

*03/30/2019 00:41:41*

*9*

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

**Лабораторна робота №9**

з дисципліни “Основи web-програмування”

тема “Браузерний JavaScript та AJAX”

|  |  |  |
| --- | --- | --- |
| Виконав  Студент II курсу  групи КП-82  Кривчук Денис Віталійович  (*прізвище, ім’я, по батькові*)  варіант № |  | Перевірив  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладач  Гадиняк Руслан Анатолійович  (*прізвище, ім’я, по батькові*) |

Київ 2019

#### **Мета**

Навчитись створювати динамічні веб-сторінки за допомогою браузерних скриптових мов.  
 Навчитись працювати із технологією AJAX для отримання даних від REST API веб-сервера.

### **Загальні вимоги до завдання**

Реалізувати:

На сторінці перегляду і пошуку сутностей:

* 1. **Пошук**: реалізувати AJAX запит до REST API сервера із динамічним оновленням вмісту поточної web-сторінки та відображення результатів пошуку.
  2. **Пагінація**: реалізувати AJAX запити із динамічним оновленням вмісту поточної web-сторінки та відображення сторінки даних.

Використати:

* чистий браузерний JavaScript та AJAX (або **FetchAPI**, але без third-party залежностей, таких як JQuery чи axios)
* браузерний шаблонізатор рядків (наприклад, [**Mustache**](https://www.google.com/url?q=https://github.com/janl/mustache.js&amp;sa=D&amp;ust=1476659636650000&amp;usg=AFQjCNHD_43DunvosTq-KaBA5CgHTmvdjg)) для створення нових DOM-елементів сторінки браузерним JavaScript. Шаблони зробити зовнішніми ресурсами і завантажувати з браузерного JavaScript.
* для авторизації деяких\* запитів до REST API з попереднього завдання додати можливість перевірки **сесії**, або **JWT**.  
  (\* якщо у завданні потрібно виконувати доступ до endpoint з авторизацією)

**Тексти коду програм**

|  |
| --- |
| **app.js** |
| const express = require('express');  const path = require('path');  const busboyBodyParser = require('busboy-body-parser');  const app = express();  const mustache = require("mustache-express");  const mongoose = require('mongoose');  const config = require('./config');  const url = config.DatabaseUrl;  const serverPort = config.ServerPort;  const connectionOptions = { useNewUrlParser: true, useUnifiedTopology: true }  const passport = require('passport');  const LocalStrategy = require('passport-local').Strategy;  const cookieParser = require('cookie-parser');  const session = require('express-session');  const bodyParser = require('body-parser');  const User = require('./models/user');  const authRouter = require('./routes/auth')  app.use(cookieParser());  app.use(session({  secret: "Some\_secret^string",  resave: false,  saveUninitialized: true  }));  app.use(passport.initialize());  app.use(passport.session());  passport.serializeUser(function (user, done) {  done(null, user.\_id);  });  passport.deserializeUser(function (id, done) {  User.getById(id)  .then((user) => done(null, user))  .catch((err) => done(err, null))  });  passport.use(new LocalStrategy((username, password, done) => {  console.log('localstrayegy');  User.getAll()  .then(users => {  let us;  for (let user of users) {  if (user.login == username && user.password == authRouter.sha512(password, config.serverSalt).passwordHash) {  us = user;  console.log(us)  }  }  if (us != undefined)  return done(null, us)  else {  return done("no such user", null)  }  })  .catch((err) => done(err, null))  }));  const BasicStrategy = require('passport-http').BasicStrategy;  passport.use(new BasicStrategy(  function (username, password, done) {  User.findOne(username)  .then((user) => {  if (!user)  return done(null, false);  if (user.password!=authRouter.sha512(password).passwordHash) {  return done(null, false);  }  return done(null, user);  })  })  )  app.use(bodyParser.urlencoded({ extended: true }));  app.use(bodyParser.json());  app.use(busboyBodyParser({ limit: '5mb' }));  mongoose.connect(url, connectionOptions)  .then(() => console.log('Database connected'))  .then(() => app.listen(serverPort))  .then(() => console.log('Running at Port 3000'))  .catch(err => console.log(err.toString()))  app.use(busboyBodyParser());  app.use('/static', express.static('public'));  app.use(express.static(path.join(\_\_dirname, './public')));  const viewsDir = path.join(\_\_dirname, 'views');  app.engine("mst", mustache(path.join(viewsDir, "partials")));  app.set('views', viewsDir);  app.set('view engine', 'mst');  const usersRouter = require('./routes/users')  app.use('/', usersRouter)  const albumsRouter = require('./routes/albums')  app.use('/', albumsRouter);  const photosRouter = require('./routes/photos')  app.use('/', photosRouter)  const userapiRouter = require('./routes/api')  app.use('/', userapiRouter)  const devRouter = require('./routes/developer')  app.use('/', devRouter)  app.use('/', authRouter)  app.get('/data/fs/:path', function (req, res) {  res.sendFile(path.join(\_\_dirname, './data/fs/' + req.params.path));  });  app.get('/', function (req, res, next) {  res.render('index', {  user: req.user  })  })  app.use('/about', function (req, res, next) {  res.render('about', { user: req.user })  })  app.use(function(req,res){  res.status(404).render('notFound');  }); |

|  |
| --- |
| **javascripts/albums.js** |
| console.log("Loading Browser JS app.js");  Promise.all([  fetch("/templates/albums.mst").then(x => x.text()),  fetch("/api/v1/albums").then(x => x.json()),  ])  .then(([templateStr, itemsData]) => {  console.log('itemsData', itemsData);  const dataObject = {items: itemsData};  const renderedHtmlStr = Mustache.render(templateStr, dataObject);  return renderedHtmlStr;  })  .then(htmlStr => {  const appEl = document.getElementById('parentElement');  appEl.innerHTML = htmlStr;  })  .catch(err => console.error(err));  function search(){  document.getElementById('page').innerText=1  let tag=document.getElementById('search').value  console.log(tag)  Promise.all([  fetch("/templates/albums.mst").then(x => x.text()),  fetch("/api/v1/albums?searching="+tag).then(x => x.json()),  ])  .then(([templateStr, itemsData]) => {  if (itemsData.length==0){  document.getElementById('heading').innerText="no results"  }  else{  if(tag!='')  document.getElementById('heading').innerText="results for '"+tag+"'"  else  document.getElementById('heading').innerText="current album list"  }  console.log('itemsData', itemsData);  const dataObject = {items: itemsData};  const renderedHtmlStr = Mustache.render(templateStr, dataObject);  return renderedHtmlStr;  })  .then(htmlStr => {  const appEl = document.getElementById('parentElement');  appEl.innerHTML = htmlStr;  })  .catch(err => console.error(err));  }  function next(){  let currentpage=document.getElementById('page').innerHTML  let nextpage=parseInt(currentpage)+1;  console.log(nextpage)  let tag=document.getElementById('search').value  console.log(tag)  Promise.all([  fetch("/templates/albums.mst").then(x => x.text()),  fetch("/api/v1/albums?searching="+tag+"&page="+nextpage).then(x => x.json()),  fetch("/api/v1/albums?searching="+tag+"&page="+(parseInt(nextpage)+1)).then(x => x.json()),  ])  .then(([templateStr, itemsData,nextData]) => {  document.getElementById('page').innerText=nextpage  if(nextpage>1)  document.getElementById('prevButton').removeAttribute("disabled")  else  document.getElementById('prevButton').setAttribute("disabled","")  document.getElementById('nextButton').removeAttribute("disabled")  console.log(nextData)  if(nextData){  if(nextData.error=="no such page")  document.getElementById('nextButton').setAttribute("disabled","")  }  if (itemsData.length==0){  document.getElementById('heading').innerText="no results"  }  else{  if(tag!='')  document.getElementById('heading').innerText="results for '"+tag+"'"  else  document.getElementById('heading').innerText="current album list"  }  console.log('itemsData', itemsData);  const dataObject = {items: itemsData};  const renderedHtmlStr = Mustache.render(templateStr, dataObject);  return renderedHtmlStr;  })  .then(htmlStr => {  const appEl = document.getElementById('parentElement');  appEl.innerHTML = htmlStr;  })  .catch(err => console.error(err));  }  function prev(){  let currentpage=document.getElementById('page').innerHTML  let nextpage=parseInt(currentpage)-1;  console.log(nextpage)  let tag=document.getElementById('search').value  console.log(tag)  Promise.all([  fetch("/templates/albums.mst").then(x => x.text()),  fetch("/api/v1/albums?searching="+tag+"&page="+nextpage).then(x => x.json()),  fetch("/api/v1/albums?searching="+tag+"&page="+(parseInt(nextpage)+1)).then(x => x.json()),  ])  .then(([templateStr, itemsData,nextData]) => {  document.getElementById('page').innerText=nextpage  if(nextpage>1)  document.getElementById('prevButton').removeAttribute("disabled")  else  document.getElementById('prevButton').setAttribute("disabled","")  document.getElementById('nextButton').removeAttribute("disabled")  console.log(nextData)    if(nextData.error=="no such page")  document.getElementById('nextButton').setAttribute("disabled","")  if (itemsData.length==0){  document.getElementById('heading').innerText="no results"  }  else{  if(tag!='')  document.getElementById('heading').innerText="results for '"+tag+"'"  else  document.getElementById('heading').innerText="current album list"  }  console.log('itemsData', itemsData);  const dataObject = {items: itemsData};  const renderedHtmlStr = Mustache.render(templateStr, dataObject);  return renderedHtmlStr;  })  .then(htmlStr => {  const appEl = document.getElementById('parentElement');  appEl.innerHTML = htmlStr;  })  .catch(err => console.error(err));  } |

|  |
| --- |
| **javascripts/photos.js** |
| console.log("Loading Browser JS app.js");  Promise.all([  fetch("/templates/photos.mst").then(x => x.text()),  fetch("/api/v1/photos").then(x => x.json()),  ])  .then(([templateStr, itemsData]) => {  console.log('itemsData', itemsData);  const dataObject = {items: itemsData};  const renderedHtmlStr = Mustache.render(templateStr, dataObject);  return renderedHtmlStr;  })  .then(htmlStr => {  const appEl = document.getElementById('parentElement');  appEl.innerHTML = htmlStr;  })  .catch(err => console.error(err));  function search(){  document.getElementById('page').innerText=1  //let heading=document.getElementById('heading').innerText  let tag=document.getElementById('search').value  console.log(tag)  Promise.all([  fetch("/templates/photos.mst").then(x => x.text()),  fetch("/api/v1/photos?searching="+tag).then(x => x.json()),  ])  .then(([templateStr, itemsData]) => {  if (itemsData.length==0){  document.getElementById('heading').innerText="no results"  }  else{  if(tag!='')  document.getElementById('heading').innerText="results for '"+tag+"'"  else  document.getElementById('heading').innerText="current photo list"  }  console.log('itemsData', itemsData);  const dataObject = {items: itemsData};  const renderedHtmlStr = Mustache.render(templateStr, dataObject);  return renderedHtmlStr;  })  .then(htmlStr => {  const appEl = document.getElementById('parentElement');  appEl.innerHTML = htmlStr;  })  .catch(err => console.error(err));  }  function next(){  let currentpage=document.getElementById('page').innerHTML  let nextpage=parseInt(currentpage)+1;  console.log(nextpage)  let tag=document.getElementById('search').value  console.log(tag)  Promise.all([  fetch("/templates/photos.mst").then(x => x.text()),  fetch("/api/v1/photos?searching="+tag+"&page="+nextpage).then(x => x.json()),  fetch("/api/v1/photos?searching="+tag+"&page="+(parseInt(nextpage)+1)).then(x => x.json()),  ])  .then(([templateStr, itemsData,nextData]) => {  document.getElementById('page').innerText=nextpage  if(nextpage>1)  document.getElementById('prevButton').removeAttribute("disabled")  else  document.getElementById('prevButton').setAttribute("disabled","")  document.getElementById('nextButton').removeAttribute("disabled")  console.log(nextData)  if(nextData){  if(nextData.error=="no such page")  document.getElementById('nextButton').setAttribute("disabled","")  }  if (itemsData.length==0){  document.getElementById('heading').innerText="no results"  }  else{  if(tag!='')  document.getElementById('heading').innerText="results for '"+tag+"'"  else  document.getElementById('heading').innerText="current photo list"  }  console.log('itemsData', itemsData);  const dataObject = {items: itemsData};  const renderedHtmlStr = Mustache.render(templateStr, dataObject);  return renderedHtmlStr;  })  .then(htmlStr => {  const appEl = document.getElementById('parentElement');  appEl.innerHTML = htmlStr;  })  .catch(err => console.error(err));  }  function prev(){  let currentpage=document.getElementById('page').innerHTML  let nextpage=parseInt(currentpage)-1;  console.log(nextpage)  let tag=document.getElementById('search').value  console.log(tag)  Promise.all([  fetch("/templates/photos.mst").then(x => x.text()),  fetch("/api/v1/photos?searching="+tag+"&page="+nextpage).then(x => x.json()),  fetch("/api/v1/photos?searching="+tag+"&page="+(parseInt(nextpage)+1)).then(x => x.json()),  ])  .then(([templateStr, itemsData,nextData]) => {  document.getElementById('page').innerText=nextpage  if(nextpage>1)  document.getElementById('prevButton').removeAttribute("disabled")  else  document.getElementById('prevButton').setAttribute("disabled","")  document.getElementById('nextButton').removeAttribute("disabled")  console.log(nextData)    if(nextData.error=="no such page")  document.getElementById('nextButton').setAttribute("disabled","")  if (itemsData.length==0){  document.getElementById('heading').innerText="no results"  }  else{  if(tag!='')  document.getElementById('heading').innerText="results for '"+tag+"'"  else  document.getElementById('heading').innerText="current photo list"  }  console.log('itemsData', itemsData);  const dataObject = {items: itemsData};  const renderedHtmlStr = Mustache.render(templateStr, dataObject);  return renderedHtmlStr;  })  .then(htmlStr => {  const appEl = document.getElementById('parentElement');  appEl.innerHTML = htmlStr;  })  .catch(err => console.error(err));  } |

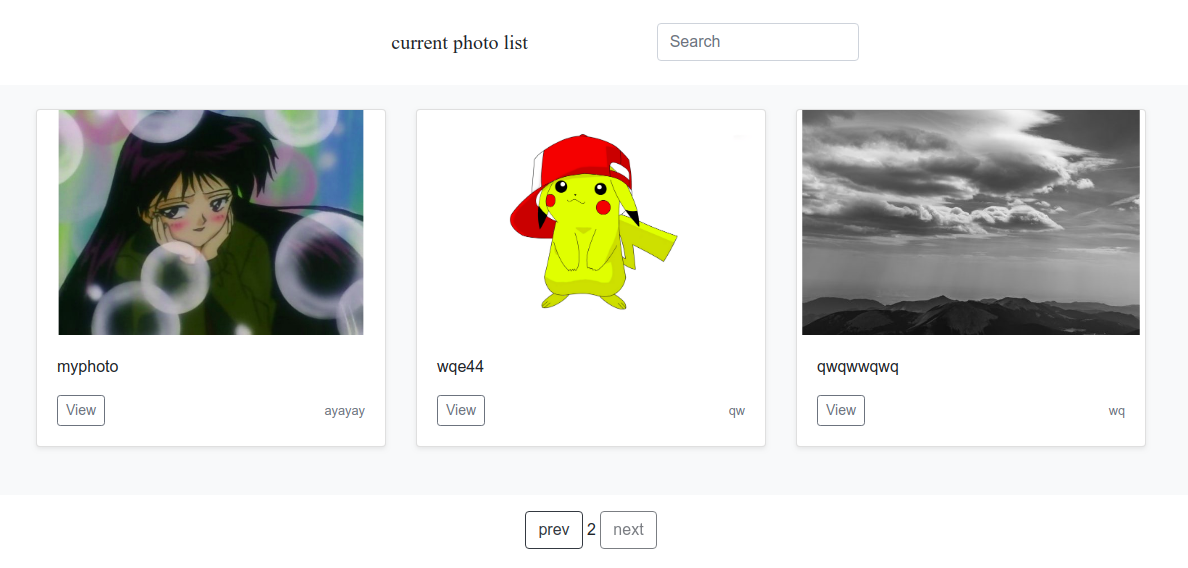
|  |
| --- |
| **templates/albums.mst** |
| {{#items}}  <div *class*="col-md-4">  <div *class*="card mb-4 shadow-sm">  <svg *class*="bd-placeholder-img card-img-top" *width*="100%" *height*="225" *xmlns*="http://www.w3.org/2000/svg" *preserveAspectRatio*="xMidYMid slice" *focusable*="false" *role*="img" *aria-label*="Placeholder: Thumbnail">  <image *xlink:href*="{{previewUrl}}" *width*="100%" *height*="100%" *x*="0" *y*="0" />  </svg>  <div *class*="card-body">  <p *class*="card-text">{{name}}</p>  <div *class*="d-flex justify-content-between align-items-center">  <div *class*="btn-group">  <a *href*="/albums/{{\_id}}" *class*="btn btn-sm btn-outline-secondary">View</a>  </div>  <small *class*="text-muted">{{author}}</small>  </div>  </div>  </div>  </div>  {{/items}} |

|  |
| --- |
| **templates/photos.mst** |
| {{#items}}  <div *class*="col-md-4" *style*="display:inline-block">  <div *class*="card mb-4 shadow-sm">  <svg *class*="bd-placeholder-img card-img-top" *width*="100%" *height*="225" *xmlns*="http://www.w3.org/2000/svg" *preserveAspectRatio*="xMidYMid slice" *focusable*="false" *role*="img" *aria-label*="Placeholder: Thumbnail">  <image *xlink:href*="{{url}}" *width*="100%" *height*="100%" *x*="0" *y*="0" />  </svg>  <div *class*="card-body">  <p *class*="card-text">{{name}}</p>  <div *class*="d-flex justify-content-between align-items-center">  <div *class*="btn-group">  <a *href*="/photos/{{\_id}}" *class*="btn btn-sm btn-outline-secondary">View</a>  </div>  <small *class*="text-muted">{{author}}</small>  </div>  </div>  </div>  </div>  {{/items}} |

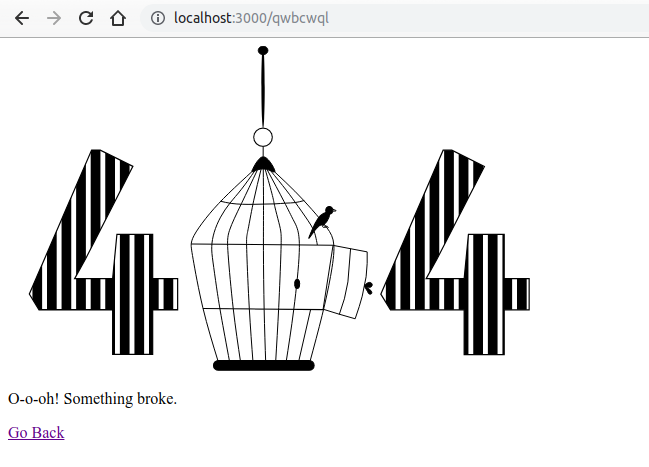
|  |
| --- |
| **views/albums.mst** |
| <!doctype *html*>  <html *lang*="en">  <head>  {{> head }}  </head>  <body>  {{> header }}  <form *class*="form-inline mt-2 mt-md-0" *style*="margin-left:25%">  <label *for*="searh">{{heading}}  <pre> </pre>  </label>  <input *class*="form-control mr-sm-2" *id*="search"*type*="text" *placeholder*="Search" *aria-label*="Search" *name*="searching">  <button *class*="btn btn-outline-success my-2 my-sm-0" *type*="submit">Search</button>  </form>  <br>  <div *class*="album py-4 bg-light">  <div *class*="container">  <div *class*="row">  {{#photos}}  <div *class*="col-md-4">  <div *class*="card mb-4 shadow-sm">  <svg *class*="bd-placeholder-img card-img-top" *width*="100%" *height*="225" *xmlns*="http://www.w3.org/2000/svg" *preserveAspectRatio*="xMidYMid slice" *focusable*="false" *role*="img" *aria-label*="Placeholder: Thumbnail">  <image *xlink:href*="{{url}}" *width*="100%" *height*="100%" *x*="0" *y*="0" />  </svg>  <div *class*="card-body">  <p *class*="card-text">{{name}}</p>  <div *class*="d-flex justify-content-between align-items-center">  <div *class*="btn-group">  <a *href*="/photos/{{\_id}}" *class*="btn btn-sm btn-outline-secondary">View</a>  </div>  <small *class*="text-muted">{{author}}</small>  </div>  </div>  </div>  </div>  {{/photos}}  </div>  </div>  </div>  <p>  <div *style*="text-align:center">  <a *href*="?page={{prevPage}}{{#tag}}&searching={{tag}}{{/tag}}" *class*="btn btn-outline-dark">prev</a>  {{page}}  <a *href*="?page={{nextPage}}{{#tag}}&searching={{tag}}{{/tag}}" *class*="btn btn-outline-dark">next</a>  <div>  {{>footer}}  </body>  </html> |

**Приклади результатів**

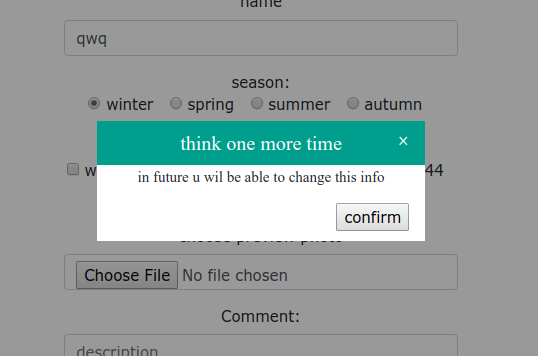
сторінка з пагінацією та динамічним пошуком:



404 page:

****

модальні вікна:

****

**Висновки**

На даній лабораторній роботі я ознайомився з браузерним javascript. Навчився створювати динамічні сторінки на основі власних api відповідей. Також за допомогою фреймворку bootstrap 4 було стилізовано всі мторінки сайту.