Факультет компьютерных технологий

Практическое задание №1

«Современные платформы программирования»

Вариант №6

Выполнил: студент гр. 981064 Ефименко Павел Викторович

Проверил: Бакунов А. М.

Минск 2021

Условие:

Реализовать WEB приложение и использование многопоточки, выполнять асинхронное обращение к серверу с обновлением контента без перезагрузки страницы. Работа с Java bean. По возможности использовать ejb.

Код:

package com.mpp.laba\_1;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication  
public class Application {  
  
 public static void main(String[] args) {  
 SpringApplication.run(Application.class, args);  
 }  
  
}

package com.mpp.laba\_1.product;  
  
import lombok.Data;  
import org.hibernate.annotations.CreationTimestamp;  
import org.hibernate.annotations.UpdateTimestamp;  
  
import javax.persistence.Entity;  
import javax.persistence.GeneratedValue;  
import javax.persistence.GenerationType;  
import javax.persistence.Id;  
import java.io.Serializable;  
import java.math.BigDecimal;  
import java.util.Date;  
  
@Entity  
  
@Data  
public class Product implements Serializable {  
 @Id  
 @GeneratedValue(strategy = GenerationType.IDENTITY)  
 private Long id;  
  
 private String name;  
  
 private String description;  
  
 private BigDecimal price;  
  
 @CreationTimestamp  
 private Date createdAt;  
  
 @UpdateTimestamp  
 private Date updatedAt;  
}

package com.mpp.laba\_1.product;  
  
import lombok.RequiredArgsConstructor;  
import lombok.extern.slf4j.Slf4j;  
import org.springframework.http.ResponseEntity;  
import org.springframework.stereotype.Controller;  
import org.springframework.web.bind.annotation.\*;  
  
import javax.validation.Valid;  
import java.util.List;  
import java.util.Optional;  
  
  
@RestController  
@RequestMapping("/api/v1/products")  
@Slf4j  
@RequiredArgsConstructor  
public class ProductAPI {  
 private final ProductService productService;  
  
 @GetMapping  
 public ResponseEntity<List<Product>> findAll() {  
 return ResponseEntity.ok(productService.findAll());  
 }  
  
 @PostMapping  
 public ResponseEntity create(@Valid @RequestBody Product product) {  
 return ResponseEntity.ok(productService.save(product));  
 }  
  
 @GetMapping("/{id}")  
 public ResponseEntity<Product> findById(@PathVariable Long id) {  
 Optional<Product> stock = productService.findById(id);  
 if (!stock.isPresent()) {  
 log.error("Id " + id + " is not existed");  
 ResponseEntity.badRequest().build();  
 }  
  
 return ResponseEntity.ok(stock.get());  
 }  
  
 @PutMapping("/{id}")  
 public ResponseEntity<Product> update(@PathVariable Long id, @Valid @RequestBody Product product) {  
 if (!productService.findById(id).isPresent()) {  
 log.error("Id " + id + " is not existed");  
 ResponseEntity.badRequest().build();  
 }  
  
 return ResponseEntity.ok(productService.save(product));  
 }  
  
 @DeleteMapping("/{id}")  
 public ResponseEntity delete(@PathVariable Long id) {  
 if (!productService.findById(id).isPresent()) {  
 log.error("Id " + id + " is not existed");  
 ResponseEntity.badRequest().build();  
 }  
  
 Runnable task = () -> {  
 productService.deleteById(id);  
 };  
 Thread thread = new Thread(task);  
 thread.start();  
  
 return ResponseEntity.ok().build();  
 }  
}

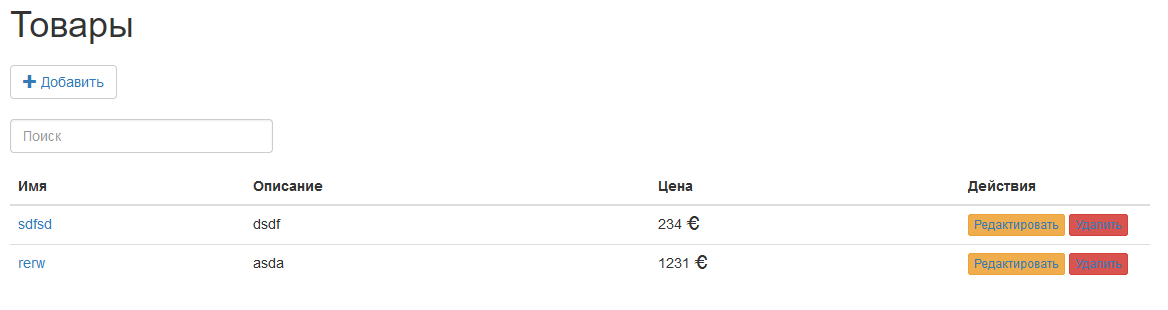
package com.mpp.laba\_1.product;  
  
import org.springframework.stereotype.Controller;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.RequestMapping;  
  
@Controller  
public class ProductController {  
 @GetMapping("")  
 public String list(){  
 return "products";  
 }  
}

package com.mpp.laba\_1.product;  
  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface ProductRespository extends JpaRepository<Product, Long> {  
}

package com.mpp.laba\_1.product;  
  
import lombok.RequiredArgsConstructor;  
import org.springframework.stereotype.Service;  
  
import java.util.List;  
import java.util.Optional;  
  
@Service  
  
@RequiredArgsConstructor  
public class ProductService {  
 private final ProductRespository productRespository;  
  
 public List<Product> findAll() {  
 return productRespository.findAll();  
 }  
  
 public Optional<Product> findById(Long id) {  
 return productRespository.findById(id);  
 }  
  
 public Product save(Product stock) {  
 return productRespository.save(stock);  
 }  
  
 public void deleteById(Long id) {  
 productRespository.deleteById(id);  
 }  
}

package com.mpp.laba\_1.controller;  
  
import org.springframework.boot.web.servlet.error.ErrorController;  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.ResponseBody;  
import org.springframework.web.bind.annotation.RestController;  
  
@RestController  
public class ErrorHandlerController implements ErrorController {  
  
 @RequestMapping("/error")  
 @ResponseBody  
 public String getErrorPath() {  
 return "<center><h1>Something went wrong</h1></center>";  
 }  
}

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">  
 <meta name="description" content="">  
 <meta name="author" content="">  
 <title>Лаба 1</title>  
 <link href="/webjars/bootstrap/3.4.0/css/bootstrap.min.css" rel="stylesheet">  
 <link href="/products.css" rel="stylesheet"/>  
 <script src="/webjars/bootstrap/3.4.0/js/bootstrap.min.js"></script>  
 <script src="/webjars/jquery/3.6.0/jquery.min.js"></script>  
</head>  
<body>  
<div class="container">  
 <h1>Товары</h1>  
 <main id="app">  
 <router-view></router-view>  
 </main>  
</div>  
  
<template id="product">  
 <div>  
 <h2>{{ product.name }}</h2>  
 <b>Описание: </b>  
 <div>{{ product.description }}</div>  
 <b>Цена:</b>  
 <div>{{ product.price }}<span class="glyphicon glyphicon-euro"></span></div>  
 <br/>  
 <span class="glyphicon glyphicon-arrow-left" aria-hidden="true"></span>  
 <a>  
 <router-link to="/">Назад</router-link>  
 </a>  
 </div>  
</template>  
  
<template id="product-list">  
 <div>  
 <div class="actions">  
 <a class="btn btn-default">  
 <router-link :to="{path: '/add-product'}">  
 <span class="glyphicon glyphicon-plus"></span>  
 Добавить  
 </router-link>  
 </a>  
 </div>  
 <div class="filters row">  
 <div class="form-group col-sm-3">  
 <input placeholder="Поиск" v-model="searchKey" class="form-control" id="search-element" requred/>  
 </div>  
 </div>  
 <table class="table">  
 <thead>  
 <tr>  
 <th>Имя</th>  
 <th>Описание</th>  
 <th>Цена</th>  
 <th class="col-sm-2">Действия</th>  
 </tr>  
 </thead>  
 <tbody>  
 <tr v-for="product in filteredProducts">  
 *<!-- tr v-for="product in products" -->  
 <!-- tr v-for="product in products | filterBy searchKey in 'name'" -->* <td>  
 <a>  
 <router-link :to="{name: 'product', params: {product\_id: product.id}}">{{ product.name }}</router-link>  
 </a>  
 </td>  
 <td>{{ product.description }}</td>  
 <td>  
 {{ product.price }}  
 <span class="glyphicon glyphicon-euro" aria-hidden="true"></span>  
 </td>  
 <td>  
 <a class="btn btn-warning btn-xs">  
 <router-link :to="{name: 'product-edit', params: {product\_id: product.id}}">Редактировать</router-link>  
 </a>  
 <a class="btn btn-danger btn-xs">  
 <router-link :to="{name: 'product-delete', params: {product\_id: product.id}}">Удалить</router-link>  
 </a>  
 </td>  
 </tr>  
 </tbody>  
 </table>  
 </div>  
</template>  
  
<template id="add-product">  
 <div>  
 <h2>Добавить новый товар</h2>  
 <form @submit="createProduct">  
 <div class="form-group">  
 <label for="add-name">Имя</label>  
 <input class="form-control" id="add-name" v-model="product.name" required/>  
 </div>  
 <div class="form-group">  
 <label for="add-description">Описание</label>  
 <textarea class="form-control" id="add-description" rows="10" v-model="product.description"></textarea>  
 </div>  
 <div class="form-group">  
 <label for="add-price">Цена, <span class="glyphicon glyphicon-euro"></span></label>  
 <input type="number" class="form-control" id="add-price" v-model="product.price"/>  
 </div>  
 <button type="submit" class="btn btn-primary">Создать</button>  
 <a class="btn btn-default">  
 <router-link to="/">Отмена</router-link>  
 </a>  
 </form>  
 </div>  
</template>  
  
<template id="product-edit">  
 <div>  
 <h2>Изменение продукта</h2>  
 <form @submit="updateProduct">  
 <div class="form-group">  
 <label for="edit-name">Имя</label>  
 <input class="form-control" id="edit-name" v-model="product.name" required/>  
 </div>  
 <div class="form-group">  
 <label for="edit-description">Описание</label>  
 <textarea class="form-control" id="edit-description" rows="3" v-model="product.description"></textarea>  
 </div>  
 <div class="form-group">  
 <label for="edit-price">Цена, <span class="glyphicon glyphicon-euro"></span></label>  
 <input type="number" class="form-control" id="edit-price" v-model="product.price"/>  
 </div>  
 <button type="submit" class="btn btn-primary">Сохранить</button>  
 <a class="btn btn-default">  
 <router-link to="/">Отмена</router-link>  
 </a>  
 </form>  
 </div>  
</template>  
  
<template id="product-delete">  
 <div>  
 <h2>Удалить товар {{ product.name }}</h2>  
 <form @submit="deleteProduct">  
 <p>Действие не может быть отменено.</p>  
 <button type="submit" class="btn btn-danger">Удалить</button>  
 <a class="btn btn-default">  
 <router-link to="/">Отмена</router-link>  
 </a>  
 </form>  
 </div>  
</template>  
  
<script src="https://unpkg.com/vue@2.5.22/dist/vue.js"></script>  
<script src="https://unpkg.com/vue-router@3.0.2/dist/vue-router.js"></script>  
<script src="/webjars/axios/0.21.4/dist/axios.js"></script>  
<script src="/products.js"></script>  
  
</body>  
</html>

Результат кода: ****

Вывод:

Были изучены возможности создания WEB приложения асинхронного и многотопочного выполнения. Были задействованы java bean и ejb.