Meilenstein 2 – Entwurf

Gruppe: Archiv mit FHIR-Schnittstelle

Gruppenmitglieder: Chizie Leitricia Moguem Foka

Dolvie Emma Kouekam

Gabriel Nathanael da Gomez

Nasreddin Abdalla A. El Awani

Said Franck Dam Noujepme Njimoluh

Torben Heller

# Inhalt

[1 UML-Anwendungsfalldiagramme 3](#_Toc77283314)

[2 UML-Aktivitätsdiagramm 5](#_Toc77283315)

[3 Datenbankkonzept 8](#_Toc77283316)

[3.1 Stärken 9](#_Toc77283317)

[3.2 Schwächen 9](#_Toc77283318)

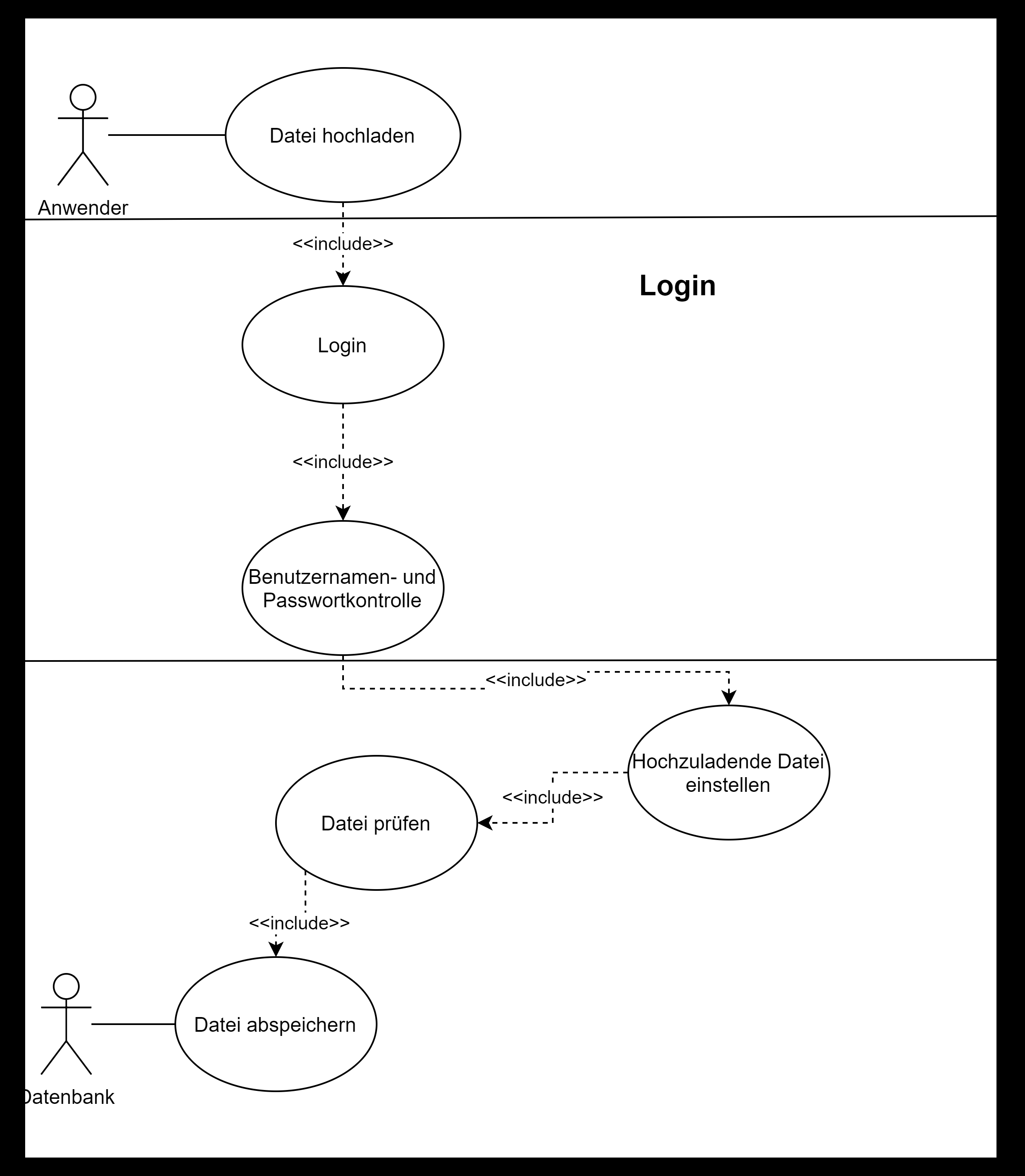
[3.3 Analyse 9](#_Toc77283319)

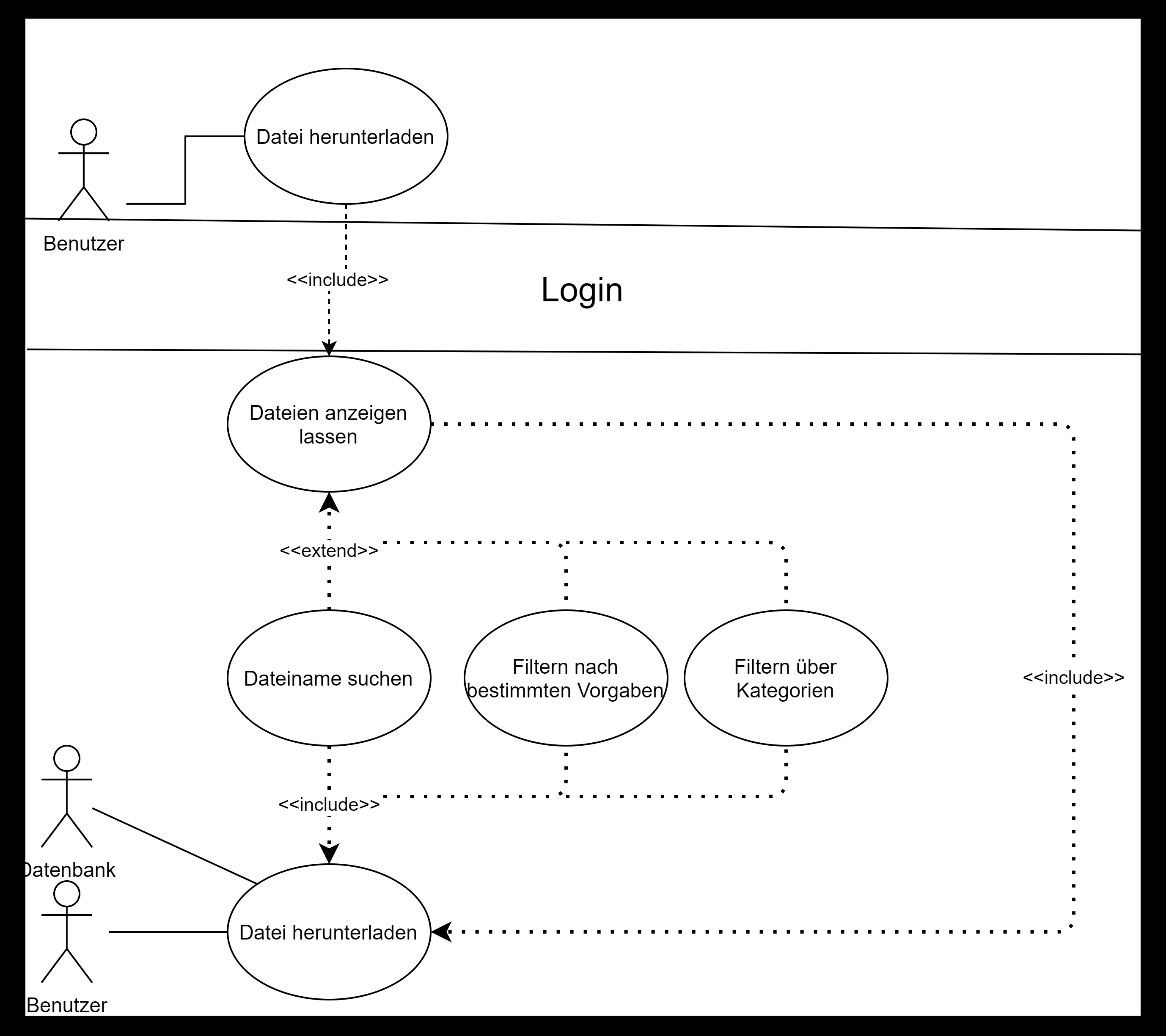
[4 ERD 10](#_Toc77283320)

[5 Mockups 11](#_Toc77283321)

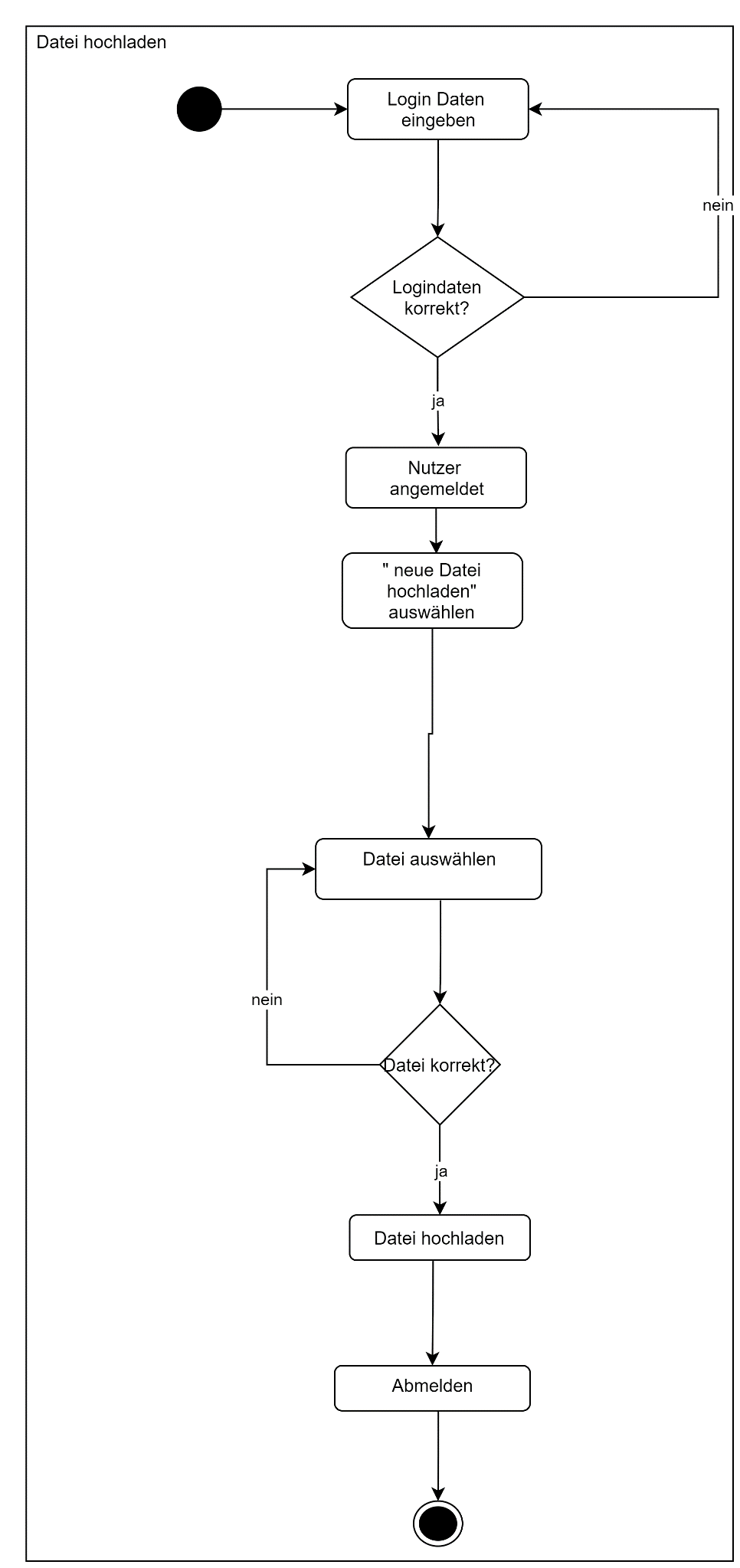
[6 Konzept – Erweiterbarkeit und Konfigurierbarkeit 13](#_Toc77283322)

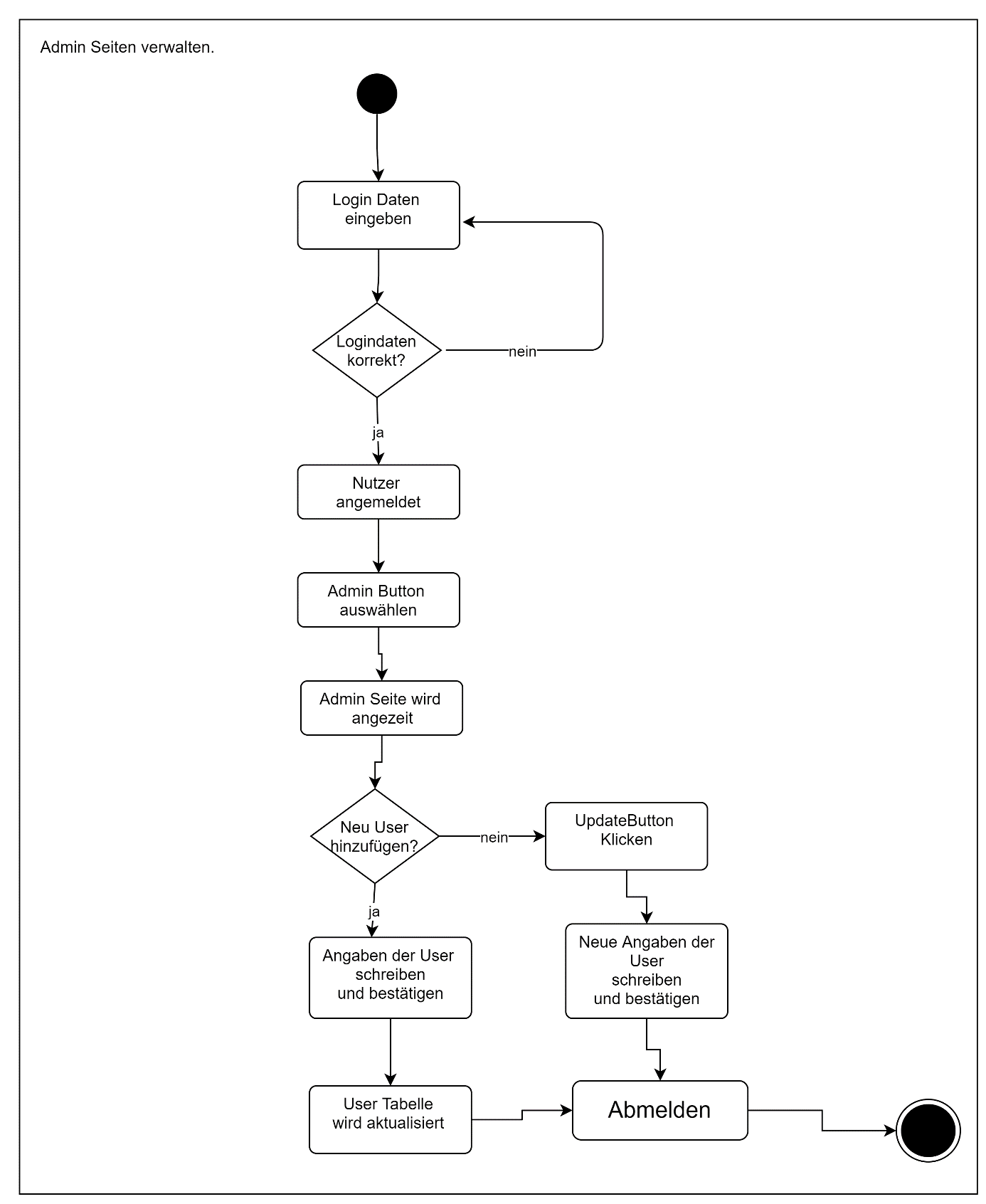
# UML-Anwendungsfalldiagramme

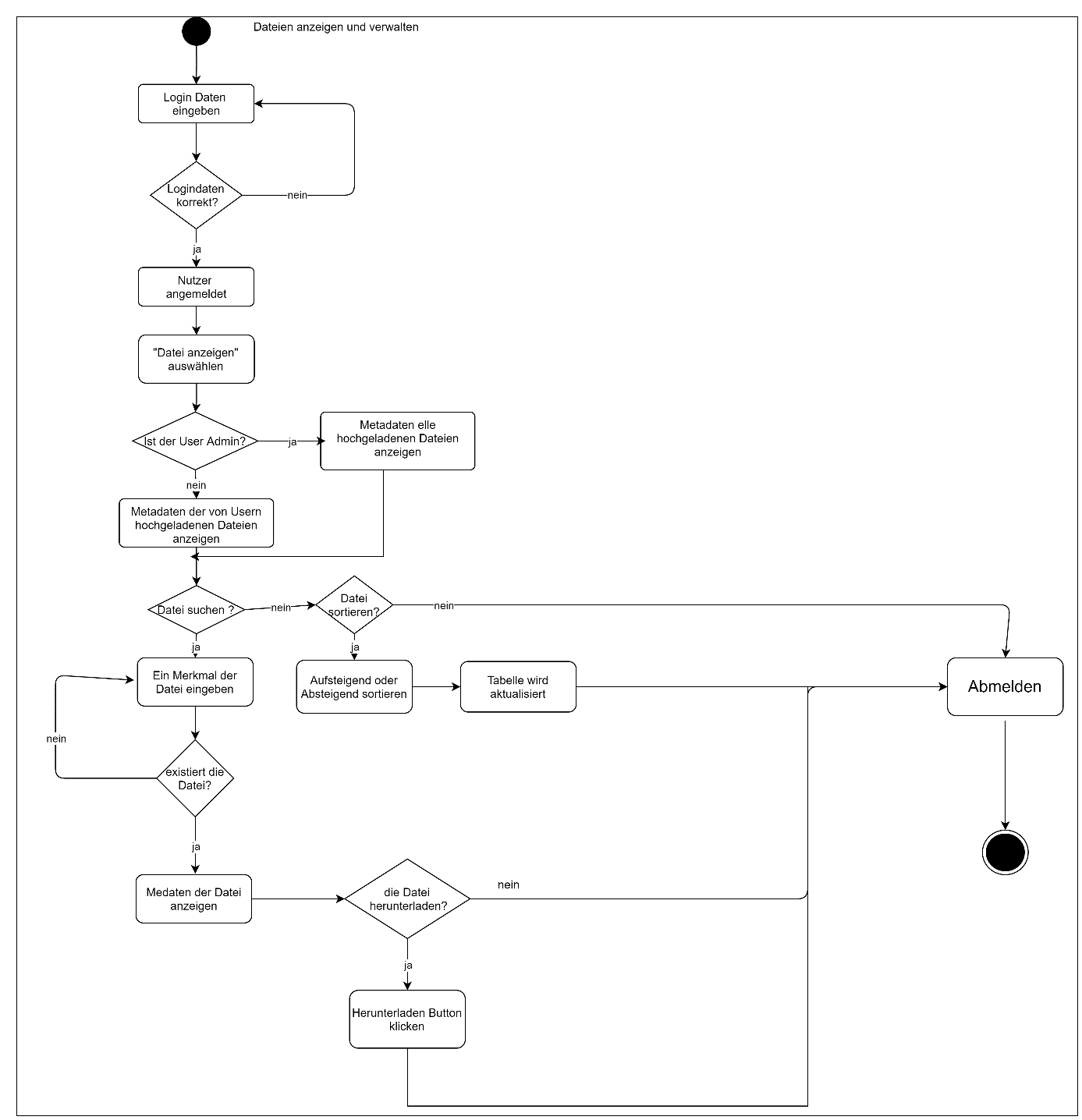




# UML-Aktivitätsdiagramm







# Datenbankkonzept

In dieser Datenbank wurde 3 Tabellen angelegt; resources, metadaten und user. In der Tabelle resources werden die Dateien mithilfe der Stored Procedure „insertionnew“ hinzugefügt.

Die Stored Procedure „insertionnew“ nimmt 4 Input Parameter und 1 Input/Output Parameter übergeben;

1. jsonTxt (INPUT) = Resourceninhalt als json(text).
2. datei\_n (INPUT) = Dateiname der zu speichernde Resource.
3. usern (INPUT) = Der Name des Benutzers, der die Resource abgegeben (hochgeladen) hat.
4. Datei\_byte (INPUT) = Die Resourcedatei in der Byte-Format.
5. resul (INPUT/OUTPUT) = Das ergebnis als Zahl von 0 bis 3, jenach was in der Stored Procedure passiert ist:

0: Wenn die Datei nicht aufgenommen wurde (Grund: Schon vorhanden),

1: Wenn die Datei mit einer Patientreferenz hinzugefügt wurde,

2: Wenn die Datei ohne eine Patientreferenz hinzugefügt wurde und

3: Wenn eine neue Version einer Datei hinzugefügt wurde.

* Die Stored Procedure sucht in der JSON-Datei nach Daten, um die verschiedenen Spalte der Tabelle Resources zu füllen gegebenfalls auch Daten für die Metadaten Tabelle.
* Die Patientreferenz wird durch den Vergleich der **Resourcen\_id mit der Stelle „Subject“ (**wie bei vielen Resourcen), oder **„Patient“** (zum Beispiel bei der Resource AllergieIntollerenz) **von der Json Datei** und es wird immer die neuste Version der Patient genommen. Damit eine Referenzierung stattfinden kann, muss der zu referenzierende Patient erst in der Datenbank vorhanden sein. Anders gesagt müssen alle Patienten erstmal hochgeladen werden, bevor die zu Ihnen passenden Dateien.
* Versionierung Erfolg durch den Vergleich der Resourcen\_id und den resourceninhalt die neue Datei, mit alle in der Datenbank gespeicherte Dateien.
* Die Metadaten werden erst hinzugefügt, nachdem eine Datei hinzugefügt wurde. Die Metadaten Tabelle beinhaltet;

metadaten\_id = Durch den Stored Procedure automatische hochgezählte Nummer,

datei\_resource = Fremdschlüssel der dazugehörigen Datei von der Resourcen Tabelle,

groesse = Die Dateigröße,

aenderungsDatum,

herkunft = Der Benutzer, der die Datei hochgeladen hat als Fremdschlüssel,

datenTyp = Default JSON,

version = Die Version der Datei. Anfangswert=1.0.

* Bei der Users Tabelle werden

user\_id (automatisch hochgezählte Nummer), username, password, active (boolean) und roles (Admin oder User).

## Stärken

* Die Primary Keys haben den Datentyp „BIGINT“, um genau 9223372036854775807 Dateien speichern zu können.
* Dateien, die mehr als 20Mb groß sind werden von der Datenbank nicht gespeichert, um Effizienz zu erhöhen.
* Die Datenbank wurde auf dem 3.Normalform normalisiert, damit die Datenbank übersichtlich bleibt

## Schwächen

* Damit eine Referenzierung stattfinden kann, muss der zu referenzierende Patient erst in der Datenbank vorhanden sein. Das heißt, bei gleichzeitige Upload der Patient und die ihn zugehörige Dokumente wird eine richtige Referenzierung auf manche Dokumenten nicht stattfinden.

## Analyse

Es sollte eine Datenbank implementiert werden, die FHIR Resourcen annehmen kann und die Metadaten speichern. Das heißt zum Beispiel Dateigröße, Version, Änderungsdatum, wer das hochgeladen hat etc. Dazu soll die Datenbank auch Benutzerdaten speichern.

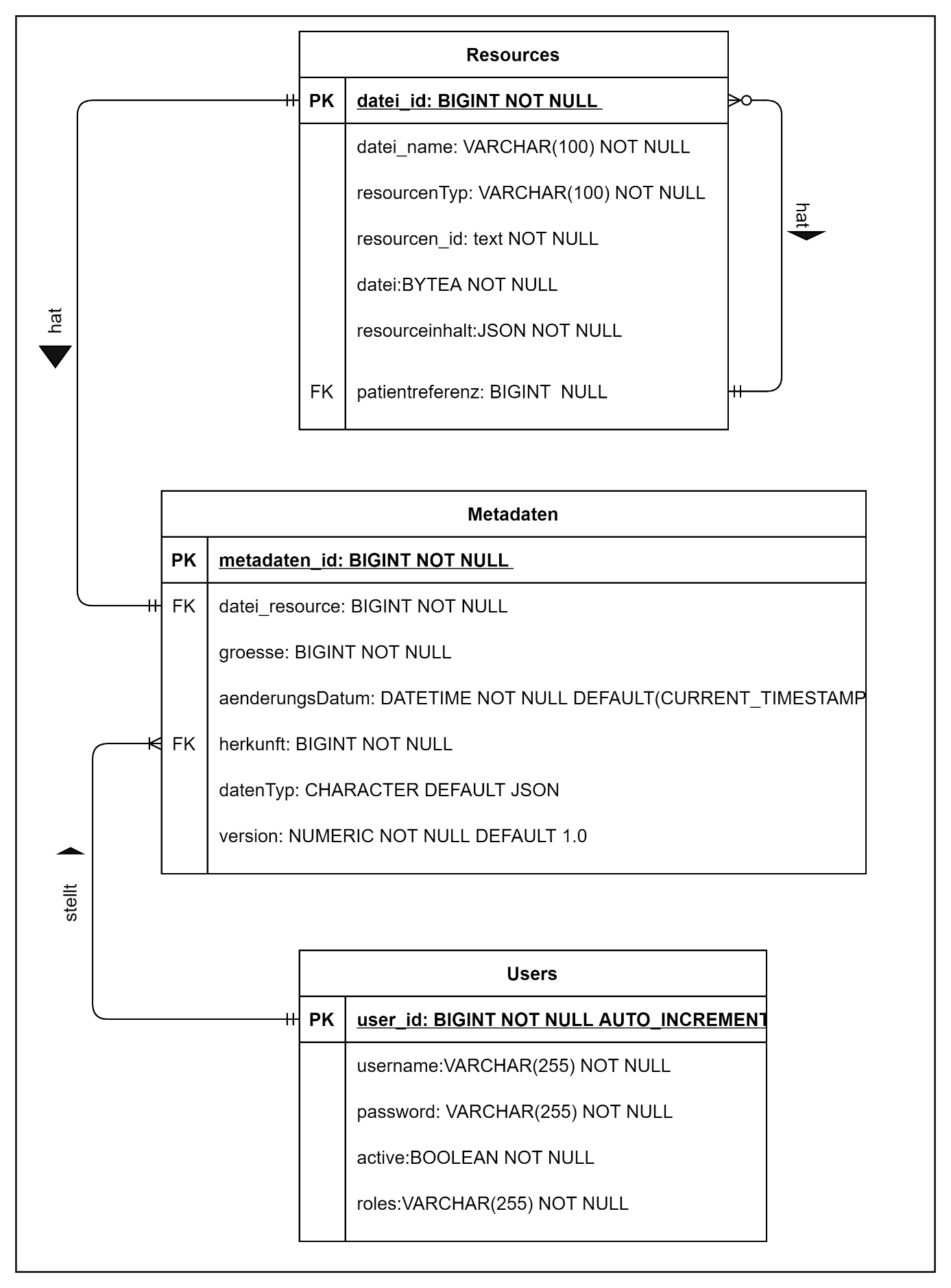
Entities mit Attributen (#=Primary Key, \*=Foreign Key)

Resources (#datei\_id, datei\_name, resourcenTyp, resourcen\_id, datei, resourceinhalt, \*patientreferenz)

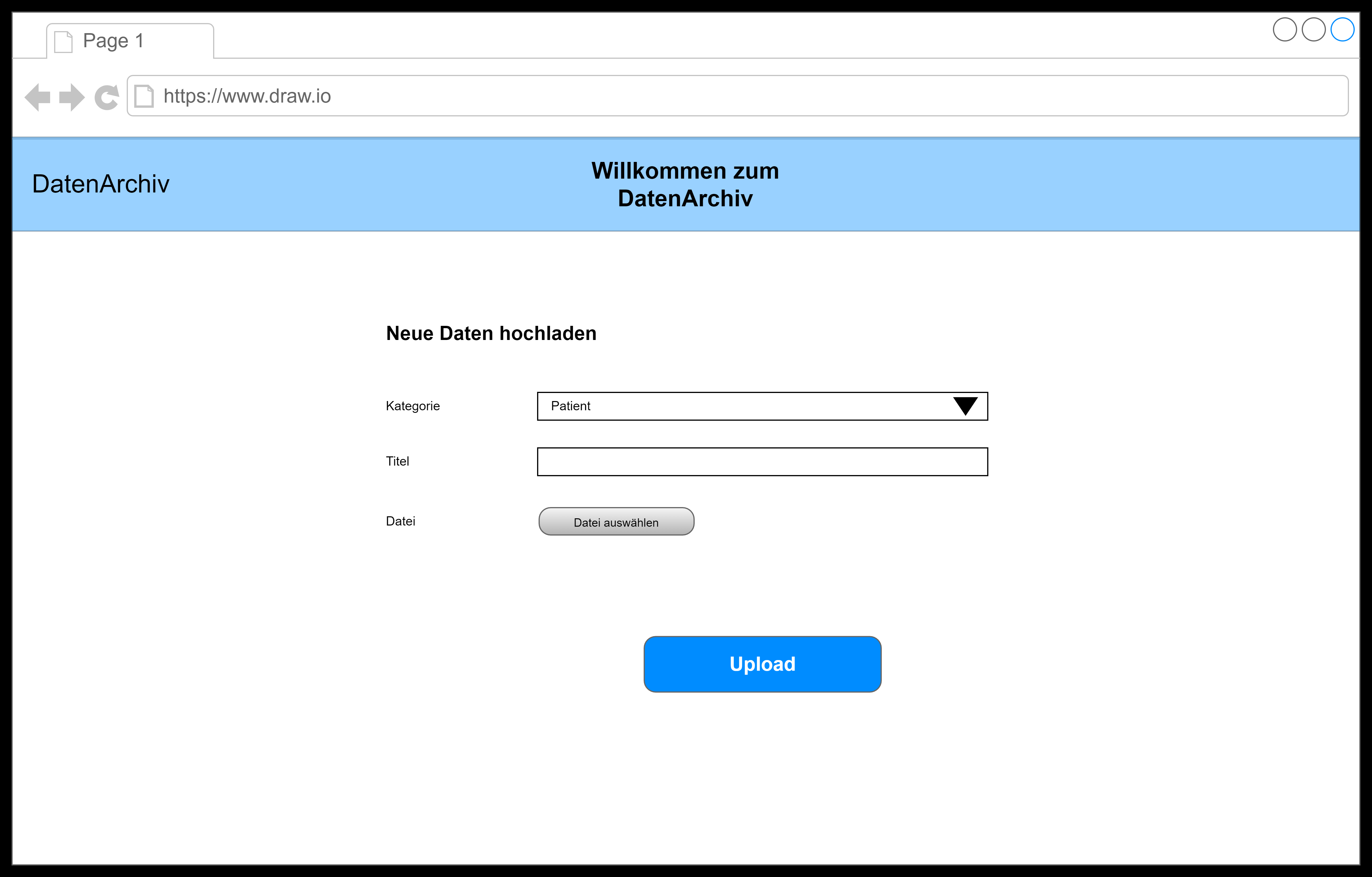
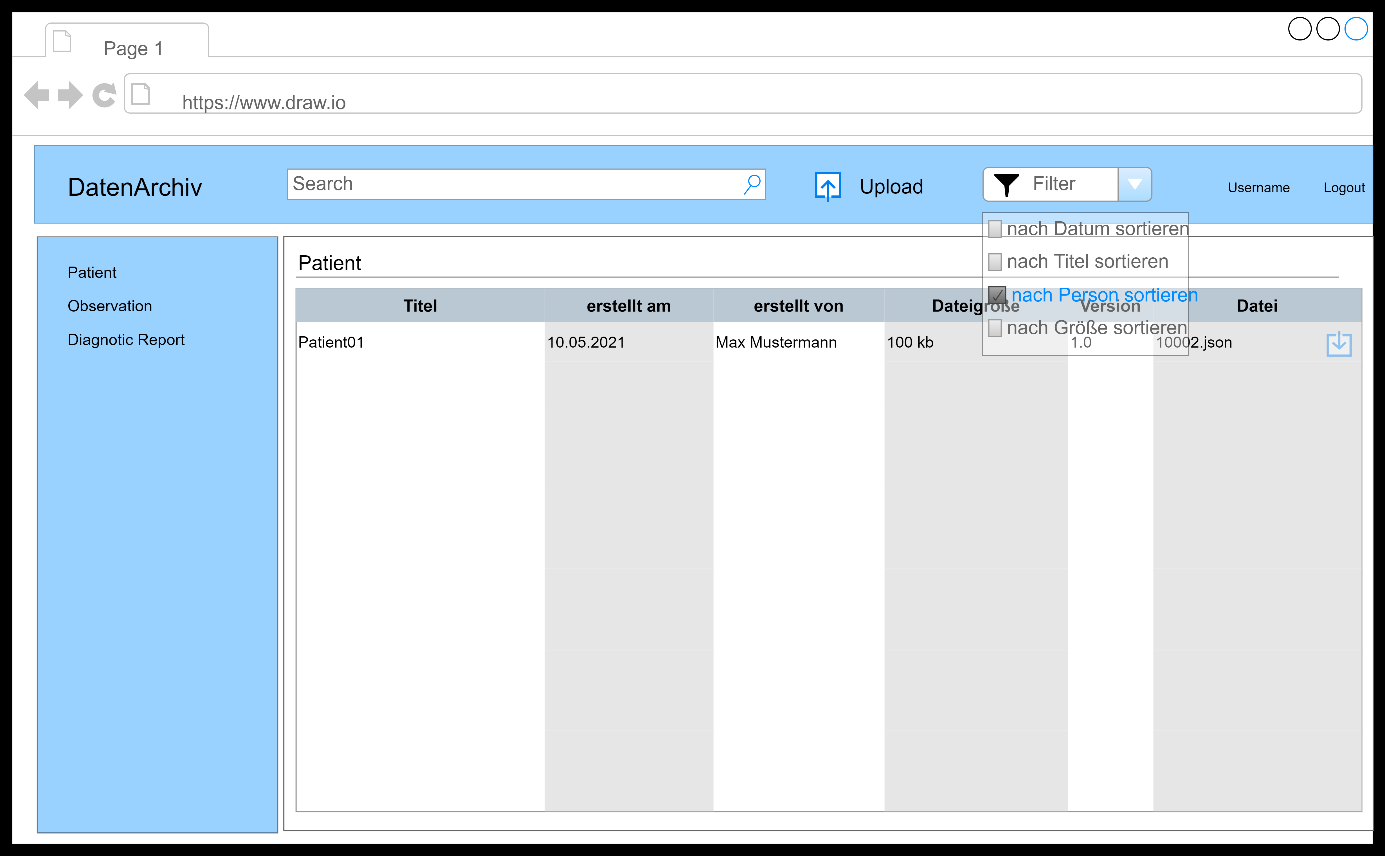
Metadaten (#metadaten\_id, \*datei\_resource, groesse, aenderungsDatum, \*herkunft, datenTyp, version)

User (#user\_id, username, password, active, roles)

# ERD



# Mockups



# Konzept – Erweiterbarkeit und Konfigurierbarkeit

Unsere Datenbank ist nicht auf die von uns gewählten Ressourcentypen begrenzt und kann theoretisch jeden Ressourcentyp aufnehmen. Sollte zu einem späteren Zeitpunkt mehr Daten aus den JSON-Files in die Metadatentabelle gespeichert werden und weitere Spalten entstehen, müssten wir nur die dafür zuständige Stored Procedure anpassen, und nicht jeden Datensatz händisch aktualisieren.

Dadurch, dass wir eine eigene Datenbank für die FHIR-Ressourcen entwickelt und keine vordefinierten Tabellen implementiert haben konnten wir sie auch genau auf unsere Belange zuschneiden und konfigurieren. Zudem lässt sich die Datenbank auch in der Zukunft bei Veränderung der benötigten Daten schnell anpassen, ohne, dass wir die Datenbank komplett neu erstellen müssen oder die enthaltenen Datensätze neu geladen werden müssen. So können wir die für uns relevanten Daten aus den Ressourcenfiles extrahieren und unwichtige Daten zurücklassen.