|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |

**Институт информационных технологий (ИТ)**

**Кафедра инструментального и прикладного программного обеспечения (ИиППО)**

**КУРСОВАЯ РАБОТА**

по дисциплине: Шаблоны программных платформ языка Джава

по профилю: Разработка программных продуктов и проектирование информационных систем

направления профессиональной подготовки: 09.03.04 «Программная инженерия»

Тема: Web-приложение «Магазин шоколадных фигурок»

Студент: Анваржонов Жавохирбек Тулкинжонович

Группа: ИКБО-20-19

Работа представлена к защите 22.05.2021 /Анваржонов Ж. Т. /

Руководитель: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ст. преподаватель Р.Г. Болбаков

Работа допущена к защите\_\_\_\_\_\_\_\_\_\_\_(дата)\_\_\_\_\_\_\_\_\_/Зорина Н. В./

Оценка по итогам защиты: \_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
**«МИРЭА - Российский технологический университет»**

**РТУ МИРЭА**



**Институт информационных технологий (ИТ)**

**Кафедра инструментального и прикладного программного обеспечения (ИиППО)**

**ЗАДАНИЕ**

**на выполнение курсовой работы**

по дисциплине: Шаблоны программных платформ языка Джава

по профилю: Разработка программных продуктов и проектирование информационных систем

Студент: Анваржонов Жавохирбек Тулкинжонович

Группа: ИКБО-20-19

Срок представления к защите: 25.04.2021

Руководитель: Болбаков Роман Геннадьевич

**Тема** Web-приложение «Магазин шоколадных фигурок»

**Исходные данные**: индивидуальное задание на разработку; документация по Spring Framework и JEE , документация по языку Java (версия не ниже 8); инструменты и технологии: JDK (не ниже 8), создание RESTful web-сервисов, Spring ORM , Maven , gitHub, IntelliJIDEA . Нормативный документ: инструкция по организации и проведению курсового проектирования СМКО МИРЭА 7.5.1/04.И.05-18.

**Перечень вопросов, подлежащих разработке, и обязательного графического материала**:

1. Провести анализ предметной области и формирование основных требований к приложению. 2. Обосновать выбор средств ведения разработки. 3. Разработать приложение с использование фреймворка Spring и выбранной технологии и инструментария. 4.Провести тестирование приложения. 5. Оформить пояснительную записку по курсовой работе 6. Провести анализ текста на антиплагиат 7. Создать презентацию по выполненной курсовой работе.

Руководителем произведён инструктаж по технике безопасности, противопожарной технике и правилам внутреннего распорядка.

И. о. зав. кафедрой ИиППО: \_\_\_\_\_\_\_\_\_\_\_ /Р. Г. Болбаков /, «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Задание на КР выдал: \_\_\_\_\_\_\_\_\_\_\_\_\_ /Н. В. Зорина /, «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Задание на КР получил: \_\_\_\_\_\_\_\_\_\_\_/Ж.Т. Анваржонов /, «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

УДК 004.4

Анваржонов Ж.Т.Веб-приложение «Магазин шоколадных фигурок»/ **Курсовая работа** по дисциплине «Шаблоны программных платформ на языке Джава» профиля «Разработка программных продуктов и проектирование информационных систем» направления профессиональной подготовки бакалавриата 09.03.04 «Программная инженерия» (4-ый семестр) / руководитель старший преподаватель Н.В. Зорина / кафедра ИППО Института ИТ МИРЭА – с. 29, табл. 6, ист. 10.

Целью работы является создание веб-магазина шоколадных фигурок.

В рамках работы осуществлены краткий анализ аналогов веб-магазина различных изделий.

Anvarzhonov Zh. T. Web-application "Shop of chocolate figurines" **/Coursework** in the discipline «Java software platform templates» of the profile «Software development and information systems design» courses undergraduate 09.03.04. "Software Engineering" (4th semester) / head associate senior lecturer N.V. Zorina / department IPPO of institute IT RTU MIREA.

The aim of the work to create a web store of chocolate figurines.

As part of the work, a brief analysis of the analogues of the web store of various products was carried out.

М. МИРЭА. Ин-т ИТ. Каф. ИиППО. 2021 г. Иванов Д.М.

**ОГЛАВЛЕНИЕ**

[СПИСОК ИСПОЛЬЗОВАННЫХ СОКРАЩЕНИЙ 5](#_Toc72602802)

[ВВЕДЕНИЕ 6](#_Toc72602803)

[1. СБОР И АНАЛИЗ ТРЕБОВАНИЙ К ПРОГРАММНОМУ ПРОДУКТУ 7](#_Toc72602804)

[1.1. Описание предметной области 7](#_Toc72602805)

[Выводы к разделу 1 9](#_Toc72602806)

[2. РАЗРАБОТКА ИНТЕРНЕТ-МАГАЗИНА 11](#_Toc72602807)

[2.1. Проектирование программного продукта 11](#_Toc72602808)

[MainController: 14](#_Toc72602809)

[Контроллер RegController: 16](#_Toc72602810)

[Контроллер UserController: 16](#_Toc72602811)

[Модель Product: 18](#_Toc72602812)

[Модель: ProductType: 18](#_Toc72602813)

[Модель User: 19](#_Toc72602814)

[Выводы к разделу 2 24](#_Toc72602816)

[3. ТЕСТИРОВАНИЕ ПРОГРАММНОГО ПРОДУКТА 24](#_Toc72602817)

[3.2.Тест кейсы 27](#_Toc72602819)

[Таблица 4.1 – Тест-кейс для тестирования регистрации на сайте 27](#_Toc72602820)

[Таблица 4.2 – Тест-кейс для тестирования формы авторизации пользователей и их выход из системы. 28](#_Toc72602821)

[Таблица 4.3 – Тест-кейс для тестирования добавления, редактирования и удаления продуктов или типов продуктов 30](#_Toc72602822)

[Таблица 4.4 – Тест-кейс для тестирования допуска пользователя в панель администратора 32](#_Toc72602823)

[Выводы к разделу 3 34](#_Toc72602824)

[ЗАКЛЮЧЕНИЕ 35](#_Toc72602825)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 36](#_Toc72602826)

# СПИСОК ИСПОЛЬЗОВАННЫХ СОКРАЩЕНИЙ

SSAO — Screen Space Ambient Occlusion (преграждение окружающего света в экранном пространстве);

ПО — Программное обеспечение;

UML — Unified Modeling Language (унифицированный язык моделирования);

ФИО — фамилия, имя, отчество;

БД — база данных;

СУБД — система правления базами данных.

# ВВЕДЕНИЕ

В настоящее время мир во многом стал зависим от услуг,

предоставляемых в сети Интернет. Большая часть современного общества

пользуется вебсайтами, а темп жизни возрос настолько, что на прочтение

накопленной в течение дня информации, которая может быть не только

интересной, но и важной для отдельного человека, может потребоваться

значительное количество свободного времени. Каждый человек, имея свои

предпочтения и интересы, стремится оставаться в курсе событий, касающихся его увлечений. Для этого создаются различные тематические сайты, с помощью которых происходит распространение большого количества информации через удобное устройство, будь то смартфон, планшетный компьютер, ноутбук или персональный компьютер.

Для того, чтобы веб-ресурс работал корректно, необходимо разработать план тестирования [1].

В рамках данной курсовой работы тестируется и проверяется на соответствие требованиям веб-ресурса шоколадных изделий на Spring Framework [2].

Основная часть курсовой работы состоит из следующих разделов:

1. Анализ тестируемого программного продукта;
2. тестирование требований заказчика программного продукта;
3. определение стадий, этапов и сроков программы и документации;
4. формирование тестового плана;
5. формирование тест кейсов;

# СБОР И АНАЛИЗ ТРЕБОВАНИЙ К ПРОГРАММНОМУ ПРОДУКТУ

## Описание предметной области

Предметной областью для данной курсовой работы является создание веб-ресурса наподобие его аналогов – магазина шоколадных изделий на заказ.

В настоящее время людям трудно удается придумать подарок для своих детей, родственников. Найти подарок не составляет труда, но вот чтобы подарок был по-настоящему оригинальным – на это уходит много времени и сил. А интернет-магазин шоколадных изделий можно запросто помочь таким отчаявшимся. Ведь тот ассортимент, который предлагается может удивить даже самого изощренного искусителя. Здесь можно найти изделия, начиная с туфельки золушки, заканчивая огромным шоколадным камазом для дальнобойщиков. Но вся прелесть на этом не заканчивается: мало того, что это оригинальный подарок, его можно ещё и съесть в кругу семьи – ведь подарки сделаны из шоколада.

У аналогов данного ресурса заметно одно отличие – практически одинаковый дизайн. Оно то и понятно, ведь это интернет- магазин, а значит здесь не должно быть потайных кнопок, замудренного дизайна. Красота – в простоте, особенно для веб-ресурса Шоколадных изделий.

Рассмотрим некоторые аналоги:

На рисунке 1.1 изображён интерфейс интернет-магазина «Confaelshop»

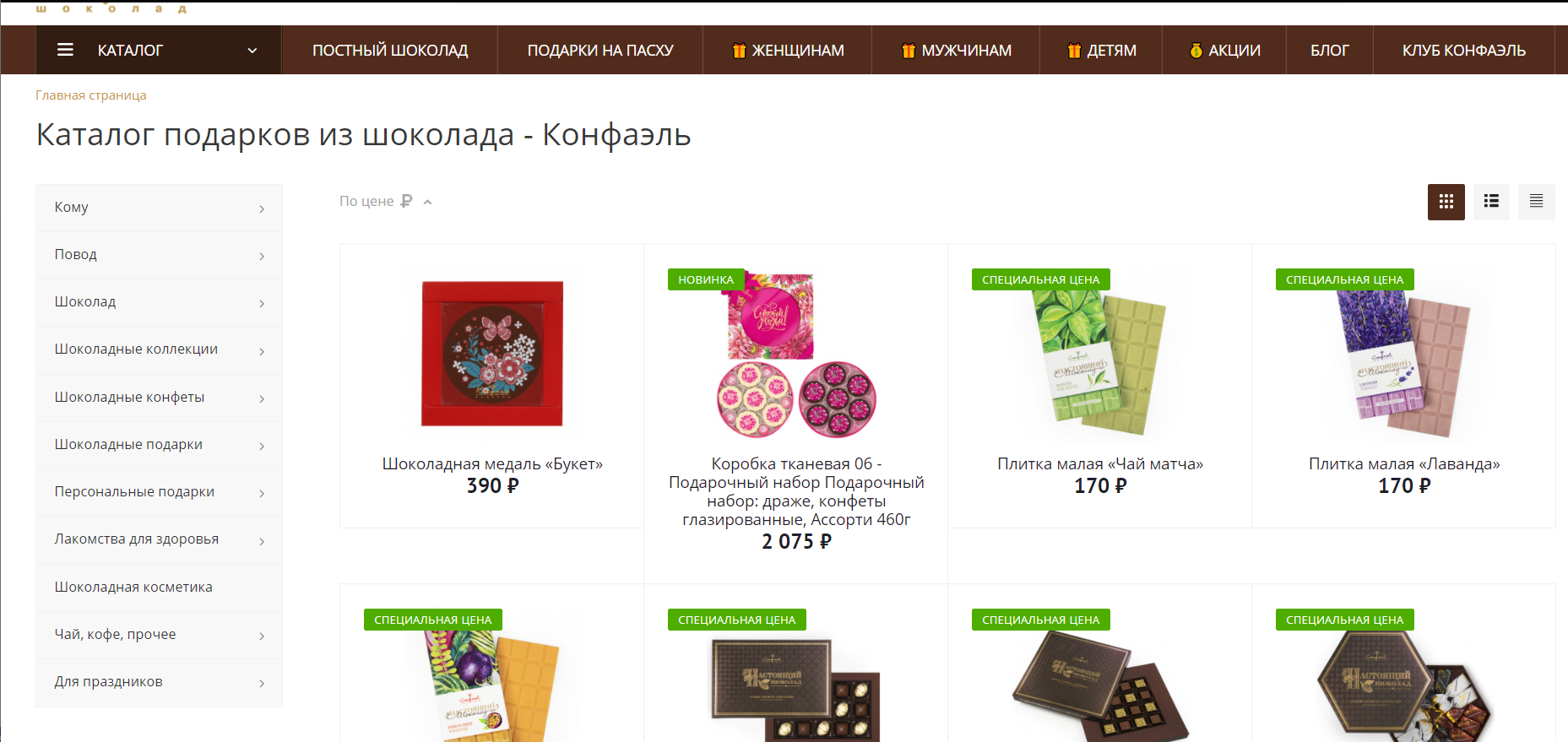


Рисунок 1.1 – Интерфейс интернет-магазина «Confaelshop»

На рисунке 1.2 изображён интерфейс интернет-магазина «El arbol de Cocoa»

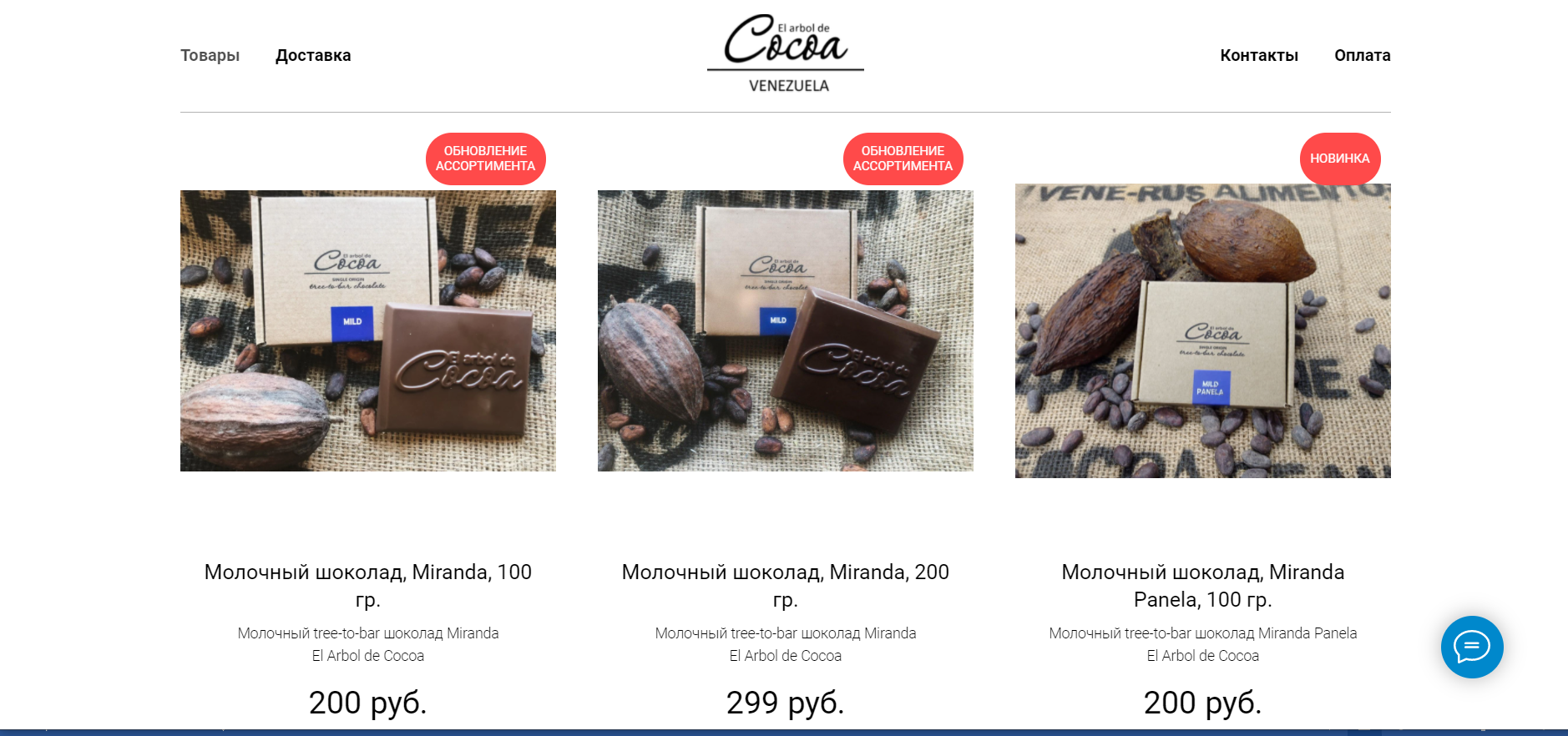


Рисунок 1.2 – интерфейс интернет-магазина «El arbol de Cocoa»

На рисунке 1.3 изображён интерфейс интернет-магазина «Frenchkiss»

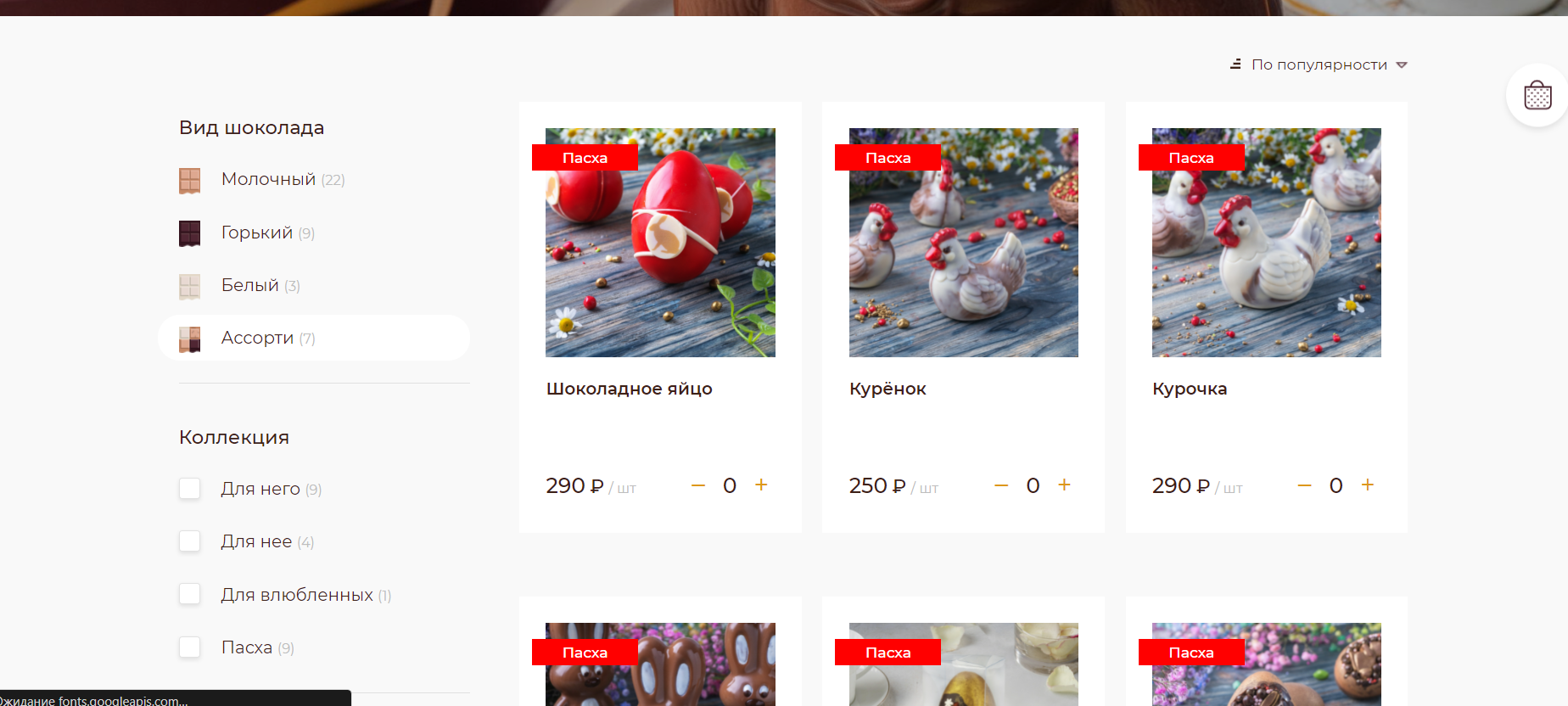


Рисунок 1.3 – Интерфейс интернет-магазина «Frenchkiss»

На рисунке 1.1 изображён интерфейс интернет-магазина «Confaelshop»

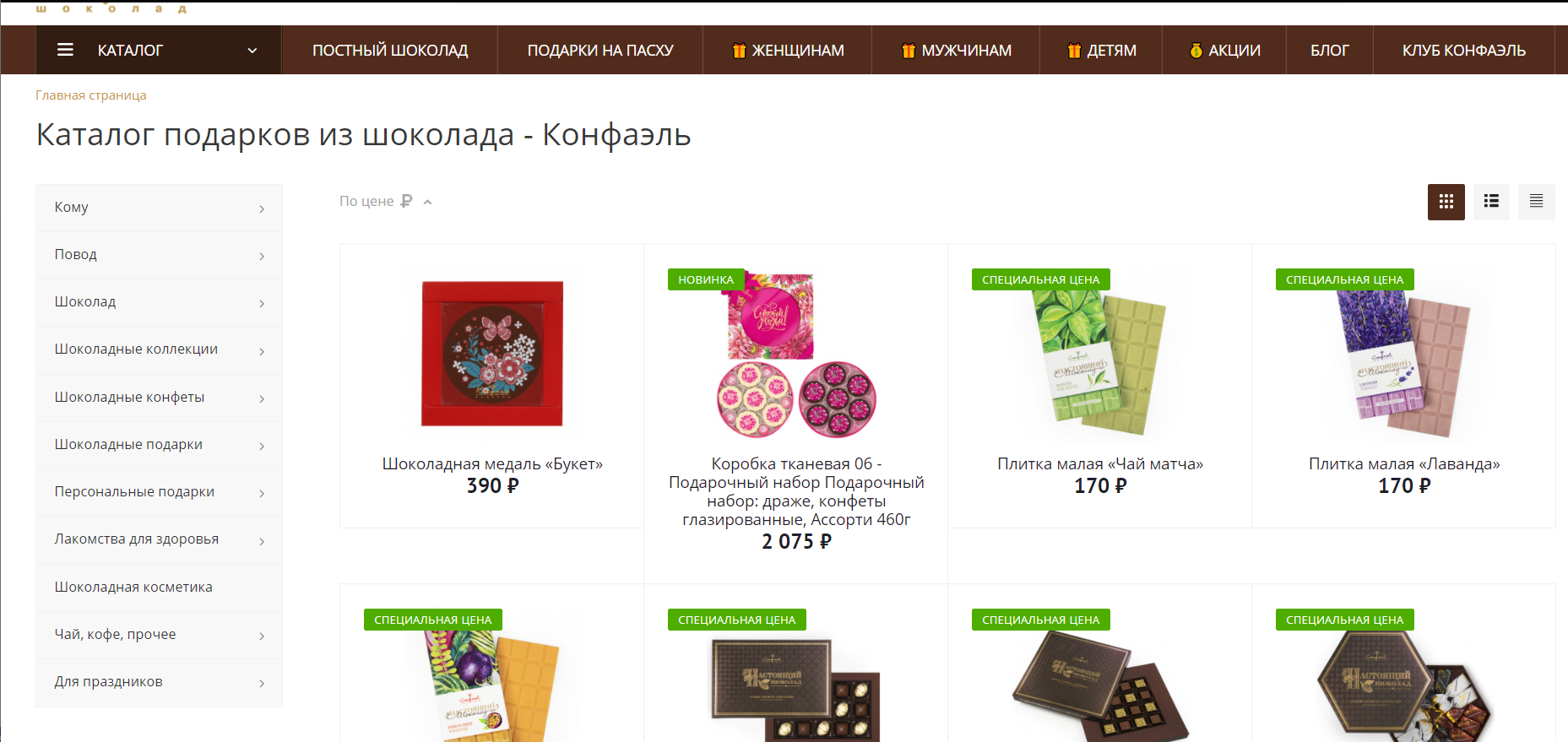


Рисунок 1.1 – Интерфейс интернет-магазина «Confaelshop»

## Выводы к разделу 1

Проведя анализ, можно сделать вывод, что для данной тематике не важен многоструктурный дизайн. Достаточно только самой важной информации – а именно:

1. Каталога товаров шоколадных изделий по тематическим темам
2. Цены одного набор шоколада
3. Различных вариантов изделий, которые подойдут для каждого

Также не стоит забывать и про администраторов и владельцев такого ресурса. Для них нужно создать панель администратора, в котором они могут:

1. видеть список авторизовавшихся пользователей;
2. добавлять типы каталогов;
3. Добавлять продукты к этим типам каталогов, их описание, цену, фотографию.

Именно данные функции должны быть подвергнуты детальному процессу тестирования с наивысшим приоритетом.

Учитывая особенности данного программного обеспечения, можно поставить следующие критически необходимые задачи в порядке их приоритете с первого до последнего:

1. Функциональность и работоспособность панели администратора
2. функции регистрации и авторизации пользователя с корректной обработкой исключений: уже зарегистрированного имени пользователя и неверных введённых данных для процесса авторизации;
3. функции отображения добавленных администратором различных продуктов на странице пользователя.

# РАЗРАБОТКА ИНТЕРНЕТ-МАГАЗИНА

## 2.1. Проектирование программного продукта

После формирования списка общих требований в анализе предметной области необходимо сформулировать список требований заказчика, которые будут отражать его видение проекта и необходимость тестирования функционала.

Список основных требований заказчика (функции, которые должны быть в ПО и быть протестированы):

1. загрузка, удаление и редактирование продуктов, их описания и тд

2. регистрация и вход (и выход) в аккаунт пользователя;

3. Возможность добавление администратором роли ADMIN обычным пользователям и недопуск обычных пользователей в панель администратора.

Данные функции могут быть разбиты на подфункции, например, тестирование регистрации и авторизация, т.к. они требуют от программного обеспечения проверять данные, введённые вручную

**2.2.Выбор средств и технологии ведения разработки**

Для разработки веб-ресурса была использованы технологии Spring boot, Thymeleaf для динамического отображения шаблонов, MySQL – для базы данных.

***2.3.Структура программного продукта***

Структура моего продукта представляет собой папки:

* Сonfig – класс, который будет читать из файлов свойств
* Контроллеры – обрабатывает запрос пользователя, создаёт соответствующую Модель и передаёт её для отображения в Вид.
* View - отвечает за отображение данных Модели, — как правило, генерируя HTML, которые мы видим в своём браузере.
* Model – этот блок инкапсулирует данные приложения.
* Repository – классы для взаимодействия с БД в Spring Data. Каждый репозиторий работает со своим классом-сущностью
* Services – классы, содержащие бизнес-логику продукта

На рисунке 2.1. отображена структура веб-ресурса

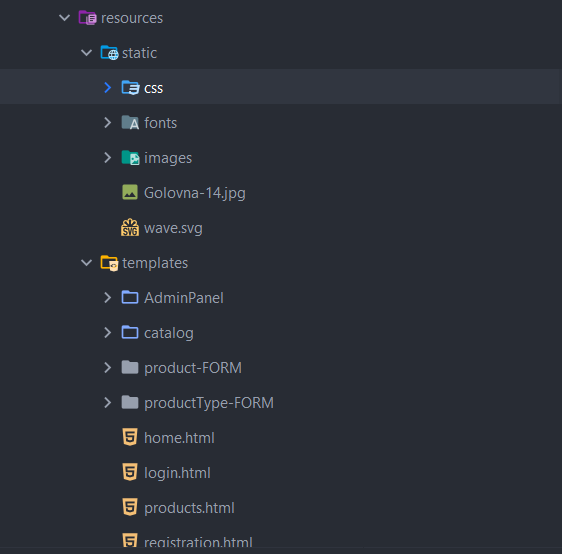
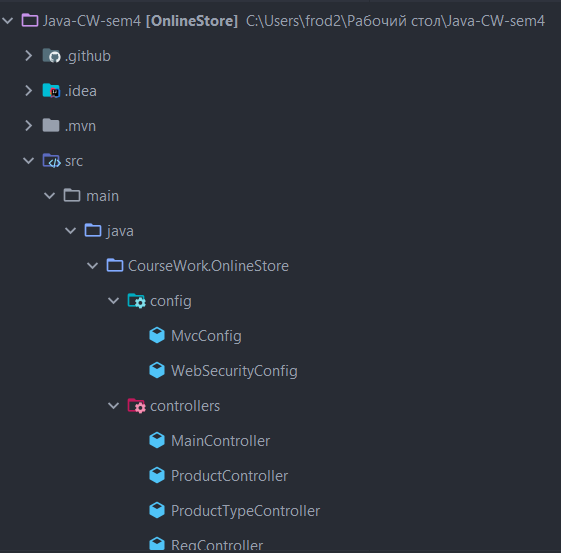


Рисунок 2.1 – Структура проекта

Ниже приведен код классов- контроллеров.

## MainController:

@Controller  
*public class* MainController {  
  
 @Autowired  
 *ProductRepository* productRepository;  
 @Autowired  
 *ProductTypeRepository* productTypeRepository;  
 @Autowired  
 *MessageRepository* messageRepo;  
  
  
 *// Главная страница* @GetMapping("/")  
 *public* String greeting(*Model* model) {  
 model.addAttribute("name", "Главная страница");  
 *return* "home";  
 }  
 *//Страница логина* @GetMapping("/login")  
 *public* String login(*Model* model){  
 model.addAttribute("name","Авторизация");  
 *return* "login";  
 }  
  
  
  
 *//Страница с каталогами* @GetMapping("/products")  
 *public* String products(*Model* model){  
  
 */\*\*  
 \* Map в котором для каждого типа продукта хранится свой список продуктов  
 \*/  
 Iterable*<ProductType> types =productTypeRepository.findAll();  
 *Map*<ProductType, *List*<Product>> map = *new* HashMap<>();  
 *// Идет по каждому типу и для каждого типа в MAP закидывается ключ и значение.* types.forEach(type -> map.put(type, productRepository.findByProductType(type)));  
 model.addAttribute("map", map);  
 *return* "products";  
 }  
  
 @GetMapping("/products/{id}")  
 *public* String productDetails(@PathVariable(value = "id") *long* id, *Model* model){  
 Product product = productRepository.findById(id).orElse(*null*);  
  
 *Iterable*<Message> messages = messageRepo.findAll();  
  
 model.addAttribute("product", product);  
 model.addAttribute("messages", messages);  
  
 *return* "catalog/product";  
 }}

Если пользователь перейдем по адресу с «/», то попадет на главную страницу веб-ресурса; На странице с «/login» хранится шаблон авторизации пользователя; На странице с «/products» содержатся все типы шоколадных изделий на выбор. На странице «/products/{id}» динамически отображается каждый продукт, который выберет пользователь, где id – номер продукта.

Приведем ProductController:

@Controller  
@RequestMapping("/admin")  
*public class* ProductController {  
 @Autowired  
 *ProductRepository* productRepository;  
 @Autowired  
 *ProductTypeRepository* productTypeRepository;  
  
 */\* -------------------------Контроллеры для редактипования и добавления типов продукта-------------------------\*/* @GetMapping("/productList")  
 *public* String productList(*Model* model) {  
 *Iterable*<Product> products = productRepository.findAll();  
 model.addAttribute("products", products);  
 *return* "product-FORM/productList";  
 }  
  
 @GetMapping("/productList/add")  
 *public* String productListAdd(*Model* model) {  
 Product product = *new* Product();  
 model.addAttribute("product", product);  
 *return* "product-FORM/productForm";  
 }  
  
 @PostMapping("/productList/add")  
 *public* String productListAddSubmit(@ModelAttribute Product product, *Model* model){  
 productRepository.save(product);  
 model.addAttribute("product", productRepository.findAll());  
 *return* "product-FORM/productList";  
 }  
  
 @GetMapping("/productList/delete/{productId}")  
 *public* String productListDelete(@PathVariable("productId") *long* id, *Model* model) {  
 productRepository.deleteById(id);  
 model.addAttribute("product", productRepository.findAll());  
 *return* "product-FORM/productList";  
 }  
  
 @GetMapping("/productList/edit/{productId}")  
 *public* String productListEdit(@PathVariable("productId") *long* id, *Model* model) {  
 Product product = productRepository.findById(id).orElse(*null*);  
 model.addAttribute("product", product);  
 *return* "product-FORM/productForm";  
 }  
 */\*==============================================================================================================\*/*}

Этот контроллер отвечает за связь администратора с панелью админа.

Чтобы, например, редактировать конкретный продукт, администратор должен перейти по URL «/admin/productList/edit/{productId}». Но ему не придется вводить все это вручную, ибо переходы на URL сделаны ссылками thymeleaf шаблона

## Контроллер RegController:

@Controller  
*public class* RegController {  
 @Autowired  
 *private UserRepository* userRepo;  
  
 @GetMapping("/registration")  
 *public* String registration() {  
 *return* "registration";  
 }  
  
 @PostMapping("/registration")  
 *public* String addUser(User user, *Map*<String, Object> model) {  
 User userFromDb = userRepo.findByUsername(user.getUsername());  
  
 *if* (userFromDb != *null*) {  
 model.put("message", "User exists!");  
 *return* "registration";  
 }  
  
 user.setActive(*true*);  
 user.setRoles(Collections.*singleton*(Role.USER));  
 userRepo.save(user);  
  
 *return* "redirect:/login";  
 }  
}

Контроль отвечает за страницу регистрации нового пользователя.

## Контроллер UserController:

@Controller  
@RequestMapping("/admin/user")  
*public class* UserController {  
 @Autowired  
 *private UserRepository* userRepo;  
  
 @GetMapping  
 *public* String userList(*Model* model) {  
 model.addAttribute("users", userRepo.findAll());  
 *return* "AdminPanel/userList";  
 }  
  
 @GetMapping("{user}")  
 *public* String userEditForm(@PathVariable User user, *Model* model) {  
 model.addAttribute("user", user);  
 model.addAttribute("roles", Role.*values*());  
  
 *return* "AdminPanel/userEdit";  
 }  
  
 @PostMapping  
 *public* String userSave(  
 @RequestParam String username,  
 @RequestParam *Map*<String, String> form,  
 @RequestParam("userId") User user  
 ) {  
 user.setUsername(username);  
  
 *Set*<String> roles = Arrays.*stream*(Role.*values*())  
 .map(Role::name)  
 .collect(Collectors.*toSet*());  
  
 user.getRoles().clear();  
  
 *for* (String key : form.keySet()) {  
 *if* (roles.contains(key)) {  
 user.getRoles().add(Role.*valueOf*(key));  
 }  
 }  
  
 userRepo.save(user);  
  
 *return* "redirect:/user";  
 }  
  
}

Данный контроллер отвечает за страницу просмотра и редактирования ролей пользователя в панели администратора.

Ниже приведен код конфигурации приложения для подключения авторизации пользователя с помощью Spring Security:

@Configuration  
*public class* MvcConfig *implements WebMvcConfigurer* {  
 *public void* addViewControllers(ViewControllerRegistry registry) {  
 registry.addViewController("/login").setViewName("login");  
 }  
}  
  
  
@Configuration  
@EnableWebSecurity  
*public class* WebSecurityConfig *extends* WebSecurityConfigurerAdapter {  
 @Autowired  
 *private DataSource* dataSource;  
 @Autowired  
 *private* UserService userService;  
  
 @Override  
 *protected void* configure(HttpSecurity http) *throws* Exception {  
 http  
 .authorizeRequests()  
 .antMatchers("/", "/registration").permitAll()  
 .antMatchers("/admin/\*\*").hasAuthority("ADMIN")  
 .anyRequest().authenticated()  
 .and()  
 .formLogin()  
 .loginPage("/login").permitAll()  
 .and()  
 .logout().permitAll();  
 }  
  
 @Override  
 *protected void* configure(AuthenticationManagerBuilder auth) *throws* Exception {  
 auth.userDetailsService(userService)  
 .passwordEncoder(NoOpPasswordEncoder.*getInstance*());  
 }  
}

В методе с аннотацией @Override configure мы разрешили доступ ко всем страницам с url «/» и «/registration», но запретили доступ заходить на страницу с начинающийся url – «/admin/\*\*» - туда вход только пользователям с ролью ADMIN.

Покажем модели, которые есть в приложении:

## Модель Product:

*package* CourseWork.OnlineStore.models;

@Entity(name = "product")  
*public class* Product {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.IDENTITY)  
 *private* Long Id;  
  
 *private* String name;  
  
 *private* Double price;  
  
 *private* String image;  
  
  
 @Column(length = 65553)  
 @Type(type = "text")  
 *private* String description;  
  
 @OneToOne(cascade = CascadeType.ALL)  
 @JoinColumn(name = "product\_type\_id")  
 *private* ProductType productType;  
  
 *public* Product() {  
 }  
  
 *public* Product(String name, Double price, String image) {  
 *this*.name = name;  
 *this*.price = price;  
 *this*.image = image;  
 }  
  
}

## Модель: ProductType:

*package* CourseWork.OnlineStore.models;  
  
*import* lombok.Data;  
*import* org.hibernate.annotations.Type;  
  
*import* javax.persistence.\*;  
  
@Entity(name = "product")  
@Data  
*public class* Product {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.IDENTITY)  
 *private* Long Id;  
  
 *private* String name;  
  
 *private* Double price;  
  
 *private* String image;  
  
  
 @Column(length = 65553)  
 @Type(type = "text")  
 *private* String description;  
  
 @OneToOne(cascade = CascadeType.ALL)  
 @JoinColumn(name = "product\_type\_id")  
 *private* ProductType productType;  
  
 *public* Product() {  
 }  
  
 *public* Product(String name, Double price, String image) {  
 *this*.name = name;  
 *this*.price = price;  
 *this*.image = image;  
 }  
  
}

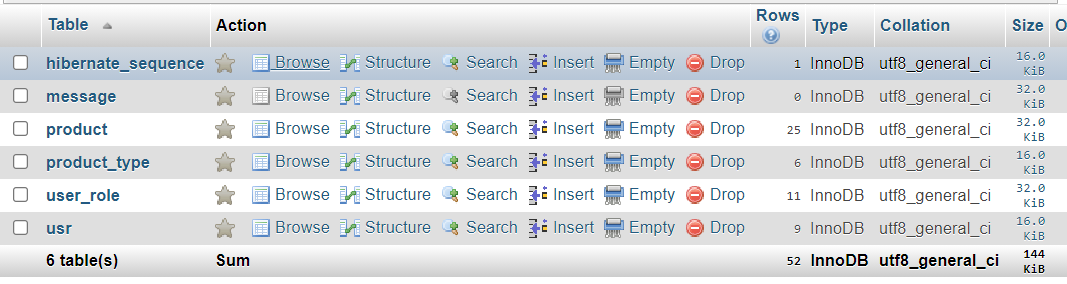
## Модель User:

*package* CourseWork.OnlineStore.models  
@Entity  
@Table(name = "usr")  
@Data  
*public class* User *implements UserDetails* {  
 @Id  
 @GeneratedValue(strategy = GenerationType.AUTO)  
 *private* Long id;  
 *private* String username;  
 *private* String password;  
 *private boolean* active;  
  
 @ElementCollection(targetClass = Role.*class*, fetch = FetchType.EAGER)  
 @CollectionTable(name = "user\_role", joinColumns = @JoinColumn(name = "user\_id"))  
 @Enumerated(EnumType.STRING)  
 *private Set*<Role> roles;  
  
 @Override  
 *public boolean* isAccountNonExpired() {  
 *return true*;  
 }  
  
 @Override  
 *public boolean* isAccountNonLocked() {  
 *return true*;  
 }  
  
 @Override  
 *public boolean* isCredentialsNonExpired() {  
 *return true*;  
 }  
  
 @Override  
 *public boolean* isEnabled() {  
 *return true*;  
 }

Аннотация @Table говорит о том, что модель представляет с собой таблицу. Эту аннотацию обязательно надо вводить, чтобы связаться с БД.

Аннотация @Data создает геттеры и сеттеры для атрибутов наших классов

Покажем данные таблицы в БД MySQL:



Как видим все таблицы успешно создались в БД MySQL.

# Покажем на рисунке 2.1 страницу с каталогами:

Рисунок 2.1. Главная страница сайта

Покажем на рисунке 2.2 страницу с полным каталогом продуктов:

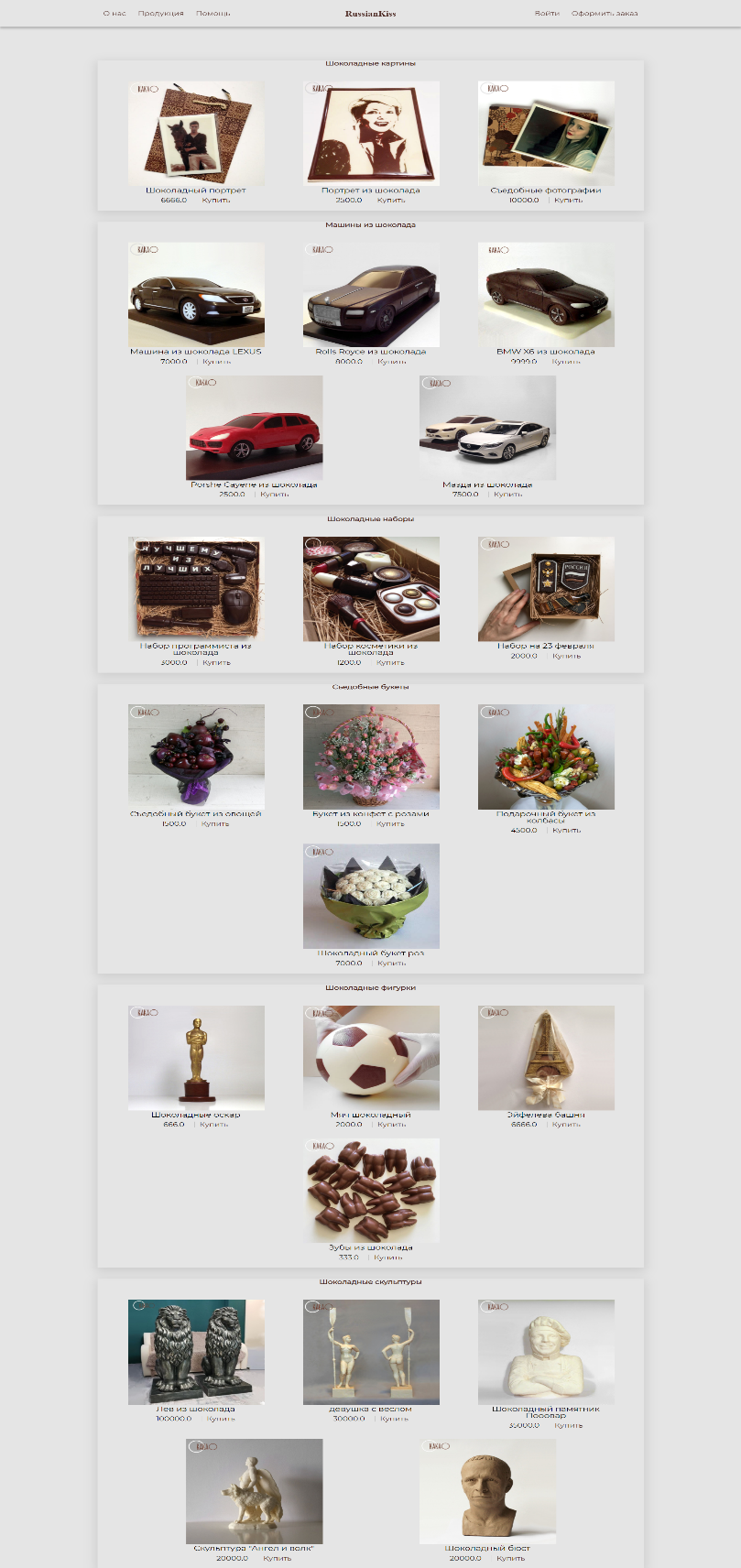


Рисунок 2.2. Страница с продуктами

Все эти отображения каталогов и их продуктов были добавлены через удобную панель администратора.

Покажем и её на рисунках 2.3, 2.4, 2.5:

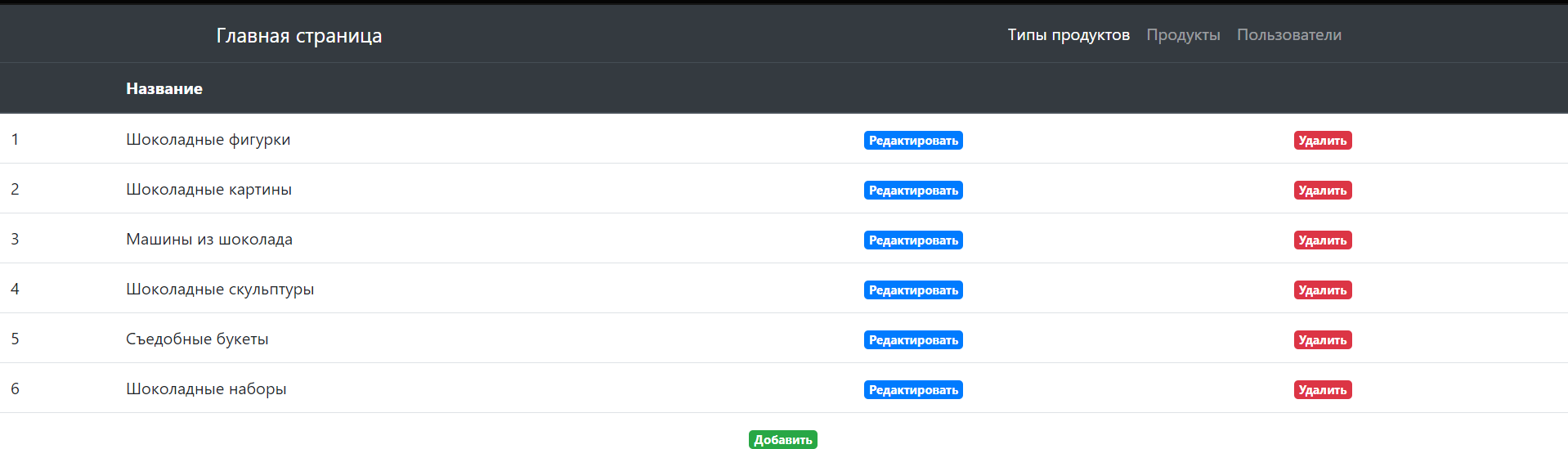


Рисунок 2.3. Страница с панелью администратора – редактирование и добавление типов пр

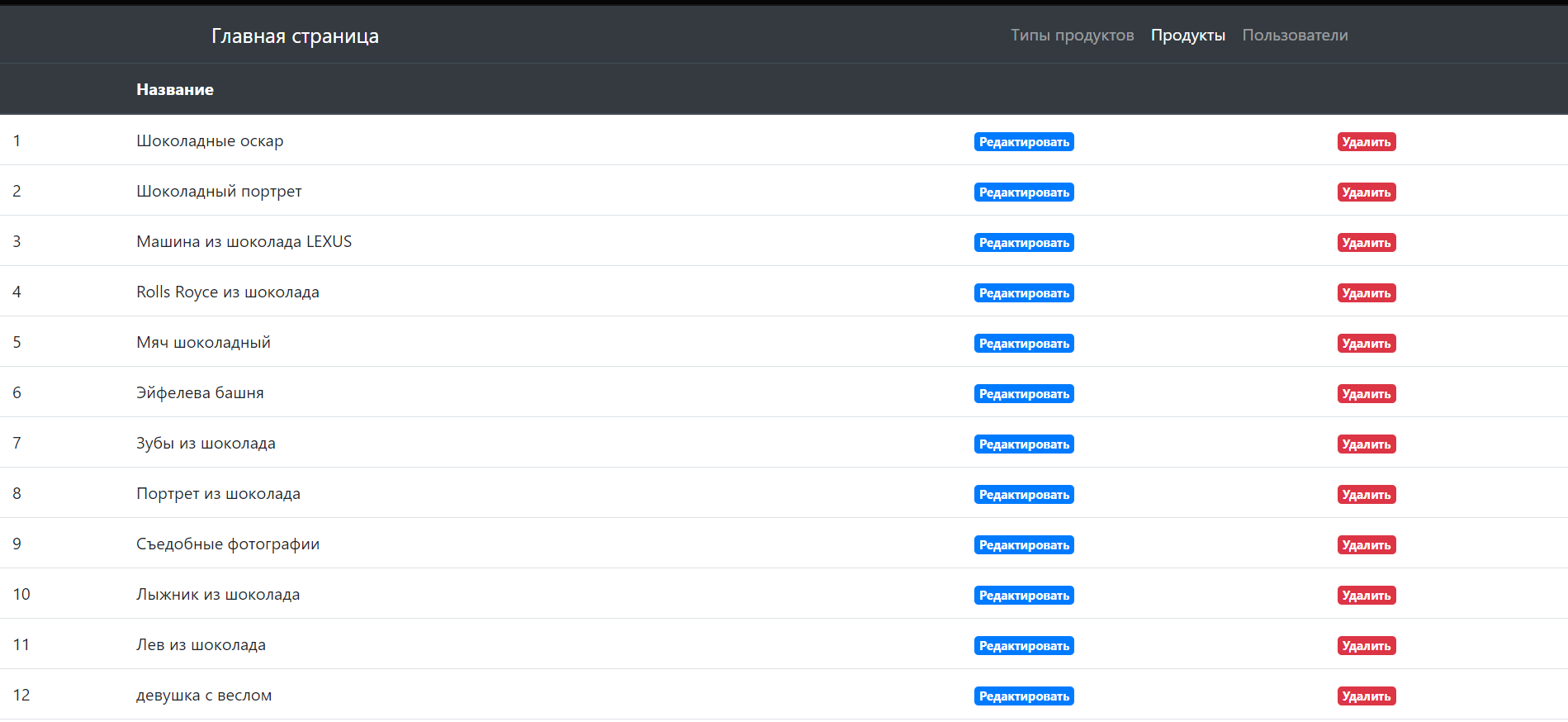


Рисунок 2.4. Страница с панелью админа – редактирование и добавление продуктов

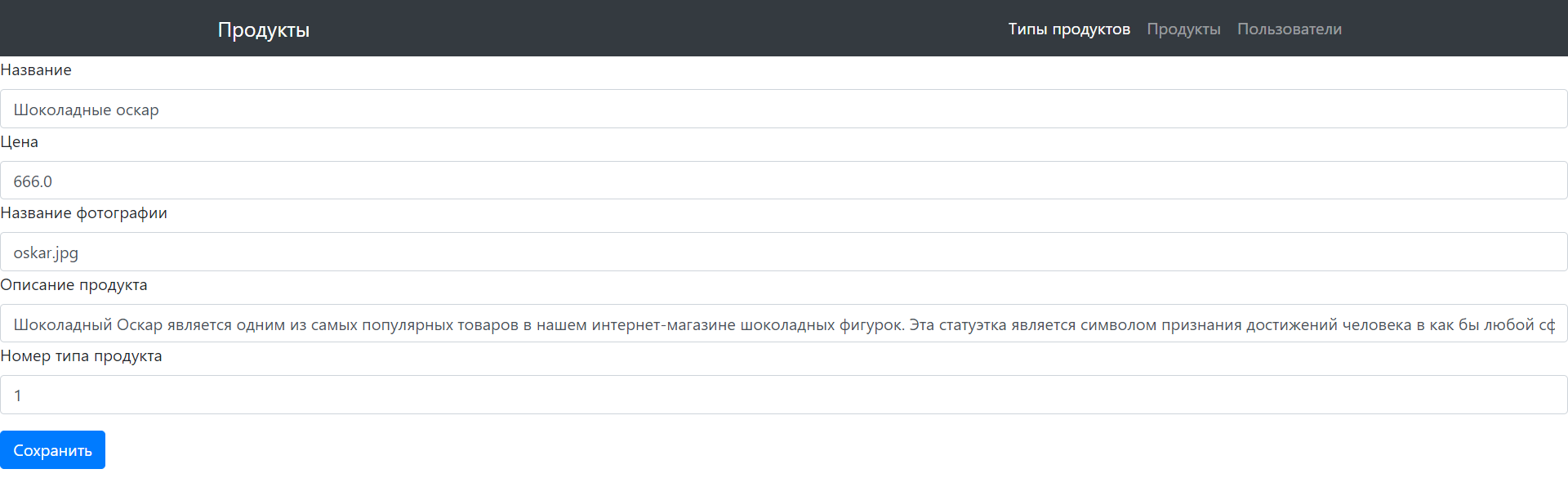


Рисунок 2.5. Страница с редактированием описания продукта, цены и тд.

## Выводы к разделу 2

В результате анализа требований заказчика к программному обеспечению были сделаны выводы о требованиях заказчика и о необходимых техниках тестирования, которые позволят выявить слабые места ПО, а также продемонстрировать корректность его работы.

Были проанализированы техники тестирования, и из них были выбраны две: «формулировка вопросов» и «формирование чек-листов или тест-кейсов».

Был составлен список уточняющих вопросов для тестирования требований заказчика, в соответствии с техникой тестирования «формулирование вопросов», с целью получить наиболее точный и достоверный список требований, необходимых к реализации в программном проекте.

# ТЕСТИРОВАНИЕ ПРОГРАММНОГО ПРОДУКТА

В этом разделе описывается тестирование программного продукта.

* 1. ***Тестовый план***

Тестовый план составляется по стандарту «IEEE 829» с удалением нескольких пунктов из-за ограниченности проводимого тестирования [9].

**3.1.1 Test Plan Identifier (идентификатор)**

Тестовый план нумеруется как «id\_1», уровень тестового плана - «plan\_level\_1», уровень программного обеспечения - «websoftware\_1». План является мастер-планом («Master Plan»).

Номер ревизии плана: 1.

Контактная информация главного составителя плана тестирования:

ФИО: Анваржонов Ж Т.

Email: anvarzhonov@mail.ru

Номер телефона: +7 (977) 386-38-17.

***3.1.2. Цель***

Целью плана является создание условий тестирования функций ПО «веб-приложения шоколадных изделий», улучшение работоспособности системы, посредством уменьшения вероятности отказа работы основных модулей программы.

Тестирование объектов тестирования необходимо для повышения удовлетворения пользователей сайта и заказчика проекта от использования веб-приложения, уменьшение количества ошибок ПО в процессе эксплуатации [10].

**3.1.3 Области, подвергаемые тестированию**

1. Регистрация пользователей в системе
2. Авторизация / выход пользователей из системы;
3. добавление, удаление и отображение продуктов;
4. Добавление администратором допуска пользователям входа в панель администратора; проследить, чтобы обычный пользователь с ролью User не смог зайти на страницу панели администратора;

**3.1.4 Области, не подвергаемые тестированию**

1) сетевая безопасность сервера;

2) бизнес-аналитика приложения;

4) программы, в которых разрабатывалось приложение.

## 3.1.5 Тестовая стратегия и подходы

Тестирование безопасности будет проходить методом «белого ящика». Остальное будет проверяться методом «серого ящика» (или «чёрного ящика», если доступа к ПО нет).

Уровни тестирования:

1) Модульное тестирование

Проводится разработчиком и одобряется руководителем. После одобрения данного вида тестирования, осуществляется системное/интеграционное тестирование.

2) Системное/интеграционное тестирование

Проводится менеджером тестирования. Системное/интеграционное тестирование происходит после того, как критические дефекты исправлены. После этого вида тестирования выполняется приёмочное тестирование.

3) Приемочное тестирование

Проводится конечными пользователями с помощью менеджера тестирования.

**3.1.6 Критерии**

Необходимо, чтобы пользователи с ролью «ADMIN» загрузил все продукты через панель администратора на веб-сайт.

После этого осуществляется проверка количества успешно загруженных моделей, которые отображаются корректно в веб-браузере.

Для каждой тестируемой метрики её показатели «выполнение тест-кейсов» и «успешное прохождение тест-кейсов» должны быть больше 60 процентов и меньше 80 процентов соответственно. В этом случае тестирование продолжается.

**3.1.7 Ресурсы**

* Операционная система: Windows 10 Home.
* Установленный Intelij Idea Ultimate, программа MAMP для запуска локального сервера на MySQL.
* Внедренные зависимости Spring для корректного отображения и работы приложения в целом.

**3.1.8 Роли и ответственность**

Программист ответственен за выполнение модульного тестирования.

Менеджер тестирования ответственен за выполнение системного/интеграционного тестирования.

Конечные пользователи ответственны за приёмочное тестирование.

***3.1.9 Метрики***

* Покрытие требований тест-кейсами – процентное отношение количества покрытых тест-кейсами требований к общему количеству требований.
* Успешное прохождение тест-кейсов – процентное отношение количества успешно выполненных тест-кейсов к общему количеству выполненных тест-кейсов.
* Выполнение тест-кейсов – процентное отношение количества выполненных тест-кейсов к количеству тест-кейсов, запланированных к выполнению.

# 3.2.Тест кейсы

В таблице 4.1 представлен сформированный тест-кейс для тестирования регистрации пользователя в системе.

## Таблица 4.1 – Тест-кейс для тестирования регистрации на сайте

|  |  |  |  |
| --- | --- | --- | --- |
| ID: 1 | Название: Регистрация на сайте | | |
| R\_1 | Анваржонов Ж | Высокий приоритет | Модуль: Форма регистрации |
| Приготовления на начало тестирования:   1. Перейти на сайт проекта и нажать на кнопку «Авторизация» в верхней панели. 2. Выбрать «Регистрация» на этой странице | | | |
| Шаги:   1. Нажать на поля ввода и ввести туда корректные требуемые данные (имя пользователя, пароль). 2. Нажать на кнопку «Зарегистрироваться». | | Ожидаемый результат:  1. Изменение внешнего вида форм на с выводом ошибки красного цвета в случае уже существующего пользователя  2. Если имя свободно, пароль введен(иначе вернуться к пункту 1 и заменить неверные данные), в БД добавляется запись в таблицу и происходит redirect на страницу авторизации. | |
| Статус «Разработано» | | Дата выполнения  20.05.2021 | Ссылка на дефекты: --- |

Покажем все описанное в таблице на рисунке 3.1:

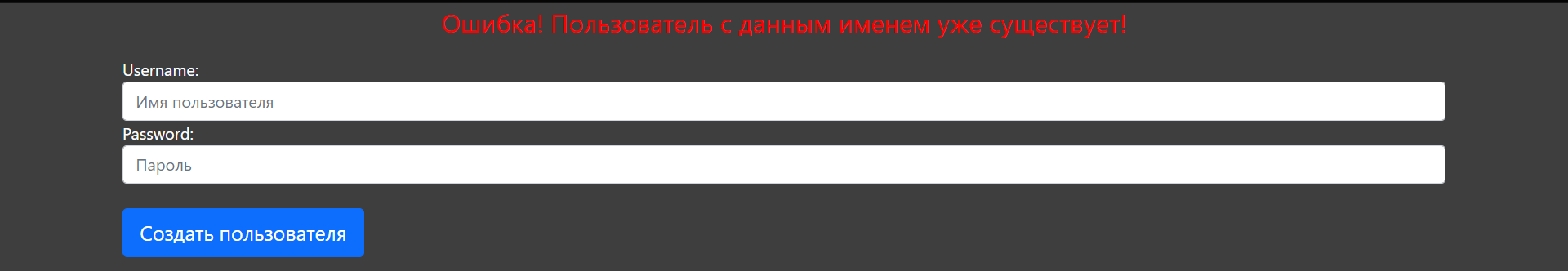


Рисунок 3.1. Страница с регистрацией пользователя при введенных данных уже существующего пользователя.

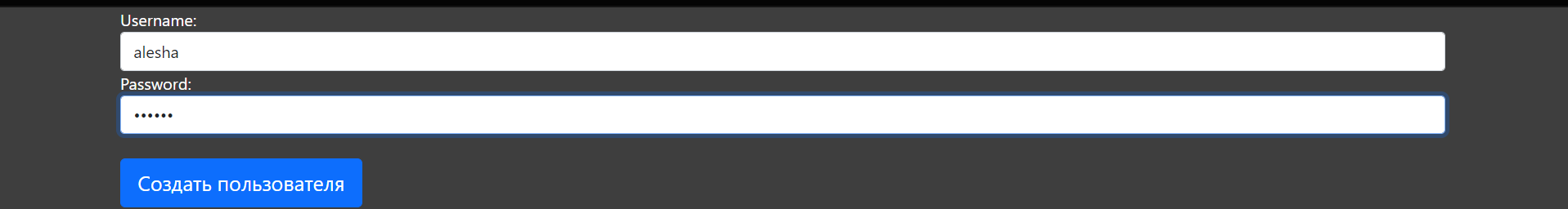


Рисунок 3.2. Создание пользователя с именем alesha

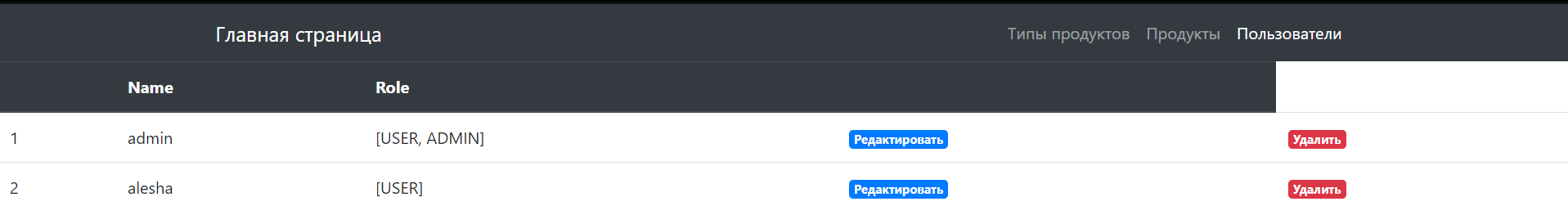


Рисунок 3.3. Отображение созданного пользователя в панели администратора.

В таблице 4.2 представлен сформированный тест-кейс для тестирования формы авторизации пользователей и их выход из системы

## Таблица 4.2 – Тест-кейс для тестирования формы авторизации пользователей и их выход из системы.

|  |  |  |  |
| --- | --- | --- | --- |
| ID: 2 | Название: Вход и выход пользователя из системы | | |
| R\_2 | Анваржонов Ж | Высокий приоритет | Модуль: Форма входа |
| Приготовления на начало тестирования:   1. Перейти на сайт проекта и нажать на кнопку «Авторизация» в верхней панели. 2. Ввести данные, которые указали в регистрации | | | |
| Шаги:  1.Перейти на сайт проекта и нажать на кнопку «Авторизация» в верхней панели.  2. Набрать имя пользователя и пароль  3. Нажать «Войти в систему» или «Выйти из системы» - если вход уже был выполнен | | Ожидаемый результат:  1. Появления ошибки о том, что данные пользователя были введены неудачно, в случае некорректных данных  2. Если данные пользователя, которые хранятся в БД, были введены корректно, то происходит redirect на главную страницу веб-сайта.  3. При нажатии на кнопку «Выход из системы» - появление сообщения о том, что был осуществлен выход из системы. | |
| Статус «Разработано» | | Дата выполнения  22.05.2021 | Ссылка на дефекты: -- |

Покажем все описанное на рисунках 3.4, 3.5, 3.6:

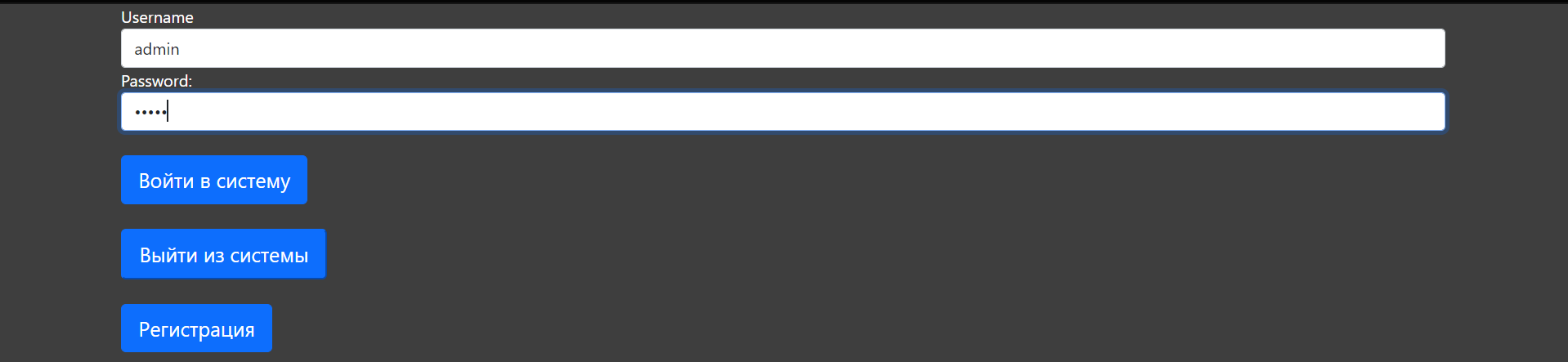


Рисунок 3.4. Форма входа в систему.

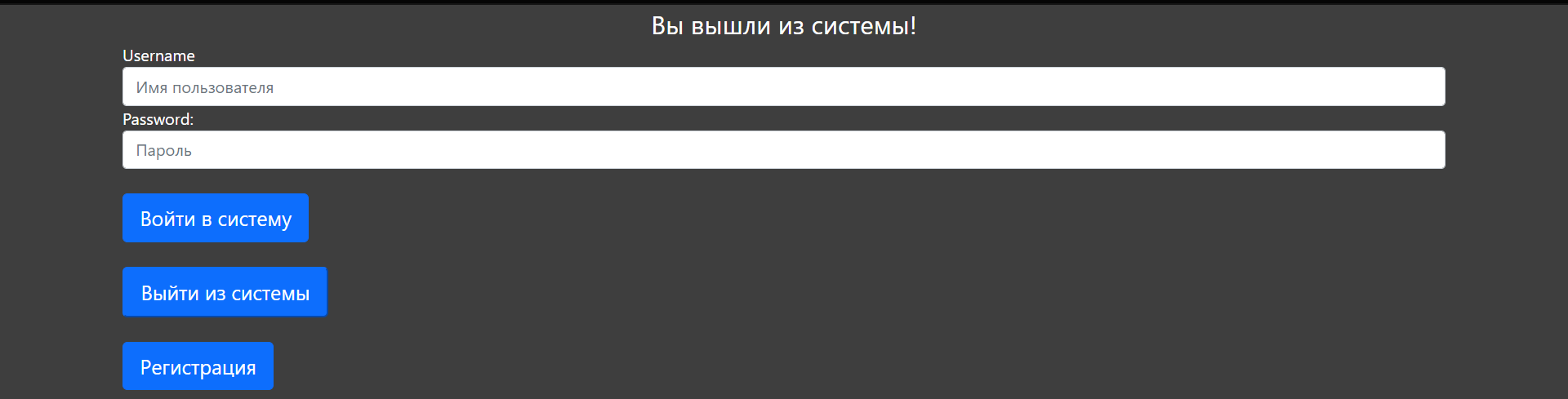


Рисунок 3.5. Сообщение при выходе из системы.



Рисунок 3.6. Сообщение при некорректно введенных данных.

В таблице 4.3 представлен сформированный тест-кейс для тестирования добавления, редактирования и удаления продуктов или типов продуктов.

## Таблица 4.3 – Тест-кейс для тестирования добавления, редактирования и удаления продуктов или типов продуктов

|  |  |  |  |
| --- | --- | --- | --- |
| ID: 2 | Название: Вход и выход пользователя из системы | | |
| R\_2 | Анваржонов Ж | Высокий приоритет | Модуль: Панель админа |
| Приготовления на начало тестирования:   1. Перейти на сайт проекта и нажать на кнопку «Авторизация» в верхней панели. 2. Ввести данные администратора для входа в систему | | | |
| Шаги:  1.Перейти на сайт проекта и нажать на кнопку «Авторизация» в верхней панели.  2. Набрать имя пользователя и пароль пользователя с ролью ADMIN  3. Нажать «Войти в систему».  4. Нажать в верхней панели сайти «панель администратора» чтобы перейти на страницу админа | | Ожидаемый результат:  1. Корректное отображение на страницах продуктов, которые были добавлены через панель администратора  2. Корректное отображение на страницах продуктов, информация которых была редактирована через панель администратора  3. Исчезновение из страниц продукта, который был удален через панель администратора | |
| Статус «Разработано» | | Дата выполнения  20.05.2021 | Ссылка на дефекты: --- |

Покажем все описанное в таблице на рисунках:

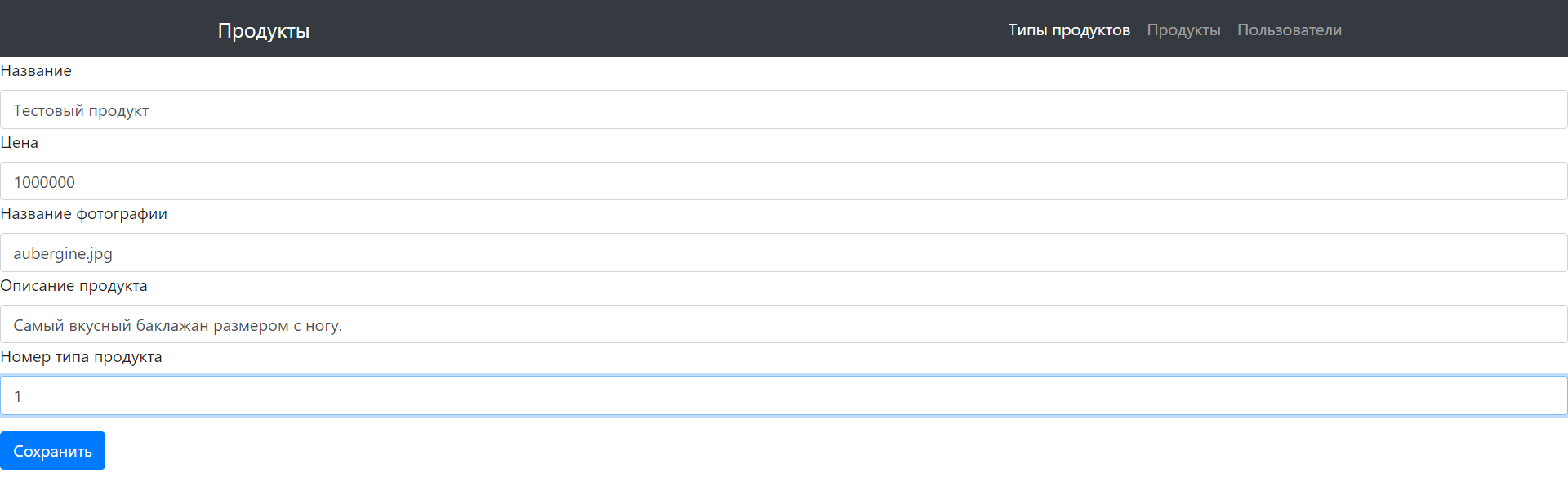


Рисунок 3.7. Форма добавления продукта через панель администратора

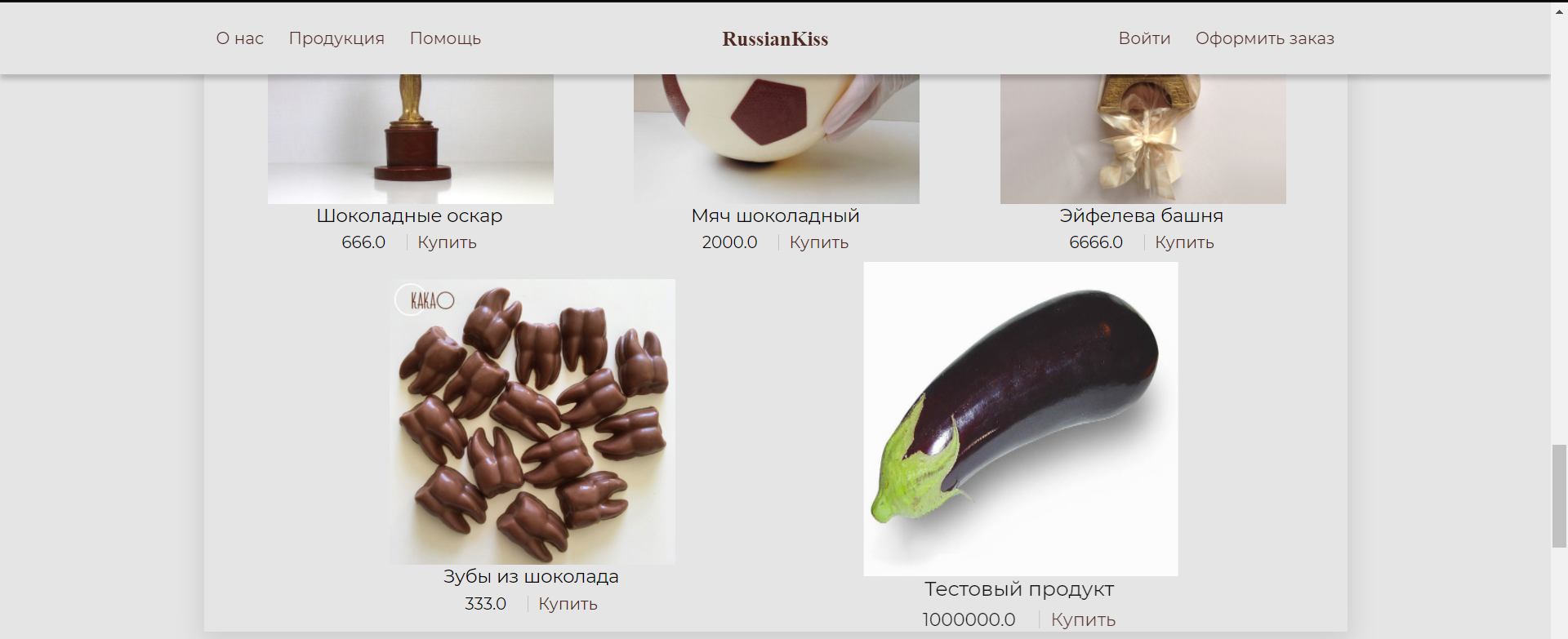


Рисунок 3.8. Отображение добавленного нового продукта через панель админа

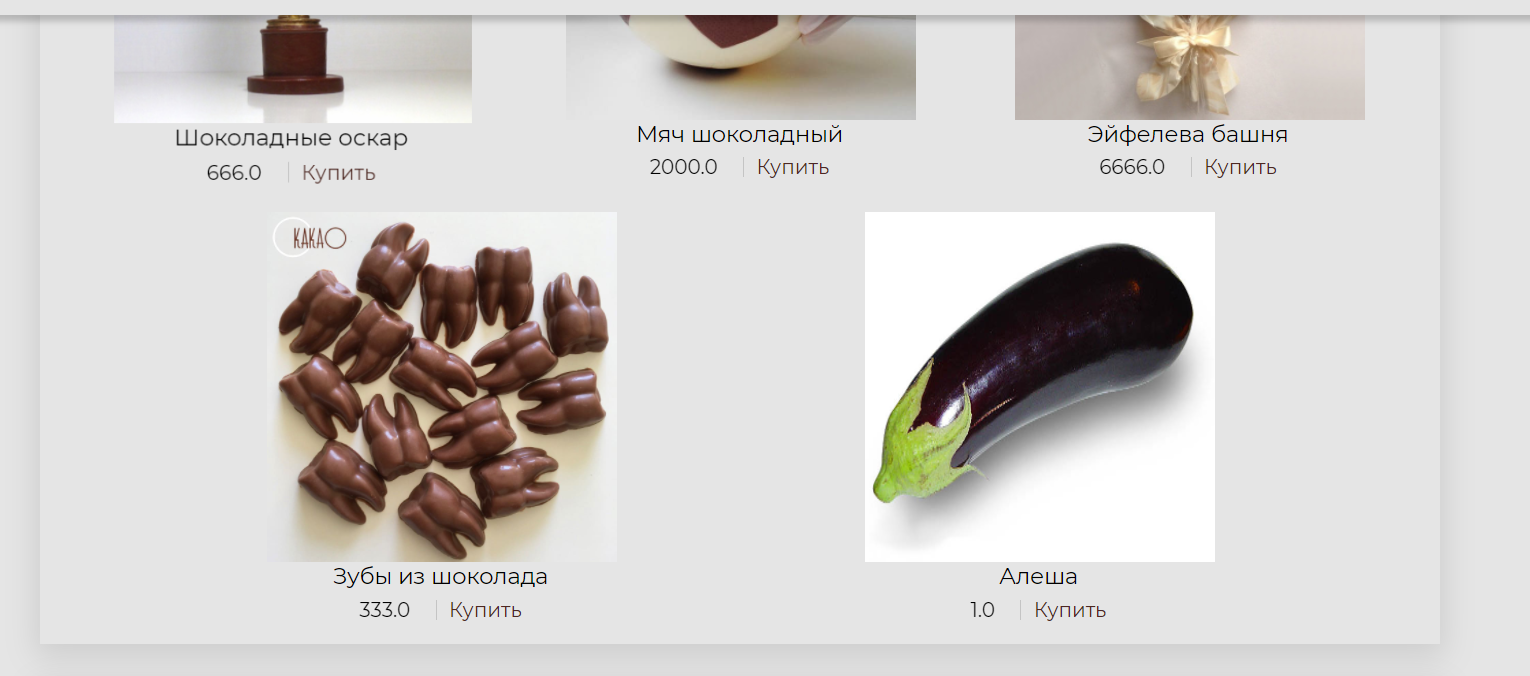


Рисунок 3.9. Отображение редактированного продукта через панель админа

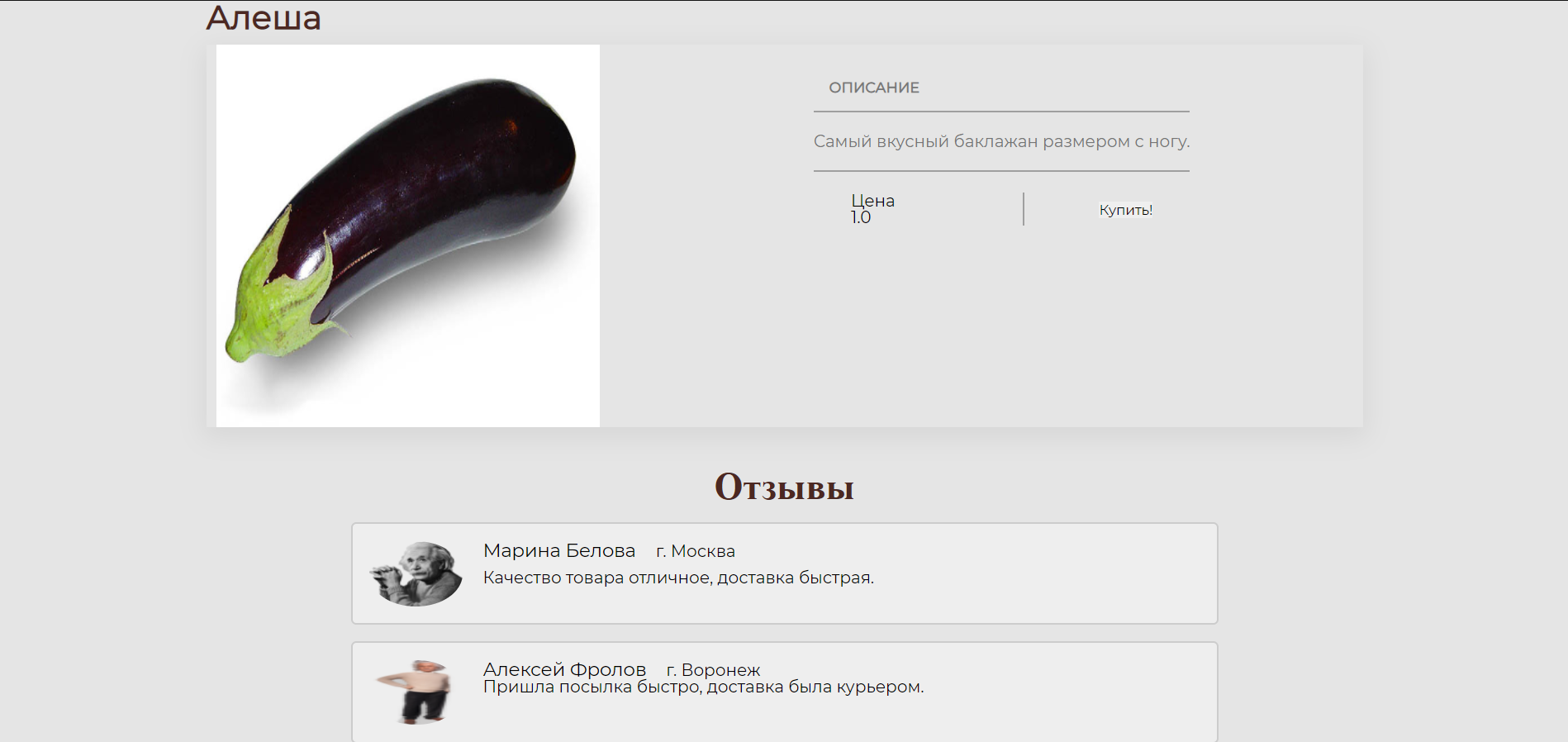


Рисунок 3.10. Отображение страницы с описанием продукта, который был добавлен через панель админа

В таблице 4.4 представлен сформированный тест-кейс для тестирования допуска пользователя в панель администратора

## Таблица 4.4 – Тест-кейс для тестирования допуска пользователя в панель администратора

|  |  |  |  |
| --- | --- | --- | --- |
| ID: 2 | Название: Вход и выход пользователя из системы | | |
| R\_2 | Анваржонов Ж | Высокий приоритет | Модуль: панель администратора |
| Приготовления на начало тестирования:   1. Перейти на сайт проекта и нажать на кнопку «Авторизация» в верхней панели. 2. Зарегистрировать нового пользователя или войти в уже существующего | | | |
| Шаги:  1.Перейти на сайт проекта и нажать на кнопку «Авторизация» в верхней панели.  2. Набрать имя пользователя и пароль  3. Нажать «Войти в систему»  4. Попыться Зайти на панель администратора | | Ожидаемый результат:  1. Если у пользователя нет роли ADMIN, то отображение ошибки при переходе на страницу панели администратора, иначе если у пользователя есть роль Admin’a, то удачный вход на страницу панели админа  2. Изменение ролей у пользователей для допуска на панель админа | |
| Статус «Разработано» | | Дата выполнения  20.05.2021 | Ссылка на дефекты: --- |

Покажем все описанное в таблице на рисунках:

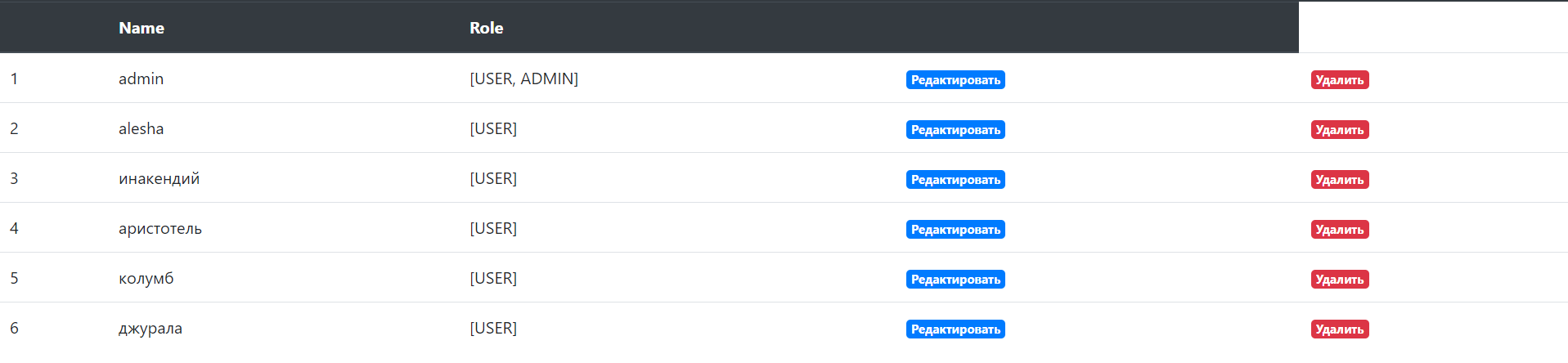


Рисунок 3.11. Страница всех пользователей, зарегистрированных на сайте.

Попробуем зайти на аккаунт инакендия и попытаемся войти в панель администратора:





Рисунок 3.12. Попытка входа в панель администратора через пользователя с обычной ролью – USER.

Дадим инакендию роль ADMIN’a и попробуем зайти на панель админа:





Рисунок 3.13. Добавили дополнительную роль к пользователю

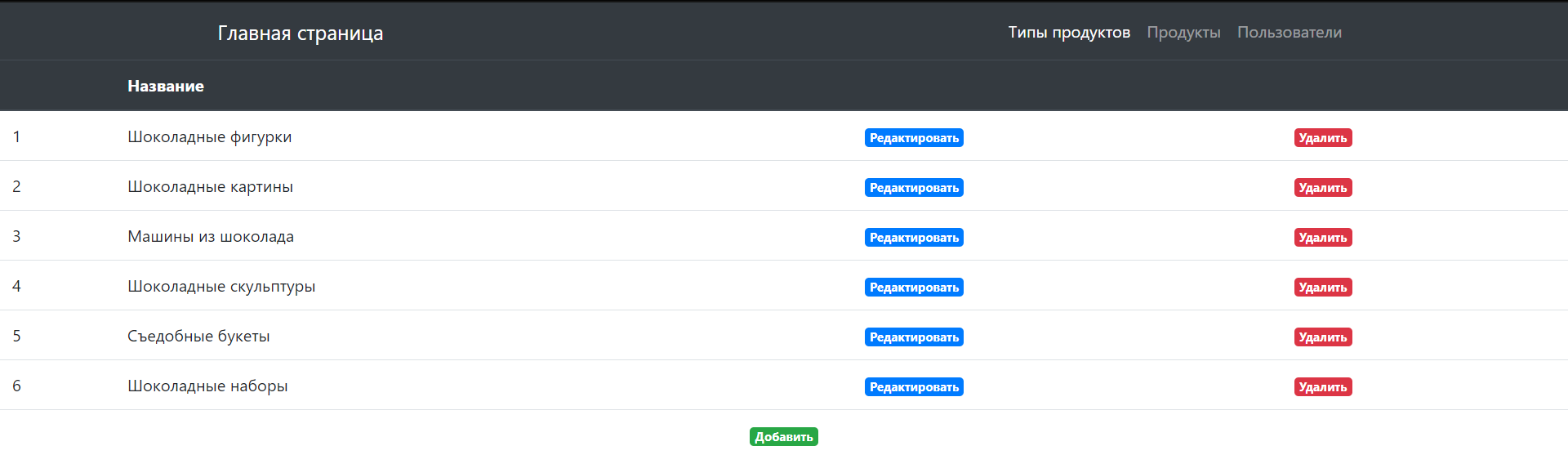
****

Рисунок 3.14. Удачный вход пользователя «инакендий» в панель администратора.

Как видим, изменение роли у пользователя позволило инакендию войти в панель админа.

## Выводы к разделу 3

В данном разделе был составлен тестовый план в соответствии со стандартом «IEEE 829», который позволит повысить отказоустойчивость и стабильность тестируемого программного продукта.

# ЗАКЛЮЧЕНИЕ

В процессе выполнения данной курсовой работы было проведено тестирование интернет-магазина шоколадных изделий. Спроектированное приложение дает пользователям видеть и пользоваться сайтом с хорошим дизайном и минимумом сложных конструкций, для администратора – возможность удобно редактировать продукты, их типы, изменять доступ обычных пользователям к суперпользователям

Был проведён анализ предметной области с выявлением базовых функций для подобного класса систем, а также определены требования, которые должны быть реализованы и протестированы в рамках системы заказчика.

Был протестированы требования заказчика, изучены техники тестирования и выбраны те их них, которые позволяют провести наиболее качественное и быстрое тестирование веб-системы.

Был разработан план тестирования для повышения качества работы департамента тест-дизайна и составлены тестовые-кейсы в виде демонстрации применения техники тестирования. Тестирование никаких ошибок не показало.

В процессе разработки курсовой работы были приобретены следующие

компетенции:

ПК-20 – способностью оценивать временную и емкостную сложность программного обеспечения.

ОПК-3 – готовностью применять основы информатики и программирования к проектированию, конструированию и тестированию программных продуктов.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Куликов С. С. Тестирование программного обеспечения. — 2017. — 356 с. [Дата обращения: 20.05.21];
2. Видео-курсы по Spring Boot. https://www.youtube.com/watch?v=R3YNPm7ZmYY&list=PL0lO. [Дата обращения: 20.05.21];
3. Официальный веб-сайт Spring [Электронный ресурс]: Режим доступа: https://spring.io/ [Дата обращения: 20.05.21];
4. Веб-сайт Spring Security Reference [Электронный ресурс]: Режим доступа: https://docs.spring.io/spring-security/site/docs/current/reference/html5/ [Дата обращения: 20.05.21];
5. Видео-курс Spring Security | FULL COURSE [Электронный ресурс]: Режим доступа: https://www.youtube.com/watch?v=her\_7pa0vrg/ [Дата обращения: 20.05.21];
6. Гэвин Кинг, Java Persistence with Hibernate [Электронный ресурс]: учебное пособие / Г. Кинг. — Электрон. дан. — М.: , 2016. — 248 с. — Режим доступа: https://www.manning.com/books/java-persistence-with-hibernate. — Загл. с экрана [Дата обращения: 20.05.21].
7. Видео-курс Spring Security | FULL COURSE [Электронный ресурс]: Режим доступа: https://www.youtube.com/watch?v=her\_7pa0vrg/ [Дата обращения: 20.05.21];
8. Официальный сайт MySQL [Электронный ресурс]: Режим доступа: https://www.mysql.com/ [Дата обращения: 20.05.21];
9. Крейг Уоллс, Spring в действии [Электронный ресурс]: учебное пособие— Электрон. дан.— 208 с. — Режим доступа: https://ru.pdfdrive.com/spring-%D0%B2-%D0%B4%D0%B5%D0%B9%D1%81%D1%82%D0%B2%D0%B8%D0%B8-e188685085.html— Загл. с экрана [Дата обращения 20.05.21];