

**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 №4**

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

**тема “ОБРОБКА ДАНИХ ВЕБ-ФОРМ”**

|  |  |  |
| --- | --- | --- |
| **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 роботи**

Навчитись створювати HTML-форми для взаємодії користувача із веб-сайтом та обробляти на сервері внесені користувачем дані.  
Вивчити основні принципи асинхронного програмування за допомогою callback-функцій.

**Постановка завдання**

### **Підготовка**

Знати:

1. HTML форми, їх призначення, принципи роботи та обмеження.
2. Типи елементів вводу у формах для різних типів даних.
3. Елемент форми для завантаження файлів.
4. Асинхронне виконання JS коду за допомогою callback-функцій.
5. Повернення даних та помилок із callback-функцій.

Потрібно:

1. Створити у навчальному репозиторії директорію проекту: labs/lab4 і перейти до неї у редакторі.
2. Скопіювати у корінь проект лабораторної роботи [ЛР№3](https://progbase.herokuapp.com/modules/webprogbase/labs/lab3) для продовження роботи над ним.
3. Ознайомитись із [асинхронним програмуванням за допомогою callback-функцій](http://progbase.herokuapp.com/posts/2/asinkhronnist-u-javascript).

### Попереднє завдання

1. Додати до скопійованого проекту веб-сайту нову веб-сторінку /{entities}/new із базовим HTML5 шаблоном.
2. Додати на сторінку /{entities} пагінацію списку основних сутностей (див. Додатки).
3. Переписати статичні синхронні функції моделей на [асинхронні аналоги з callback-функціями](https://docs.google.com/document/d/17d93c-kxsFSJgKXFP1tK210ijlQ_vg9dZ60VNlfqB18/edit" \l "heading=h.a1uny8dlwyp), які використовують стандартні асинхронні функції для роботи з файловою системою.

### Завдання

1. Пошук сутностей по назві
   1. Модифікувати веб-сторінку /{entities} з сутностями так, щоби можна було шукати сутності по частковому співпадіння строки назви за допомогою форми із GET запитом.
   2. Сторінка має містити спеціальні компоненти для інформування користувача про те, для якого пошукового запиту відображаються результати.
   3. Якщо в результаті пошуку не було знайдено жодних об'єктів, додавати на сторінку спеціальне повідомлення про це.
2. Створення нових сутностей
   1. Додати на сторінку /{entities}/new HTML форму для створення нової сутності (за варіантом) за допомогою POST запиту.
   2. Дана форма обов’язково повинна мати одне поле для завантаження на сервер файлу (зображення чи ін.), що пов'язаний із типом об'єкта.
   3. Завантажені файли розміщувати у директорії data/fs/. Додати у веб-сервер обробник GET запитів на отримання файлів із data/fs/. У поля сутностей зберігати URL для отримання цих файлів через GET запити до веб-сервера.
   4. Після створення нової сутності і присвоєння їй ідентифікатора, перенаправити на сторінку нової сутності (/{entities}/:id).
3. Видалення сутностей
   1. На сторінці перегляду інформації про окрему сутність (/{entities}/:id) додати форму (яка виглядатиме як кнопка, посилання чи зображення) для видалення поточної сутності за допомогою POST запиту.

**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');  }); |

**Приклади результатів**

|  |
| --- |
|  |
|  |
|  |

**Висновки**

Виконавши дану лабораторну роботу ми навчилися працювати із НТМL-формами та сallback-функціями.

Компіляція всього коду відбувалася за допомогою node app.js .

Після цього на локальному сервері доступ можна отримати за посиланням [http://localhost:7000/](http://localhost:3000/).

Асинхронність – це процес обробки введення/виводу, що дозволяє продовжити обробку інших завдань, не чекаючи завершення попереднього завдання. Сallback-функція у JS чекає виконання потрібних функцій і лише потім починають виконуватися. Середовище виконання JS має подійно-орієнтовану монопоточну характеристику, що й спричиняє асинхронність.

Які стандартні асинхронні функції для роботи із файловою системою ви використовували при виконанні лабораторної роботи? В чому особливість їх використання у порівнянні із синхронними функціями?