

Galeria - dodawanie obrazków,
przeglądanie galerii

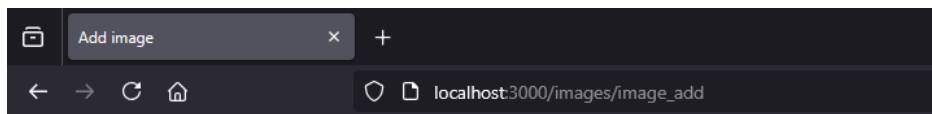
Zadanie 1

Dodaj do aplikacji "Gallery" możliwość dodawania obrazków do galerii i ich przeglądania w następujący sposób:

- a) dodawać obrazki może tylko zalogowany użytkownik i tylko do swoich galerii (admin może dodawać obrazki do wszystkich galerii). Dla uproszczenia, na początek dodawanie obrazka może polegać na dodaniu informacji o nim (ścieżki) do bazy. Przykładowe pliki obrazków dla uproszczenia należy wgrać ręcznie do podkatalogu projektu: [./public/images](#). Potem można dodać upload (np. korzystając z pakietu *formidable*).
- b) przeglądać obrazki w danej galerii może dowolny, zalogowany, użytkownik, po uprzednim jej wybraniu.

Zadanie 1a (dodawanie obrazków)

Dodawanie obrazka. Użytkownik-właściciel galerii musi być zalogowany.



- [Home](#)
- [Stats](#)
- [Users](#)
- [Galleries](#)
- [Images](#)
- [Browse gallery](#)

Add image

Image Name:

Description:

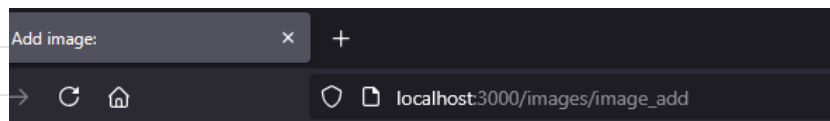
Select file:

 Porsche_911_Carrera_S.jpg

Select gallery:

- [Add user](#)
- [Add gallery](#)
- [Add image](#)

- [Login \(logged as jkowalski\)](#)
- [Logout](#)



- [Home](#)
- [Stats](#)
- [Users](#)
- [Galleries](#)
- [Images](#)
- [Browse gallery](#)

- [Add user](#)
- [Add gallery](#)
- [Add image](#)

- [Login \(logged as jkowalski\)](#)
- [Logout](#)

Add image:

Image Name:

Description:

Select file:

 Nie wybrano pliku.

Select gallery:

- Image added!

Zadanie 1b (przeglądanie galerii)

Przeglądać galerię może dowolny, zalogowany użytkownik.

View gallery: x +

localhost3000/galleries/gallery_browse

- [Home](#)
- [Stats](#)
- [Users](#)
- [Galleries](#)
- [Images](#)
- [Browse gallery](#)

- [Add user](#)
- [Add gallery](#)
- [Add image](#)


- [Login \(logged as jkowalski\)](#)
- [Logout](#)

View gallery:

Select gallery:


Auta

Submit




Aston Martin

Description: Aston Martin



Corvette

Description: Chevrolet Corvette



Porsche

Description: Porsche

Zadanie 1a - pomoc (dodawanie obrazków)

1) W **./routes/images.js** należy dodać ścieżki GET i POST do obsługi dodawania obrazka.

```
//Wyświetlanie formularza dodawania obrazka - GET.  
router.get("/image_add", authenticate, image_controller.image_add_get);  
  
//Przetwarzanie danych formularza dodawania obrazka - POST.  
router.post("/image_add", authenticate, image_controller.image_add_post);
```

Zadanie 1a - pomoc

2) W `./controllers/imageControler.js` należy dodać odpowiednie funkcje (GET i POST) do obsługi wyświetlania i przetwarzania formularza dodawania obrazka.

```
// Import modeli gallery i user - potrzebne do doawania obrazka
const gallery = require("../models/gallery");
const user = require("../models/user");

// OBSŁUGA DODAWANIA OBRAZKA
// Kontroler wyświetlania formularza dodawania obrazka - GET.
exports.image_add_get = asyncHandler(async (req, res, next) => {
  //utworzyć kod podobny jak przy wyświetlaniu formularza dodawania Galerii - GET.
})

...

// Import funkcji walidatora.
const { body, validationResult } = require("express-validator");

// OBSŁUGA DODAWANIA OBRAZKA
// Kontroler przetwarzania danych formularza dodawania obrazka - POST.
exports.image_add_post = [
  //utworzyć kod podobny jak przy przetwarzaniu danych formularza dodawania Galerii - POST.
]
```

Zadanie 1a - pomoc

3) Przykładowy szablon dodawania obrazka ./views/image_form.

```
extends layout

block content
  h1=title

  form(method='POST')
    div.form-group
      label(for='i_name') Image Name:
      input#i_name.form-control(type='text', placeholder='min. 2 litery' name='i_name' required value=(undefined===image ? '' :
image.name) )
      label(for='i_description') Description:
      input#i_description.form-control(type='text', placeholder='min. 2 litery' name='i_description' required value=(undefined===image
? '' : image.description))
      label(for='i_path') Select file:
      input#i_path.form-control(type='file', placeholder='min. 3 litery' name='i_path' required value="")
      label(for='i_gallery') Select gallery:
      // lista rozwijana wyboru galerii
      select#i_gallery.form-control(name='i_gallery' required)
        option(value='') --Select a gallery--
        for gallery in galleries
          if image
            if gallery._id.toString()===image.gallery._id.toString()
              option(value=gallery._id selected) #{gallery.name}
            else
              option(value=gallery._id) #{gallery.name}
          else
            option(value=gallery._id) #{gallery.name}
      button.btn.btn-primary(type='submit') Submit

  if messages
    ul
      for message in messages
        li!= message
```

Zadanie 1b - pomoc (przeglądanie galerii)

1) W aplikacji głównej **./app.js** dla ułatwienia można podpiąć pod ścieżkę *'/galleries'* katalog plików statycznych *'public/images'*.

```
app.use('/galleries', express.static(path.join(__dirname, 'public/images')));
```


Zadanie 1b - pomoc

2) W **./routes/galleries.js** należy dodać kolejne ścieżki GET i POST do obsługi przeglądania galerii (ścieżka `/galleries/gallery_browse`).

```
// Wyświetlanie formularza przeglądania galerii GET (/galleries/gallery_browse)
router.get("/gallery_browse", authenticate, gallery_controller.gallery_browse_get);

// Przetwarzanie danych formularza przeglądania galerii POST
(/galleries/gallery_browse)
router.post("/gallery_browse", authenticate, gallery_controller.gallery_browse_post);
```

Zadanie 1b - pomoc

3) W `./controllers/galleriesControler.js` należy dodać odpowiednie funkcje GET i POST do obsługi ścieżki przeglądania galerii (w przykładzie użyto tego samego szablonu `'gallery_browse'` dla GET i POST).

```
// Import modelu Image
const image = require("../models/image");

// Kontroler wyświetlania formularza GET gallery_browse - wyświetla formularz wyboru galerii
exports.gallery_browse_get = asyncHandler(async (req, res, next) => {
  // Pokaż formularz wyboru galerii
  const all_galleries = await gallery.find({}).exec();
  res.render("gallery_browse", { title: "Select gallery:", galleries: all_galleries, loggedUser: req.loggedUser
});
});

// Kontroler przetwarzania formularza POST gallery_browse - wyświetla brazki, ale też formularz wyboru galerii
exports.gallery_browse_post = asyncHandler(async (req, res, next) => {
  // Pokaż listę obrazków wybranej galerii
  const all_galleries = await gallery.find({}).exec();

  let gallery_images = [];
  let sel_gallery = null

  if (req.body.s_gallery) {
    gallery_images = await image.find({ gallery: req.body.s_gallery }).exec();
    sel_gallery = req.body.s_gallery
  }

  res.render("gallery_browse", { title: "View gallery:", galleries: all_galleries, images: gallery_images,
sel_gallery: sel_gallery, loggedUser: req.loggedUser});
});
```

Zadanie 1b - pomoc

4) Przykładowy szablon ./views/gallery_browse.

```
extends layout

block content
  h1=title

  form(method='POST')
    div.form-group
      label(for='s_gallery') Select gallery:
      //- lista rozwijana wyboru galerii
      select#s_gallery.form-control(name='s_gallery' required)
        for gallery in galleries
          if (sel_gallery == gallery._id)
            option(value=gallery._id selected) #{gallery.name}
          else
            option(value=gallery._id) #{gallery.name}
      button.btn.btn-primary(type='submit') Submit
  if images
    div(style="display:flex; flex-wrap:wrap;")
      each image in images
        div(style='display:flex; flex-direction:column; width:20%; height:100%; margin:20px;')
          img(src=image.path alt=image.path style='max-width:100%; object-fit:scale-down;')
          h2 #{image.name}
          a Description: #{image.description}
  else
    p There are no any images.
  if messages
    ul
      for message in messages
        li!= message
```