

**МIНIСТЕРСТВО ОСВIТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ**

**ІМЕНІ ІГОРЯ СІКОРСЬКОГО”**

**Факультет прикладної математики**

**Кафедра програмного забезпечення комп’ютерних систем**

**Лабораторна робота №7**

**з дисципліни “ Основи web-програмування ”**

**тема “РЕЄСТРАЦІЯ ТА АВТОРИЗАЦІЯ КОРИСТУВАЧІВ”**

|  |  |  |
| --- | --- | --- |
| **Виконав**  **студент II курсу**  **групи КП-83**  **Мортіков Владислав Євгенович** |  | **Перевірив**  **“\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.**  **викладач**  **Гадиняк Руслан Анатолійович**  **(прізвище, ім’я, по батькові)** |

**Київ 2019**

#### **Мета**

Реалізувати персоналізацію вмісту сайту за допомогою користувачів і авторизації із використанням сесій.

### Підготовка

Знати:

1. Реєстрація, ідентифікація, аутентифікація та авторизація.
2. HTTP Cookies.

Потрібно:

1. Створити у навчальному репозиторії директорію проекту labs/lab7 і перейти до неї у редакторі.
2. Скопіювати у корінь проект попередньої лабораторної роботи для продовження роботи над ним.
3. Видалити з проекту приховану директорію .git.
4. Встановити пакети: passport, passport-local, cookie-parser, express-session.

### Попереднє завдання

1. Додати шаблони сторінок register.mst та login.mst.
2. Додати пусті обробники для запитів (винести у окремий Router для /auth):

GET /auth/register

POST /auth/register

GET /auth/login

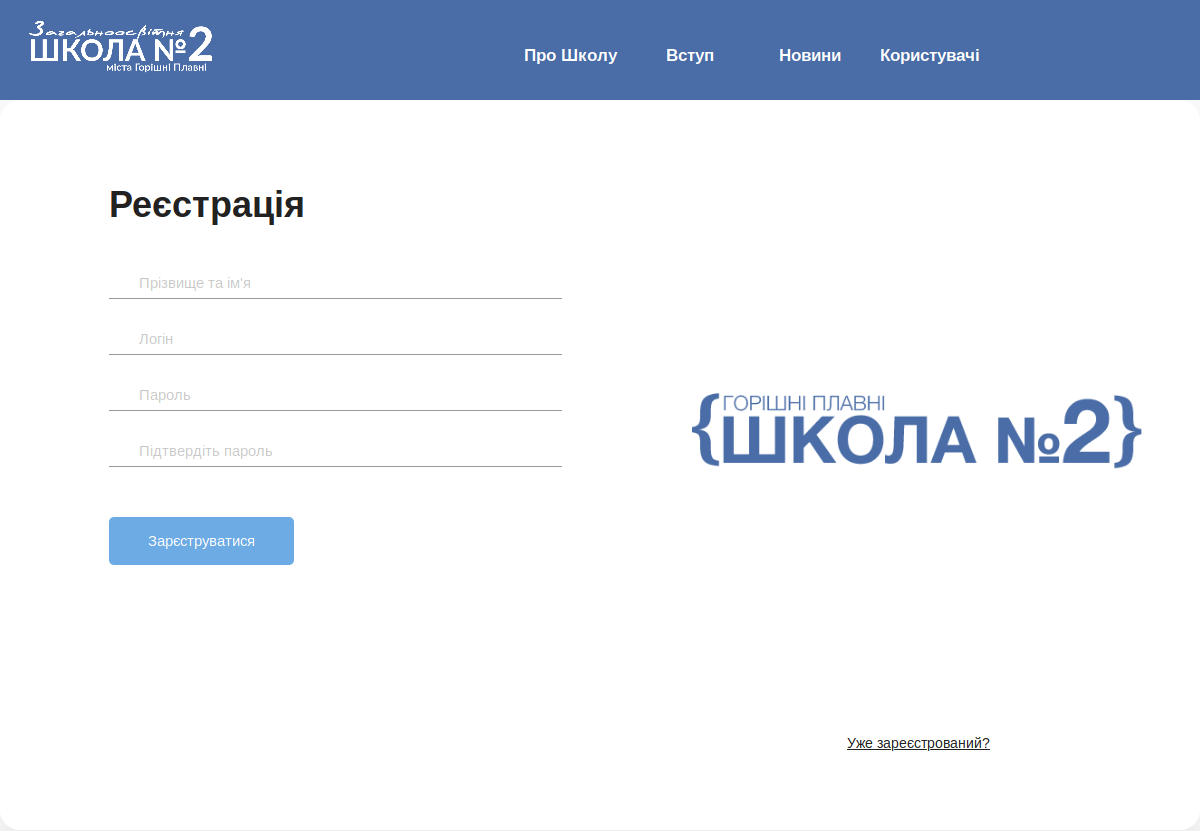
POST /auth/login

POST /auth/logout

1. Налаштувати рендер сторінок register та login у відповідних обробниках.

### Завдання

1. Додати у БД відмітки про ролі користувачів сайту:
   * Стандартний користувач
   * Адміністратор
2. Реалізувати реєстрацію користувачів на сайті по шляху /auth/register:
   * Реєстрація по унікальному логіну і паролю.
   * За допомогою HTML атрибутів задати мінімальний та максимальний розміри строк вводу та їх обов'зковість.
   * На форму реєстрації розмістити два поля для вводу паролю (ввід прихований) для перевірки співпадіння паролю (поки на сервері). Якщо реєстрація пройшла успішно - перевести користувача на сторінку логіну (/auth/login) у іншому випадку - знову на сторінку реєстрації, але з помилкою у аргументах URL (наприклад, /auth/register?error=Username+already+exists).
   * Перед формою виводити повідомлення про помилку із аргументів URL, якщо така існує.
   * Роль створеного користувача "Стандартний користувач".
3. За допомогою клієнтських сесій і HTTP Cookies реалізувати логін (/auth/login) та логаут (/auth/logout) користувачів на сайті. У шапці всіх сторінок відображати ім’я авторизованого користувача або повідомлення про те, що користувач не увійшов на сайт.
4. Дозволити гостям (неавторизованим користувачам) доступ лише до таких сторінок:
   * головної сторінки (/)
   * сторінки реєстрації (/auth/register)
   * входу (/auth/login)
   * та допоміжних сторінок (/about та ін.).
5. Обмежити доступ до сторінки перегляду списку зареєстрованих користувачів (/users).  
   Лише користувачі-адміністратори:
   * бачать посилання на цю сторінку у шапці сайту
   * можуть переглядати цю сторінку
   * можуть змінювати роль користувача на його сторінці (/users/:id)



**Тексти коду програм**

|  |
| --- |
| **app.js** |
| const express = require('express');  const path = require('path');  const User = require("./models/user.js");  const Items = require("./models/item.js");  const mustache = require("mustache-express");  const app = express();  const viewsDir = path.join(\_\_dirname, 'views');  app.engine("mst", mustache(path.join(viewsDir, "partials")));  app.set('views', viewsDir);  app.set('view engine', 'mst');  // will open public/ directory files for http requests  app.use(express.static('public'));  app.get('/', function (req, res) {  res.render('index');  });  app.get('/users', function (req, res) {  let userList = {};  userList.users = User.getAll();  res.render('users', userList);  });  app.get('/users/:id', function (req, res) {  let id = parseInt(req.params.id);  const user = User.getById(id);  if (typeof (user) === "undefined") {  res.sendStatus(404);  }  else {  res.render('user', user);  }  });  app.get('/items', function (req, res) {  let itemsList = {};  itemsList.items = Items.getAll();  res.render('items', itemsList);  });  app.get('/items/:id', function (req, res) {  let id = parseInt(req.params.id);  const item = Items.getById(id);  if (typeof (item) === "undefined") {  res.sendStatus(404);  }  else {  res.render('item', item);  }  });  app.get('/philosophy', function (req, res) {  res.render('philosophy');  });  app.get('/handbook', function (req, res) {  res.render('handbook');  });  app.get('/enrolment', function (req, res) {  res.render('enrolment');  });  app.get('/api/users', function (req, res) {  const users = User.getAll();  res.send(users);  });  app.get("/api/user/:id", function (req, res) {  const self = User.getById(parseInt(req.params.id));  if (typeof (self) === "undefined") {  res.sendStatus(404);  }  else {  res.send(self);  }  });  app.listen(3000, function () {  console.log('Example app listening on port 3000');  }); |

**Висновки**

Виконавши дану лабораторну роботу ми продовжили навчання роботі із базами даних. Тепер ми вміємо реєструвати та авторизувати користувачів, ділити їхні ролі.