Міністерство освіти і науки України
Вінницький національний технічний університет
Вінницький навчально-науковий інститут економіки ТНЕУ
ВП НУБіП України «Бережанський агротехнічний інститут»
Кіровоградська льотна академія Національного авіаційного університету
Мозирський педагогічний університет ім. І. П. Шамякіна (Республіка Білорусь)
Одеська військова академія

ІННОВАЦІЙНІ ТЕХНОЛОГІЇ В ПРОЦЕСІ ПІДГОТОВКИ ФАХІВЦІВ

Матеріали IV Міжнародної науково-практичної інтернет-конференції

28-29 березня 2019 року

Збірник наукових праць

Вінниця
ВНТУ
2019

2. VR в медицине [Електронний ресурс]. Режим доступу: https://blog.mednote.life/articles/vr-v-medicine.

Воловик Богдан Петрович, студент групи 2ПІ-18м, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, Вінниця, b.volovyk@gmail.com

Науковий керівник*:* ***Кобилянська Ірина Миколаївна***, кандидат педагогічних наук, доцент кафедри безпеки життєдіяльності та педагогіки безпеки, Вінницький національний технічний університет, Вінниця, akobilanskiy@gmail.com.

Bohdan Volovyk, student of 2PI-18m group, Faculty of Information Technology and Computer Engineering, Vinnytsia National Technical University, Vinnytsia, b.volovyk@gmail.com

Scientific supervisor: ***Iryna Kobylyanska***, Candidate of Pedagogical Sciences, Associate Professor of the Department of Health and Safety Studies, Vinnytsia National Technical University, Vinnytsia, aKobilanskiy@gmail.com.

УДК 37.018.43:004

О. Д. Азаров
О. І. Черняк
В. В. Залізецький

Використання інтерактивної документації в процесі підготовки фахівців

Вінницький національний технічний університет

Анотація. *В статті розглянуто особливості фільтрації запитів по тегам, або по іншим ознакам для здійснення поетапного навчання чи інших цілей, наприклад, обмеження доступу особам, які не мають дозволу на перегляд та виконання окремих запитів.*

Ключові слова: Swagger; інтерактивна документація; REST; API.

The using of interactive documentation in a training process

Abstract: In the article considers the features of filtration of queries by tags, or other features for step-by-step learning or other purposes, such as limiting access to individuals who do not have permission to view and perform individual queries.

Keywords: Swagger; interactive documentation; REST; API.

Сучасні тенденції в розробці програмного забезпечення диктують необхідність переходу від статичної документації до більш ефективної інтерактивної. Є різні варіанти реалізації такого підходу, наприклад Swagger, що являє собою набір скриптів, які генерують документацію для Web-додатків з REST API [1]. Swagger активно використовується при розробці системи, що описана в [2,3] та інших публікаціях авторів. Система досить складна, має розгалужену хмарну інфраструктуру, складається з різних модулів та сервісів, тому є сенс здійснювати поетапне навчання основам роботи з нею. Для цього можна скористатись напрацюваннями, описаними в [4-7], разом з можливостями, що надає Swagger.

Відомо, що можливо налаштувати генерацію документації для кожного сервісу окремо, а також на шлюзі, що проксує на всі інші сервіси і має інформацію про всі кінцеві точки інших сервісів. Авторами пропонується рішення, що дозволяє фільтрувати запити по тегам, або по іншим ознакам. Це потрібно для здійснення поетапного навчання або інших цілей, наприклад, обмеження доступу особам, що не мають дозволу на перегляд та виконання окремих запитів.

Для реалізації даного рішення створено клас API шаблону, що містить опис, які запити потрібно відобразити, а які приховати:

```
case class ApiTemplatePath(include: Option[Seq[String]] = None, exclude: Option[Seq[String]] = None)
object ApiTemplatePath {
 implicit val format: OFormat[ApiTemplatePath] = Json.format[ApiTemplatePath]
}

case class ApiTemplate(name: String, paths: ApiTemplatePath)

object ApiTemplate {
 implicit val format: OFormat[ApiTemplate] = (
  (__ \ "_id").format[String] and
   (__ \ "paths").format[ApiTemplatePath]
  )(ApiTemplate.apply, unlift(ApiTemplate.unapply))

 def simple(name: String = "api", paths: ApiTemplatePath = ApiTemplatePath(Some(Seq(".*")))) : ApiTemplate = ApiTemplate(name,paths)
}
```

У фрагменті коду, що подано вище, описано immutable case клас ApiTemplate, а також об'єкт компаньйон, який має доступ до закритих членів класу і фактично реалізує статичні значення та методи цього класу [8]. Метод simple використовується для створення екземпляру класу без фільтрації. Це потрібно для того, щоб за замовчуванням дозволявся доступ до всіх запитів. Для більшої зручності та гнучкості списки регулярних виразів для відображення або приховання запитів не є обов'язковими, тобто можна вказати лише те, що потрібно відображати, або лише те, що потрібно фільтрувати.

Наступним кроком створюється CRUD сервіс для ApiTemplate, щоб можна було працювати з базою даних:

```
class ApiTemplateService @Inject()(reactiveMongoApi: ReactiveMongoApi)(implicit ctx: ExecutionContext) {
.....
 private def apiCollection = db.map { db =>
  val collection = db.collection[JSONCollection]("swagger")
  collection.find(Json.obj(id -> defaultApi.name)).one[ApiTemplate].map(_.getOrElse(collection.insert(ApiTemplate.simple())))
  collection
 }

 def retrieve(name: String): Future[ApiTemplate] = {
  apiCollection.flatMap(
   _.find(Json.obj(id -> name)).one[ApiTemplate]
    .map(_.getOrElse(throw new AppException(ResponseCode.ENTITY_NOT_FOUND, s"Api specs '$name' not found in db")))
  )
 }

 def save(apiTemplate: ApiTemplate): Future[ApiTemplate] = {
  apiCollection.flatMap(
   _.insert(apiTemplate).map(_ => apiTemplate)
    .recover(MongoErrorHandler.processError)
  )
 }
.....
}
```

Описаний фрагмент коду демонструє приклад реалізації сервісу для роботи з базою даних. Якщо в базі даних немає жодного ApiTemplate, то реалізується варіант без фільтрування. Цей сервіс дає змогу створювати нові шаблони для фільтрації запитів. Далі при використанні різних адрес API-сторінок підвантажується необхідний фільтр.

Для побудови компактного працюючого коду створено функцію вищого порядку, яка перетворює функцію з двома аргументами на функцію з один аргументом, який у свою чергу повертає функцію, що використовує другий аргумент.

```
 def filterSwaggerPaths(api: ApiTemplate)(fullPaths: JsObject): JsObject = {
  JsObject(
   fullPaths.fields.filter(field => api.paths.include.fold(true)(include => include.exists(incl => field._1.matches(incl)))).
    filterNot(filteredField => api.paths.exclude.fold(false)(exclude => exclude.exists(excl => filteredField._1.matches(excl))))
  )
 }
```

Коли стає відомо яку сторінку запитує користувач, з бази даних вибирається шаблон і передається в метод filterSwaggerPaths(api), а далі цей метод передається в інший метод, який збирає інформацію з усіх сервісів та вказується у ньому, що на вхід має надійти метод transformer: (JsObject) => JsObject.

Висновки

Запропоноване рішення дозволяє фільтрувати запити по тегам, або по іншим ознакам для здійснення поетапного навчання чи інших цілей, наприклад, обмеження доступу особам, які не мають дозволу на перегляд та виконання окремих запитів. Для більшої зручності та гнучкості списки регулярних виразів для відображення або приховання запитів не є обов'язковими, тобто можна вказати лише те, що потрібно відображати, або лише те, що потрібно фільтрувати.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. The Best APIs are Built with Swagger Tools. [Електронний ресурс]. Режим доступу: https://swagger.io. Дата звернення: Бер. 8, 2019.

2. О. Д. Азаров, О. І. Черняк, В. В. Залізецький, "Програмне забезпечення для віддаленого виконання арифметичних і логічних операцій в кодах золотої пропорції", на *Всеукраїнській науково-практичній інтернет-конференції Молодь в науці: дослідження, проблеми, перспективи (МН-2018)*, Вінниця, Україна: ВНТУ, 2018. [Електронний ресурс]. Режим доступу: https://conferences.vntu.edu.ua/index.php/mn/mn2018/paper/viewFile/3756/3145.

3. О. Д. Азаров, О. І. Черняк, В. В. Залізецький, "Програмне забезпечення для опрацювання даних дистанційно-розподілених систем та пошуку об'єктів на місцевості", *ІТКІ*, vol 42, № 2, с. 10-15. 2018. [Електронний ресурс]. Режим доступу: https://itce.vntu.edu.ua/index.php/itce/article/view/706.

4. В. В. Залізецький, "Інтерактивний онлайн курс з основ SQL", на *XLIII регіональній науково-технічній конференції професорсько-викладацького складу, співробітників та студентів університету з участю працівників науково-дослідних організацій та інженерно-технічних працівників підприємств м. Вінниці та області*, Вінниця, Україна: ВНТУ, 2014. [Електронний ресурс]. Режим доступу: http://conf.vntu.edu.ua/allvntu/2014/initki/txt/Zalizetskyy.pdf.

5. С. В. Хрущак, В. В. Залізецький, "Інтерактивна система для дистанційного навчання роботі з СУБД", на *Четвертій міжнародної науково-практичної конференції Інформаційні технології та комп'ютерна інженерія*, Вінниця, Україна: ВНТУ, 2014. с. 126-128. [Електронний ресурс]. Режим доступу: https://ir.lib.vntu.edu.ua/handle/123456789/23010.

6. О. Д. Азаров, Л. В. Крупельницький, О. І. Черняк, В. В. Залізецький, "Система дистанційної колективної самопідготовки", *ІТКІ*, vol 36, № 2, с. 15-20. 2016. [Електронний ресурс]. Режим доступу: https://itce.vntu.edu.ua/index.php/itce/article/view/498.

7. О. Д. Азаров, О. І. Черняк, В. О. Михальченко, "Програмне забезпечення для інтерактивного навчального тестування студентів з дисциплін програмування", на *XLVIII науково-технічній конференції підрозділів ВНТУ*, Вінниця, Україна: ВНТУ, 2019, [Електронний ресурс]. Режим доступу: https://conferences.vntu.edu.ua/index.php/all-fitki/all-fitki-2019/paper/view/6662.

8. The Scala Programming Language. [Електронний ресурс]. Режим доступу: https://www.scala-lang.org. Дата звернення: Бер. 8, 2019.

Азаров Олексій Дмитрович, доктор технічних наук, професор кафедри обчислювальної техніки, декан факультету інформаційних технологій і комп'ютерної інженерії, Вінницький національний технічний університет, Вінниця.

Черняк Олександр Іванович, кандидат технічних наук, доцент кафедри обчислювальної техніки, Вінницький національний технічний університет, Вінниця.

Залізецький Василь Володимирович, аспірант кафедри обчислювальної техніки, Вінницький національний технічний університет, Вінниця, zwww@i.ua.

Oleksiy Azarov, Doctor of Engineering Sciences, Professor of the Department of Computer Engineering, Dean of the Faculty of Information Technology and Computer Engineering, Vinnytsia National Technical University, Vinnytsia.

Olexander Chernyak, Candidate of Engineering Science, Associate Professor of the Department of Computer Engineering, Vinnytsia National Technical University, Vinnytsia.

Vasyl Zalizetskyi, post-graduate student of the Department of Computer Engineering, Vinnytsia National Technical University, Vinnytsia, zwww@i.ua.