

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 №** **3**

з дисципліни “ Основи веб-програмування ”

тема “**Інтеграція із базами даних**”

|  |  |  |
| --- | --- | --- |
| Виконав(ла)  студент(ка) II курсу  групи КП-83  Симонюк Володимир Павлович  (*прізвище, ім’я, по батькові*)  варіант № 21 |  | Перевірив  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладач  Гадиняк Руслан Анатолійович  (*прізвище, ім’я, по батькові*) |

Київ 2018

**Мета роботи**

Навчитися працювати із базами даних у веб-сервері.

Ознайомитись із Promise-об'єктами для їх використання при написанні асинхронного коду.

**Постановка завдання**

1. Переписати реалізацію моделей для взаємодії із обраною СКБД замість JSON-файлів.
2. Композиційна сутність
   1. Створити нову модель **Position** у модулі models/Positions.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**

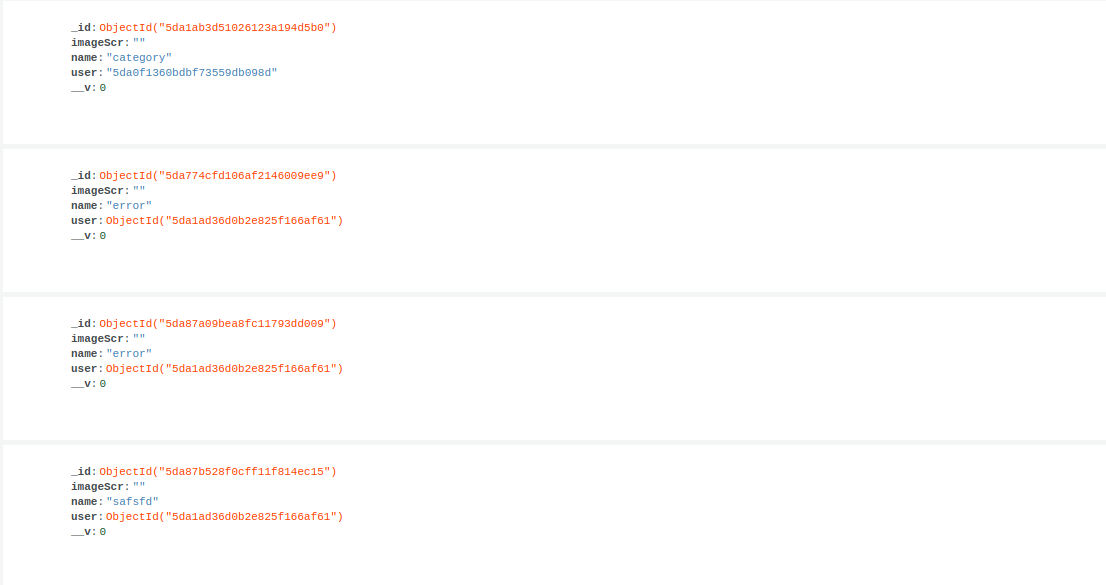
|  |
| --- |
| **Підключення бази даних** |
| mongoose.connect(mongoLink)  .then(() => console.log('Mongo connected.'))  .catch(error => console.log(error)) |

|  |
| --- |
| **Приклади моделей створених за допомогою mongoose схем** |
| const mongoose = require('mongoose')  const Schema = mongoose.Schema  const userSchema = new Schema( {  email: {  type: String,  required: true,  unique:true  },  password: {  type: String,  required: true  },  role: {  type: String,  default: 'customer'  },  ordersList: [  {  date: {  type: Date  },  waiter: {  ref: 'users',  type: Schema.Types.ObjectId  },  customer: {  ref: 'users',  type: Schema.Types.ObjectId  },  table: {  ref: 'tables',  type: Schema.Types.ObjectId  }  }  ]  } )    module.exports = mongoose.model('users', userSchema)  const mongoose = require('mongoose')  const Schema = mongoose.Schema  const positionSchema = new Schema( {  name: {  type: String,  required: true  },  cost: {  type: Number,  required: true  },  category: {  ref: 'categories',  type: Schema.Types.ObjectId  },  user: {  ref: 'users',  type: Schema.Types.ObjectId  },  } )    module.exports = mongoose.model('positions', positionSchema) |

|  |
| --- |
| **Взаємодія з віддаленю базою даних за допомогою методів mongoose** |
| module.exports.getByCategoryId = async function(req, res) {  try {  const positions = await Position.find({  category: req.params.categoryId,  name: {$regex : `.\*${req.query.filter}.\*`}  })  res.status(200).json(positions)  } catch (error) {  res.status(500).json({  success: false,  message: error.message ? error.message : error  })  }  }  module.exports.create = async function(req, res) {  try {  const user = await User.findOne({email: "empty"})  const position = new Position({  name: req.body.name,  cost: req.body.cost,  category: req.body.category,  user: user.id  })  await position.save()  res.status(201).json(position)  } catch (error) {  res.status(500).json({  success: false,  message: error.message ? error.message : error  })  }  } |

**Приклади результатів**

**Приклади моделей категорій**

****

**Приклади моделей позицій**

****

**Висновки**

Виконавши дану лабораторну роботу я навчився створювати моделі за допомогою mongoose схем, підключати до проетку віддалену базу даних та використовувати різноманітні mongoose методи для зручної взаємодії з базою даних.