Dokumentation

Marko Livajusic

28. Juni 2024

Contents

1 Einführung **1**
1.1 Tech-Stack 2
1.2 Projekt-Struktur 2
1.3 Schwierigkeiten / Design-Entscheidungen 2

2 Datenbankdesign **2**
2.1 Entitätstypen (Tabellen) 3
2.1.1 AuroraUsers 3
2.1.2 Roles 3
2.1.3 AuroraGifs 3
2.1.4 GifCategory 4
2.1.5 BelongsTo 4
2.1.6 Follows 4
2.1.7 Comments 4
2.1.8 Likes 5
2.1.9 ProfilePictures 5
2.1.10 Settings 5
2.1.11 Notifications 6
2.2 Beziehungen 6
2.2.1 Relationenmodell 7

1 Einführung

Mithilfe von Aurora lassen sich GIFs in 2D und 3D erstellen. Diese GIFs können für Memes verwendet und anschließend gespeichert werden. Die erstellten GIFs sind herunterladbar und teilbar. Aurora bietet eine Plattform, um die Kreationen mit der Welt zu teilen. Es ist eine einfache und kreative Möglichkeit, unterhaltsame Inhalte zu erstellen und zu verbreiten.

1.1 Tech-Stack

Für die Umsetzung des Projekts habe ich die Programmiersprache Java verwendet, das Spring Boot Framework für die App selbst und Vaadin für die Benutzeroberfläche und die grafische Darstellung. Für das Datenbankmanagementsystem habe ich PostgreSQL verwendet und das gesamte Projekt wird mit Maven kompiliert.

1.2 Projekt-Struktur

Aurora setzt sich aus den folgenden Ordnern zusammen (`/src/main/java/com/livajusic/marko`):

1. `/configuration`: In diesem Ordner wird Spring Security modifiziert, um Anfragen zuzulassen und die von den Benutzern hochgeladenen Inhalte so zu regulieren, wie sie vom Server bereitgestellt werden sollen.
2. `/db_repos`: Interfaces aus diesem Ordner erweitern JpaRepository und ermöglichen so den Zugriff auf die jeweiligen Repositories.
3. `/services`: Klassen aus diesem Ordner interagieren mit den Repositories aus (`/db_repos`) und aktualisieren die Werte in den Datenbanktabellen.
4. `/tables`: Klassen aus diesem Ordner verwenden JPA, um die von Spring erstellten Tabellen zu modellieren.
5. `/views`: Klassen aus diesem Ordner verwenden Vaadin, um die App im Web anzuzeigen.

1.3 Schwierigkeiten / Design-Entscheidungen

Bisher bereitete mir die Anzeige jeglicher Bilder (sei es die Anzeige von GIFs, die von Nutzern hochgeladen wurden, oder von Profilbildern) Probleme, da ich sie zunächst als statische Dateien im Spring-Ordner `/resource/` speicherte und nur ihre Pfade in der Datenbank hinterlegte.

Trotz verschiedener Konfigurationen, die versuchten, die Erzeugung statischer Inhalte zu regeln, wurden nur die "alt"-Elemente von HTML angezeigt, da die Bilder nicht verfügbar waren. Erst nach einem Neustart der App konnten alle Bilder korrekt angezeigt werden. Da ich dies aber nicht wollte, habe ich die Datenbank so geändert, dass die Bilder als BLOBS gespeichert werden, was zwar zu einem Leistungsverlust führt, aber für den Benutzer viel angenehmer ist.

2 Datenbankdesign

In `index.html` ist das ER-Diagramm von Aurora zu sehen. Dabei sind folgende Entitätstypen vorhanden:

2.1 Entitätstypen (Tabellen)

2.1.1 AuroraUsers

Die Tabelle AuroraUsers speichert Informationen über die Benutzer der Plattform, die sich auf `/register` registrieren und damit ein Konto anlegen können. Jede Zeile in dieser Tabelle repräsentiert einen einzelnen Benutzer und enthält die folgenden Spalten:

1. `userId` (Primärschlüssel): Eindeutige Identifikationsnummer für jeden Benutzer.
2. `username`: Der Benutzername des Benutzers.
3. `email`: Die E-Mail-Adresse des Benutzers. Während bei der Anmeldung (`/login`) der Nutzername gebraucht wird, soll die E-Mail-Addresse für das Zurücksetzen des Passworts dienen.
4. `password`: Das Passwort des Benutzers, welches gehasht gespeichert wird.

2.1.2 Roles

Die Tabelle Roles speichert die verschiedenen Rollen, die ein Benutzer haben kann. Jede Zeile in dieser Tabelle repräsentiert eine Rolle und enthält die folgenden Spalten:

- `role`: Ein Teil des Primärschlüssel (also als Composite-Key), ein String, welches eine Rolle repräsentiert.
- `user_id`: Der andere Teil des Primärschlüssel, referenziert `user_id` aus `AuroraUsers`

2.1.3 AuroraGifs

Die Tabelle AuroraGifs speichert die GIFs, die von Benutzern erstellt bzw. veröffentlicht wurden. Jede Zeile in dieser Tabelle repräsentiert ein einzelnes GIF und enthält die folgenden Spalten:

1. `gif_id` (Primärschlüssel): Eindeutige Identifikationsnummer für jedes GIF.
2. `user_id` (Fremdschlüssel zu AuroraUsers): Die ID des Benutzers, der das GIF erstellt bzw. veröffentlicht hat.
3. `license`: Die Lizenz des GIFs.
4. `publish_date`: Das Veröffentlichungsdatum des GIFs.
5. `image_data`: Die Bilddaten des GIFs, gespeichert als `bytea`.

2.1.4 GifCategory

Die Tabelle GifCategory speichert die verschiedenen Kategorien, in die GIFs eingeordnet werden können. Jede Zeile in dieser Tabelle repräsentiert eine Kategorie und enthält die folgenden Spalten:

- `category_id`: Primärschlüssel, eindeutige Identifikationsnummer für jede Kategorie.
- `category`: Der Name der Kategorie, also die Kategorie selbst.

2.1.5 BelongsTo

Die Tabelle BelongsTo verknüpft GIFs mit Kategorien. Jede Zeile in dieser Tabelle repräsentiert eine Zuordnung von einem GIF zu einer Kategorie und enthält die folgenden Spalten:

- `gif_id`: Fremdschlüssel zu AuroraGifs, d.h. die ID des GIFs.
- `category_id`: Fremdschlüssel zu GifCategory, die ID der Kategorie.

2.1.6 Follows

Die Tabelle Follows repräsentiert die "Folgen"-Beziehungen zwischen Benutzern. Jede Zeile in dieser Tabelle stellt eine Folgebeziehung dar und enthält die folgenden Spalten:

- `user_id`: Fremdschlüssel zu AuroraUsers, die ID des Benutzers, der einem anderen Benutzer folgt.
- `follows_user_id`: Fremdschlüssel zu AuroraUsers, die ID des Benutzers, der gefolgt wird.
- `followed_at`: Das Veröffentlichungsdatum, an welchem ein Benutzer einem anderen Benutzer gefolgt ist.

2.1.7 Comments

Die Tabelle Comments speichert Kommentare, die von Benutzern zu GIFs abgegeben werden. Jede Zeile in dieser Tabelle repräsentiert einen einzelnen Kommentar und enthält die folgenden Spalten:

- `comment_id`: Primärschlüssel, eindeutige Identifikationsnummer für jeden Kommentar.
- `gif_id`: Fremdschlüssel zu AuroraGifs, die ID des GIFs, zu dem der Kommentar abgegeben wurde.
- `user_id`: Fremdschlüssel zu AuroraUsers, die ID des Benutzers, der den Kommentar abgegeben hat.
- `comment_text`: Der Text des Kommentars.
- `created_at`: Das Erstellungsdatum des Kommentars.

2.1.8 Likes

Die Tabelle Likes speichert die "Gefällt mir"-Angaben von Benutzern zu GIFs. Jede Zeile in dieser Tabelle repräsentiert eine "Gefällt mir"-Angabe und enthält die folgenden Spalten:

- `like_id`: Primärschlüssel, eindeutige Identifikationsnummer für jede "Gefällt mir"-Angabe.
- `user_id`: Fremdschlüssel zu AuroraUsers, die ID des Benutzers, der das GIF geliked hat.
- `gif_id`: Fremdschlüssel zu AuroraGifs, die ID des GIFs, das geliked wurde.

2.1.9 ProfilePictures

Die Tabelle ProfilePictures speichert die Profilbilder der Benutzer. Jede Zeile in dieser Tabelle repräsentiert ein Profilbild und enthält die folgenden Spalten:

- `picture_id`: Primärschlüssel, eindeutige Identifikationsnummer für jedes Profilbild.
- `user_id`: Primärschlüssel und Fremdschlüssel zu AuroraUsers, die ID des Benutzers, zu dem das Profilbild gehört.
- `image_data`: Das Profilbild des Benutzers, , gespeichert als `bytea`.

2.1.10 Settings

Die Tabelle Settings speichert benutzerdefinierte Einstellungen. Jede Zeile in dieser Tabelle repräsentiert eine Einstellung und enthält die folgenden Spalten:

- `id`: Primärschlüssel und Fremdschlüssel zu AuroraUsers, die ID des Benutzers, zu dem die Einstellung gehört.
- `user_id`: Der Schlüssel der Einstellung.
- `language`: Der Wert der Einstellung.
- `theme`: Der Wert der Einstellung.
- `others_can_see_my_followers`: Ein Boolean, welches repräsentiert, ob andere Benutzer sehen können, welche Benutzer dem Benutzer folgen.
- `others_can_see_whoiam_following`: Ein Boolean, welches repräsentiert, ob andere Benutzer sehen können, welche Benutzern der Benutzer folgt.

2.1.11 Notifications

Die Tabelle Notifications speichert Benachrichtigungen, die die Benutzer erhalten. Jede Zeile in dieser Tabelle repräsentiert eine Benachrichtigung und enthält die folgenden Spalten:

- `notificationId`: Primärschlüssel, eindeutige Identifikationsnummer für jede Benachrichtigung.
- `created_at`: Das Erstellungsdatum der Benachrichtigung.
- `message`: Die Nachricht, die dem Nutzer beim Öffnen der Benachrichtigung gezeigt werden soll.
- `user_id`: Referenziert den `user_id` aus `AuroraUsers` (Fremdschlüssel).

2.2 Beziehungen

1. **AuroraUsers erstellt AuroraGifs**
   - Ein Benutzer **kann n** GIFs erstellen.
   - Ein GIF **muss** von genau einem Benutzer erstellt werden (1:n).
2. **AuroraUsers folgt AuroraUsers**
   - Ein Benutzer **kann n** anderen Benutzern folgen.
   - Ein anderer Benutzer kann **m** Benutzern folgen (m:n).
3. **AuroraUsers kommentiert AuroraGifs**
   - Ein Benutzer **kann n** Kommentare zu verschiedenen GIFs abgeben.
   - Ein GIF **kann m** Kommentare von verschiedenen Benutzern erhalten (m:n).
4. **AuroraUsers gefällt AuroraGifs**
   - Ein Benutzer **kann n** GIFs liken.
   - Ein GIF **kann** von **m** Benutzern geliked werden (n:m).
5. **AuroraUsers hat ProfilePictures**
   - Ein Benutzer **muss 1** Profilbild haben.
   - Ein Profilbild **muss 1** Benutzer gehören (1:1).
6. **AuroraUsers verwaltet Settings**
   - Ein Benutzer **muss 1** Einstellung haben.
   - Eine Einstellung **muss 1** Benutzer gehören (1:1).
7. **AuroraGifs gehört zu GifCategory**

- Ein GIF **kann n** Kategorien zugeordnet sein.
- Eine Kategorie **kann m** GIFs zugeordnet sein (n:m).

8. **AuroraUsers hat Roles**
   - Ein Benutzer **muss n**, wobei m i.d.R **1** ist, Rolle(n) haben.
   - Eine Rolle **muss 1** Benutzer zugeordnet sein (1:n).

2.2.1 Relationenmodell

Unoptimiert:

```
AuroraUser(user_id, username, email, password)
folgt(user_id, follows_user_id)
erstellt(user_id, gif_id)
AuroraGIF(gif_id, license, publish_date, image_data)
gefällt(user_id, gif_id)
schreibt(user_id, comment_id)
Comment(comment_id, created_at, comment_text)
Comment.gehört_zu(comment_id, gif_id)
AuroraGIF.gehört_zur(gif_id, category_id)
Kategorie(category_id, category_id)
erhält(user_id, notification_id)
Notification(notification_id, created_at, message_text)
hat(user_id, role, r.user_id)
Role(role, role_user_id)
ProfilePicture(picture_id, image_data)
verwaltet(user_id, settings_id)
Settings(settings_id, language, theme, others_can_see_my_followers,
 others_can_see_whoiam_following)
```

Optimiert:

```
AuroraUser(user_id, username, email, password)
AuroraGIF(gif_id, user_id, publish_date, path, license)
Kommentar(comment_id, user_id, gif_id, created_at, comment_text)
folgt(follows_user_id, user_id)
gefällt(user_id, gif_id)
gehört_zur(gif_id, category_id)
ist(auroraUserId, user_id)
Rolle(user_id, role)
Notification(notification_id, created_at, message_text, user_id)
Role(role, role_user_id)
ProfilePicture(picture_id, image_data, user_id)
Settings(settings_id, language, theme, others_can_see_my_followers,
 others_can_see_whoiam_following, user_id)
```