ЦЕНТРАЛЬНЫЙ УНИВЕРСИТЕТ

УТВЕРЖДЕНА

Решением Ученого совета
АНО ВО «Центральный университет»
«25» мая 2026 г.
Протокол № 2

Рабочая программа дисциплины (модуля)
«Разработка на Python»

Направление подготовки: *38.03.05 Бизнес-информатика*

Направленность (профиль) подготовки: *Бизнес и аналитика*

Квалификация (степень) выпускника: *бакалавр*

Форма обучения: *очная*

Срок освоения программы: *4 года*

Год набора: *2026*

ДОКУМЕНТ ПОДПИСАН
ЭЛЕКТРОННОЙ ПОДПИСЬЮ

Сертификат: 0215B0C000D1B3FEB44A37EE631E0C27D4
Владелец: Ивашкевич Евгений Васильевич
Действителен: с 13.01.2026 до 13.04.2027

Электронный документ

Москва
2026

Содержание

1. Краткая характеристика дисциплины (модуля)

Рабочая программа дисциплины (модуля) «Разработка на Python» составлена в соответствии с федеральным государственным образовательным стандартом высшего образования – бакалавриат по направлению подготовки 38.03.05 Бизнес-информатика, профиль Бизнес и аналитика, утвержденный приказом Министерства науки и высшего образования Российской Федерации № 838 от 29.07.2020 года.

Изучение дисциплины (модуля) «Разработка на Python» формирует у обучающегося практические навыки разработки масштабируемых и производительных веб-сервисов, востребованных в современной IT-индустрии. Освоение инструментов FastAPI, GitLab, Redis и loguru обеспечивает готовность к участию в реальных проектах в составе команды под руководством опытного специалиста.

Место дисциплины (модуля) в структуре образовательной программы

Настоящая дисциплина (модуль) включена в учебный план по программе подготовки бакалавриата по направлению 38.03.05 Бизнес-информатика, профиль Бизнес и аналитика и входит в часть, формируемую участниками образовательных отношений, Блока 1, как дисциплина по выбору.

Дисциплина (модуль) изучается на 3 или 4 курсе в 6,7 или 8 семестре на выбор. Доступна к изучению после успешного освоения одной из дисциплин (модулей): «Язык программирования Python. Базовый», «Язык программирования Python. Основной», «Язык программирования Go», «Язык программирования Java» или «Язык программирования C++».

Цель изучения дисциплины (модуля): подготовить обучающегося к разработке и сопровождению современных веб-приложений на Python с использованием асинхронных технологий, инструментов версионирования и интеграции с внешними сервисами.

Задачи изучения дисциплины (модуля):

— освоить принципы клиент-серверной архитектуры и организацию взаимодействия между фронтендом и бэкендом;

— научиться использовать Git и GitLab для командной разработки, включая ветвление, коммиты, merge request и разрешение конфликтов;

— освоить основы HTTP и разработку API с помощью FastAPI, включая маршрутизацию, обработку запросов и возврат ответов;

— научиться применять асинхронное программирование, кеширование с Redis и структурированное логирование с loguru для повышения производительности и надежности приложений;

— развить навыки автоматизации процессов через реализацию периодических задач с использованием apscheduler и BackgroundTasks FastAPI.

В результате освоения дисциплины (модуля), обучающийся должен:

знать:

— принципы работы клиент-серверной архитектуры: роль фронтенда и бэкенда;

— основы организации проекта: структура каталогов, работа с;

— основные команды Git: создание репозиториев, ветвление, коммиты, слияние и разрешение конфликтов;

— основы работы с GitLab: создание проектов, управление ветками, MR;

— принципы работы HTTP: методы запросов (GET, POST, PUT, DELETE), структура запроса и ответа;

— основные возможности FastAPI: асинхронные маршруты, работа с запросами, возвращение ответов;

— основы асинхронного программирования: что такое async/await, задачи асинхронности;

— что такое middleware, их назначение и основные примеры использования в FastAPI;

— основы кеширования: зачем оно нужно, типы кеширования (в памяти, распределенное), как кеширование влияет на производительность;

— основные принципы логирования: уровни логов, структурированное логирование, назначение логов в разработке;

— что такое периодические задачи и как их использовать для автоматизации процессов;

— основы работы с планировщиком и FastAPI BackgroundTasks для реализации периодических задач.

уметь:

— работать с Git для управления версионностью проекта: выполнять коммиты, переключаться между ветками, работать с удаленными репозиториями;

— настраивать проект на FastAPI: создавать основные маршруты, разделять файлы по назначению;

— взаимодействовать с внешними API через HTTP-запросы с использованием библиотеки httpx;

— обрабатывать данные из сторонних API, передавать их фронтенду через маршруты FastAPI;

— использовать асинхронные функции для работы с длительными операциями;

— реализовывать middleware в FastAPI;

— настраивать кеширование с использованием Redis для повышения производительности приложений;

— использовать loguru для добавления логирования в приложение;

— настраивать формат вывода логов;

— управлять уровнями логирования;

— логировать исключения и критические события;

— создавать периодические задачи с использованием apscheduler и BackgroundTasks FastAPI;

— автоматизировать выполнение фоновых операций.

владеть:

— навыками создания и ведения проекта в GitLab;

— навыками разработки проектов в команде с использованием Git: создавать MR, проводить код-ревью, разрешать конфликты при слиянии;

— навыками написания бэкенда для готового фронтенда, обеспечив корректное взаимодействие клиентской и серверной частей;

— навыками организации проектной структуры для комплексного FastAPI-приложения, разделяя логику API, бизнес-логику и обработку данных;

— навыками реализации взаимодействия с несколькими сторонними API, комбинируя их данные в одном сервисе;

— навыками интеграции асинхронной обработки данных в проект, обеспечивая стабильность и производительность приложения;

— навыками разработки middleware;

— навыками настройки кеширования с продвинутыми функциями: инвалидация кеша, настройка TTL;

— навыками организации централизованного структурированного логирования с использованием loguru;

— навыками реализации комплексных периодических задач, используя комбинацию планировщика и FastAPI BackgroundTasks, обеспечивая стабильную и предсказуемую работу сервиса;

— навыками связи результатов логирования, кеширования и периодических задач для мониторинга и улучшения производительности приложения.

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-1.	Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач	УК-1.1.	Знает методы поиска и анализа информации в области разработки, основные принципы критической оценки источников информации и их релевантности.
		УК-1.2.	Умеет критически оценивать источники информации и синтезировать данные из различных источников для решения задач, применять системный подход к анализу и решению комплексных проблем.
		УК-1.3.	Имеет практический опыт работы с современными инструментами и технологиями для обработки информации, формулировании и структурировании задач на основе полученной информации.
ОПК-2.	Способен проводить исследование и анализ рынка информационных систем и информационно-коммуникационных технологий, выбирать рациональные решения для управления бизнесом	ОПК-2.1.	Знает основные тенденции и характеристики рынка информационных систем и информационно-коммуникационных технологий
		ОПК-2.2.	Умеет проводить исследование и анализ рыночной информации для оценки потребностей бизнеса и выбора оптимальных решений
		ОПК-2.3.	Имеет практический опыт в разработке и внедрении стратегий управления бизнесом на основе анализа рынка информационных технологий
ПК-2.	Способен использовать соответствующий математический аппарат и инструментальные средства для обработки, анализа и систематизации информации по теме исследования для решения задач профессиональной деятельности	ПК-2.1.	Знает основные математические методы и инструментальные средства, применяемые для обработки и анализа информации
		ПК-2.2.	Умеет эффективно использовать математический аппарат для систематизации данных и решения профессиональных задач
		ПК-2.3.	Имеет практический опыт работы с инструментами анализа информации в рамках исследовательских проектов

ПК-3.	Способен готовить научно-технические отчеты, презентации, научные публикации по результатам выполненных исследований	ПК-3.1.	Знает требования и стандарты оформления научно-технических отчетов, презентаций и публикаций
		ПК-3.2.	Умеет структурировать и представлять результаты исследований в ясной и доступной форме
		ПК-3.3.	Имеет практический опыт подготовки и публикации научных материалов, отражающих результаты выполненных исследований
ПК-8.	Способен под руководством специалиста более высокой категории осуществлять планирование и организацию проектной деятельности на основе стандартов управления проектами	ПК-8.1.	Знает принципы и стандарты управления проектами
		ПК-8.2.	Умеет разрабатывать планы и организовывать проектную деятельность в соответствии с установленными стандартами
		ПК-8.3.	Имеет практический опыт участия в проектной работе, включая планирование и координацию задач

3. Тематический план

№ п/п	Наименование раздела дисциплины (модуля)	Трудоемкость, академические часы					ТКУ (текущий контроль успеваемости)
		Очная форма					
		Аудиторная работа			Контроль	Самостоятельная работа	
		Лекции	Семинары	Консультации			
1	Базовые инструменты и взаимодействие с сервисами	4	4	2		20	Домашнее задание
2	Проектирование веб-приложений на FastAPI	10	10	6	2	50	Домашнее задание Контрольная работа
3	Инженерные практики и промышленная разработка	10	10	6	4	50	Домашнее задание Контрольная работа Проект
	Зачет с оценкой				2		
	Итого:	***24***	***24***	***14***	***8***	***120***	
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	Базовые инструменты и взаимодействие с сервисами	Git и GitLab Основы HTTP и работа с API
2	Проектирование веб-приложений на FastAPI	FastAPI Асинхронные маршруты и обработка запросов Dependency Injection Взаимодействие с внешними API через асинхронные библиотеки MongoDB
3	Инженерные практики и промышленная разработка	Основы логирования и использование loging Основы кеширования и использование Redis Продвинутое кеширование и инвалидация кеша Периодические задачи в FastAPI

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. Чернышев, С. А. Основы программирования на Python : учебник для вузов / С. А. Чернышев. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2026. — 349 с. — (Высшее образование). — ISBN 978-5-534-17139-6. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/588667.

2. Виафоре, П. Надежный Python : практическое руководство / П. Виафоре, М. Райтман. - Санкт-Петербург : БХВ-Петербург, 2023. - 352 с. - ISBN 978-5-9775-1174-2. - Текст : электронный. - URL: https://znanium.com/catalog/product/2123392.

Дополнительная литература:

1. Бейдер, Д. Чистый Python. Тонкости программирования для профи : практическое руководство / Д. Бейдер. - Санкт-Петербург : Питер, 2021. - 288 с. - (Серия «Библиотека программиста»). - ISBN 978-5-4461-0803-9. - Текст : электронный. - URL: https://znanium.ru/catalog/product/1766370.

2. Хиллард, Д. Секреты Python Pro / Д. Хиллард. - Санкт-Петербург : Питер, 2021. - 320 с. - ISBN 978-5-4461-1684-3. - Текст : электронный. - URL: https://znanium.ru/catalog/product/2142837.

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.	Научная электронная библиотека elibrary.ru библиотека	https://elibrary.ru/defaultx.asp
2.	База данных для IT-специалистов	https://habr.com
3.	База данных ScienceDirect	https://www.sciencedirect.com
4.	Официальный сайт Министерства науки и высшего образования Российской Федерации	https://minobrnauki.gov.ru/
5.	Федеральный портал «Российское образование"	https://www.edu.ru/
6.	Информационная система "Единое окно доступа к образовательным ресурсам"	http://window.edu.ru/
7.	Единая коллекция цифровых образовательных ресурсов	http://school-collection.edu.ru/
8.	Федеральный центр информационно - образовательных ресурсов	http://fcior.edu.ru/

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

Среды разработки:		
Visual Studio Code	зарубежное	свободно распространяемое
Bash (Unix shell)	зарубежное	свободно распространяемое
Anaconda	зарубежное	свободно распространяемое
Robotic Operating System	зарубежное	свободно распространяемое
CopelliaSim	зарубежное	свободно распространяемое
Google Colaboratory	зарубежное	свободно распространяемое
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) «Разработка на Python» в рамках текущего контроля успеваемости используются такие виды учебной работы, как лекции, семинары, консультации, домашние задания, контрольные работы, проект, 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 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.

Консультации – структурированные встречи, на которых преподаватели предоставляют индивидуальную или групповую помощь в освоении учебного материала, обсуждении вопросов и решении проблем, возникающих в процессе обучения.

Консультации могут включать разъяснение сложных тем, подготовку к экзаменам и помощь в выполнении проектных работ, что способствует более глубокому пониманию предмета и улучшению академической успеваемости.

Домашнее задание – набор задач 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.

Контрольная работа – письменная работа с набором задач, которые нужно решить за ограниченное время.

Цель контрольной работы – получить специальные знания по одной или нескольким темам дисциплины и продемонстрировать навыки их практического применения.

Проект – исследовательская работа по дисциплине (модулю) и презентация результатов.

Для успешной подготовки к проекту рекомендуется: четко определить цели и задачи проекта; составить план работы, разбив проект на этапы с указанием сроков выполнения каждого из них; использовать разнообразные источники информации и инструменты для исследования темы; регулярно проверять прогресс и вносить коррективы в план, если это необходимо.

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 «Разработка на Python»

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

Промежуточная аттестация по дисциплине (модулю) осуществляется в форме ***зачета с оценкой***, 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, если иное не указано в силлабусе дисциплины:

Десятибалльная оценка	Пятибалльная оценка	Общая характеристика результата обучения по дисциплине (модулю)
10	Отлично	Студент полностью владеет знаниями, изложенными в рабочей программе, и глубоко осмысляет дисциплину. 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
9	Отлично	
8	Отлично	

Десятибалльная оценка	Пятибалльная оценка	Общая характеристика результата обучения по дисциплине (модулю)
		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.
7	Хорошо	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. 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.
6	Хорошо	
5	Удовлетворительно	Студент обладает базовыми знаниями по дисциплине, 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	
3	Не сдан	Студент не овладел обязательным минимумом знаний по предмету и не может ответить на вопросы, даже если преподаватель задает дополнительные наводящие вопросы.
2	Не сдан	
1	Не сдан	

Дисциплина (модуль) «Разработка на Python» оценивается следующим образом:

Активность	Вес	Количество	Описание
Домашнее задание	15%	11	Набор задач по темам недели
Проект	25%	1	Исследовательская работа по дисциплине (модулю) и презентация результатов
Контрольная работа	30%	3	Письменная работа с набором задач, которые нужно решить за ограниченное время
Зачет с оценкой	30%	1	Письменная или устная работа над заданием, направленным на проверку полученных знаний и навыков по дисциплине (модулю)

Электронный документ

Формула расчёта итоговой оценки по дисциплине (модулю) «Разработка на Python»: «0,25 ×проект + 0,15 × среднее за домашние задания + 0,3 × среднее за контрольные работы + 0,3 × зачет с оценкой».

Текущий контроль успеваемости обучающихся по дисциплине (модулю)

Примерные домашние задания

Домашнее задание 1: Работа с Git и основы взаимодействия с API

1. Установите Git и настройте учётную запись на GitLab;
2. Создайте новый проект в GitLab, инициализируйте локальный репозиторий и свяжите его с удалённым;
3. Создайте ветку feature/api-request, перейдите в неё;
4. Напишите Python-скрипт, который отправляет GET-запрос к публичному API с использованием библиотеки httpx;
5. Выведите статус ответа и тело запроса в консоль;
6. Добавьте комментарий в код: # Запрос к внешнему API;
7. Закоммитьте изменения, отправьте ветку в GitLab и создайте Merge Request в main с описанием проделанной работы.

Домашнее задание 2: Разработка асинхронного API на FastAPI

1. Создайте FastAPI-приложение с асинхронным маршрутом /ping, возвращающим {"status": "ok"};
2. Добавьте маршрут /posts, который асинхронно получает данные с https://jsonplaceholder.typicode.com/posts и возвращает первые 10 постов;
3. Реализуйте параметризованный маршрут /posts/{post_id}, который возвращает данные конкретного поста по ID;
4. Используйте Dependency Injection для выделения логики запроса к внешнему API в отдельную функцию;
5. Добавьте обработку ошибок: если пост не найден, возвращайте HTTP 404;
6. Сохраните полученные данные в MongoDB (или в JSON-файл, если MongoDB недоступна);
7. Оформите код с комментариями и выложите в GitLab в новой ветке feature/fastapi-app.

Домашнее задание 3: Логирование, кеширование и фоновые задачи

1. Настройте структурированное логирование в приложении с помощью loguru: логируйте входящие запросы, ошибки и ключевые события;
2. Подключите Redis и реализуйте кеширование для маршрута /posts/{post_id}: при первом запросе данные берутся из внешнего API и сохраняются в Redis с TTL 60 секунд, при повторном — из кеша;
3. Добавьте механизм инвалидации кеша: при ручном запросе на /cache/clear кеш очищается;
4. Реализуйте периодическую задачу с помощью apscheduler, которая каждые 2 минуты асинхронно получает один случайный пост и сохраняет его в MongoDB;
5. Добавьте логирование запуска и результата фоновой задачи;
6. Выложите реализацию в ветку feature/caching-logging-scheduler и создайте Merge Request.

Примерное описание задания и критерии оценивания к проекту

Цель проекта: разработать полнофункциональное FastAPI-приложение, демонстрирующее навыки работы с асинхронностью, внешними API, системами хранения данных, кешированием, логированием и периодическими задачами, а также применение современных практик командной разработки.

Задачи, которые необходимо выполнить студенту:

— создать репозиторий в GitLab, настроить структуру проекта и организовать работу с ветками;

— реализовать асинхронные маршруты FastAPI для получения и обработки данных из публичного REST API (например, https://jsonplaceholder.typicode.com);

— использовать Dependency Injection для разделения логики маршрутов и бизнес-логики;

— интегрировать MongoDB (или другой источник данных) для хранения результатов;

— подключить Redis и реализовать кеширование ответов с возможностью инвалидации по запросу;

— настроить структурированное логирование с помощью loguru: логировать запросы, ошибки и фоновые операции;

— реализовать периодическую задачу с использованием apscheduler, которая автоматически получает данные с внешнего API и сохраняет их в базу каждые 2–5 минут;

— обеспечить обработку ошибок и возврат корректных HTTP-статусов;

— оформить код в соответствии с PEP 8, добавить комментарии и документацию (описание endpoints в Swagger);

— выложить проект в GitLab, создать Merge Request и представить результаты.

Критерии оценивания проекта:

— корректная реализация асинхронных маршрутов и взаимодействие с внешним API с использованием асинхронных библиотек;

— применение принципов инжиниринга: логирование, кеширование с инвалидацией, фоновые задачи и работа с Redis;

— использование современных практик разработки: структура проекта, Dependency Injection, обработка ошибок, работа с MongoDB;

— соблюдение стандартов командной разработки: работа с Git и GitLab, ветвление, коммиты, MR, читаемость кода;

— функциональность и стабильность приложения: корректная работа всех компонентов, отсутствие критических ошибок при запуске и выполнении задач.

Примерные задания для контрольной работы

1. Напишите асинхронный маршрут FastAPI @app.get("/health"), который возвращает JSON-ответ {"status": "ok"} при успешном запуске приложения.
2. Объясните, зачем используется async и await в FastAPI. Приведите пример асинхронной функции, выполняющей HTTP-запрос к внешнему API с помощью httpx.
3. Реализуйте зависимость (dependency), которая проверяет наличие заголовка X-API-Key в запросе. Если ключ отсутствует или не равен secret123, возвращается ошибка 401.
4. Напишите код, который с помощью httpx.AsyncClient получает данные с https://jsonplaceholder.typicode.com/posts/1 и возвращает только поле title.
5. Опишите, как подключить MongoDB в FastAPI-приложении. Приведите пример асинхронного сохранения документа (например, {"post_id": 1, "title": "Test"}) в коллекцию posts.
6. Как настроить логирование в FastAPI с помощью loguru? Напишите код, который логирует каждый входящий запрос (метод и путь) с уровнем INFO.
7. Объясните, зачем используется Redis в веб-приложениях. Напишите функцию, которая сохраняет данные в Redis с TTL 30 секунд и проверяет их наличие.

8. Реализуйте кеширование для маршрута /posts/{post_id}: при первом запросе данные берутся из внешнего API и кешируются, при повторном — возвращаются из Redis. Укажите, как можно инвалидировать кеш по запросу /cache/clear.

9. Напишите пример периодической задачи с использованием BackgroundTasks и apscheduler, которая каждые 5 минут логирует сообщение "Выполнение фоновой задачи...".

10. Опишите, как реализовать graceful shutdown FastAPI-приложения, чтобы завершить все фоновые задачи и корректно отключиться от Redis и MongoDB.

Задания для промежуточной аттестации по дисциплине (модулю)

№ п/п	Задание	Ответ	Компетенция
1	Объясните, как асинхронность в FastAPI позволяет эффективно обрабатывать запросы к медленным внешним API. Приведите пример с **async** / **await** и **httpx**.	Использование асинхронных вызовов предотвращает блокировку сервера; пример: **async with httpx.AsyncClient() as client**.	УК-1
2	Опишите принцип Dependency Injection в FastAPI. Приведите пример зависимости, которая возвращает соединение с базой данных.	DI позволяет внедрять зависимости (например, БД) в маршруты; пример: **def get_db(): yield db**.	УК-1
3	Как системный подход помогает при проектировании сервиса, объединяющего данные из нескольких внешних API? Перечислите этапы анализа и интеграции.	Анализ API, проектирование обработки ошибок, нормализация данных, объединение, кэширование.	УК-1
4	Напишите асинхронный маршрут FastAPI, который получает данные с **https://jsonplaceholder.typicode.com/posts/1** и возвращает только **title**.	**@app.get("/title") async def get_title(): async with httpx.AsyncClient() as client: r =**	УК-1

		await client.get(...); return r.json()["title"].	
5	Объясните, зачем используется кеширование в веб-приложениях. Опишите алгоритм сохранения и чтения данных из Redis с TTL.	Уменьшает нагрузку на API и БД; данные сохраняются по ключу с временем жизни, TTL=60.	ОПК-2
6	Как логирование с **loguru** помогает в отладке и мониторинге приложения? Напишите пример логирования запроса к **/user/{id}**.	Позволяет отслеживать поведение; пример: **logger.info(f "GET /user/{user_id}")**.	ОПК-2
7	Опишите, как периодические задачи могут использоваться для автоматического обновления кеша. Приведите пример задачи, запускаемой каждые 3 минуты.	**@scheduler.scheduled_job("interval", minutes=3)**; задача обновляет кеш из внешнего API.	ОПК-2
8	Какие математические методы можно применить для анализа частоты обращений к API и оптимизации TTL кеша? Приведите пример расчёта.	Статистический анализ: среднее время между запросами, прогнозирование пиковой нагрузки.	ОПК-2
9	Объясните, как Git и GitLab поддерживают системный подход в разработке. Какие практики (ветвление, MR, CI) помогают контролировать качество кода?	Позволяют тестировать изменения, проводить ревью, автоматизировать сборку и деплой.	ПК-2

10	Разработайте структуру FastAPI-приложения с разделением на **routers**, **dependencies**, **services**, **models**. Обоснуйте выбор архитектуры.	Модульность упрощает сопровождение, тестирование и масштабирование.	ПК-2
11	Как асинхронные функции влияют на производительность при одновременной обработке множества запросов? Приведите пример сценария, где это критично.	Позволяют обрабатывать сотни запросов без блокировок; например, микросервис для сбора данных.	ПК-2
12	Напишите зависимость, которая проверяет наличие заголовка **Authorization: Bearer <token>**. При отсутствии — возвращается 401.	**async def verify_token (authorization: str = Header(...)): if not valid: raise HTTPException(...)**.	ПК-2
13	Опишите, как можно реализовать инвалидацию кеша при обновлении данных. Приведите пример маршрута **/cache/clear**.	**@app.post(" /cache/clear") async def clear_cache() : await redis.flushdb ()**.	ПК-3
14	Как логирование помогает в выявлении узких мест в производительности? Опишите, какие события стоит логировать.	Логировать время выполнения запросов, ошибки, обращения к API и БД.	ПК-3
15	Объясните, зачем используется MongoDB в FastAPI-приложениях. Приведите пример асинхронного сохранения документа.	Для хранения гибких JSON-документов; **await db.posts.inse rt_one({"titl e": "Test"})**.	ПК-3
16	Как системный подход помогает при проектировании отказоустойчивого API? Перечислите меры по	Таймауты, повторные	ПК-3

	обработке ошибок и резервированию.	запросы, fallback-ответы, мониторинг, резервные источники данных.	
17	Напишите маршрут, который возвращает HTTP 404, если пост с заданным ID не найден во внешнем API.	Проверка статуса ответа; **if r.status_code == 404: raise HTTPException(...)**.	ПК-8
18	Объясните, как использование **BackgroundTasks** в FastAPI помогает отделить долгие операции от основного запроса. Приведите пример.	Например, отправка уведомления после регистрации: **background_tasks.add_task(send_email, ...)**.	ПК-8
19	Как можно использовать статистические данные о запросах для оптимизации структуры кеша? Опишите модель распределения нагрузки.	Анализ пиковых часов, популярных эндпоинтов; настройка TTL и размера кеша.	ПК-8
20	Разработайте концепцию веб-сервиса, который агрегирует данные из двух API, кеширует результат, логирует операции и обновляет кеш каждые 5 минут. Опишите архитектуру.	FastAPI + httpx + Redis + MongoDB + apscheduler + loguru; модульная структура.	ПК-8