**SnapSort**

# **General idea:**

## Photo collection management system. In contrast to e.g. instagram the collections are private, as in visible only to the logged user. The main functionality for the user is being able to upload and download photos from the application.

# **Links:**

* <https://github.com/patkub2/SnapSort>
* <https://trello.com/invite/b/taLT5C14/ATTI3fbb7b1104d328580a9f17df2338122cBB943B44/untitled-board>
* <https://docs.google.com/document/d/1kCaM3dTO0v237yhWdcmlG3YiMHLyUrToSRPlUYrED-M/edit?usp=sharing>

# **Features:**

## Core:

* uploading photos
* filter the photos according to the selected criteria (specific attributes / keywords in the description)
* add a description to photos, (comments, tags)
* creation of account ( the user will be able to see the pictures only after logging in ) using email and a password
* sharing the gallery (e.g. using a link or to another account)

## Additional:

* viewing the photo thumbnail
* download the selected photo(s)
* grouping pictures in custom groups/folders (manually)
* automatically grouping variations of the same picture into a cluster (group)
* responsive web design - media queries
* logging in with a google/facebook account

# **Technology:**

## Front-end

* Next.js
* Typescript
* Styled Components

## Back-end

* Java Spring Boot
* MySql
* JDBC
* Thymeleaf
* Lombok

# **Team members:**

| **Name and surname** | **Role** |
| --- | --- |
| Patryk Kubala | Front-end dev |
| Radek Gąsior | Back-end dev |
| Łukasz Rak | Back-end dev |
| Marcel Filejecki | Front-end dev |
| Bartosz Wójcik | Back-end dev |
| Mateusz Kies | Back-end dev |

At the moment the work is split into back-end/front-end. We have two members who have experience with JS/React or Next.js. The rest of us (4 members) will focus on implementation of the back-end of the web application. Moreover, we will later split the work into more detailed tasks assigned to each of the individual members.

# **Road map:**

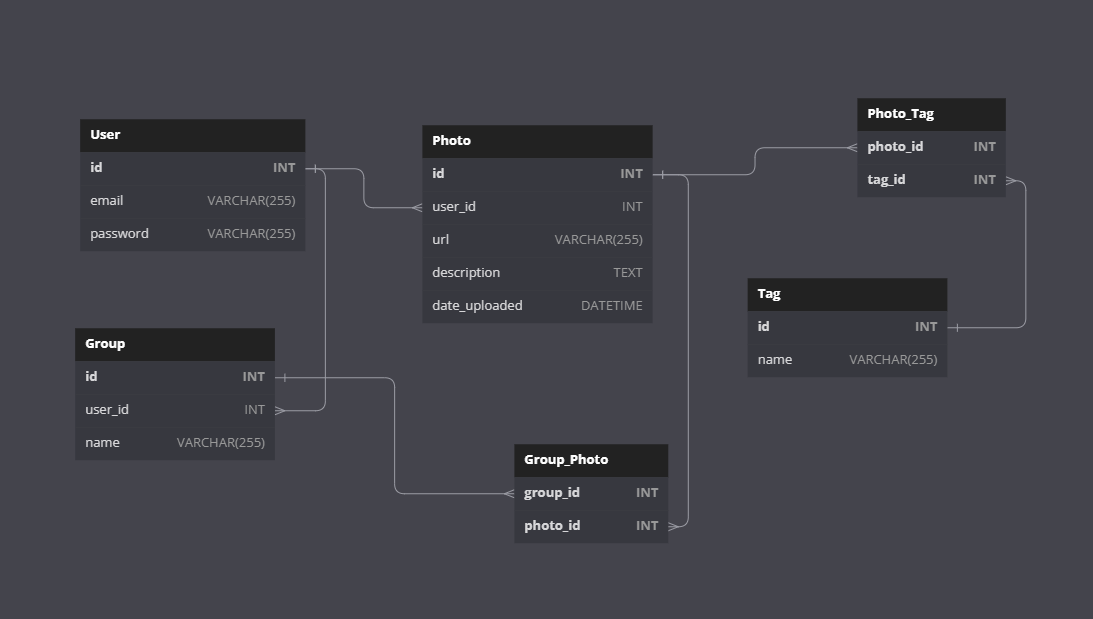
## Backend:

1. Creating accounts (with validation).
2. Creating the DB - enabling the uploading of photos to the application.
3. Adding tags/descriptions to photos.
4. Filtering the photos - downloading selected photos.
5. Sharing the gallery/photos (using a link or to another account).
6. Creating manual & automatic grouping of photos.

## Frontend:

1. Create the basic page design in figma app.
2. Implement the basic design from figma.
3. Implementing the registration with validation.
4. Connecting with the database.
5. Taking care of API and db requests.

# **Basic database scheme:**

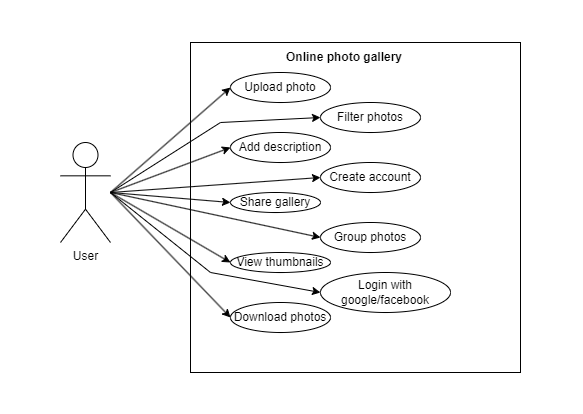
****

# 

# **Use cases & views:**

| **Use case** | **Actor** | **Description** |
| --- | --- | --- |
| Upload Photo | User | User uploads a photo to their private gallery. |
| Filter Photos | User | User filters photos in their gallery based on specific attributes or keywords in the description. |
| Add Description | User | User adds a description, comments, or tags to their photos. |
| Create Account | User | User creates an account using their email and a password. |
| Share Gallery | User | User shares their gallery with others via a link or to another account. |
| View Thumbnails | User | User views photo thumbnails in their gallery. |
| Download Photos | User | User downloads selected photos from their gallery. |
| Group Photos | User | User groups photos into custom groups or folders manually. |
| Cluster Photos | System | System automatically groups variations of the same photo into a cluster. |
| Login with Google/Facebook | User | User can log in using their Google or Facebook account |

**Use case diagram:**



The UI design can be found in the link below:

* <https://www.figma.com/file/rFFoqQOhokYt8vAeXgCPod/SnapSort>

