МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»**

Кафедра Математического и компьютерного моделирования

Разработка интернет-магазина на основе NOSQL

АВТОРЕФЕРАТ МАГИСТЕРСКОЙ РАБОТЫ

студента 2 курса 247 группы

направление 09.04.03 — Прикладная информатика

механико-математического факультета

Шведова Евгения Валерьевича

Научный руководитель
профессор,д.э.н.,профессор. Л. В Кальянов

Зав. кафедрой
зав. каф., д.ф.-м.н., доцент Ю.А. Блинков

Саратов 2022

Введение. Магистерская работа посвящена изучению электронной коммерции и разработке собственного интернет магазина по продаже электронной техники с использование нереляционной базы данных.

С каждый годом количество интернет пользователей только увеличивается. Уже сейчас продажи через интернет в крупных городах достигают 25%, при этом специалисты подчеркивают, что тенденция увеличения онлайн продаж сохранится в долгосрочной перспективе. Ежегодно количество интернет-магазинов увеличивается, так как это действительно прибыльно и удобно для покупателя, не говоря о экономии бюджета и времени.

Таким образом, создание интернет-магазинов является довольно важной и актуальной задачей, так как у них есть ряд преимуществ, по сравнению с обычными магазинами:

- интернет-магазин работает круглые сутки и может продавать определенные товары в автоматическом режиме без участия продавца;
- не надо закупать товар заранее, а это существенная экономия, на складских помещениях;
- нужно лишь договориться с поставщиками, и в нужный момент, просто выкупить товар, который у вас закажут;
- по сравнению с обычным магазином, территория продаж которого ограничивается населением города или района, территория охвата интернет-магазина увеличивается на всю Россию и русскоязычную аудиторию в других странах, ведь товар можно доставлять не только курьерской службой, но и почтой.

Целью данной магистерской работы является разработка интернет-магазина по продаже электронной техники с использованием NOSQL БД MongoDB.

Для реализации проекта необходимо выполнить следующие задачи:

- рассмотреть особенности электронной коммерции;
- рассмотреть особенности NOSQL баз данных;
- разработать основной концепт интернет-магазина с дизайном
- обеспечить систему безопасности авторизации;
- реализовать функционал интернет-магазина

Основная часть. Магистерская работа состоит из введения, 2 разделов, заключения и списка используемой литературы. Во введении описывается важность создания интернет-магазинов и их преимущества по сравнению с обычными магазинами, а также формулируется цель работы, и ставятся задачи. Первая глава посвящена теории и выбору инструментов для решения задачи.

Под термином «электронная коммерция» понимается прежде всего предоставление товаров и платных услуг через глобальные информационные сети.

Термин «веб-сервис» имеет множество различных и расплывчатых определений, но на практике оно должно быть достаточным для понимания и написания кода, который состоит из сервера и клиента, также известного как consumer или requester, что хорошо описывается книгах М. Калина «Java Web Services: Up and Running» и Т.С. Машнина «Web-сервисы Java».

REST — это архитектурный стиль для проектирования распределенных систем. Он не является стандартом, но определяет ограничения, такие как отсутствие состояний, клиент-серверная взаимосвязь и унифицированный интерфейс. REST не связан строго с HTTP, но чаще всего его ассоциируют именно с ним.

Spring Framework обеспечивает комплексную модель разработки и конфигурации для современных бизнес-приложений на Java — на любых платформах. Ключевой элемент Spring — поддержка инфраструктуры на уровне приложения: основное внимание уделяется «водопроводу» бизнес-приложений, поэтому разработчики могут сосредоточиться на бизнес-логике без лишних настроек в зависимости от среды исполнения.

Spring MVC модуль отлично подходит для реализации RESTFul web-сервиса. В основе модуля Spring MVC лежит стандарт MVC, который состоит из трех компонентов:

- Models — объекты на которую возлагается бизнес логика;
- Views — представление данных (html, jsp, json, xml и т.д.);
- Controller — своего рода мост для отображения на views того, что происходит в models.

Задача Spring Data состоит в том, чтобы предоставить знакомую и согласованную модель программирования на основе Spring для доступа к данным, сохраняя при этом особые черты базового хранилища данных.

Spring Security — это модуль, который сфокусирован на обеспечение как аутентификации, так и авторизации в Java-приложениях. Как и все Spring проекты, настоящая сила Spring Security в том, что он может быть легко дополнен нужным функционалом.

Spring Boot — инструмент для упрощения процесса конфигурации Spring приложений. Он не является средством автоматической генерации кода, а представляет собой плагин для системы автоматизации сборки проектов таких как Maven или Gradle.

HTML (Hypertext Markup Language) — это код, который используется для структурирования и отображения веб-страницы и её контента. Например, контент может быть структурирован внутри множества параграфов, маркированных списков или с использованием изображений и таблиц данных. HTML состоит из ряда элементов, которые используются, чтобы вкладывать или оборачивать различные части контента, чтобы заставить контент отображаться или действовать определённым образом.

Thymeleaf — современный серверный механизм Java-шаблонов для веб- и автономных сред, способный обрабатывать HTML, XML, JavaScript, CSS и даже простой текст.

Docker — это открытая платформа для разработки приложений, созданная для поддержки DevOps и разработчиков. Используя Docker, разработчики могут легко создавать, упаковывать, доставлять и запускать приложения в виде легких, портативных, самодостаточных контейнеров, которые могут работать практически где угодно.

Как документо-ориентированная СУБД MongoDB — это довольно-таки обобщённое NoSQL решение. Её так же можно рассматривать как альтернативу реляционным СУБД. Подобно реляционным СУБД, она также может выигрышно дополняться более специализированными NoSQL решениями.

Java — язык программирования общего назначения. Относится к объектно-ориентированным языкам программирования, к языкам с сильной типизацией. Создатели реализовали принцип WORA: write once, run

anywhere или «пиши один раз, запускай везде». Это значит, что написанное на Java приложение можно запустить на любой платформе, если на ней установлена среда исполнения Java (JRE, Java Runtime Environment).

Maven был выбран в качестве системы автоматической сборки проекта серверной части и панели администратора. Это инструмент для сборки Java проекта: компиляции, создания jar, создания дистрибутива программы, генерации документации. Простые проекты можно собрать в командной строке. Maven используется для упрощения написания bat/sh скриптов, которые зависят от платформы.

Во второй главе описывается процесс разработки интрент-магазина по продаже электронной техники. Приводится описание архитектуры приложения, разработка базы данных, описание страниц и реализуемого функционала на них. Также описана настройка конфигурации готового приложения.

Разработка производилась на языке программирования Java с использованием Spring Boot и необходимых библиотек.

Первым делом необходимо подключить MongoDB. Это можно сделать напрямую, установив MongoDB на компьютер, но в таком случае нужно будет вручную отслеживать состояние БД и также вручную отслеживать и менять версию СУБД на более актуальную. Поэтому для большего удобства было решено поставить MongoDB через платформу Docker.

В предыдущей работе в базе данных MongoDB были созданы коллекции:

- *product* — набор продуктов
- *receiptSale* — набор чеков
- *productSale* — набор позиций товаров в чеке

Для текущей работы, помимо вышеупомянутых, необходимо создать следующие коллекции:

- *user* — набор пользователей
- *contactMessage* — набор сообщений в поддержку

Первым делом, перед началом работы со Spring Boot необходимо указать основные конфигурации в файле *application.yaml*, где указывается адрес базы данных и её название, а так же стартовый пользователь с ролью администратора.

Для обеспечении конфиденциальности данных хорошо подходит Spring Security. На сервере будет храниться объект сессии для каждого бизнес пользователя, который будет содержать всю необходимую информацию. А на стороне клиента браузер будет хранить cookie, которые будут являться идентификатором на сессию бизнес пользователя. Для настройки Spring Security был написан класс `WebSecurityConfig`

Далее необходимо реализовать уже функционал самого интернет-магазина:

1. Главная страница

   Вывод страницы реализован в контроллере `String index(Model model)`, вызываемый `HTTP GET` запросом «/» или «`/main`». В нём в модель добавляется список товаров отсортированных по полулярности.

   Если нажать на категорию товара слева на странице, то на экран выведутся товары только выбранной категории. Контроллер с сортировкой вызывается с помощью `HTTP GET` запроса «`/pruduct?productType=`», где в `productType` указывается категория товара.

   Также на данной странице можно произвести поиск товаров. Поиск вызывается с помощью `HTTP POST` запроса «`/search?search=`», где в `search` указывается введённый в право-верхнем углу текст.

2. Регистрация

   Вывод страницы регистрации реализован в контроллере `String newUser(Model model)`, вызываемый `HTTP GET` запросом «`/logUp`», после нажатая на раздел «Зарегистрироваться»:

   После корректного заполнения необходимых полей и нажатия на кнопку «Зарегистрироваться» произойдёт вызов контроллера `String saveUser(UserLogUp userLogUp, Model model)` с помощью `HTTP POST` запроса «`/product/new`», сохраняющего нового пользователя в базе данных с ролью «CLIENT».

3. Авторизация

   При нажатии на раздел «Профиль», произойдёт вызов `HTTP GET` запроса «`/login`», для которого реализован контроллер `String login(String error, Model model)`.

В случае, если пользователь авторизован, то для него откроется страница корзины. Если же покупатель ещё не авторизовался, то произойдёт переход на страницу авторизации. Так же на страницу подаётся сообщение об ошибке, если это была не первая попытка авторизации пользователя.

После ввода логина и пароля и нажатия на кнопку «Войти», произойдёт вызов встроенного в Spring Security контроллера с помощью `HTTP POST` запроса «`/auth`».

После получения данных покупателя Spring Security проверяет корректность пароля по хеш ключу. Если пароль оказался не корректным, то пользователя возвращает на страницу авторизации и выводит сообщение о неверном логине или пароле.

4. Данные о товаре

   Вывод страницы с данными о товаре реализован в контроллере `String details(String ean, Model model)`, который вызывается `HTTP GET` запросом «`/product/details?ean=`», после нажатая на товар в списке товаров. В нём происходит поиск товара в базе данных по его EAN. После чего в html коде благодаря thymeleaf можно взять данные о выбранном товаре и вывести их на экран.

   На странице показаны следующие данные о товаре:

   - изображение товара
   - название товара
   - описание товара
   - изначальная цена
   - цена с учётом скидки, если скидка имеется

   Так же на данной странице можно добавить товар в корзину, нажав на кнопку «Добавить». Стоит ещё отметить, что тот же функционал вызывается, если нажать на изображение корзины в кратком описании товара на главной странице.

   Для добавления товара в корзину выполняется контроллер `addToBusket(String ean, HttpServletRequest servletRequest)`, вызываемый с помощью `HTTP GET` запроса «`/product/addToBasket?ean=`». В нём сначала осуществляется поиск

необходимого товара в базе данных по его EAN, после найденный товар добавляется в объект класса `Basket`.

5. Корзина

   Как было упомянуто выше, при нажатии на раздел «Профиль», а так же при нажатии на изображение в окне корзины произойдёт переход на страницу корзины. Для открытия данной страницы происходит вызов `HTTP GET` запроса «`/basket`», для которого реализован контроллер `basketPage(HttpServletRequest servletRequest, Model model)`. В нём из локального хранилища достаётся объект класса `Basket`. После чего в html коде благодаря thymeleaf можно взять данные о добавленных в корзину товарах и вывести их на экран.

   На странице представлена краткая информация о товаре и общая стоимость покупки. Товар можно убрать из корзины, нажав на кнопку «Удалить» у товара.

   Удаление товара выполняется в контроллере `String deleteProduct(String ean, HttpServletRequest servletRequest)` после вызова `HTTP GET` запроса «`/basket/deleteProduct?ean=`». В нём из локального хранилища берётся объект класса `Basket`. Затем из него удаляется выбранный товар, после чего объект пересохраняется в локальном хранилище.

   Так же на данной странице можно произвести покупку всех выбранных товаров. При нажатии на кнопку «оплатить», произойдёт вызов `HTTP GET` запроса «`/basket/payment`», для которого реализован контроллер `String payment(HttpServletRequest servletRequest)`.

   Контроллер выполняет следующий порядок действий:

   (a) Получение из локального хранилища данных о товарах в корзине;
   (b) Если корзина пуста, то работа контроллера прекращается;
   (c) Создаётся объект класса `ReceiptSale`;
   (d) Объект заполняется данными о чеке;
   (e) Затем для каждого товара:
      i. Создаётся объект класса `ProductSale`;
      ii. Объект заполняется данными о товаре и его позиции в чеке;
      iii. Происходит сохранение позиции товара в чеке в базу данных;

iv. Обновляется количество продаж у товара в базе данных.

(f) Происходит сохранение чека в базу данных;

(g) Очищается корзина в локальном хранилище;

(h) Происходит переход на главную страницу.

6. Помощь

Если у пользователя возникли какие-то проблемы или предложения, он поможет их отправить через раздел «Помощь». Открытие страницы помощи реализовано в контроллере `String contactPage()`, вызываемом с помощью `HTTP GET` запроса «`/contact`».

На странице пользователь указывает:

- Имя
- email, на который пользователь хочет получить ответ
- телефон, если пользователь не хочет указывать email
- сообщение с вопросом или предложением

После заполнения необходимых полей и нажатия на кнопку «Отправить» произойдёт вызов контроллера `String send(ContactMessage contactMessage)` с помощью `HTTP POST` запроса «`/contact/send`», сохраняющего сообщение в базе данных.

7. Добавление товара

Если авторизоваться в магазине пользователем с ролью «ADMIN», то в верхней панели меню появиться специальный раздел для администраторов «Добавить товар». Открытие страницы добавления товара реализовано в контроллере `creatingProductPage()`, который вызывается с помощью `HTTP GET` запроса «`/product/creatingPage`».

На странице пользователь указывает:

- EAN товара
- название товара
- описание товара
- цену, без учёта скидок
- процент скидки
- тип товара
- гарантийный период (в месяцах)
- изображение товара

После заполнения полей и нажатия на кнопку «Отправить» произойдёт вызов контроллера `String newProduct(Product product, @RequestParam("file") MultipartFile file)` с помощью `HTTP POST` запроса «`/product/new`», сохраняющий продукт в базе данных:

Заключение. В ходе данной магистерской работы были выполнены следующие задачи:

- рассмотрены особенности электронной коммерции;
- рассмотрены особенности NOSQL баз данных;
- разработан основной концепт интернет-магазина с дизайном
- обеспечена система безопасности авторизации;
- реализован функционал интернет-магазина

Итогом работы является интернет-магазин по продаже электронной техники, разработанный с использованием фреймворка Sring на основе нереляционной базы данных MongoBD