3D Library

## Inhaltsverzeichnis

[Inhaltsverzeichnis 1](#_Toc34070723)

[Einführung 2](#_Toc34070724)

[Overview 2](#_Toc34070725)

[Detailview 3](#_Toc34070726)

[Login 3](#_Toc34070727)

[Upload 4](#_Toc34070728)

[Technische Dokumentation 5](#_Toc34070729)

[Overview 5](#_Toc34070730)

[Detailview (3D Viewer) 5](#_Toc34070731)

[Login 5](#_Toc34070732)

[Upload 5](#_Toc34070733)

[API Backend 6](#_Toc34070734)

[Reflexion 6](#_Toc34070735)

[Fazit 6](#_Toc34070736)

## Einführung

Wie der Name schon andeutet, handelt es sich bei diesem Projekt um eine Bibliothek von 3D Modellen. Hier lassen sich 3D Modelle in unterschiedlichen Formaten hochladen, betrachten, und herunterladen. Die Bibliothek besteht aus folgenden Bereichen:

### Overview

Repräsentiert den Einstieg in die 3D Library und bietet eine Übersicht über alle verfügbaren Modelle (siehe Abbildung 1). Hier wird für jedes vorhandene Modell eine Kachel mit Namen und Vorschaubild angezeigt. Die Kacheln lassen sich anklicken, und leiten auf die Detailview weiter.

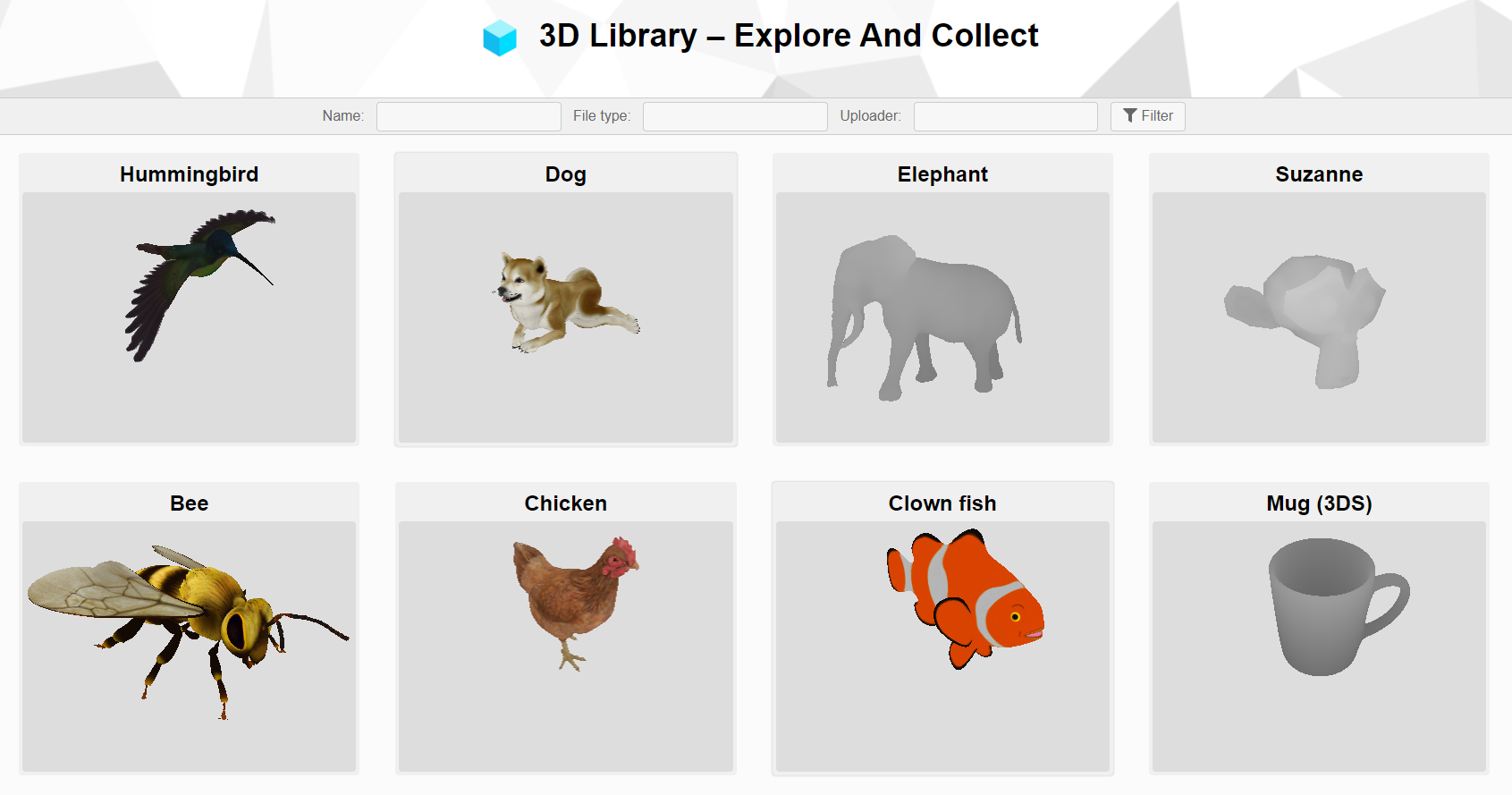


Abbildung 1: Overview

Die Overview bietet ausserdem eine Suchfunktion mit einem Filter (siehe Abbildung 2). Dieser kann genutzt werden, um die angezeigten Modelle einzuschränken (siehe Abbildung 3).

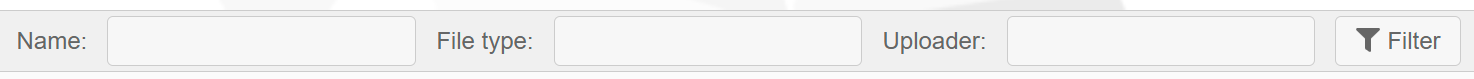


Abbildung : Filter

Die Filterung wird angewendet, sobald der entsprechende Knopf gedrückt wird.

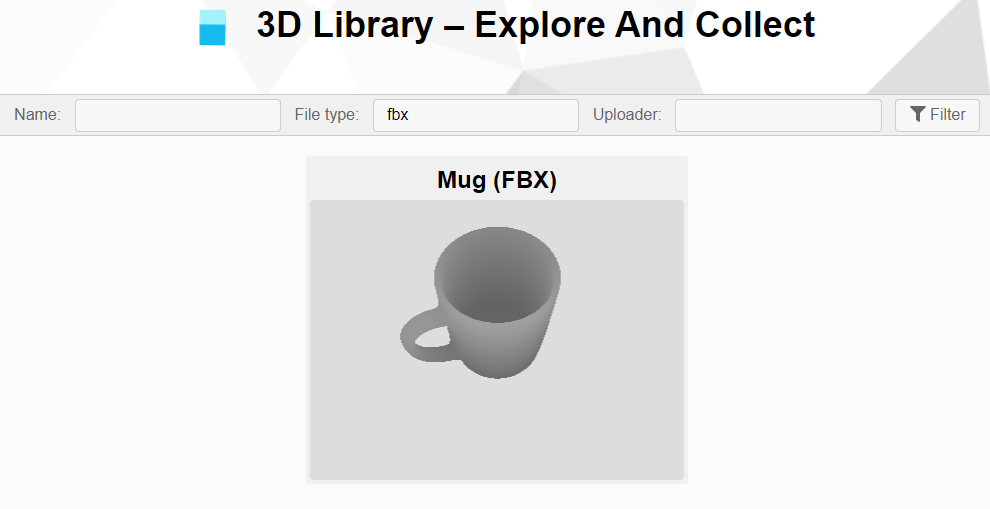


Abbildung 3: Anzeige mit Filter

### Detailview

Ermöglicht die dreidimensionale Betrachtung der Modelle (siehe Abbildung 4). Sie besteht aus einer grossen Kachel, welche den 3D-Viewer, Informationen zum Modell, sowie einen Knopf für den Download beinhaltet. Ist der aktuelle Benutzer eingeloggt, wird ein weiterer Knopf bereitgestellt, um das Modell zu löschen.

Das Modell kann im Viewer durch Halten der Maustaste gedreht werden. Zoomen geschieht durch Scrollen per Mausrad, oder einer äquivalenten Geste auf Notebooks. Falls das Modell animiert ist, kann eine Animation im Dropdown ausgewählt werden, die dann sofort abgespielt wird.

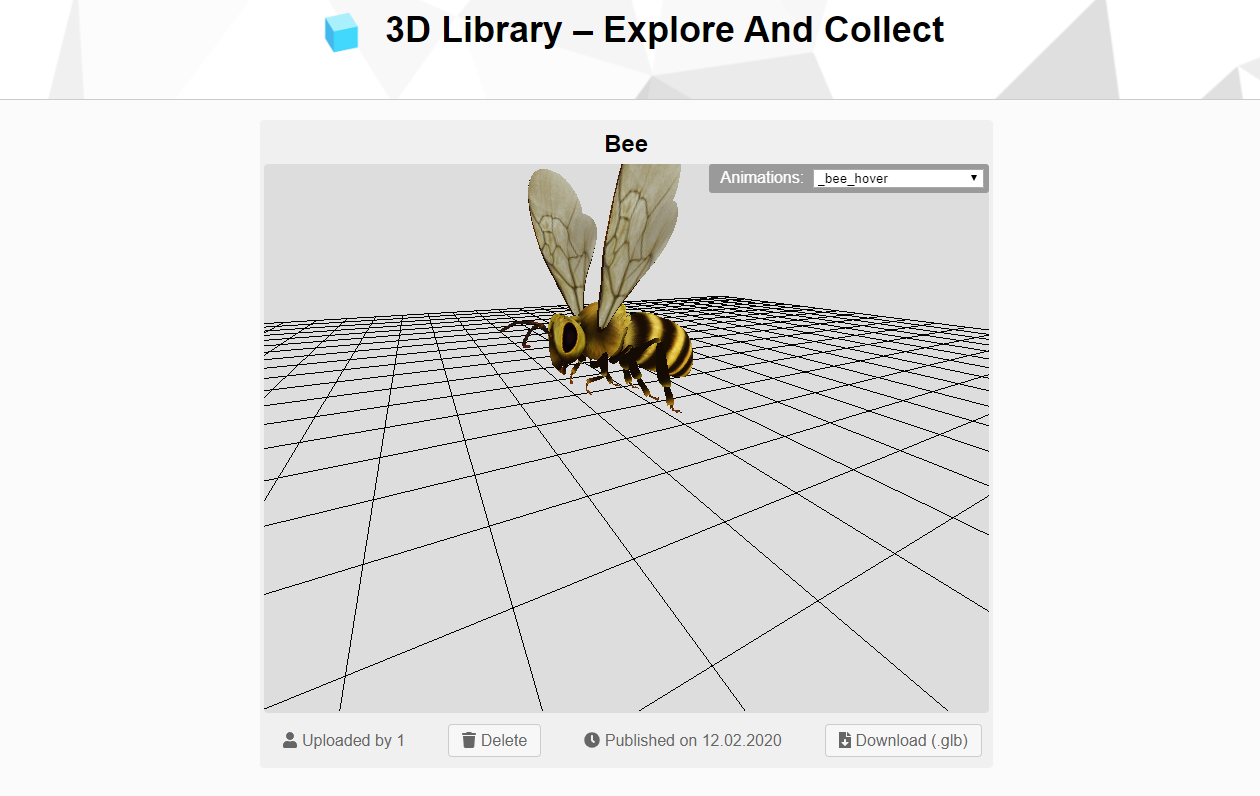


Abbildung 4: Detailview

### Login

Der gleichnamige Knopf für das Login befindet sich im Banner und führt zur Loginmaske (siehe Abbildung 5). Das Login erlaubt das Hochladen und Löschen von Modellen.

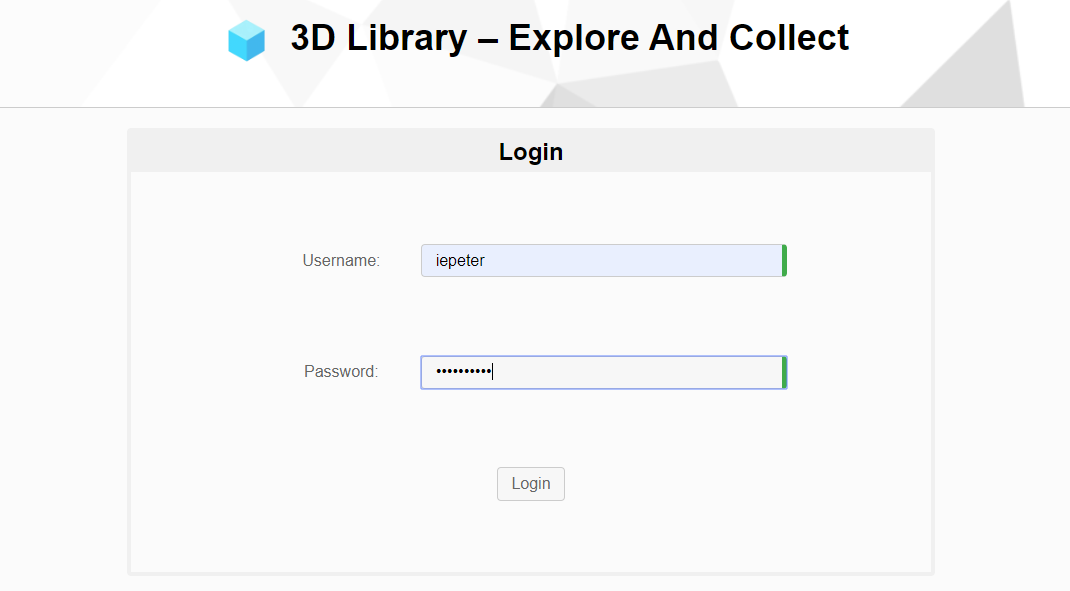


Abbildung 5: Loginmaske

Ist der Benutzer eingeloggt, befindet sich anstelle des Knopfes für das Login einen, um neue Modelle hochzuladen und einen, um sich wieder auszuloggen.

### Upload

Beim Hochladen werden nur unterstützte Datentypen erlaubt. Wird eine Datei ausgewählt, erscheint der Viewer, mit dem eine Vorschau generiert wird (siehe Abbildung 6). Es handelt sich hierbei um denselben Viewer, wie in der Detailview. Der Benutzer kann für das perfekte Vorschaubild im Viewer frei rotieren und zoomen, sowie die gewünschte Animation (falls vorhanden) auswählen.

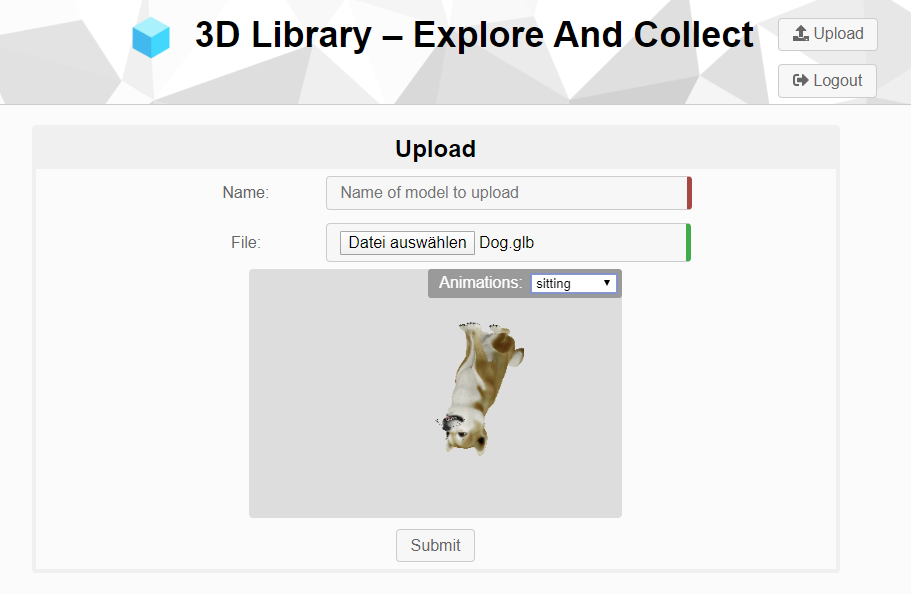


Abbildung 6: Uploadmaske

## Technische Dokumentation

Die 3D Library besteht aus einem Angular Webserver, einem Express Datenbankserver, und einer MongoDB Datenbank (siehe Abbildung 7**Error! Reference source not found.**). Web Server und DB Server laufen auf der gleichen Maschine, während die Datenbank in der Cloud liegt.

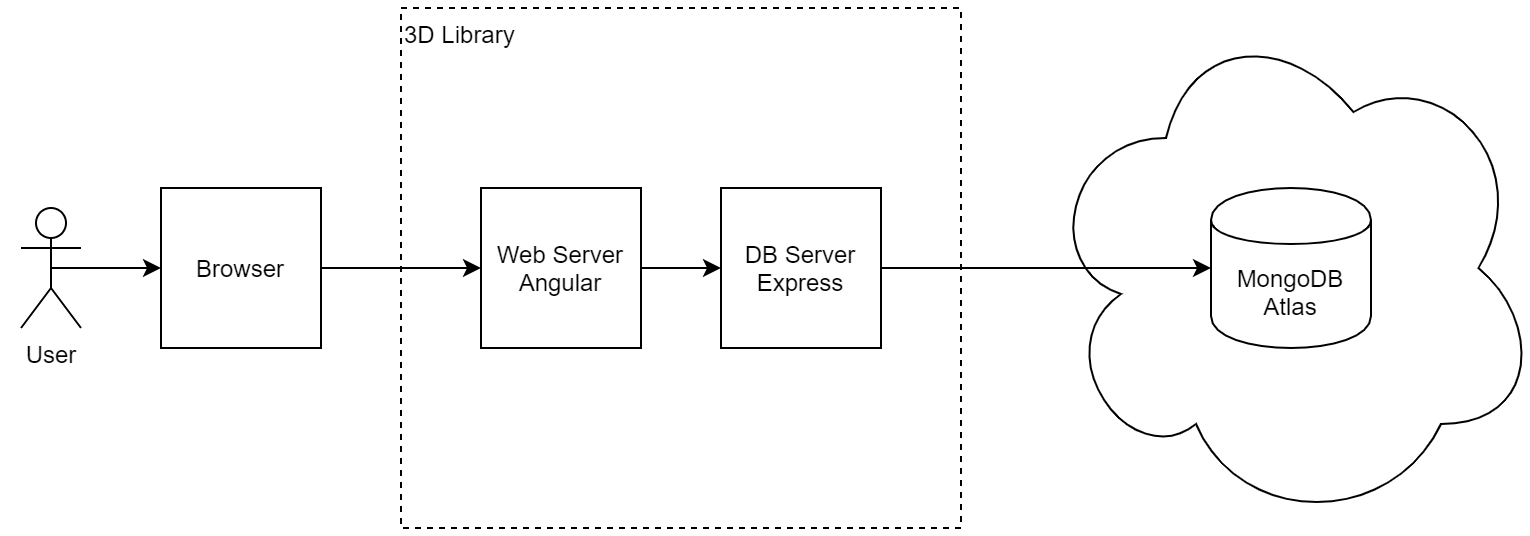


Abbildung : Architekturdiagramm

### Overview

Beim Aufruf der Overview holt der Datenbankserver alle Einträge aus der Datenbank, sofern keine Filter definiert wurden. Für jeden Eintrag erstellt der Webserver eine Kachel mit Vorschaubild.

Beim Suchfilter können mehrere Filter gleichzeitig angewendet werden, jedoch nicht mehrere Werte pro Filter. Die Werte werden beim Aufruf des Datenbankservers in der URL mitgeschickt und vom Datenbankserver für die Datenbankabfrage übersetzt und angewendet.

### Detailview (3D Viewer)

Der Viewer nutzt WebGL über das three.js Framework, um 3D Modelle direkt im Browser darzustellen. Unterstützt werden alle Browser, die WebGL unterstützen, inklusive IE 11 und Safari. Dank des selbst entwickelten «LoaderManager» können insgesamt 6 unterschiedliche Formate geladen und dargestellt werden: GLB, OBJ, STL, DAE, FBX und 3DS. Der LoaderManager ist zudem für weitere Formate erweiterbar.

### Login

Der Datenbankserver erhält die Logindaten und überprüft dessen Richtigkeit. Aus Sicherheitsgründen werden neben dem Benutzernamen nur der Hashwert ihrer Passwörter auf der Datenbank gespeichert. Bei erfolgreichem Login stellt der Datenbankserver ein JSON Web Token aus, welches im lokalen Speicher des Benutzers abgelegt wird. Bei allen weiteren Aufrufen der Seite wird das Token im Header mitgeschickt. Versucht der Benutzer ein Modell hochzuladen oder zu löschen, so wird das Token wiederum vom Datenbankserver validiert.

### Upload

Der Upload erfolgt über ein HTML-Form. Der Datenbankserver erhält dabei einen Namen, eine Datei mit dem 3D Modell, und ein vom Viewer erstelltes Vorschaubild als Data URL. 3D Modell und Vorschaubild bleiben dabei auf dem Filesystem, und werden nicht an die Datenbank übertragen.

### API Backend

|  |  |  |
| --- | --- | --- |
| Methode | Route | Rückgabewert |
| GET | /models | DatabaseModel[] |
| GET | /model/:modelId | DatabaseModel |
| DELETE | /model/:modelId | { success: boolean, errors: string[] } |
| POST | /upload | DatabaseModel |
| POST | /login | {  idToken: string,  expiresIn: number,  username: string  }  oder  { message: string } |

## Reflexion

Die Reihenfolge und Zeitpunkt der einzelnen Arbeitspakete, als auch die Arbeitsteilung haben wir im Trello-Board teilweise nur lose festgelegt. Dadurch wurden gewisse Arbeiten in einer fragwürdigen Reihenfolge abgearbeitet. Es gab zum Beispiel noch keine Funktion, um Objekte hochzuladen, bevor die Overview und die Detailview implementiert wurden. Dadurch mussten Beispieldaten eigenhändig in die Datenbank eingespeist werden.

Das Datenbankschema inklusive alle Beispieldaten musste während dem Projekt mehrmals angepasst werden, da es nicht für die neuen Anforderungen geeignet war. Beim nächsten Mal würden wir das Schema am Anfang so definieren, dass es auch für die späteren Anforderungen geeignet ist.

Wir haben uns bewusst auf die wichtigsten Features konzentriert und diese auch erfolgreich implementiert. Von den optionalen Anforderungen konnten wir ebenfalls einige umsetzen. Es gibt jedoch noch einige Dinge, welche noch erweitert werden könnten. Deshalb haben wir die Webseite so aufgebaut, dass neue Anforderungen problemlos integriert werden können.

## Fazit

Wir haben viel neues gelernt, besonders im Bereich der Authentifikation und Sicherheit von Webseiten. Dabei war es uns ein Bedürfnis, die Nutzernamen, Passwörter und Schlüssel nicht einfach in den Code zu schreiben oder in das Repository zu pushen, sondern wie in produktiven Projekten zu schützen.

Sehr interessant war auch das Arbeiten mit der automatischen Pipeline von Gitlab. Wir hatten zuvor noch keine Erfahrungen damit gemacht. Nachdem wir die Schwierigkeit mit den Umgebungsvariabeln gelöst hatten, war sie eine verlässliche Hilfe, um die Qualität des Codes hoch zu halten.