Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

(СибГУТИ)

09.03.01 Информатика и вычислительная техника

код и наименование направления подготовки

**ОТЧЕТ**

по производственной практике

по направлению 09.03.01 «Информатика и вычислительная техника»,

направленность (профиль) – «Программное обеспечение средств вычислительной техники и автоматизированных систем», квалификация – бакалавр,

программа академического бакалавриата,

форма обучения – очная, год начала подготовки (по учебному плану) – 2017

|  |  |  |
| --- | --- | --- |
| Выполнил:  студент гр. ИП-712  «11» июля 2020 г. | \_\_\_\_\_\_\_\_\_\_\_\_ | /Алексеев С.В./ |
| Оценка «\_\_\_\_\_\_\_\_\_\_\_\_\_» |  |  |
| Руководитель практики от университета  доцент Кафедры ПМиК  «11» июля 2020 г. | \_\_\_\_\_\_\_\_\_\_\_\_ | /Янченко Е.В./ |

Новосибирск 2020

ПЛАН-ГРАФИК ПРОВЕДЕНИЯ ПРОИЗВОДСТВЕННОЙ ПРАКТИКИ

Тип практики: производственная практика по получению профессиональных умений и опыта профессиональной деятельности

Способ проведения практики: стационарная

Форма проведения практики: дискретно по периодам проведения практики

Тема: Разработка интернет-магазина на языке Джава с помощью фреймворка Spring.

Содержание практики

|  |  |
| --- | --- |
| Наименование видов деятельности | Дата (начало – окончание) |
| 1. Общее ознакомление со структурным подразделением предприятия, вводный инструктаж по технике безопасности | 03.02.2020–15.02.2020 |
| 2. Выдача задания на практику, деление студентов на группы (если необходимо), определение конкретной индивидуальной темы, формирование плана работ | 17.02.2020–22.02.2020 |
| 3. Работа с библиотечными фондами структурного подразделения или предприятия, сбор и анализ материалов по теме практики | 24.02.2020–21.03.2020 |
| 4. Выполнение работ в соответствии с составленным планом:  – Изучение языка основ джава по книге Хорстманна “Java Core”.  – Изучение фреймворка Spring по видео-роликам, книгам.  – Разработка приложения. | 23.03.2020 – 28.03.2020  06.04.2020 - 30.05.2020 |
| 5. Анализ полученных результатов и произведенной работы | 01.06.2020–06.06.2020 |
| 6. Составление отчета по практике, защита отчета | 29.06.2020–11.07.2020 |

Согласовано:

|  |  |  |
| --- | --- | --- |
| Руководитель практики от университета  доцент Кафедры ПМиК | \_\_\_\_\_\_\_\_\_\_\_\_ | /Янченко Е.В./ |

Оглавление

[«Сибирский государственный университет телекоммуникаций и информатики» 1](#_Toc43215280)

[(СибГУТИ) 1](#_Toc43215281)

[**ОТЧЕТ** 1](#_Toc43215282)

[Новосибирск 2020 1](#_Toc43215283)

[ЗАДАНИЕ НА ПРАКТИКУ 3](#_Toc43215284)

[ВВЕДЕНИЕ 3](#_Toc43215285)

[основная часть 4](#_Toc43215286)

[ЗАКЛЮЧЕНИЕ 10](#_Toc43215287)

[СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 11](#_Toc43215288)

[ПРИЛОЖЕНИЯ 12](#_Toc43215289)

# ЗАДАНИЕ НА ПРАКТИКУ

Написать интернет-магазин на языке джава, изучив необходимые для этого технологии.

# ВВЕДЕНИЕ

*Разработка приложений становится всё более высокоуровневой. В итоге появляются надстройки над языками программирования, такие, как фреймворк Спринг. Огромное число библиотек, необходимых для работы веб-приложения подключить вручную очень сложно. Объединить объекты друг с другом также бывает сложно. Также большому сложному приложению нужна структура, позволяющая выстроить взаимосвязи между частями программы по определённым правилам, чтобы развитие и поддержка проекта были возможны с привлечением новых разработчиков. Всё это предлагается решить с помощью Спринг.*

# основная часть

Первым делом изучались компоненты, необходимые для работы приложения: сам язык Джава, бины, аннотации, перечисления, лямбда-выражения.

Изучал среду разработки Spring Tool Suite.

Также изучался Спринг, система сборки Maven, зависимости, контроллеры, Spring MVC – концепция контроллера, класса, принимающего запросы и отвечающего какой-либо информацией. У веб-приложения в частности котроллер наполняет веб страницу информацией.

Шаги на практике:

1. Создал новый проект Spring Starter Project, добавил нужные зависимости. Вначале это были DevTools(включает автоматический перезапуск приложения, когда код меняется, перезапуск браузера, когда меняются ресурсы, автоматическое отключение кэшей шаблонов, Встроенная консоль базы данных H2 на случай использования БД), Thymeleaf, Web. Данные зависимости позволяют выводить простые представления в виде веб-страниц.

Mvnw, mvnw.cmd – скрипты обёртки Maven, их можно использовать для сборки проетка даже если Мавен не установлен на компьютере.

Pom.xml – спецификация для сборки проекта Мавен.

PancakeCloudAppliction – главный класс объединяющий весь проект.

Папка resources/static – для любых статичных(не меняющихся с течением времени работы программы ресурсов).

Папка resources/templates – для объектов, которые будут отображать нужное содержимое в браузере.

Нужно пнимать, что Spring Boot не требует установки сервера, он сам его устанавливет, в моём случае Tomcat.

1. Создаю класс Pancake с айди, полем имени, даты, списком ингредиентов.

Создаю доменные объекты Джава(обладающие определённым набором качеств). Для этого определяю класс Ingredient. При этом доменным его явно не делаю. Для этого использую библиотеку lombok. Для этого прописываю соостветствующие зависимости в pom.xml, импортирую аннотации этой библиотеки(@Data, @RequireArgConstructor), создающие автоматически все конструкторы, геттеры и сеттеры.

Создаю design.html для отображения меню начинок для блинов. Вся работа основана на движке Thymeleaf:

<**h3**>Выберите тип блина:</**h3**>

<**div** th:each=*"ingredient : ${wrap}"*><!-- для каждого элемента с энамом типа wrap: создаю чекбокс -->

<**input** name=*"ingredients"* type=*"checkbox"* th:value=*"${ingredient.id}"* /><!-- Высвечиваю имя Здесь ingredients - поле класса

Pancake.-->

<**span** th:text=*"${ingredient.name}"*>INGREDIENT</**span**><**br**/><!-- INGREDIENT - это просто плейсхолдер. Когда шаблон передан в html, тело(плейсхолдер) элемента span будет заменено значением атрибута запроса сервлета, чей ключ ingredient.name. span нужен для создания ярлыка.

1. Репозитории в спринге – специальные коллекции, которые могут быть легко портированы в большое число видов баз данных.

Создал классы и репозитории для хранения соответствующих объектов: IngredientRepository, OrderRepository, PancakeRepository, UserRepository.

package pancake.data;

/\*CrudRepository (Create, Read, Update, Destroy)

Объявляет дюжину методов, которые поддерживают вышеупомянутые операции.

\*

\*/

import org.springframework.data.repository.CrudRepository;

import pancake.Ingredient;

public interface IngredientRepository

extends CrudRepository<Ingredient, String> {

}

1. Создал формы регистрации и авторизации: RegistrationController, RegistrationForm, SecurityConfig, UserRepositoryUserDetailsService.

package pancake.security;

import org.springframework.security.crypto.password.PasswordEncoder;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import pancake.data.UserRepository;

@Controller//эта аннотация будет отсканирована с помощью @ComponentScan

@RequestMapping("/register")//теперь адреса всех методов этого класса должны иметь префикс /register

public class RegistrationController {

private UserRepository userRepo;

private PasswordEncoder passwordEncoder;

public RegistrationController(

UserRepository userRepo, PasswordEncoder passwordEncoder) {

this.userRepo = userRepo;

this.passwordEncoder = passwordEncoder;

}

@GetMapping //нет параметров=> путь по умолчанию. Данный метод д.б. вызван, когда кто-то вызывает метод GET на пути /register

public String registerForm() {

return "registration";

}

@PostMapping

public String processRegistration(RegistrationForm form) {

userRepo.save(form.toUser(passwordEncoder));

return "redirect:/login";

}

}

1. Создал Контроллеры в связке с .html файлами: DesignPancakeController, DiscountController, IngredientByIdConverter, OrderController, OrderProps, WebConfig, design.html, discount.html. discountList.html, home.html, login.html, orderForm.html, orderList.html, registration.html.

package pancake.web;

import java.security.Principal;

import java.util.ArrayList;

import java.util.List;

import java.util.stream.Collectors;

import javax.validation.Valid;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.validation.Errors;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.SessionAttributes;

import pancake.Ingredient;

import pancake.Ingredient.Type;

import pancake.Order;

import pancake.Pancake;

import pancake.User;

import pancake.data.IngredientRepository;

import pancake.data.PancakeRepository;

import pancake.data.UserRepository;

@Controller//сообщает, что данный класс является контроллером. Эта аннотация наследуется от @Component,

//поэтому @ComponentScan работает с @Controller так же, как и просто с @Component. Когда @ComponentScan видит

//аннотацию @Controller, она создаёт бин из этого помеченного java класса. Т.е. @Controller - тот же @Component,

//но с дополнительными возможностями. Внутри контроллеров находятся методы. Обычно, (но не всегда) каждый метод

//соответсвует одному URL. Обычно(но не всегда), методы возвращают строку (String) - название представления,

//которое надо показать пользователю. Эта аннотация особо много и не делает. Её основнное назначение -

//идентифицировать класс как компонент для сканирования компонентов. Т.о. результатом сканирования будет создание

//объекта bean в контексте приложения Спринг.

/\*

\* Если код такой:

@Controller

@RequestMapping("/people")

public class PersonController{

То здесь для обращения к любому методу класса сначала надо добавлять префикс /people

\*/

@RequestMapping("/design")//для работы этого класса адрес должен содержать /design.

@SessionAttributes("order")

public class DesignPancakeController {

private final IngredientRepository ingredientRepo;

private PancakeRepository tacoRepo;

private UserRepository userRepo;

@Autowired

public DesignPancakeController(

IngredientRepository ingredientRepo,

PancakeRepository tacoRepo,

UserRepository userRepo) {

this.ingredientRepo = ingredientRepo;

this.tacoRepo = tacoRepo;

this.userRepo = userRepo;

}

@ModelAttribute(name = "order")

public Order order() {

return new Order();

}

@ModelAttribute(name = "design")

public Pancake design() {

return new Pancake();

}

@GetMapping

public String showDesignForm(Model model, Principal principal) {

List<Ingredient> ingredients = new ArrayList<>();

ingredientRepo.findAll().forEach(i -> ingredients.add(i));

Type[] types = Ingredient.Type.values();

for (Type type : types) {

model.addAttribute(type.toString().toLowerCase(),

filterByType(ingredients, type));

}

String username = principal.getName();

User user = userRepo.findByUsername(username);

model.addAttribute("user", user);

return "design";

}

@PostMapping

public String processDesign(

@Valid Pancake taco, Errors errors,

@ModelAttribute Order order) {

if (errors.hasErrors()) {

return "design";

}

Pancake saved = tacoRepo.save(taco);

order.addDesign(saved);

return "redirect:/orders/current";//Реагирую на нажатие кнопки Подтвердите в меню дизайна

}

private List<Ingredient> filterByType(

List<Ingredient> ingredients, Type type) {

return ingredients

.stream()

.filter(x -> x.getType().equals(type))

.collect(Collectors.toList());

}

}

# ЗАКЛЮЧЕНИЕ

В целом получены хорошие навыки веб-разработки на джава. Проект выглядит сыровато, но суть обучения была в изучении бэк-энда, что я успешно выполнил. В дальнейшем буду продолжать в том же духе. Все новые технологии можно изучить, нужна просто дисциплинированность и методичность в обучении.

# СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

<http://www.iprbookshop.ru/epd-reader?publicationId=45047>

Хорстманн К. Java Библиотека профессионала

Beginning Spring 5 Joseph B. Ottinger

<https://www.youtube.com/watch?v=FyZFK4LBjj0>

<https://www.youtube.com/watch?v=msXL2oDexqw&list=PLmbC-xnvykcghOSOJ1ZF6ja3aOgZAgaMO>

# ПРИЛОЖЕНИЯ

Листинг можно посмотреть в приложенных файлах.

Скриншоты:

