**Темы, рассматриваемые в курсе, и задачи, которые будут выполнены в рамках пройденных тем\***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ПП модуля** | | **Тема модуля** | | **Задачи в рамках курсового проекта** | | **Краткое (или не очень) описание** |
| **Модуль 1** | | | | | | |
| 2 | | Spring Boot 3.x и его возможности. | | Создать работающее Web приложение, моделирующую работу автомойки | | Включает в себя Starter пакеты (например контекст Spring)+ дополнительно можно включить в них сервер Tomcat и обработчик JSON строк. В целом все, что нужно для запуска web приложения |
| 3. | | Создание и запуск Spring Boot приложения | |
| 4. | | Аннотации и XML конфигурации. | | Используя аннотации будем создавать объекты в контексте Spring для управления приложением. | | Работа будет содержать DAO объекты с сервисными методами (методами бизнес-логики) для управления приложением такие, как findAll, findById и т.д. |
| **Модуль 2** | | | | | | |
| 1. | | Многопоточность в Spring. | | «*Бронь, пользователь которой не явился заблаговременно, автоматически аннулировалась, если она не была аннулирована вручную*»  «*Бронь должна подтверждаться в течении 15 минут, по ссылке, в противном случае бронь должна автоматически аннулироваться.*» | | Задачи аннулирования брони должны запускаться по определенному времени. Это задача может быть решена через планировщик задач. |
| 2. | | Планировщик Spring. | |
| 3 | | Java Mail Sender | | *«Функционал отправки письма на почту следует имитировать выводом в консоль»* | | Можно попробовать создать реальный функционал отправки письма, создав пару тройку почтовых ящиков |
| 4 | | Логирование в Spring. | | Здесь четких требований по логированию и уровню нет | | Можно основные действия по типу регистрация пользователя или запись на мойку записывать в виде логов в контексте Spring, а не JavaCore+консоль. Также логировать кому какое сообщение отправлено |
| **Модуль 3** | | | | | | |
| 1. | | JDBC API и JdbcTemplate для работы с реляционными (SQL) БД. | | В рамках курсовой работы будет создана реляционная БД с сущностями, необходимыми для выполнения функционала (таблица услуг, таблица пользователей и т.д.). | | Для работы с реляционной БД будет использован PostgeSQL. Как прикрутить нереляционную пока не понятно, т.к. и особого представлегния работы с ней нет. |
| 2. | | Spring Data JPA для работы с реляционными (SQL) и нереляционными (NoSQL) БД. | |
| **Модуль 4** | | | | | | |
| 1. | Spring Web и разработка Rest приложений. | | Предполагается написание контроллеров, которые бы отдавали объекты по средствам HTTP запросов | | Например есть задача: вывести список услуг, предоставляемых автомойкой. Результат работы метода должен выглядеть следующем образом:  GET запрос, результат которого возвращает JSON строку в браузере. | |
| 2. | Библиотека Jackson. | |
| 3. | Валидация данных и обработка ошибок | | Все JavaBeans объекты буду провалидированы через Spring. | |  | |
| 4. | Spring Security | | *«функционал идентификации/аутентификации пользователей следует исполнить, используя Spring Security, сессии должны быть Stateless, например, через использование JWT токена вместо JSESSION. Реализовать PasswordEncoder»* | |  | |
| **Модуль 5** | | | | | | |
| 1. | Реактивное программирование: спецификация Reactive Streams, Reactor, Mongo и Flux | | В MongoDB будут храниться шаблоны сообщений, которые будут отправляться пользователям. Отправка сообщений будет асинхронная. | | Для отправки сообщений будет реализовано взаимодействие микросервиса отправки сообщений с первым основным микросервисом, чтобы дергать юзеров для отправки сообщений | |
| 2 | Spring WebFlux и архитектура микросервисов. | | В курсовой работе будет реализовано 2 микросервиса:   1. Будет управлять взаимодействием пользователя с сервисов автомойки 2. Будет заниматься рассылкой сообщений | | 2 этих сервиса имеют слабую связь, поэтому не должны влиять на работу друг-другу | |
| 3 | Gateway API, балансировщик нагрузки, фильтрация запросов. | |  | |  | |
| 4 | Брокер сообщений: ActiveMQ и Kafka. | | Для взаимодействия двух микросервисов будет использован брокер сообщений, но пока имеется очень слабое представление на этот счет | |  | |
| 5 | Работа с БД из Spring WebFlux | |  | |  | |
| 6 | Spring Security в Spring WebFlux | |  | |  | |
| **Модуль 6** | | | | | | |
| 1 | Знакомство со Spring Integration. | | *Пока не очень понятно, для чего это нужно* | |  | |
| 2 | Подготовка к продакшн и деплой приложений. | |  | |  | |

***\****[*Курс "Java: Spring Framework Developer" (itmo.ru)*](https://cdpo.itmo.ru/spring?ysclid=lx21r4c3ze411734963)

**План работ на курсовую работу с учетом этапов освоение программы курса**

|  |  |
| --- | --- |
| **№ шага** | **Описание** |
| 1 | Выделение сущностей из ТЗ, конструирование БД |
| 2 | Создание Entity классов на базе имеющихся сущностей, создание отношений между сущностями |
| 3 | Подключение Spring Data, настройка отображения сущностей из БД на Java объекты. Верификация схемы из БД с Entity классами |
| 4 | Задание ограничений на свойства Java Beans объекты |
| 5 | Подключение Spring Web, написание нескольких CRUD методов, требуемых по заданию, попытки создание первых Bean объектов/методов (инъекция заивисимостей) и попытки получения результатов их работы через HTML запросы |
| 6 | Подключение инструмента логирования через Spring, попытки добавить некоторые уровни логирования и добавления записи в файл. Логирование должно быть реализовано в новом потоке (умозрительно). Здесь понадобятся знания из многопоточности в Spring в core – это бал бы обычный Thread. |
| 7 | Подключение TaskSheduler, попытка реализовать функционал подтверждения записи и прибытия на автомойку заранее |
| 8 | Создание бОльшей части DAO классов и превращение их в Bean сущности для реализации бизнес логики |
| 9 | Подключение Spring Validator и реализация части функционала системы (именно на систему, как отдельную сущность ложится ответственность по валидации данных) |
| 10 | Попытки реализации функционала аунтификации через Spring Security (идентификация вроде бы должна реализовываться достаточно просто через добавление данных в БД) |
| 11 | Попытки создания инструмента рассылки сообщений через Java Mail Sender. В качестве бонуса, можно подтверждение учетной записи сделать через email. Это значительно облегчает валидацию почты, т.к. по факту ее вообще не нужно валидировать. Если почта написана некорректно, то и анонимный пользователь не создаст учетную запись в БД. Также функционал восстановления пароля тоже может быть реализован через отправку сообщения и перехода по ссылке. |
| 1 – 11 | Попутное написание тестов для тестирования создаваемого функционала |
| 12 | При наличии свободного времени можно еще что-нибудь прикрутить по мере изучения тем в курсе. |