 МІНІСТЕРСТВО ОСВІТИ I НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ   
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ

КАФЕДРА БІОМЕДИЧНОЇ КІБЕРНЕТИКИ

**Комп’ютерний практикум №9**

з дисципліни «Веб-технології та веб-дизайн-1»

**Виконав:**

студентка гр. БС-81

Кисельова І.Д.

**Перевірив:**

ас. каф. БМК Матвійчук О.В.

Зараховано від \_\_\_.\_\_\_.\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис викладача)

Київ-2020

**Завдання:**

**Створити динамічну галерею зображень (index.html).**

* Вибрати фреймворк для реалізації галереї (Vue, React, Angular)
* Реалізувати галерею зображень як у лабораторній роботі #6 (включаючи режим preview)
* в Preview режимі відображати картинки сусіди (по 1 з кожної сторони, на кільцевому списку. В галереї в кожен момент часу має бути не менше 3х зображень).
* Додати функціонал з видалення зображень
* Додати функціонал з розширення галереї (форму для додавання нових зображень)
* Галерею реалізувати за паттерном Model-View-ViewModel

Посилання на github: <https://github.com/Irines/WEB_commits>

# DragDropInput.js

import React from "react";

class DragDropInput extends React.Component {

  state = {

    drag: false,

  };

  dragCounter = 0;

  dropRef = React.createRef();

  inputRef = React.createRef();

  handleDrag = (e) => {

    e.preventDefault();

    e.stopPropagation();

  };

  handleDragIn = (e) => {

    e.preventDefault();

    e.stopPropagation();

*this*.dragCounter++;

    if (e.dataTransfer.items && e.dataTransfer.items.length > 0) {

*this*.setState({ drag: true });

    }

  };

  handleDragOut = (e) => {

    e.preventDefault();

    e.stopPropagation();

*this*.dragCounter--;

    console.log(*this*.dragCounter);

    if (*this*.dragCounter === 0) {

*this*.setState({ drag: false });

    }

  };

  handleDrop = (e) => {

    e.preventDefault();

    e.stopPropagation();

*this*.setState({ drag: false });

    if (e.dataTransfer.files && e.dataTransfer.files.length > 0) {

*this*.props.handleDrop(e.dataTransfer.files[0]);

      e.dataTransfer.clearData();

*this*.dragCounter = 0;

    }

  };

  componentDidMount() {

    let div = *this*.dropRef.current;

    div.addEventListener("dragenter", *this*.handleDragIn);

    div.addEventListener("dragleave", *this*.handleDragOut);

    div.addEventListener("dragover", *this*.handleDrag);

    div.addEventListener("drop", *this*.handleDrop);

  }

  componentWillUnmount() {

    let div = *this*.dropRef.current;

    div.removeEventListener("dragenter", *this*.handleDragIn);

    div.removeEventListener("dragleave", *this*.handleDragOut);

    div.removeEventListener("dragover", *this*.handleDrag);

    div.removeEventListener("drop", *this*.handleDrop);

  }

  render() {

    return (

      <div

        style={{

          width: "300px",

          height: "150px",

          display: "flex",

          alignItems: "center",

          justifyContent: "center",

          position: "relative",

        }}

        ref={*this*.dropRef}

      >

        <input

          ref={*this*.inputRef}

          id="image"

          name="image"

          type="file"

          onChange={(e) => {

*this*.props.changeHandler(e);

*this*.inputRef.current.files = null;

          }}

        />

        {*this*.state.drag && (

          <div

            style={{

              border: "dashed grey 4px",

              backgroundColor: "rgba(255,255,255,.8)",

              position: "absolute",

              top: 0,

              bottom: 0,

              left: 0,

              right: 0,

              zIndex: 9999,

              marginBottom: "15px",

            }}

          >

            <div

              style={{

                position: "absolute",

                top: "50%",

                transform: "translateY(-50%)",

                right: 0,

                left: 0,

                textAlign: "center",

                color: "grey",

                fontSize: 36,

              }}

            >

              <div>drop here :)</div>

            </div>

          </div>

        )}

        {*this*.props.children}

      </div>

    );

  }

}

export default DragDropInput;

# Slider.js

import React, { useEffect, useState } from "react";

const classes = require("./Slider.module.scss");

export const Slider = ({ images, initImg, setInit }) => {

  const [currentIdx, setCurrentIdx] = useState(null);

  useEffect(() => {

    setCurrentIdx(images.indexOf(initImg));

  }, [images, initImg]);

  const getPrevImage = () => {

    return currentIdx - 1 > -1

      ? images[currentIdx - 1]

      : images[images.length - 1];

  };

  const getNextImage = () => {

    return currentIdx + 1 < images.length ? images[currentIdx + 1] : images[0];

  };

  if (!initImg) {

    return <></>;

  }

  return (

    <div className={classes.Slider\_\_wrapper} onClick={() => setInit(null)}>

      <div

        className={classes.Slider\_\_body}

        onClick={(e) => e.stopPropagation()}

      >

        <div

          className={classes.Slider\_\_arrow\_prev}

          onClick={() => setInit(getPrevImage())}

        />

        <img src={getPrevImage()} alt="" className={classes.Slider\_\_prev\_img} />

        <img src={images[currentIdx]} alt="" className={classes.Slider\_\_img} />

        <img src={getNextImage()} alt="" className={classes.Slider\_\_next\_img} />

        <div

          className={classes.Slider\_\_arrow\_next}

          onClick={() => setInit(getNextImage())}

        />

      </div>

    </div>

  );

};

# Picturelist.js

import React, { useState } from "react";

import DragDropInput from "../UI/DragDropInput/DragDropInput";

const classes = require("./PictureList.module.scss");

const { PictureCard } = require("../PictureCard/PictureCard");

export const PictureList = ({ images, onAdd, onRemove }) => {

  const [uploadImg, setUploadImg] = useState(null);

  const changeHandler = (event) => {

    setUploadImg(event.target.files[0]);

  };

  return (

    <>

      <div className={classes.PictureList\_\_wrapper}>

        {images &&

          images.map((img, idx) => {

            return <PictureCard key={idx} imageSrc={img} onRemove={onRemove} />;

          })}

      </div>

      <div className={classes.user\_upload}>

        <DragDropInput

          changeHandler={changeHandler}

          handleDrop={(file) => {

            onAdd(file);

          }}

        />

        <button

          className={classes.PictureList\_\_button\_add}

          onClick={() => onAdd(uploadImg)}

        >

          Добавить

        </button>

      </div>

    </>

  );

};

# gallery.routes.js

const { Router } = require("express");

const router = Router();

const fs = require("fs");

const http = require("http");

const https = require("https");

const path = require("path");

const multer = require("multer");

const upload = multer({ dest: "static/" });

router.get("/loadImages", async (req, resp) => {

  const images = fs.readdirSync(path.join(\_\_rootdir, "static"));

  resp.json({ images });

});

router.post("/uploadImage", upload.single("image"), async (req, resp) => {

  const image = req.file;

  const newImagePath = path.join(\_\_rootdir, "/static", image.originalname);

  fs.rename(image.path, newImagePath, function (err) {

    if (err) {

      console.log(err);

      resp.send(500);

    } else {

      resp.json({

        message: "File uploaded successfully",

        filename: newImagePath.split("/").slice(-1)[0],

      });

    }

  });

});

router.post("/uploadImageUrl", async (req, resp) => {

  const imageUrl = req.body.uploadImgUrl;

  const file = fs.createWriteStream(

    path.join(\_\_rootdir, "static", imageUrl.split("/").slice(-1)[0])

  );

  try {

    http.get(imageUrl, (response) => {

      response.pipe(file);

    });

  } catch (e) {

    if (e.message.indexOf("Protocol") > -1) {

      https.get(imageUrl, (response) => {

        response.pipe(file);

      });

    }

  }

  resp.json({

    message: "File uploaded successfully",

  });

});

router.post("/removeImage", async (req, resp) => {

  const imageSrc = req.body.imageSrc;

  fs.unlink(path.join("static", imageSrc), () => {

    resp.json({

      message: "File was removed successfully",

    });

  });

});

module.exports = router;

