ЦЕНТРАЛЬНЫЙ УНИВЕРСИТЕТ

УТВЕРЖДЕНА

Приказом Ректора АНО ВО
«Центральный университет»
Е.В. Ивашкевич
от «26» июня 2025 г. № 0626.32

Рабочая программа дисциплины (модуля) «Java Spring (Разработка веб-приложений на Java с использованием Spring)» дополнительной профессиональной программы – программы профессиональной переподготовки «Академия data science»

Траектория: Продуктовый менеджмент

Москва
2025

Электронный документ

Содержание

1. Краткая характеристика дисциплины (модуля)

Изучение дисциплины (модуля) «Java Spring (Разработка веб-приложений на Java с использованием Spring)» имеет важное значение для слушателей, поскольку оно позволяет им создавать сложные веб-приложения с высокими требованиями к производительности, масштабируемости и безопасности. Кроме того, знания Spring и Java являются одними из наиболее востребованных навыков на рынке труда, что делает их изучение перспективным и актуальным для будущей карьеры в области разработки веб-приложений.

Цель изучения дисциплины (модуля): формирование у слушателей навыков и знаний по разработке веб-приложений на Java с использованием фреймворка Spring, работа с базами данных, управление зависимостями и конфигурацией приложения.

Задачи изучения дисциплины (модуля):

- изучить основные принципы сетевого взаимодействия и концепции неблокирующего ввода-вывода в Java для создания производительных приложений;
- освоить методы работы с реляционными базами данных на разных уровнях абстракции с использованием JDBC, Spring JDBC, Jdbi и jOOQ;
- познакомиться с принципами организации обмена сообщениями и интеграции через Spring JMS и Spring AMQP;
- научиться проектировать и реализовывать REST и gRPC сервисы на базе Spring Boot с настройкой и оптимизацией логирования;
- развить навыки настройки и интеграции систем обмена сообщениями (ActiveMQ, RabbitMQ, Kafka) в составе Spring-приложений.

В результате освоения дисциплины (модуля) обучающийся должен:

знать:

- основные принципы работы сетей, а также концепции неблокирующего ввода-вывода (Non Blocking IO) в Java;
- способы взаимодействия с реляционными БД с помощью JDBC, Spring JDBC, а также современных библиотек Jdbi и jOOQ;
- принципы работы с очередями сообщений, интеграцию через Spring JMS/Spring AMQP.

уметь:

- проектировать и реализовывать REST и gRPC сервисы на базе Spring Boot, настраивать и оптимизировать логирование;
- создавать взаимодействие приложений с реляционными БД на разных уровнях абстракции — от «сырых» JDBC-запросов до декларативного маппинга с помощью jOOQ;
- настраивать и интегрировать системы обмена сообщениями и событий (ActiveMQ, RabbitMQ, Kafka) в составе Spring-приложений.

владеть:

- навыками настройки веб-приложения с помощью Spring Boot и реализации REST/gRPC сервисов;
- навыками настройки подключения и взаимодействия с различными реляционными СУБД, выбора подходящего способа работы с БД;

навыками организации обмена сообщениями через очереди/брокеры, интеграции приложения с помощью Spring JMS/Spring AMQP

2. Тематический план

№ п/п	Наименование раздела дисциплины (модуля)	Трудоемкость, академические часы				ТКУ (текущий контроль успеваемости)
		Очная форма				
		Аудиторная работа		Контроль	Самостоятельная работа	
		Лекции	Семинары (Практические занятия)			
1	Основы сетевого программирования	2	2		13	Домашнее задание
2	Основы многопоточности	4	4		13	Домашнее задание
3	Введение в Spring	6	6		13	Домашнее задание
4	Взаимодействие с сетью	6	6		13	Домашнее задание
5	Интеграция с базами данных	4	4		14	Домашнее задание
6	Взаимодействие с очередями сообщений	5	6		14	Домашнее задание
7	Продвинутые возможности Spring	6	6		14	Домашнее задание
	Зачет			***4***		
Итого:		***33***	***34***	***4***	***94***	
Объем дисциплины (модуля) (в ак. ч.)		***165***				

3. Содержание дисциплины (модуля)

№п/п	Наименование раздела дисциплины (модуля)	Содержание дисциплины (модуля) по темам
1	Основы сетевого программирования	Клиент-серверная архитектура. Реализация простейшего сервера
2	Основы многопоточности	Потоки исполнения (threads). Многопоточный сервер Thread pool и повторное использование потоков
3	Введение в Spring	Инверсия управления, внедрение зависимостей, введение в Spring Жизненный цикл бина, типы бинов Расширенные возможности управления бинами. Properties. Profiles
4	Взаимодействие с сетью	Spring Boot” HTTP, Spring Web. Rest Controller Основные протоколы интеграции по сети: REST client, gRPC (protobuf)
5	Интеграция с базами данных	Интеграция с реляционными БД: JDBC, Spring JDBC Spring JDBC Core Classes. Transactions. DB migration
6	Взаимодействие с очередями сообщений	JDBI, jOOQ, Testcontainersr Интеграция с очередями сообщений: JMS, AMQP
7	Продвинутые возможности Spring	Интеграция с Apache Kafka Spring Advanced: Production-ready, Security, AOP

4. Учебно-методическое обеспечение

Университет располагает полным набором лицензионного и свободно распространяемого программного обеспечения, включая продукты отечественного производства.

Каждый слушатель в течение всего периода обучения получает индивидуальный неограниченный доступ к электронно-библиотечной системе и электронной информационно-образовательной среде университета. Эти системы предоставляют возможность доступа к ресурсам из любой точки, где есть подключение к сети Интернет, как на территории университета, так и за его пределами.

Слушателям обеспечен удаленный доступ к современным профессиональным базам данных и информационным справочным системам.

Основная литература:

1. Кубенский, А. А. Функциональное программирование : учебник и практикум для вузов / А. А. Кубенский. — Москва : Издательство Юрайт, 2025. — 348 с. — (Высшее образование). — ISBN 978-5-9916-9242-7. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/561074.

2. Тузовский, А. Ф. Проектирование и разработка web-приложений : учебник для вузов / А. Ф. Тузовский. — Москва : Издательство Юрайт, 2025. — 219 с. — (Высшее образование). — ISBN 978-5-534-16300-1. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/561176.

Дополнительная литература:

1. Зыков, С. В. Программирование : учебник и практикум для вузов / С. В. Зыков. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2025. — 285 с. — (Высшее образование). — ISBN 978-5-534-16031-4. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/560815.

5. Материально-техническое обеспечение

Университет располагает материально-технической базой, соответствующей действующим противопожарным правилам и нормам и обеспечивающей проведение всех видов дисциплинарной и междисциплинарной подготовки, практической и научно-исследовательской работ обучающихся, предусмотренных учебным планом.

Помещения, которые представляют собой учебные аудитории для проведения занятий лекционного типа, занятий семинарского (практического) типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, а также помещения для самостоятельной работы и помещения для хранения и профилактического обслуживания учебного оборудования. Помещения укомплектованы специализированной мебелью и техническими средствами обучения, служащими для представления учебной информации большой аудитории.

Изучение дисциплины (модуля) обеспечивается в учебных аудиториях, оснащенных:

— столами и стульями;

— компьютерной техникой;

— механическими калькуляторами;

— специализированным оборудованием, включая демонстрационное оборудование.

Помещения для самостоятельной работы обучающихся, в том числе приспособленные для использования инвалидами и лицами с ограниченными возможностями здоровья, оснащены компьютерной техникой с возможностью подключения к сети «Интернет» и обеспечением доступа к в электронную информационно-образовательную среду Университета.

Обучающимся предоставляется доступ (в том числе удаленный) к ресурсам информационно-телекоммуникационной сети «Интернет», электронным ресурсам (в том числе электронным библиотечным системам, современным профессиональным базам данных и информационным справочным системам):

№	Наименование портала (издания, курса, документа)	Ссылка
1	Катастрофы, стихийные бедствия, аварии, эпидемии. Солнечная и геомагнитная активность. /ежедневный обзор	http://www.disasters.chat.ru
2	Каталог по безопасности жизнедеятельности	http://www.eun.chat.ru
3	Научная электронная библиотека elibrary.ru библиотека	https://elibrary.ru/defaultx.asp
4	База данных для IT-специалистов	https://habr.com
5	База данных ScienceDirect	https://www.sciencedirect.com
6	Официальный сайт Министерства науки и высшего образования Российской Федерации	https://minobrnauki.gov.ru/
7	Федеральный портал «Российское образование"	https://www.edu.ru/
8	Информационная система "Единое окно доступа к образовательным ресурсам"	http://window.edu.ru/
9	Единая коллекция цифровых образовательных ресурсов	http://school-collection.edu.ru/
10	Федеральный центр информационно - образовательных ресурсов	http://fcior.edu.ru/
11	Сайт различных плагинов	https://maven.apache.org/plugins/
12	Maven central repository - хранилище библиотек и фреймворков	https://mvnrepository.com/repos/central

Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), в том числе комплект лицензионного программного обеспечения, современные профессиональные базы данных и информационные справочные системы:

Наименование ПО	Производство	Лицензионное / свободно распространяемое
Операционные системы:		
Microsoft Imagine (Windows Client, Server)	зарубежное	лицензионное
Браузеры:		
Яндекс.Браузер	отечественное	свободно распространяемое
Google Chrome	зарубежное	свободно распространяемое
Офисные приложения:		
Microsoft Imagine (Visio, OneNote)	зарубежное	лицензионное
TeXstudio	зарубежное	свободно распространяемое
Adobe Acrobat Reader	зарубежное	свободно распространяемое
Программное обеспечение для планирования и учета времени:		
Toggle app	зарубежное	свободно распространяемое
Системы управления проектами:		
Microsoft Imagine (Project)	зарубежное	лицензионное
Системы управления базами данных:		
Microsoft Imagine (SQL Server)	зарубежное	лицензионное
Системы резервного копирования (backup):		
Acronis Backup Advanced for HyperV	зарубежное	лицензионное
Справочно-правовые системы:		

КонсультантПлюс: справочно-правовая система	отечественное	лицензионное
Средства антивирусной защиты:		
Kaspersky Endpoint Security для бизнеса Стандартный Russian Edition	отечественное	лицензионное
Пакеты программных средств и библиотек:		
AutoPsy	зарубежное	свободно распространяемое
Interactive Disassembler (IDA)	зарубежное	свободно распространяемое
Системы управления библиографической информацией:		
Zotero	зарубежное	свободно распространяемое
Сервисы и службы:		
Bind	зарубежное	свободно распространяемое
Docker	зарубежное	свободно распространяемое

6. Методические и оценочные материалы

Методические указания для обучающихся по освоению дисциплины (модуля)

В процессе изучения дисциплины (модуля) «Java Spring (Разработка веб-приложений на Java с использованием Spring)» в рамках текущего контроля успеваемости используются такие виды учебной работы, как лекции, семинары, домашние задания, а также различные виды самостоятельной работы обучающихся по заданию преподавателя, направленные на развитие навыков профессиональной лексики, закрепление практических профессиональных компетенций, поощрение инициатив.

Лекция – систематическое, последовательное, монологическое изложение преподавателем учебного материала, как правило, теоретического характера.

В процессе лекций рекомендуется вести конспект лекций: кратко и схематично фиксировать основные идеи, выводы и обобщения лекции; выделять важные мысли, ключевые слова и термины. Необходимо отметить вопросы или материалы, которые вызывают затруднения, и попытаться найти ответы в рекомендованной литературе. Если разобраться в материале не удается, следует сформулировать вопрос и задать его преподавателю на консультации или во время семинарского (практического) занятия.

Семинар — это форма учебной деятельности, проводимая в учебном заведении под руководством преподавателя, где слушатели активно участвуют в обсуждениях, практических заданиях и других формах взаимодействия.

Для успешной подготовки к семинару рекомендуется заранее ознакомиться с темой занятия и основными материалами, чтобы иметь возможность активно участвовать в обсуждении. Также полезно подготовить вопросы и идеи для обсуждения, что поможет глубже понять материал и продемонстрировать заинтересованность.

Домашнее задание – набор заданий по темам недели.

При работе над домашними заданиями важно внимательно ознакомиться с требованиями и сроками выполнения. Рекомендуется разбивать задания на этапы, чтобы избежать перегрузки и лучше усвоить материал. Использовать различные источники информации, включая учебники и онлайн-ресурсы, для более глубокого понимания темы.

Самостоятельная работа – работа слушателей, направленная на углубленное изучение отдельных тем и вопросов учебной дисциплины (модуля).

В процессе самостоятельной работы слушатели взаимодействуют с рекомендованными материалами при минимальном участии преподавателя. Задачи слушателя включают работу с конспектами лекций (обработка текста), повторное изучение учебных материалов планов и тезисов ответов, изучение дополнительных тем, выполнение учебно-исследовательских заданий и другое.

Система оценивания результатов обучения по дисциплине (модулю)

Оценивание уровня учебных достижений обучающихся по дисциплине (модулю) осуществляется в виде текущего контроля успеваемости.

Промежуточная аттестация по дисциплине (модулю) осуществляется в форме ***зачета***.

Для оценивания текущего контроля успеваемости и промежуточной аттестации используется десятибалльная шкала оценивания, которая соотносится с традиционной пятибалльной шкалой следующим образом:

Десятибалльная оценка	**Пятибалльная оценка**	**Оценка за зачет**	**Общая характеристика результата обучения по дисциплине (модулю)**
10	Отлично	Зачтено	Слушатель полностью владеет знаниями, изложенными в рабочей программе, и глубоко осмысляет дисциплину (модуль). Он самостоятельно и логически последовательно отвечает на все вопросы, акцентируя внимание на наиболее важном. Умеет анализировать, сравнивать, классифицировать, обобщать, конкретизировать и систематизировать изученный материал, выделяя ключевые моменты и устанавливая причинно-следственные связи. Четко формулирует ответы, уверенно интерпретирует результаты анализов и других исследований, а также решает сложные задачи. Слушатель хорошо знаком с методами исследования, необходимыми для практической деятельности, и умеет связывать теоретические аспекты дисциплины (модуля) с практическими задачами.
9	Отлично	Зачтено	
8	Отлично	Зачтено	
7	Хорошо	Зачтено	Слушатель обладает знаниями предмета почти в полном объеме рабочей программы и самостоятельно, логически последовательно и всесторонне отвечает на все вопросы, акцентируя внимание на наиболее значимых моментах. Он умеет анализировать, сравнивать, классифицировать, обобщать, конкретизировать и систематизировать изученный материал, выделяя его ключевые аспекты и устанавливая причинно-следственные связи. Формулирует свои ответы, уверенно интерпретирует результаты анализов и других исследований, а также решает сложные ситуационные задачи. Слушатель хорошо знаком с методами
6	Хорошо	Зачтено	

Десятибалльная оценка	Пятибалльная оценка	Оценка за зачет	Общая характеристика результата обучения по дисциплине (модулю)
			исследования, необходимыми для практической деятельности, и умеет связывать теоретические аспекты предмета с практическими задачами.
5	Удовлетворительно	Зачтено	Слушатель обладает базовыми знаниями по дисциплине (модулю), но испытывает трудности при самостоятельных ответах и использует неточные формулировки. В ходе ответов он допускает ошибки, касающиеся сути вопросов. Слушатель способен решать только самые простые задачи и владеет лишь минимальным набором методов исследования.
4	Удовлетворительно	Зачтено	
3	Не сдан	Не зачтено	Слушатель не овладел обязательным минимумом знаний по предмету и не может ответить на вопросы, даже если преподаватель задает дополнительные наводящие вопросы.
2	Не сдан	Не зачтено	
1	Не сдан	Не зачтено	

Дисциплина (модуль) «Java Spring (Разработка веб-приложений на Java с использованием Spring)» оценивается следующим образом:

Активность	Вес	Описание
Домашние задания	70%	Оцениваются по критериям. Можно набрать максимум 10 баллов за каждое из заданий.
Зачет	30%	Письменная или устная работа над заданием, направленным на проверку полученных знаний и навыков по дисциплине (модулю)

Формула расчёта итоговой оценки по дисциплине (модулю) «Java Spring (Разработка веб-приложений на Java с использованием Spring)»: «0,7 × среднее за домашние задания + 0,3 × зачет».

Текущий контроль успеваемости обучающихся по дисциплине (модулю)

Примерные домашние задания

Домашнее задание

Чат

Напишите два консольных приложения: сервер и клиент, реализующие простой чат.

Условие

Серверное приложение в аргументах требует порт, через который будет происходить взаимодействие с клиентами по протоколу TCP. Клиентское -- адрес и порт сервера, имя клиента.

Клиент после подключения печатает в стандартный вывод сообщения о подключении и отключении к чату других клиентов, новые сообщения этих клиентов, а также ожидает в стандартном вводе сообщения пользователя. После получения сразу отправляет его на сервер.

Формат новых сообщений в терминале: `[mm:ss.SSS] client_name: message`, каждое с новой строки, где `mm:ss.SSS` -- серверное время сообщения (минуты, секунды, миллисекунды), `client_name` -- имя отправителя, `message` -- его сообщение.

Формат сообщений о подключении и отключении:

- `client_name joined the chat`
- `client_name left the chat`

Поддержите только однострочные сообщения. Уникальность имен клиентов поддерживать не нужно, могут совпадать.

Поддержка нескольких чатов

Поддержите возможность подключения к разным чатам. Четвертым опциональным аргументом в параметрах клиентского приложения можно передать номер чата -- положительное число. Если клиент подключается к чату `n`, то только клиентам этого чата отправляются сообщения о новом клиенте и сами сообщения клиентов. Если в параметрах приложения не указан номер чата, это означает, что клиент подключается к чату по умолчанию. Чат по умолчанию -- это отдельный чат, сообщения в котором видны только клиентам этого чата.

Смену чата у запущенного клиента поддерживать не нужно.

Критерии оценивания

- 4 балла: реализован сервер
- +3 балла: реализован клиент
- +3 балла: поддержана возможность подключения к разным чатам

Если вы реализуете только сервер, добавьте файл README.md, в котором укажите формат взаимодействия сервера и клиента.

Формат решения

Мультимодульный gradle проект, расположенный в директории `week_03/hometask`. Внутри два модуля: `server`, `client`, gradle wrapper, `settings.gradle` (или `settings.gradle.kts`) с подключение дочерних модулей.

В зависимостях проекта не должно быть сторонних библиотек кроме тестовых (написание самих тестов на ваше усмотрение).

Для поддержки запуска приложения через gradle в корневых модулях подключите плагин `application` https://docs.gradle.org/current/userguide/application_plugin.html.

В результате `cd week_03/hometask && ./gradlew server:run --args='port'` должно запуститься серверное приложение, слушающее порт `port`.

В результате `cd week_03/hometask && ./gradlew client:run --args='host:port client_name chat_id'` должно запуститься клиентское приложение, которое будет подключаться к серверу с адресом `host`, портом `port`. После успешного подключения будут приходить сообщения участников чата `chat_id`. У других участников этого чата сообщения клиента будут отображаться с именем `client_name`.

Домашнее задание

Spring Cached Proxy

Напишите библиотеку для кэширования результатов вызова методов бина.

Требования к решению

Ваша библиотека должна:

- Обеспечивать автоматическое кэширование в оперативной памяти результатов методов, аннотированных `@Cached`.
- При первом вызове метода выполнять реальный вызов.

• При последующих вызовах с теми же аргументами (равными по `Object#equals`) возвращать ранее сохранённый результат.

Обязательные компоненты

1. Аннотация `hometask.cached.annotations.Cached`
   - Поддерживает необязательные параметры:
   - `long ttl` — время жизни кэша (по умолчанию бесконечно).
   - `java.time.temporal.ChronoUnit unit` — единица измерения TTL (по умолчанию `ChronoUnit.SECONDS`).
2. Конфигурационный класс `hometask.cached.configuration.CachedConfiguration`
   - При подключении с помощью `@Import(CachedConfiguration.class)` в Spring Context должен автоматически включать механизм кэширования.

Ограничения

• Аннотация `@Cached` будет применяться только к публичным нефинальным методам с возвращаемым значением (не `void`).

• В одном бине может быть несколько `@Cached` методов. Считайте, что `@Cached` методы одного бина не вызывают друг друга.

Формат решения

• Gradle-модуль `hometask.spring:spring-cached:1.0` в директории `week_06/hometask/spring_cached`.

• MR в своем приватном форке с названием `[hometask] cached`

• Для создания прокси-классов используйте `org.springframework.cglib.proxy.Enhancer` и `org.springframework.cglib.proxy.MethodInterceptor`.

Дополнительные задачи

1. Реализация TTL (время жизни кэша) [+3 балла]

Добавьте возможность автоматического удаления устаревших кэшированных данных после заданного времени:

• При очередном вызове метода результат должен браться из кэша, если срок его хранения (TTL) не истёк, иначе должен происходить вызов реального метода с последующим сохранением в кэш.

2. Реализация `@CacheEvict` [+2 балла]

Добавьте аннотацию `hometask.cached.annotations.CacheEvict` и ее поддержку:

• Перед вызовом метода с `@CacheEvict` необходимо очистить кэш для всех `@Cached` методов в этом же бине.

• Считайте, что аннотация будет применяться только к публичным нефинальным `void` методам без параметров.

3. Покрыть код контекстными тестами

Напишите интеграционные тесты с поднятием Spring-контекста, проверяющие, что кэширование работает корректно

• Добавьте минимум 2 теста на основной функционал `@Cached` и минимум по 2 на дополнительные задачи при условии их реализации.

Критерии оценивания

• 4 баллов — реализованы `@Cached` и `CachedConfiguration`.

• +2 балла — реализована логика TTL.

• +2 балла — реализованы `@CacheEvict` и его поддержка.

• +2 балла — код покрыт контекстными тестами

Домашнее задание

Сервис для организации встреч

Разработайте REST API для управления встречами между пользователями и реализуйте его с помощью Spring Boot приложения. Все данные должны храниться в памяти в подходящих коллекциях.

Требования к API

API должно соответствовать принципам REST:

- Используйте HTTP-методы по назначению
- Возвращайте корректные HTTP-коды в ответах
- Используйте понятные URL-энпоинты
- Работайте с JSON в запросах и ответах.
- Подробнее: https://restfulapi.net/resource-naming

Создание встречи

- Встреча имеет название, организатора, список участников, дату и время начала, дата и время окончания, продолжительность.
- Поддержите формат ISO-8601 для даты и времени с оффсетом (`2020-01-02T03:04:05Z`, `2020-01-02T03:04:05+06:00`). Используйте классы из пакета `java.time`.
- Единственная информация про пользователей, с которой вы работаете, — это их `id`. Сами пользователи в приложении не хранятся и никак не валидируются.
- Поддержите валидацию параметров:
  - нельзя одновременно задать дату и время окончания встречи и ее продолжительность
  - дата и время окончания должны быть позже начала
  - встреча не должна пересекаться по времени с уже существующими встречами любого из участников
  - нельзя создавать встречи в прошлом
  - нельзя в списке участников передать организатора
- Если встреча нарушает одно из этих правил, приложение возвращает ошибку с подходящим HTTP кодом.

Получение полной информации по встрече

- в ответе должны быть название, организатор, участники, дата и время начала, дата и время окончания.

Получение списка встреч

- Можно получить список всех встреч пользователя.
- Можно получить список всех встреч пользователя, где он является организатором.
- Можно получить список всех встреч пользователя за указанный период.
- Можно получить только предстоящие встречи пользователя (начиная с текущего момента).
- Поддержите возможность использования любых вышеперечисленных комбинаций, кроме противоречащих (предстоящие встречи, но за прошедшие даты). В этом случае приложение должно возвращать ошибку с подходящим HTTP кодом.
- Во всех случаях возвращается полная информация о встречах.

Обновление встречи

- Можно изменить название встречи.
- Можно изменить дату и время встречи и ее продолжительность, если новый таймслот не конфликтует с другими встречами участников.
- Можно изменить список участников, если у новых в это время нет других встреч.
- Можно в одном запросе изменять любые вышеперечисленные комбинации.
- Встречу можно изменить, если она еще не началась.
- Организатора менять нельзя.
- При попытке некорректного изменения встречи приложение должно возвращать ошибку с подходящим HTTP кодом.

Удаление встречи

- Если удаляется встреча, она исчезает у организатора и всех участников.
- Встречу можно удалить, если она еще не началась, иначе приложение должно возвращать подходящую ошибку.

Дополнительные задачи

1. Поддержка properties (+2 балла)

Поддержите возможность задавать параметры приложения:

- максимальное количество встреч для одного пользователя в день.
- максимальное количество участников в одной встрече.
- минимальная продолжительность встречи в минутах.
- максимальная продолжительность встречи в минутах.

Добавьте файл `application.properties`/`application.yaml` с настройками по умолчанию.

Параметры должны загружаться в класс `@ConfigurationProperties`.

Если вы не реализуете этот пункт, считайте, что у приложения нет перечисленных выше ограничений.

2. Тестирование (+3 балла)

Протестируйте основные сценарии с помощью Spring Boot Test, JUnit и MockMvc.

Минимально должны быть покрыты тестами:

- успешное создание встречи
- получение полной информации по встрече
- получение списка встреч с разными комбинациями параметров
- обновление встречи разными вариантами
- успешное удаление
- все ошибочные сценарии
- нарушение ограничений из предыдущего пункта при условии поддержки properties

3. Логирование (+1 балла)

- Покройте логами info ключевые действия (создание, обновление, удаление).
- Добавьте логи error при ошибочных сценариях.
- Используйте SLF4J и стандартную имплементацию логгера Spring Boot.

Формат решения

- Spring Boot Application в директории `week_08/hometask/meeting`. В результате `cd week_08/hometask/meeting && ./gradlew bootRun` должно запуститься приложение, способное принимать HTTP запросы по порту 8088
- MR в своем приватном форке с названием `[hometask] meeting`

Критерии оценивания

- 4 балла: реализация API
- +2 балла: поддержка properties
- +3 балла: тестирование
- +1 балл: покрытие логами

Примерные вопросы для подготовки к зачету

Тема 1: Основы многопоточности и работы с сетью

1. Что такое клиент-серверная архитектура и как она работает?
2. Как реализовать простейший сервер на Java?
3. Что такое потоки исполнения (threads) и зачем они нужны?
4. Как работает многопоточный сервер и какие преимущества он предоставляет?
5. Что такое Thread pool и как он используется для повторного использования потоков?
6. Какие есть основные проблемы при работе с многопоточностью в Java?

7. Как использовать класс ServerSocket для создания сервера?
8. Как использовать класс Socket для создания клиента?

Тема 2: Введение в Spring

9. Что такое Dependency Injection и как оно используется в Spring?
10. Что такое фреймворк и как он отличается от библиотеки?
11. Как работает внутреннее устройство Spring и что такое Spring context?
12. Что такое бин и какие типы бинов существуют в Spring?
13. Как использовать аннотацию @Autowired для внедрения зависимостей?
14. Как использовать аннотацию @Component для создания бинов?
15. Что такое Spring Boot и как он упрощает создание приложений?

Тема 3: Взаимодействие с сетью

16. Что такое Spring Web и как он используется для создания веб-приложений?
17. Как использовать аннотацию @RestController для создания REST контроллера?
18. Как использовать Spring Boot для создания веб-приложений?
19. Что такое REST client и как он используется для взаимодействия с REST API?
20. Как использовать gRPC и protobuf для взаимодействия с сервисами?

Тема 4: Интеграция с базами данных

21. Что такое JDBC и как он используется для взаимодействия с базами данных?
22. Как использовать Spring JDBC для упрощения работы с базами данных?
23. Что такое JPA и как он используется для работы с базами данных?
24. Как использовать Liquibase и Flyway для управления схемой базы данных?
25. Как использовать Redis для хранения данных?

Тема 5: Взаимодействие с очередями сообщений

26. Что такое JMS и как он используется для взаимодействия с очередями сообщений?
27. Как использовать Active MQ и RabbitMQ для взаимодействия с очередями сообщений?
28. Что такое Kafka и как он используется для взаимодействия с очередями сообщений?
29. Как использовать Spring для взаимодействия с очередями сообщений?
30. Как использовать аннотацию @JmsListener для обработки сообщений из очереди?