

**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 №5**

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

**тема “ІНТЕГРАЦІЯ ІЗ БАЗАМИ ДАНИХ”**

|  |  |  |
| --- | --- | --- |
| **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**

Навчитися працювати із базами даних у веб-сервері.  
Ознайомитись із Promise-об'єктами для їх використання при написанні асинхронного коду.

### Підготовка

Знати:

1. [Основи баз даних](https://docs.google.com/document/d/1eUSkjpnFdsBIcmFfr3VuYU5NYDZWcWqJcokqozLJNHk/edit?usp=sharing)
2. [JavaScript Promises](https://github.com/PublicHadyniak/webprogbase/wiki/JavaScript-Promises)

Потрібно:

1. Обрати NoSQL (MongoDB - рекомендовано) або реляційну (PostgreSQL) базу даних.
2. Встановити сервер обраної бази даних та GUI клієнт ([Robo 3T](https://robomongo.org/) або [PgAdmin](https://www.pgadmin.org/download/)).
3. Встановити локальний пакет: fs.promised (чи аналог) та пакети, відповідно до обраної БД (mongo, mongoose)
4. Змоделювати за допомогою GUI клієнта бази даних об'єкти сутностей із попередніх лабораторних робіт та додати нову композиційну сутність ({composite}) відповідно до варіанту (див. Додаток "Структура бази даних").  
   Називати таблиці\колекції сутностей у множині англійською мовою.  
   Всі сутності мають містити унікальні ключі-ідентифікатори.  
   Сутності користувачів мають мати зв'язок із основною або композиційною суністю (залежить від сутностей).

### Попереднє завдання

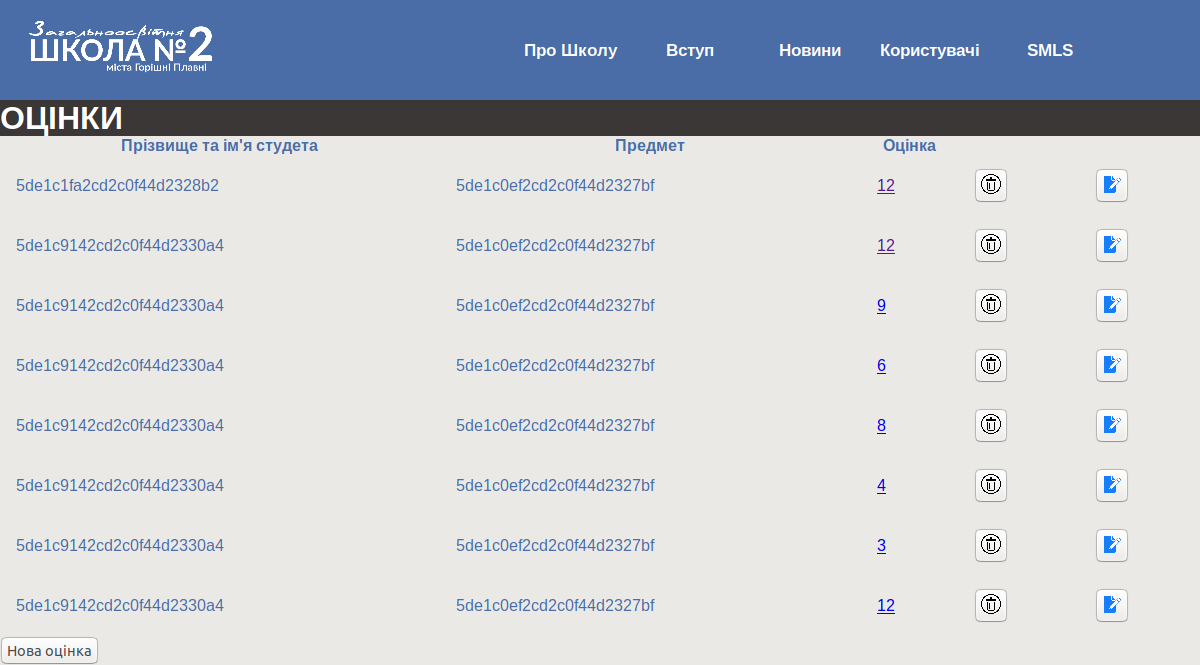
1. Переписати статичні асинхронні функції моделей на [аналоги з використанням Promise-об'єктів](https://docs.google.com/document/d/17d93c-kxsFSJgKXFP1tK210ijlQ_vg9dZ60VNlfqB18/edit" \l "heading=h.xzozu0if7re9), які використовують асинхронні функції пакета для роботи з файловою системою.

### Завдання

1. Переписати реалізацію моделей для взаємодії із обраною СКБД замість JSON-файлів.
2. Композиційна сутність
   1. Створити нову модель {Composite} у модулі models/{composite}.js для композитних сутностей (за варіантом, замість {Composite} використовувати назву сутності англійською мовою). Реалізувати асинхронні CRUD операції (Promised) для взаємодії із базою даних.
   2. Додати відповідні веб-сторінки, що дозволяють керувати композиційними сутностями.
3. Видалити з директорії /data всі JSON файли з даними, які тепер будуть розміщені у БД.  
   Директорію /data/fs залишити для зберігання файлів.

**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');  }); |

****

СУТНОСТІ МОЖНА РЕДАГУВАТИ

І ВИДАЛЯТИ, СТВОРЮВАТИ, ЧИТАТИ.

ВІДБУВАЄТЬСЯ ВЗАЄМОДІЯ ІЗ БД

**Висновки**

Виконавши дану лабораторну роботу ми продовжили навчання роботі із базами даних.

Компіляція всього коду відбувалася за допомогою node app.js .

Для того, щоб підняти базу даних, у терміналі варто ввести команду $ mongod --dbpath "/home/vladislav/Downloads/mongodb-linux-x86\_64-ubuntu1804-4.2.1/bin/data/db". Веб-сервер працює на хості 4000.