

УТВЕРЖДЕНА

Решением Ученого совета
АНО ВО «Центральный университет»
«24» июня 2025 г.
Протокол № 2

**Рабочая программа дисциплины (модуля)
«Java Spring (Разработка веб-приложений на Java с использованием Spring)»**

Направление подготовки: 02.04.01 Математика и компьютерные науки

Направленность (профиль) подготовки: Продуктовый менеджмент

Квалификация (степень) выпускника: магистр

Форма обучения: очная

Срок освоения программы: 2 года

Год набора: 2025

**Москва
2025**

Содержание

1. Краткая характеристика дисциплины (модуля)	3
2. Перечень планируемых результатов обучения.....	5
3. Тематический план.....	7
4. Содержание дисциплины (модуля).....	7
5. Учебно-методическое обеспечение	8
6. Материально-техническое обеспечение	8
7. Методические и оценочные материалы	10

1. Краткая характеристика дисциплины (модуля)

Рабочая программа дисциплины (модуля) «Java Spring (Разработка веб-приложений на Java с использованием Spring)» составлена в соответствии с федеральным государственным образовательным стандартом высшего образования – магистратура по специальности 02.04.01 Математика и компьютерные науки, профиль Продуктовый менеджмент, утвержденный приказом Министерства науки и высшего образования Российской Федерации № 810 от 23.08.2017 года.

Изучение дисциплины (модуля) «Java Spring (Разработка веб-приложений на Java с использованием Spring)» имеет важное значение для студентов, 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.

Место дисциплины (модуля) в структуре образовательной программы

Настоящая дисциплина (модуль) включена в учебный план по программе подготовки магистратуры по направлению 02.04.01 Математика и компьютерные науки, профиль Продуктовый менеджмент и входит в вариативную часть Блока 1, формируемую участниками образовательных отношений, как дисциплина по выбору.

Дисциплина (модуль) изучается на 2 курсе в 3 семестре, доступна для прохождения при условии успешного завершения дисциплин (модулей) «Java Core (Основы разработки на языке Java)», «Инструменты разработчика»

Цель изучения дисциплины (модуля): освоение современных подходов к разработке масштабируемых веб-приложений на Java с использованием экосистемы Spring, включая внедрение зависимостей, работу с базами данных, построение RESTful API и реактивных систем.

Задачи изучения дисциплины (модуля):

- разбраться в концепциях Inversion of Control (IoC), Dependency Injection (DI), Spring Context и бинов, отличать Spring Framework от Spring Boot;
- научиться выбирать между JDBC и JPA/Hibernate, реализовывать маппинг сущностей, использовать Spring Data JPA для создания репозиторий, писать кастомные запросы через @Query и понимать ограничения и альтернативы (например, JOOQ);
- создавать корректные и масштабируемые эндпоинты, обрабатывать HTTP-запросы (GET, POST, PUT, DELETE), работать с DTO, сериализацией JSON, обработкой исключений и загрузкой файлов с использованием Spring Web;
- инициализировать проект через Spring Initializr, настраивать конфигурации, управлять жизненным циклом бинов, организовывать структуру проекта и развертывать готовое backend-приложение в локальной и облачной среде;
- изучить основы реактивного программирования в Spring (Mono/Flux), понять принципы работы с асинхронными потоками данных и применять Kafka для построения событийно-ориентированных систем (EDA) в экосистеме Spring.

В результате освоения дисциплины (модуля) обучающийся должен:

знать:

- базовые принципы и компоненты экосистемы Spring: для чего нужны Inversion of Control (IoC), Dependency Injection (DI), Spring Context, бины, и чем Spring Boot отличается от Spring Framework;
- ключевые подходы к работе с данными, разницу между JDBC и JPA (Hibernate);
- принципы реализации и работы RESTful API;

- основы реактивной парадигмы (Mono/Flux) и событийно-ориентированной архитектуры (EDA/Kafka).

уметь:

- создавать приложения на Spring Boot, конфигурировать их, управлять бинами;
- реализовывать RESTful API: создавать эндпоинты для обработки запросов и формирования ответов (включая работу с DTO, файлами, исключениями);
- подключаться к базам данных и работать с ними из Spring-приложений (JDBC/JPA, миграции).

владеть:

- навыками создания полноценных приложений на Spring Boot "с нуля": от инициализации проекта через Spring Initializr до развертывания рабочего backend-приложения;
- навыками применения продвинутых возможностей Spring Data JPA для эффективной работы с БД: создание запросов с помощью методов репозитория, аннотаций @Query, понимание их ограничения и знание альтернативы (JOOQ).

2. Перечень планируемых результатов обучения

Компетенции, формируемые в результате освоения дисциплины (модуля) при проведении учебных занятий в форме контактной работы обучающихся с педагогическими работниками Университета и в форме самостоятельной работы обучающихся:

Компетенция	Содержание компетенции	Индикатор компетенции	Перечень планируемых результатов обучения по дисциплине (модулю)
УК-6.	Способен определять и реализовывать приоритеты собственной деятельности и способы ее совершенствования на основе самооценки	УК-6.1.	Знает основные методы самооценки и анализа своей деятельности, а также принципы управления временем и целеполагания
		УК-6.2.	Умеет ставить реалистичные и достижимые цели, определять приоритеты в своей деятельности, а также разрабатывать и внедрять планы по совершенствованию своих навыков и компетенций на основе полученной самооценки
		УК-6.3.	Имеет практический опыт применения методов самооценки в своей профессиональной деятельности, включая участие в тренингах, семинарах и проектах, направленных на развитие личной эффективности и профессионального роста
ОПК-2.	Способен создавать и исследовать новые математические модели в естественных науках, совершенствовать и разрабатывать концепции, теории и методы	ОПК-2.1.	Знает основные математические модели и методы, используемые в естественных науках, включая статистическое моделирование, дифференциальные уравнения и численные методы, а также современные подходы к исследованию и анализу данных
		ОПК-2.2.	Умеет разрабатывать и адаптировать математические модели для решения конкретных проблем в естественных науках, проводить их анализ и верификацию, а также интерпретировать полученные результаты в контексте научных исследований
		ОПК-2.3.	Имеет практический опыт создания и исследования математических моделей в

			рамках научных проектов или исследований, включая участие в публикациях, конференциях или коллаборациях, где были разработаны и апробированы новые концепции и методы
ПК-3.	Способен решать задачи профессиональной деятельности в области продуктового менеджмента, формулировать результаты анализа и выявлять последствия полученных данных для принятия обоснованных решений и оптимизации продуктов	ПК-3.1.	Знает методы и инструменты продуктового менеджмента
		ПК-3.2.	Умеет применять аналитические инструменты и программное обеспечение для обработки и визуализации данных, а также формулировать выводы на основе проведенного анализа
		ПК-3.3.	Имеет опыт работы над реальными проектами в области продуктового менеджмента, включая анализ пользовательского поведения и оптимизацию продуктов на основе полученных данных
ПК-4.	Способен публично представлять собственные и известные научные результаты	ПК-4.1.	Знает основные принципы эффективного публичного выступления, методы визуализации данных и основные требования к научным презентациям, включая структуру и содержание
		ПК-4.2.	Умеет четко и логично формулировать свои научные результаты, адаптируя их для различных аудиторий, а также использовать визуальные средства для улучшения восприятия информации
		ПК-4.3.	Имеет практический опыт участия в научных конференциях, семинарах или других мероприятиях, где успешно представлял свои и известные научные результаты, получая обратную связь и взаимодействуя с аудиторией

3. Тематический план

№ п/п	Наименование раздела дисциплины (модуля)	Трудоемкость, академические часы				ТКУ (текущий контроль успеваемости)
		<i>Очная форма</i>				
		Аудиторная работа		Контр оль	Самосто ятельна я работа	
Лекции	Семинары (Практическ ие занятия)					
1	Введение в Spring. Spring Boot	6	6		26	Домашнее задание
2	Работа с API и базами данными	14	14		56	Домашнее задание
3	Тестирование и профилирование	4	4		18	Домашнее задание
4	Реактивный Spring	6	6		26	Домашнее задание
	<i>Зачет</i>			4		
	Итого:	30	30	4	126	
	Объем дисциплины (модуля) (в ак. ч.)	190				
	Объем дисциплины (модуля) (в зач. ед.)	5				

4. Содержание дисциплины (модуля)

№п/п	Наименование раздела дисциплины (модуля)	Содержание дисциплины (модуля) по темам
1	Введение в Spring. Spring Boot	Экосистема Spring. Spring Framework. Spring Boot Работа с бинами. Жизненный цикл бинов Стереотипные аннотации. Конфигурирование
2	Работа с API и базами данными	RESTful API. Spring MVC. Обработка запросов RESTful API. Spring MVC. Формирование ответов Работа с БД. Spring JDBC. JdbcTemplate Работа с БД. Spring Data JPA. Hibernate Продвинутые возможности Spring Data JPA Spring Security Выполнение запросов. RestTemplate. HttpClient
3	Тестирование и профилирование	Тестирование Spring-приложений Профилирование и мониторинг Spring-приложений
4	Реактивный Spring	Реактивный Spring. Spring WebFlux Event-driven architecture. Работа с Kafka Асинхронная работа с базами данных

5. Учебно-методическое обеспечение

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

Каждый студент 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.

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

Основная литература:

1. Уоллс, К. Spring в действии : практическое руководство / К. Уоллс ; пер. с англ. А. Н. Киселева. - 6-е изд. - Москва : ДМК Пресс, 2022. - 544 с. - ISBN 978-5-93700-112-2. - Текст : электронный. - URL: <https://znanium.com/catalog/product/2110012>.

2. Хеклер, М. Spring Boot по-быстрому : практическое руководство / М. Хеклер. - Санкт-Петербург : Питер, 2022. - 352 с. - (Бестселлеры O'Reilly). - ISBN 978-5-4461-3942-2. - Текст : электронный. - URL: <https://znanium.com/catalog/product/2122953>.

Дополнительная литература:

1. Гутьеррес, Ф. Spring Boot 2: лучшие практики для профессионалов : практическое руководство / Ф. Гутьеррес. - Санкт-Петербург : Питер, 2021. - 464 с. - (Серия «Библиотека программиста»). - ISBN 978-5-4461-1587-7. - Текст : электронный. - URL: <https://znanium.com/catalog/product/1733697>.

2. Карнелл, Д. Микросервисы Spring в действии : практическое руководство / Д. Карнелл, И. Уайлупо Санчес ; пер. с англ. А. Н. Киселева. - Москва : ДМК Пресс, 2022. - 490 с. - ISBN 978-5-97060-971-2. - Текст : электронный. - URL: <https://znanium.ru/catalog/product/2155906>.

6. 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 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/plugin/s/
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

7. 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 удастся, 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 студенты 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 студентов, направленная на углубленное изучение отдельных тем и вопросов учебной дисциплины (модуля).

В процессе самостоятельной работы студенты взаимодействуют с рекомендованными материалами при минимальном участии преподавателя. Задачи студента 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)

Критерии получения уровня и оценивания сформированности компетенций по дисциплине (модулю) «Java Spring (Разработка веб-приложений на Java с использованием Spring)».

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

Промежуточная аттестация по дисциплине (модулю) осуществляется в форме *зачета*, при этом проводится оценка компетенций, сформированных по дисциплине.

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	Студент 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. Студент 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	Студент 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.
6	Хорошо	Зачтено	

Десятибалльная оценка	Пятибалльная оценка	Оценка за зачет	Общая характеристика результата обучения по дисциплине (модулю)
			Формулирует свои ответы, уверенно интерпретирует результаты анализов и других исследований, а также решает сложные ситуационные задачи. Студент хорошо знаком с методами исследования, необходимыми для практической деятельности, и умеет связывать теоретические аспекты предмета с практическими задачами.
5	Удовлетворительно	Зачтено	Студент 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. Студент способен решать только самые простые задачи и владеет лишь минимальным набором методов исследования.
4	Удовлетворительно	Зачтено	
3	Не сдан	Не зачтено	Студент 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)»: « $0,7 \times \text{среднее за домашние задания} + 0,3 \times \text{зачет}$ ».

Текущий контроль успеваемости обучающихся по дисциплине (модулю)

Примерные домашние задания

Домашнее задание

Сервис бронирования билетов

Разработайте GRPC-сервис для бронирования билетов на мероприятия. Все данные должны храниться в базе данных PostgreSQL. Приложение должно быть реализовано на Spring Boot с использованием JDBI или JOOQ для доступа к базе данных.

Поведение системы:

Система должна позволять:

- создавать мероприятия и билеты к ним

- просматривать мероприятия и доступные билеты
- бронировать билеты пользователями
- подтверждать или отменять бронь
- повторно бронировать билеты, если предыдущая бронь не была подтверждена в течение заданного времени

Мероприятия:

- Каждое мероприятие имеет название, дату и время начала.
- При создании мероприятия задаётся список номеров билетов (например: ["A1" , "A2" , "A3"]), которые создаются как доступные к бронированию. Эти номера должны быть уникальны в рамках данного мероприятия.
- Нельзя создавать мероприятие с датой начала в прошлом.
- Для добавления мероприятий должен быть реализован GRPC-метод.

Просмотр мероприятий и билетов:

- Пользователь может получить список всех мероприятий.
- Пользователь может получить список всех **свободных** билетов на выбранное мероприятие.
- Пользователь может получить список своих билетов: на предстоящие мероприятия или все

Бронирование билетов:

- Пользователь может забронировать билет на мероприятие, указав только свой `userId` и номер билета.
- Если билет уже забронирован или подтверждён другим пользователем, бронь невозможна.
- Бронь доступна для подтверждения в течение ограниченного времени (в минутах).
- Время, отведённое на подтверждение брони, должно задаваться через файл настроек (`application.yaml`) и считываться с помощью `@ConfigurationProperties`.

Подтверждение и отмена:

- Пользователь может **подтвердить** свою бронь.
- Пользователь может **отменить** свою бронь до её подтверждения.
- Подтверждённый билет не может быть отменён или забронирован повторно.
- Если бронь не была подтверждена в течение разрешённого времени, она автоматически считается **истекшей** и не препятствует новым бронированиям.
- Удаление или фоновая очистка истекших броней **не требуется в рамках задания**.

Пользователи

- Пользователи в системе **не хранятся**, используется только `userId` (тип `UUID`).
- Никакая дополнительная информация о пользователях не валидируется и не проверяется.

Требования к реализации:

1. GRPC API

- Все взаимодействия (создание мероприятия, просмотр, бронирование, подтверждение, отмена) реализуются через GRPC-сервис.

- Протокол должен быть описан в `.proto`-файле.
- UUID должен передаваться как `string` в формате UUID.

2. База данных

- Используйте PostgreSQL, схему `public`.
- Используйте `optimistic locking` для защиты от гонки при подтверждении бронирования.
- Если нужной схемы таблиц ещё нет, она должна автоматически накатываться при старте с помощью Liquibase.

3. Тестирование (+4 балла)

Покройте основные сценарии тестами с использованием `TestContainers`.

Минимально должны быть покрыты:

- создание мероприятия
- получение мероприятий и билетов
- успешное бронирование, подтверждение и отмена
- повторная бронь после истечения времени
- ошибки при конфликтующих действиях

Формат решения:

- Spring Boot приложение в директории `week_12/hometask/ticket_service`. В результате `cd week_12/hometask/ticket_service && ./gradlew bootRun` должно запускать приложение, готовое принимать GRPC-запросы на порту 9090. Приложение должно подключаться к БД: `host: localhost, port: 5432, database: tickets, user: ticket_user, password: ticket_password`.

- MR в приватном форке с названием `[hometask] ticket_service`.

Критерии оценивания:

- 6 баллов: реализация функциональности
- +4 балла: тесты с использованием `TestContainers`

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

Задания для промежуточной аттестации по дисциплине (модулю)

№ п/п	Задание	Ответ	Компетенция
1.	Опишите, как вы определяете приоритеты в своей работе при разработке программного обеспечения. Какие методы вы используете для самооценки своей продуктивности?	Матрица приоритетов	УК-6
2.	Какой метод самооценки вы используете для улучшения своей деятельности?	Рефлексия	УК-6
3.	Как вы разрабатываете математические модели для оптимизации многопоточного сервера?	Алгоритмы	ОПК-2
4.	Какую модель используют для анализа производительности REST API	Модель нагрузки	ОПК-2
5.	Как вы представляете свои научные результаты на конференции?	Презентация	ПК-4

6.	Какие данные вы используете для визуализации результатов тестирования?	Графики	ПК-3
7.	Укажите способ определения приоритетов в выборе типа бина в Spring.	Анализ зависимостей	УК-6
8.	Назовите математическую модель клиент-серверной архитектуры.	Граф состояний	ОПК-2
9.	Укажите метод исследования модели транзакций в базах данных.	Симуляция	ОПК-2
10.	Назовите элемент самооценки в интеграции с базами данных.	Тестирование запросов	УК-6
11.	Укажите способ публичного представления REST клиента.	Демонстрация	ПК-4
12.	Укажите следствие внедрения зависимостей в Spring.	Легкость тестирования	ПК-3
13.	Назовите формат презентации жизненного цикла бина.	Диаграмма	ПК-4
14.	Укажите технику совершенствования работы с очередями сообщений.	Оптимизация потоков	УК-6
15.	Назовите вывод из интеграции с JDBC.	Стабильность соединений	ПК-3
16.	Назовите элемент презентации миграции базы данных.	Код-ревью	ПК-4
17.	Укажите подход к разработке модели REST API.	Диаграмма последовательности	ОПК-2
18.	Укажите результат работы с Apache Kafka.	Асинхронная обработка	ПК-3
19.	Назовите концепцию моделирования многопоточности.	Модель Акторов	ОПК-2
20.	Назовите результат решения задачи создания многопоточного сервера.	Thread pool	ПК-3
21.	Назовите теорию моделирования очередей сообщений.	Теория очередей	ОПК-2
22.	Укажите метод демонстрации работы с JMS.	Видео-туториал	ПК-4
23.	Назовите следствие использования profiles в Spring.	Гибкость развертывания	ПК-3