Pada final project ini, kalian akan diminta untuk membuat suatu aplikasi bernama MyGram, yang dimana pada aplikasi ini kalian dapat menyimpan foto maupun membuat comment untuk foto orang lain. Aplikasi ini akan dilengkapi dengan proses CRUD dengan table dan alur yang akan dijelaskan berikut ini:

1. Project ini bebas dikerjakan dengan library apapun. Namun agar proses pengerjaannya lebih cepat dan mudah, disarankan untuk menggunakan framework Gin Gonic dan orm Gorm.

2. Berikut merupakan library/package yang wajib digunakan

- github.com/dgrijalva/jwt-go
- golang.org/x/crypto

3. Dalam project ini akan memerlukan 4 table. Berikut merupakan table-table dan field-field yang diperlukan dalam project ini.

User

- id (Primary key)
- username (string)
- email (string)
- password (string)
- age (integer)
- created_at (Date)
- updated_at (Date)

Photo

- id (Primary key)
- title (string)
- caption (string)
- photo_url (string)
- user_id (Foreign Key Of User Table)
- created_at (Date)
- updated_at (Date)

Comment

- id (Primary Key)
- user_id (Foreign Key Of User Table)
- photo_id (Foreign Key Of Photo Table)
- message (string)
- created_at (Date)
- updated_at (Date)

SocialMedia

- id (Primary Key)
- name (String/ varchar)
- social_media_url (String/ Text)
- UserId(Foreign Key Of User Table)
- created_at (Date)
- updated_at (Date)

4. Keempat table tersebut harus mempunyai validasi-validasi pada tiap field-field nya. Validasi boleh dibuat sendiri ataupun menggunakan package seperti Go Validator. Berikut merupakan penjelasan validasinya:

A. Validasi untuk table User

1. Field email

- Validasi pengecekan format email yang valid

- Validasi agar dapat menjadi unique index

- Validasi agar field email tidak boleh kosong atau harus terisi

2. Field username

- Validasi agar dapat menjadi unique index

- Validasi agar field username tidak boleh kosong atau harus terisi

3. Field password

- Validasi agar field password tidak boleh kosong

- Validasi agar field password minimal memiliki panjang sebanyak 6 karakter

4. Field age

- Validasi agar field age tidak boleh kosong atau harus terisi

- Validasi agar field age minimal memiliki nilai diatas 8

B. Validasi Untuk Table Photo

1. Field title

- Validasi agar field title tidak boleh kosong

2. Field photo_url

-Validasi agar field photo_url tidak boleh kosong atau harus terisi

C. Validasi Untuk Table SocialMedia

1. Field name

-Validasi agar field name tidak boleh kosong atau haru terisi

2. Field social_media_url

-Validasi agar field social_media_url tidak boleh kosong atau harus terisi

D. Validasi untuk Table Comment

1.Field message

-Validasi agar field message tidak boleh kosong atau harus terisi

4. Berikut merupakan alur dari aplikasinya:

- Endpoint-endpoint untuk mengakses data pada table SocialMedia, Photo, dan Comment harus melalui proses autentikasi terlebih dahulu, dan proses autentitakasinya wajib menggunakan JsonWebToken.

- Untuk endpoint-endpoint yang berguna untuk memodifikasi data kepemilikan seperti **Update** atau **delete** maka harus melalui proses autorisasi.

5. Berikut merupakan endpoint-endpoint yang harus dibuat beserta request body dan response data yang harus diikuti **(Wajib)**

- Users

POST /users/register

Request

- body:

```
{
    "age": "integer",
    "email": "string",
    "password": "string",
    "username": "string"
}
```

Response:

- status 201
- data:

Notes: Password user harus di hash menggunakan package **Bcrypt** sebelum di simpan ke database.

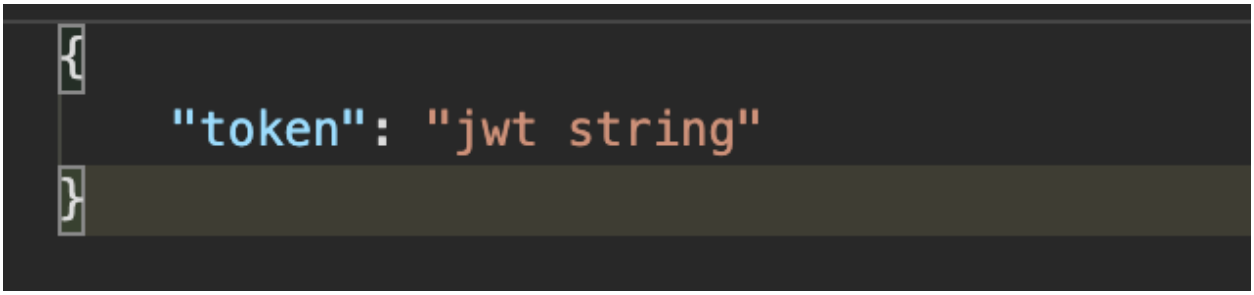
POST /users/login

Request

- body:

Response:

- status 200
- data:



Notes: Pada endpoint ini, wajib melakukan logika user login yang dimana harus melakukan pengecekan email dan password user. Pengecekan password wajib dilakukan dengan bantuan library/package **Bcrypt**.

PUT /users

Request:

- headers: Authorization (Bearer token string)
- params: userId (integer)
- body:

Response:

- status 200
- data:

```
{
    "id": "integer",
    "email": "string",
    "username": "string",
    "age": "interger",
    "updated_at": "date"
}
```

Notes: Endpoint ini perlu melewati proses autentikasi terlebih dahulu. Proses autentikasi wajib dilakukan dengan bantuan package/library **JsonWebToken**.

DELETE /users

Request

- headers: Authorization (Bearer token string)

Response:

- status 200
- data:

```
{
    "message": "Your account has been successfully deleted"
}
```

Notes: Endpoint ini perlu melewati proses autentikasi terlebih dahulu. Proses autentikasi wajib dilakukan dengan bantuan package/library **JsonWebToken**.

- Photos

POST /photos

Request

- headers: Authorization (Bearer token string)
- body:

```
{
    "title": "string",
    "caption": "string",
    "photo_url": "string"
}
```

Response:

- status 201
- data:

Notes: Endpoint ini perlu melewati proses autentikasi terlebih dahulu. Proses autentikasi wajib dilakukan dengan bantuan package/library **JsonWebToken**.

GET /photos

Request

- headers: Authorization (Bearer token string)

Response:

- status 200
- data:

```
[
    {
        "id": "integer",
        "title": "string",
        "caption": "string",
        "photo_url": "string",
        "user_id": "integer",
        "created_at": "date",
        "updated_at": "date",
        "User": {
            "email": "string",
            "username": "string"
        }
    },
        {
        "id": "integer",
        "title": "string",
        "caption": "string",
        "photo_url": "string",
        "user_id": "integer",
        "created_at": "date",
        "updated_at": "date",
        "User": {
            "email": "string",
            "username": "string"
        }
    }
]
```


Notes: Endpoint ini perlu melewati proses autentikasi terlebih dahulu. Proses autentikasi wajib dilakukan dengan bantuan package/library **JsonWebToken**.

PUT /photos/:photoId

Request

- headers: Authorization (Bearer token string)
- params: photoId (string)
- body:

```
{
    "title": "string",
    "caption": "string",
    "photo_url": "string"
}
```

Response:

- status 200
- data:

```
{
    "id": "integer",
    "title": "string",
    "caption": "string",
    "photo_url": "string",
    "user_id": "integer",
    "updated_at": "date"
}
```

Notes: Endpoint ini perlu melewati proses autentikasi dan autorisasi terlebih dahulu. Proses autentikasi wajib dilakukan dengan bantuan

package/library **JsonWebToken**. Dan alur proses autorisasinya adalah user hanya boleh mengupdate data photo miliknya sendiri.

DELETE /photos/:photoId

Request

- headers: Authorization (Bearer token string)
- params: photoId (integer)

Response:

- status 200
- data:

```
{
    "message": "Your photo has been successfully deleted"
}
```

Notes: Endpoint ini perlu melewati proses autentikasi dan autorisasi terlebih dahulu. Proses autentikasi wajib dilakukan dengan bantuan package/library **JsonWebToken**. Dan alur proses autorisasinya adalah user hanya boleh menghapus data photo miliknya sendiri.

- Comments

POST /comments

Request

- headers: Authorization (Bearer token string)
- body:

```
{
    "message": "string",
    "photo_id": "integer"
}
```

Response:

- status 201
- data:

```
{
    "id": "integer"
    "message": "string",
    "photo_id": "integer",
    "user_id": "integer",
    "created_at": "date"
}
```

Notes: Endpoint ini perlu melewati proses autentikasi dan autorisasi terlebih dahulu. Proses autentikasi wajib dilakukan dengan bantuan package/library **JsonWebToken**.

GET /comments

Request

- headers: Authorization (Bearer token string)

Response:

- status 200
- data:


```
[
    {
        "id": "integer",
        "message": "string",
        "photo_id": "integer",
        "user_id": "integer",
        "updated_at": "date",
        "created_at": "date",
        "User": {
            "id": "integer"
            "email": "string",
            "username": "string"
        },
        "Photo": {
            "id": "integer",
            "title": "string",
            "caption": "string",
            "photo_url": "string",
            "user_id": "integer",
        }
    }
]
```

Notes: Endpoint ini perlu melewati proses autentikasi dan autorisasi terlebih dahulu. Proses autentikasi wajib dilakukan dengan bantuan package/library **JsonWebToken**.

PUT /comments/:commentId

Request

- headers: Authorization (Bearer token string)
- params: commentId (integer)
- body:

```
{
    "message": "string"
}
```

Response:

- status 200
- data:

```
{
    "id": "integer",
    "title": "string",
    "caption": "string",
    "photo_url": "string",
    "user_id": "integer",
    "updated_at": "date"
}
```

Notes: Endpoint ini perlu melewati proses autentikasi dan autorisasi terlebih dahulu. Proses autentikasi wajib dilakukan dengan bantuan package/library **JsonWebToken**. Dan alur proses autorisasinya adalah user hanya boleh mengupdate data comment miliknya sendiri.

DELETE /comments/:commentId

Request

- headers: Authorization (Bearer token string)
- params: commentId (integer)

Response:

- status 200
- data:

```
{
    "message": "Your comment has been successfully deleted"
}
```

Notes: Endpoint ini perlu melewati proses autentikasi dan autorisasi terlebih dahulu. Proses autentikasi wajib dilakukan dengan bantuan package/library **JsonWebToken**. Dan alur proses autorisasinya adalah user hanya boleh menghapus data comment miliknya sendiri.

- SocialMedias

POST /socialmedias

Request

- headers: Authorization (Bearer token string)
- body:

Response:

- status: 201
- body:

Notes: Endpoint ini perlu melewati proses autentikasi dan autorisasi terlebih dahulu. Proses autentikasi wajib dilakukan dengan bantuan package/library **JsonWebToken**.

GET /socialmedias

Request

- headers: Authorization (Bearer token string)

Response:

- status: 200
- body:

```
{
    "social_medias": [
        {
            "id": "integer",
            "name": "string",
            "social_media_url": "string",
            "UserId": "integer",
            "createdAt": "date",
            "updatedAt": "date",
            "User": {
                "id": "integer",
                "username": "string",
                "profile_image_url": "string"
            }
        }
    ]
}
```

Notes: Endpoint ini perlu melewati proses autentikasi dan autorisasi terlebih dahulu. Proses autentikasi wajib dilakukan dengan bantuan package/library **JsonWebToken**.

PUT /socialmedias/:socialMediaId

Request

- headers: Authorization (Bearer token string)
- params: socialMediaId (integer)
- body:

Response:

- status: 200
- body:

Notes: Endpoint ini perlu melewati proses autentikasi dan autorisasi terlebih dahulu. Proses autentikasi wajib dilakukan dengan bantuan package/library **JsonWebToken**. Dan alur proses autorisasinya adalah user hanya boleh mengupdate data social media miliknya sendiri.

DELETE /socialmedias/:socialMediaId

Request

- headers: Authorization (Bearer token string)
- params: socialMediaId (integer)

Response:

- status: 200
- body:

```
{
    "message": "Your social media has been successfully deleted"
}
```

Notes (Wajib)

-Seluruh routing endpoint diatas harus diikuti dengan betul

-Seluruh request body, headers maupun request params harus diikuti dengan betul.

-Seluruh response status, maupun response data nya harus diikuti dengan betul.

- Mohon untuk diperhatikan notes-notes yang telah diberikan diatas seperti endpoint-endpoint yang harus melewati proses autentikasi dan yang tidak. Begitu juga dengan proses autorisasinya.

-Perlu diingat disini bahwa proses autorisasi dilakukan setelah proses autentikasi, bukan sebaliknya.