5	3	3,15	3	3,15	4	4,2	0	0
		Gesamt	21	100	13	102,9	13	102,9	12	84	5	51,45
Performance	2	CPU	0,4	20	2	0,8	4	1,6	3	1,2	5	2
		Memory	0,4	20	3	1,2	2	0,8	2	0,8	5	2
		Geschwindigkeit	1,2	60	4	4,8	2	2,4	4	4,8	5	6
		Gesamt	2	100	9	6,8	8	4,8	9	6,8	15	10
Aktualität	14	Contributors	2,1	15	4	8,4	5	10,5	3	6,3	4	8,4
		Letztes Update	7,42	53	2	14,84	5	37,1	1	7,42	3	22,26
		Alter/Commits	4,48	32	5	22,4	5	22,4	1	4,48	3	13,44
		Gesamt	14	100	11	45,64	15	70	5	18,2	10	44,1
Addins	10	Eingebaut	6,5	65	3	19,5	5	32,5	3	19,5	0	0
		Hinzufügbar	3,5	35	1	3,5	5	17,5	5	17,5	5	17,5
		Gesamt	10	100	4	23	10	50	8	37	5	17,5
Entwicklungsgeschwindigkeit	18	Lerngeschwindigkeit (Komp	3,6	20	5	18	5	18	5	18	3	10,8
		Menge des Codes	9	50	5	45	4	36	1	9	4	36
		Menge Autogenerierbar	5,4	30	2	10,8	4	21,6	5	27	3	16,2
		Gesamt	18	100	12	73,8	13	75,6	11	54	10	63
Wissen	16	Erfahrung	9,6	60	1	9,6	5	48	5	48	0	0
		Theoretisches Wissen	4	25	2	8	3	12	0	0	1	4
		Akzeptanz	2,4	15	4	9,6	5	12	3	7,2	2	4,8
		Gesamt	16	100	7	27,2	13	72	8	55,2	3	8,8
Features	19	Einfachkeit	8,55	45	5	42,75	4	34,2	4	34,2	4	34,2
		Menge	4,75	25	5	23,75	5	23,75	1	4,75	3	14,25
		Funktionalität	5,7	30	5	28,5	5	28,5	0	0	4	22,8
		Gesamt	19	100	15	95	14	86,45	5	38,95	11	71,25
Summe	100		100		71	374,34	86	461,75	58	294,15	59	266,1
Der Rang ist eine bewertung von 0 bis 5 wobei 5 100% und 0 0% entspricht				REIHUNG	2		1		3		4	

8.2.4 Dokumente

Dokumente	Gewichtung	Teilkriterien	Gewichtung	Teil Gewichtung	Word		Editor		Latex	
					Rang	G*R	Rang	G*R	Rang	G*R
Performance	**20**	Speicherplatz	6	30	4	24	5	30	2	12
		Geschwindigkeit	14	70	2	28	5	70	1	14
		Gesamt	20	100	6	52	10	100	3	26
Kompatibilität	**40**	Formatunterstützungen	28	70	4	112	1	28	3	84
		Standardisierung	12	30	2	24	1	12	4	48
		Gesamt	40	100	6	136	2	40	7	132
Benutzerfreundlichkeit	**15**	Direkte Arbeit	7,5	50	5	37,5	3	22,5	3	22,5
		Features	4,5	30	5	22,5	1	4,5	3	13,5
		Allg. Verständnis	1,8	12	5	9	5	9	1	1,8
		UI	1,2	8	5	6	1	1,2	2	2,4
		Gesamt	15	100	20	75	10	37,2	9	40,2
Erfahrung	**20**	Aktzeptanz	12	60	5	60	1	12	1	12
		Praktisches Wissen	6	30	4	24	4	24	0	0
		Theoretisches Wissen	2	10	3	6	5	10	1	2
		Gesamt	20	100	12	90	10	46	2	14
Aktualität	**5**	Letztes Update	3	60	5	15	3	9	4	12
		Marktanteil	2	40	4	8	1	2	2	4
		Gesamt	5	100	9	23	4	11	6	16
Summe	100		100		53	376	36	234,2	27	228,2
Der Rang ist eine bewertung von 0 bis 5 wobei 5 100% und 0 0% entspricht				REIHUNG	1		2		3	

8.2.5 Datenbankzugriff

Datenbankzugriffe	Gewichtung	Teilkriterien	Gewichtung	Teil Gewichtung	Jooq		JDBC		QueryDSL	
					Rang	G*R	Rang	G*R	Rang	G*R
Kompatibilität	**20**	Standardisierung	18	90	5	90	5	90	4	72
		Allg. Datenbanksystemen	2	10	5	10	5	10	4	8
		Gesamt	20	100	10	100	10	100	8	80
Benutzerfreundlichkeit	**40**	Koplexität	8	20	2	16	3	24	1	8
		UI	8	20	5	40	5	40	5	40
		Features	4	10	5	20	2	8	3	12
		Code Menge	20	50	5	100	4	80	4	80
		Gesamt	40	100	17	176	14	152	13	140
Erfahrung	**30**	Praktisches Wissen	18	60	4	72	2	36	2	36
		Theoretisches Wissen	9	30	3	27	1	9	3	27
		Aktzeptanz	3	10	5	15	2	6	5	15
		Gesamt	30	100	12	114	5	51	10	78
Aktualität	**10**	Neuartigkeit	3	30	3	9	1	3	5	15
		Letztes Update	7	70	5	35	2	14	5	35
		Gesamt	10	100	8	44	3	17	10	50
Summe	100		100		47	434	32	320	41	348
Der Rang ist eine bewertung von 0 bis 5 wobei 5 100% und 0 0% entspricht				REIHUNG	1		3		2	

8.2.6 Datenformate

Datenformate	Gewichtung	Teilkriterien	Gewichtung	Teil Gewichtung	Excel .xlsx		JSON .json		XML .xml		YAML .yml	
					Rang	G*R	Rang	G*R	Rang	G*R	Rang	G*R
Performance	20	Verwaltungsoverhead	10	50	2	20	5	50	3	30	2	20
		Speicherverbrauch	10	50	4	40	5	50	3	30	4	40
		Gesamt	20	100	6	60	10	100	6	60	6	60
Kompatibilität	20	Standardisierung	10	50	4	40	4	40	5	50	3	30
		Unabhängigkeit	10	50	3	30	5	50	5	50	5	50
		Gesamt	20	100	7	70	9	90	10	100	8	80
Softwareunterstützung	20	verfügbare Sprachunterstützung	9	45	3	27	5	45	5	45	4	36
		verfügbare Parser/Writer	9	45	3	27	5	45	5	45	3	27
		Zusatzbibliotheken	2	10	1	2	4	8	4	8	3	6
		Gesamt	20	100	7	56	14	98	14	98	10	69
Benutzerfreundlichkeit	14	Human-readable	2,1	15	0	0	3	6,3	4	8,4	4	8,4
		Popularität	3,5	25	4	14	4	14	4	14	2	7
		Features	8,4	60	5	42	3	25,2	5	42	3	25,2
		Gesamt	14	100	9	56	10	45,5	13	64,4	9	40,6
Erfahrung	16	Praktische Erfahrung	6,4	40	3	19,2	5	32	5	32	2	12,8
		Theoretisches Wissen	6,4	40	3	19,2	4	25,6	4	25,6	1	6,4
		Akzeptanz	3,2	20	4	12,8	5	16	5	16	4	12,8
		Gesamt	16	100	10	51,2	14	73,6	14	73,6	7	32
Aktualität	10	Letzte Standardaktualisierung	5	50	2	10	4	20	3	15	3	15
		Marktanteil	5	50	3	15	4	20	4	20	2	10
		Gesamt	10	100	5	25	8	40	7	35	5	25
Summe	100		100		44	318,2	65	447,1	64	431	45	306,6
Der Rang ist eine bewertung von 0 bis 5 wobei 5 100% und 0 0% entspricht				REIHUNG	3		1		2		4	

8.2.7 Datenbank

Datenbank	Gewichtung	Teilkriterien	Gewichtung	Teil Gewichtung	MySQL		SQL2O		PostgreSQL	
					Rang	G*R	Rang	G*R	Rang	G*R
Performance	**20**	Schnelligkeit	14	70	3	42	3	42	4	56
		CPU	2	10	2	4	3	6	3	6
		Speicherverbrauch	4	20	5	20	4	16	5	20
		Gesamt	20	100	10	66	10	64	12	82
Kompatibilität	**15**	Standardisierung	12	80	5	60	4	48	5	60
		Unabhängigkeit	3	20	4	12	4	12	3	9
		Gesamt	15	100	9	72	8	60	8	69
Softwarefeatures	**20**	Sicherheit	13,5	90	5	67,5	2	27	4	54
		Autogenerierung	1,5	10	4	6	3	4,5	3	4,5
		Gesamt	15	100	9	73,5	5	31,5	7	58,5
Softwareunterstützung	**5**	Sprachen unterstützung	5	100	55	275	5	25	4	20
		Gesamt	5	100	55	275	5	25	4	20
Benutzerfreundlichkeit	**10**	Komplexität	6	60	3	18	5	30	4	24
		Menge des Codes	3	30	4	12	1	3	3	9
		Lerngeschwindigkeit	1	10	2	2	1	1	3	3
		Gesamt	10	100	9	32	7	34	10	36
Erfahrung	**20**	Praktisches Wissen	8	40	4	32	0	0	0	0
		Theoretisches Wissen	8	40	4	32	1	8	1	8
		Aktzeptanz	4	20	5	20	2	8	3	12
		Gesamt	20	100	13	84	3	16	4	20
Aktualität	**10**	Neuartigkeit	4	40	2	8	5	20	4	16
		Update Frequenz	6	60	4	24	3	18	4	24
		Gesamt	10	100	6	32	8	38	8	40
Summe	100		80		102	561	41	237	46	267
Der Rang ist eine bewertung von 0 bis 5 wobei 5 100% und 0 0% entspricht				REIHUNG	1		3		2	

8.2.8 Core Testing

Core Testing	Gewichtung	Teilkriterien	Gewichtung	Teil Gewichtung	Junit		Selenium		Cucumber	
					Rang	G*R	Rang	G*R	Rang	G*R
Hilfestellung	21	Dokumentation/Beispiele	8,4	40	5	42	4	33,6	3	25,2
		Comunity	11,55	55	5	57,75	4	46,2	4	46,2
		Official Support	1,05	5	5	5,25	3	3,15	5	5,25
		Gesamt	21	100	15	105	11	82,95	12	76,65
Performance	4	CPU	0,8	20	4	3,2	4	3,2	4	3,2
		Memory	0,8	20	4	3,2	3	2,4	3	2,4
		Geschwindigkeit	2,4	60	4	9,6	3	7,2	4	9,6
		Gesamt	4	100	12	16	10	12,8	11	15,2
Aktualität	18	Contributors	2,7	15	5	13,5	4	10,8	3	8,1
		Letztes Update	9,54	53	4	38,16	4	38,16	4	38,16
		Alter/Commits	5,76	32	4	23,04	4	23,04	3	17,28
		Gesamt	18	100	13	74,7	12	72	10	63,54
Entwicklungsgeschwindigke	18	Lerngeschwindigkeit (Komp	3,6	20	5	18	5	18	5	18
		Menge des Codes	9	50	5	45	4	36	1	9
		Menge Autogenerierbar	5,4	30	3	16,2	4	21,6	5	27
		Gesamt	18	100	13	79,2	13	75,6	11	54
Wissen	20	Erfahrung	12	60	5	60	4	48	3	36
		Theoretisches Wissen	5	25	5	25	3	15	2	10
		Akzeptanz	3	15	5	15	4	12	4	12
		Gesamt	20	100	15	100	11	75	9	58
Features	19	Einfachkeit	8,55	45	5	42,75	4	34,2	4	34,2
		Menge	4,75	25	5	23,75	5	23,75	1	4,75
		Funktionalität	5,7	30	5	28,5	5	28,5	0	0
		Gesamt	19	100	15	95	14	86,45	5	38,95
Summe	100		100		83	469,9	71	404,8	58	306,34
Der Rang ist eine bewertung von 0 bis 5 wobei 5 100% und 0 0% entspricht				REIHUNG	1		2		3	

8.2.9 Authentifikations Model

Authentifications Modell	Gewichtung	Teilkriterien	Gewichtung	Teil Gewichtung	HTTP Basic Authentication		API-Key		HMAC		OAuth	
					Rang	G*R	Rang	G*R	Rang	G*R	Rang	G*R
Sicherheit	25	Keine Schwachstellen	8,25	33	2	16,5	4	33	3	24,75	4	33
		Sicherheit vor Dritten	8,25	33	2	16,5	3	24,75	4	33	4	33
		Validierung	8,5	34	2	17	3	25,5	4	34	4	34
		Gesamt	25	100	6	50	10	83,25	11	91,75	12	100
Performance	15	Geschwindigkeit	6,75	45	3	20,25	4	27	4	27	3	20,25
		Geringer Verwaltungsaufwand	6,75	45	5	33,75	3	20,25	3	20,25	2	13,5
		Ausfallsicherheit	1,5	10	4	6	4	6	4	6	3	4,5
		Gesamt	15	100	12	60	11	53,25	11	53,25	8	38,25
Erfahrung	15	Theoretisches Wissen	6	40	3	18	4	24	2	12	2	12
		Praktische Erfahrung	6	40	3	18	4	24	3	18	3	18
		Akzeptanz	3	20	0	0	5	15	4	12	2	6
		Gesamt	15	100	6	36	13	63	9	42	7	36
Aktualität	15	Erscheinungsjahr	1,5	10	3	4,5	4	6	2	3	5	7,5
		Entspricht modernen Anforderungen	10,5	70	0	0	3	31,5	4	42	5	52,5
		Marktanteil	3	20	0	0	4	12	3	9	3	9
		Gesamt	15	100	3	4,5	11	49,5	9	54	13	69
Komplexität	15	Technischer Aufbau	3	20	5	15	3	9	3	9	4	12
		Implementierung	6	40	5	30	4	24	4	24	4	24
		Auth-Prozess für Clients	6	40	5	30	5	30	5	30	3	18
		Gesamt	15	100	15	75	12	63	12	63	11	54
Entwicklung	15	Implementierungsdauer	4,95	33	5	24,75	4	19,8	3	14,85	2	9,9
		Verfügbare Softwarebibliotheken	4,95	33	0	0	4	19,8	4	19,8	4	19,8
		Sprach-Support	5,1	34	5	25,5	4	20,4	4	20,4	4	20,4
		Gesamt	15	100	10	50,25	12	60	11	55,05	10	50,1
Summe	100		100		52	275,75	69	372	63	359,05	61	347,35
Der Rang ist eine bewertung von 0 bis 5 wobei 5 100% und 0 0% entspricht				REIHUNG	4		1		2		3	

8.2.10 API Testing

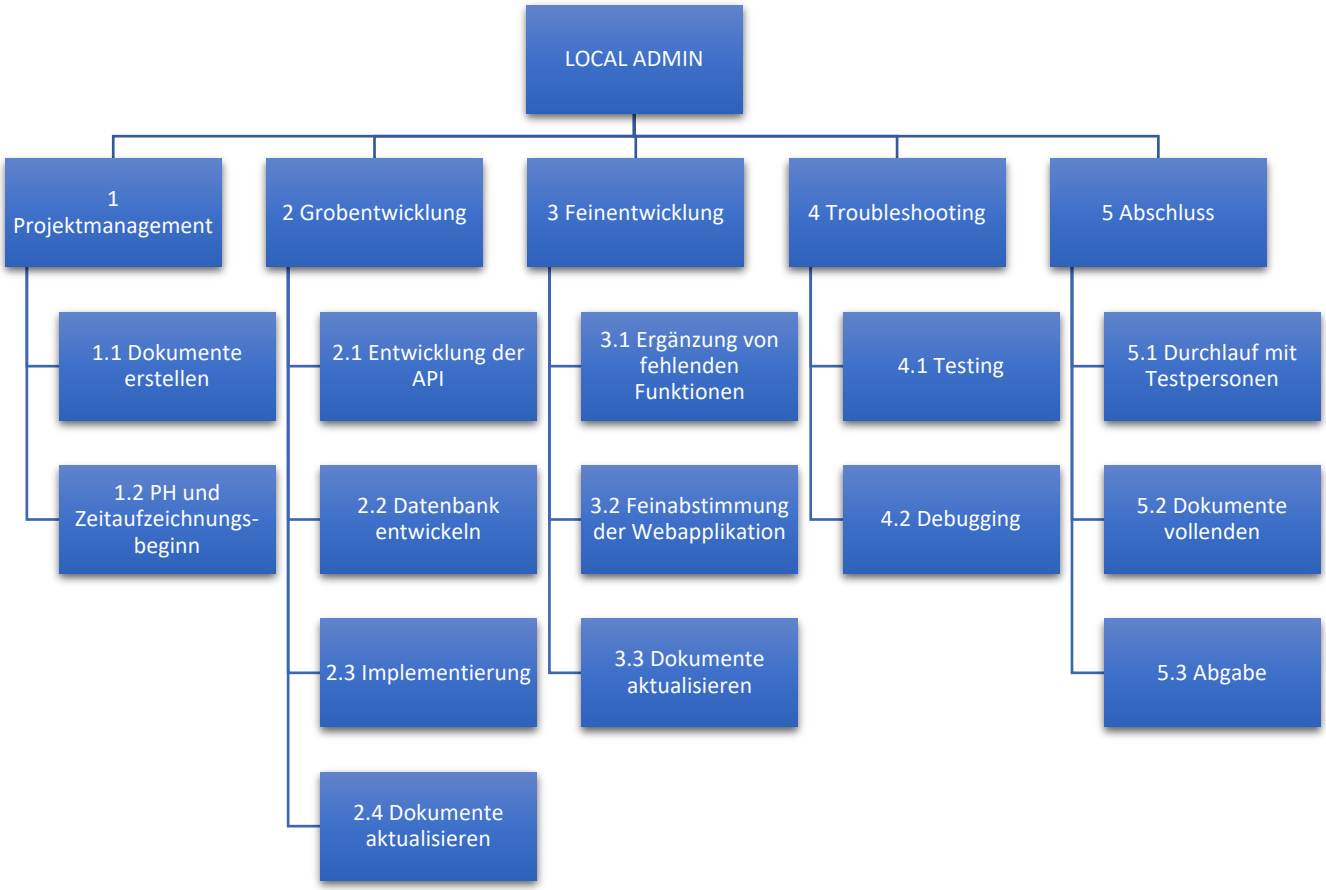
API Testing	Gewichtung		Gewichtung	Teil Gewichtung	REST-Assured		SoapUI		Stoplight	
					Rang	G*R	Rang	G*R	Rang	G*R
Hilfestellung	21	Dokumentation/Beispiele	8,4	40	4	33,6	3	25,2	5	42
		Comunity	11,55	55	3	34,65	2	23,1	3	34,65
		Official Support	1,05	5	4	4,2	3	3,15	4	4,2
		Gesamt	21	100	11	72,45	8	51,45	12	80,85
Kompatiblität	18	Java	9	50	4	36	3	27	4	36
		Andere Komponenten	9	50	2	18	2	18	5	45
		Gesamt	18	100	6	54	5	45	9	81
Aktualität	8	Contributors	1,2	15	4	4,8	2	2,4	4	4,8
		Letztes Update	4,24	53	2	8,48	3	12,72	4	16,96
		Alter/Commits	2,56	32	5	12,8	2	5,12	4	10,24
		Gesamt	8	100	11	26,08	7	20,24	12	32
Entwicklungsgeschwindigkeit	18	Lerngeschwindigkeit (Komp	3,6	20	5	18	5	18	5	18
		Menge des Codes	9	50	5	45	4	36	1	9
		Menge Autogenerierbar	5,4	30	4	21,6	4	21,6	5	27
		Gesamt	18	100	14	84,6	13	75,6	11	54
Wissen	16	Erfahrung	9,6	60	1	9,6	1	9,6	4	38,4
		Theoretisches Wissen	4	25	0	0	0	0	3	12
		Akzeptanz	2,4	15	4	9,6	3	7,2	5	12
		Gesamt	16	100	5	19,2	4	16,8	12	62,4
Features	19	Einfachkeit	8,55	45	5	42,75	2	17,1	4	34,2
		Menge	4,75	25	4	19	3	14,25	4	19
		Funktionalität	5,7	30	5	28,5	3	17,1	3	17,1
		Gesamt	19	100	14	90,25	8	48,45	11	70,3
Summe	100		100		61	346,58	45	257,54	67	380,55
Der Rang ist eine bewertung von 0 bis 5 wobei 5 100% und 0 0% entspricht				REIHUNG	2		3		1	

8.2.11 Design Testing

API Design	Gewichtung	Teilkriterien	Gewichtung	Teil Gewichtung	Visual-Paradigm		Stoplight Studio		Swagger Tools	
					Rang	G*R	Rang	G*R	Rang	G*R
Hilfestellung	23	Dokumentation/Beispiele	6,9	30	2	13,8	5	34,5	4	27,6
		Comunity	14,95	65	2	29,9	5	74,75	4	59,8
		Official Support	1,15	5	3	3,45	3	3,45	4	4,6
		Gesamt	23	100	7	47,15	13	112,7	12	92
Aktualität	18	Contributors	2,7	15	2	5,4	5	13,5	4	10,8
		Letztes Update	11,7	65	3	35,1	5	58,5	4	46,8
		Alter/Commits	3,6	20	2	7,2	5	18	4	14,4
		Gesamt	18	100	7	47,7	15	90	12	72
Entwicklungsgeschwindigkeit	18	Lerngeschwindigkeit (Komp	3,6	20	2	7,2	5	18	5	18
		Menge des Codes	9	50	2	18	4	36	4	36
		Menge Autogenerierbar	5,4	30	1	5,4	4	21,6	5	27
		Gesamt	18	100	5	30,6	13	75,6	14	81
Wissen	21	Erfahrung	14,7	70	0	0	5	73,5	2	29,4
		Theoretisches Wissen	3,15	15	0	0	4	12,6	1	3,15
		Akzeptanz	3,15	15	1	3,15	5	15,75	4	12,6
		Gesamt	21	100	1	3,15	14	101,85	7	45,15
Features	20	Einfachheit	9	45	2	18	4	36	4	36
		Menge	5	25	1	5	5	25	4	20
		Funktionalität	6	30	2	12	5	30	5	30
		Gesamt	20	100	5	35	14	91	13	86
Summe	100		100		25	163,6	69	471,15	58	376,15
Der Rang ist eine bewertung von 0 bis 5 wobei 5 100% und 0 0% entspricht				REIHUNG	3		1		2	

9 Projektplanung

9.1 Projektstrukturplan

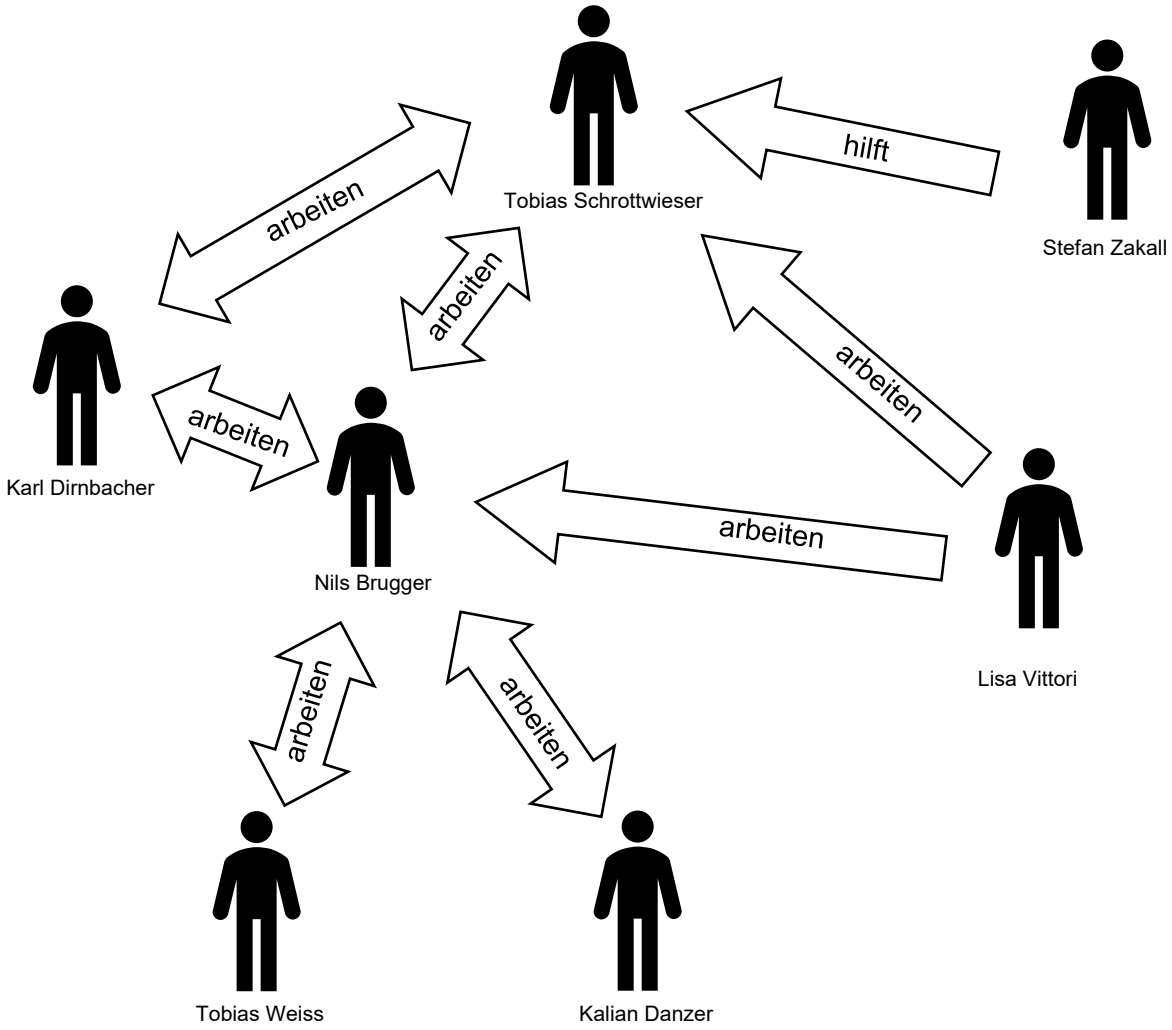


9.2 Meilensteinplanung

Meilenstein	Deliverable	Datum
Planungsabschluss	Alle Dokumente samt vorbereitetem Projektteam und Hardware	07.11.2019
Grobentwicklung abgeschlossen	80% der geforderten Funktionen	20.11.2019
Feinentwicklung abgeschlossen	100% der geforderten Funktionen samt Implementierung in das Webinterface und Erstellung der Datenbank	05.12.2019
Troubleshooting	Debugging der Website und Backend Funktion	19.12.2019
Projektabschluss	Vorführung des Produkts und Abnahme durch Projektauftraggeber	20.12.2019

10 Projektorganisation

Der Grund dafür, dass dieses Projekt ins Leben gerufen wurde, ist der Projektauftraggeber Karl Dirnbacher. Der Projektleiter ist Tobias Schrottwieser der Datenbankentwickler bestens mit der Struktur des Projektes vertraut ist und daher am besten als Projektmanager geeignet ist. Das Team besteht zusätzlich noch aus drei Backend-Entwickler. Der Lead-Entwickler heißt Nils Brugger uns ist bereits seit 5-6 Jahren mit der Programmierung von Java und diversen anderen Programmiersprachen und Scriptsprachen vertraut. Die beiden anderen Backend-Entwickler heißen Kalian Danzer und Tobias Weiss die ebenfalls schon 3-4 Jahre praktische Erfahrung mit diversen Programmiersprachen gemacht haben.



11 Management Summary

In vielen Situationen des Lebens möchte man einfach nur den Kopf in den Sand stecken, da man den Überblick über seine Zeitplanung und sein Management verliert. Man hat Stress durch Unübersichtlichkeit und zeitaufwändige Suche in den Stapeln aus Papier und Ordnern.

Es wird nun allgemein eine Lösung gesucht, die diese Probleme beheben soll.
Der Lösungsansatz wurde in der Softwarebranche gesucht, da keines der bestehenden Programme den Kriterien entsprach, soll eine neue Software entwickelt/erweitert werden.
Die Umsetzung für die Filialen ist sowohl technisch als auch wirtschaftlich sowie persönlich machbar.

Insgesamt soll das Projekt Gesamtkosten von 9.800€

Das gesamte Team und die Arbeitsmittel sind ausreichend, für die bevorstehende Aufgabe gerüstet und weitreichend einsatzbereit. Das Knowhow im Feld von Java ist vorhanden und wird für das Projekt mehr als ausreichend sein, um all die festgelegten Ziele zu erreichen. Was den Aufwand, der möglicherweise auftritt, angeht ist dieser nicht existent da in unser Teamplanung jeglicher extra Aufwand schon eigeplant wurde. Mögliche Kosten könnten anfallen im Bereich der Hardware, da Möglicherweise etwas ersetzt werden muss. Das komplette Projekt wird ca. 4 Monate benötigen. Die genauen Termine sind in der Meilensteinplanung ersichtlich.

Wenn man alle Aspekte in Betracht zieht ist dieses Projekt in diesem Umfeld und den derzeitigen wissen der Lage machbar.

12 Glossar

Begriff	Begriffsklärung
Sysadmin	Der „Sysadmin“ oder auch Systemadministrator ist eine Instanz des Systems welche alle Rechte hat. Jedoch hat diese im laufenden System keine Aufgabe da der Systemadministrator lediglich aus Wartungszwecken existiert.
Performanceproblemen	Ineffiziente Datenverwaltung bzw. interne Abläufe welche nicht das maximum der möglichen Leistung ausnutzen.
Daten	Unter Daten werden alle gespeicherten Daten der Nutzer bezeichnet. Da unser System relativ dynamisch ist kann der Benutzer seine einzugebenden Daten selbst bestimmen. Zum Beispiel Dienstzeiten, Ausgaben oder Einnahmen.
Lead-Entwickler	Der erfahrenste Programmierer, welche die Aufgaben verteilt und bei Fragen den anderen backend Entwicklern zur Seite steht.