ЦЕНТРАЛЬНЫЙ УНИВЕРСИТЕТ

УТВЕРЖДЕНА

Решением Ученого совета
АНО ВО «Центральный университет»
«07» марта 2024 г.
Протокол №1

Рабочая программа дисциплины (модуля) «Промышленная разработка»

Направление подготовки: *38.03.05 Бизнес-информатика*

Направленность (профиль) подготовки: *Бизнес-аналитика*

Квалификация (степень) выпускника: *бакалавр*

Форма обучения: *очная*

Срок освоения программы: *4 года*

Год набора: *2024*

Москва
2024

Электронный документ

Содержание

1. Краткая характеристика дисциплины (модуля)

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

Изучение дисциплины (модуля) «Промышленная разработка» позволяет овладеть ключевыми методами и технологиями, необходимыми для эффективного проектирования и внедрения промышленных решений. Это формирует практические навыки и системное мышление, способствующие успешной профессиональной деятельности в инженерной и производственной сферах.

Место дисциплины (модуля) в структуре образовательной программы

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

Дисциплина (модуль) является выборной и доступна для изучения на 3 или 4 курсе в 5, 6, 7, 8 семестрах на выбор.

Цель изучения дисциплины (модуля): формирование базовых знаний и практических навыков, необходимых для проектирования, анализа и внедрения промышленных технологий и процессов.

Задачи изучения дисциплины (модуля) направлены на формирование у студентов следующий знаний, умений и навыков:

- знание концепций ООП и их применение на практике;
- знание механизмов клиент-серверного взаимодействия приложений;
- знание процессов разработки в команде;
- умение интегрировать свою часть кода в командный проект;
- умение покрывать код различными видами тестов;
- умение разрабатывать свои классы с использованием наследования;
- умение взаимодействовать с базами данных программным образом;
- навык декомпозиции проекта по используемым технологиям и подзадачам;
- навык реализации полноценного веб-приложения с хранением данных и разграничением доступа.

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 управлять своим временем, выстраивать и реализовывать траекторию саморазвития на основе принципов образования в течение всей жизни	УК-6.1.	Знает основные принципы самовоспитания и самообразования, профессионального и личностного развития, исходя из этапов карьерного роста и требований рынка труда
		УК-6.2.	Умеет планировать свое рабочее время и время для саморазвития. формулировать цели личностного и профессионального развития и условия их достижения, исходя из тенденций развития области профессиональной деятельности, индивидуально-личностных особенностей
		УК-6.3.	Имеет практический опыт получения дополнительного образования, изучения дополнительных образовательных программ
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	ПК-2.1.	Знает основные математические методы и инструментальные средства, применяемые для обработки и анализа информации

	систематизации информации по теме исследования для решения задач профессиональной деятельности	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

3. Тематический план

№п/п	Наименование раздела дисциплины (модуля)	Трудоемкость, академические часы				ТКУ (текущий контроль успеваемости)
		Очная форма				
		Контактная работа		Контроль	Самостоятельная работа	
		Лекции	Семинары (практические занятия)			
1	Командная разработка	7	7		32	Подготовка к семинару, Домашние задания, Контрольная работа
2	Проектирование ПО	7	7		32	Подготовка к семинару, Домашние задания, Контрольная работа
3	Разработка веб-API	7	7		34	Подготовка к семинару, Домашние задания, Контрольная работа
4	Frontend-разработка	7	7		32	Подготовка к семинару, Домашние задания, Контрольная работа
	Зачет с оценкой			4		Проект
	Итого:	***28***	***28***	***4***	***130***	
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	Командная разработка	Работа в команде с использованием Git и GitLab Flow. Написание и запуск unit-тестов.
2	Проектирование ПО	Принципы объектно-ориентированного программирования (ООП). Обработка ошибок и исключений. Использование mock-объектов в тестировании.
3	Разработка веб-API	Создание и документирование HTTP API. Работа с базами данных и ORM. Развертывание приложений с помощью Docker и Docker Compose. Настройка Continuous Integration (CI). Проведение интеграционного тестирования.
4	Frontend-разработка	Основы HTML, CSS и JavaScript. Инструменты разработчика (DevTools) в браузере. Концепция Single Page Applications (SPA). Работа с React.

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. Принципы, паттерны и методологии разработки программного обеспечения : учебник для вузов / С. А. Чернышев. — Москва : Издательство Юрайт, 2025. — 176 с. — (Высшее образование). — ISBN 978-5-534-14383-6. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/567946.

Дополнительная литература:

1. Чернышев, С. А. Основы программирования на Python : учебник для вузов / С. А. Чернышев. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2025. — 349 с. — (Высшее образование). — ISBN 978-5-534-17139-6. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/567821.

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;

— 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.	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:		

Электронный документ

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) «Промышленная разработка» в рамках текущего контроля успеваемости используются такие виды учебной работы, как лекции, семинары, контрольные работы, домашние задания, 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.

Участие в семинаре (аудиторная работа) – активная работа студента на семинаре, его ответы на вопросы преподавателя и участие в дискуссии.

Для успешного участия в семинаре студентам рекомендуется заранее ознакомиться с темой обсуждения, прочитать необходимые материалы и подготовить вопросы. Важно активно слушать и вовлекаться в дискуссию, высказывая свои мнения и аргументируя их. При ответах на вопросы преподавателя стоит быть уверенным, четким и логичным, опираясь на изученный материал. Также полезно поддерживать диалог с однокурсниками, чтобы обогатить обсуждение и расширить свои знания.

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 курсу и презентация результатов.

Для успешной подготовки к проекту: четко определите цели и задачи проекта, распределите роли и обязанности между участниками, а также установите сроки выполнения каждой части работы. Регулярно проводите встречи для обсуждения прогресса и решения возникающих вопросов.

Самостоятельная работа – работа студентов, направленная на углубленное изучение отдельных тем и вопросов учебной дисциплины.

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) «Промышленная разработка»

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:

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. 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 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
6	Хорошо	Зачтено	

Десятибалльная оценка	Пятибалльная оценка	Оценка за зачет	Общая характеристика результата обучения по дисциплине (модулю)
			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.
5	Удовлетворительно	Зачтено	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	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) «Промышленная разработка» оценивается следующим образом:

Активность	Вес	Количество	Описание
Домашние задания	20%	13	Набор задач по темам недели
Аудиторная работа	15%	14	Активная работа студента на семинаре
Контрольные работы	30%	3	Письменная работа с набором задач, которые нужно решить за ограниченное время
Зачет с оценкой	35%	1	Защита итогового проекта

Формула расчёта итоговой оценки по дисциплине «Промышленная разработка»: «0,2 × среднее за домашние задания + 0,15 × аудиторная работа + 0,3 × среднее за контрольные работы + 0,35 × зачет с оценкой».

Текущий контроль успеваемости обучающихся по дисциплине (модулю)

Примерные домашние задания

Домашнее задание: Командная разработка

1. **Создание репозитория и веток**
   - Создайте репозиторий на GitLab.
   - Сделайте main ветку защищенной.
   - Создайте feature/calculator ветку и добавьте простой калькулятор (сложение, вычитание).

2. **Unit-тесты**
   - Напишите 2 unit-теста для функций калькулятора (например, test_addition(), test_subtraction()).
   - Запустите тесты локально и убедитесь, что они проходят.

3. **Merge Request (MR) и CI**
   - Создайте Merge Request из feature/calculator в main.
   - Назначьте MR на преподавателя.
   - Настройте .gitlab-ci.yml для запуска тестов при push.

4. **Code Review**
   - Внесите правки, если преподаватель оставил комментарии.
   - Перезапустите pipeline после изменений.

5. **Слияние и итог**
   - После аппрува выполните слияние (merge).
   - Убедитесь, что изменения появились в main.

Домашнее задание: Проектирование ПО

1. **Класс с ООП**
   - Создайте класс BankAccount с методами deposit(), withdraw() и get_balance().
   - Используйте инкапсуляцию (приватные поля).

2. **Обработка исключений**
   - Добавьте проверки:
     - withdraw() не должен списывать больше, чем есть на счету.
     - deposit() не должен принимать отрицательные значения.
   - Выбрасывайте исключения (InsufficientFundsError, InvalidDepositError).

3. **Mock-тестирование**
   - Напишите тест, где BankAccount зависит от Logger (mock Logger).
   - Проверьте, что при withdraw() вызывается log_transaction().

4. **Тесты на исключения**
   - Напишите тесты, которые проверяют:

     - Вызов withdraw(1000) при балансе 500.
     - Вызов deposit(-100).

5. **Рефакторинг по SOLID**

— Выделите интерфейс TransactionLoggable.
— Проверьте, что тесты работают после изменений.

Домашнее задание: Веб-API + Frontend

1. **REST API**
   — Создайте 3 эндпоинта:
      - GET /tasks — список задач.
      - POST /tasks — добавить задачу.
      - DELETE /tasks/{id} — удалить задачу.
   — Используйте FastAPI/Flask/Express.
2. **ORM и БД**
   — Подключите SQLAlchemy/TypeORM/Sequelize.
   — Создайте модель Task (id, title, status).
   — Реализуйте запросы:
      - Все задачи со статусом "не выполнено".
      - Задачи по ID.
3. **Документация и Docker**
   — Добавьте Swagger/OpenAPI.
   — Напишите Dockerfile и docker-compose.yml (API + БД).
4. **CI/CD**
   — Настройте GitLab CI:
      - Запуск тестов.
      - Сборка Docker-образа.
5. **Frontend (React)**
   — Создайте компонент TaskList, который:
      - Делает GET /tasks и выводит список.
      - Имеет кнопку "Удалить" (вызывает DELETE /tasks/{id}).

Примерные вопросы для подготовки к семинарам

Командная разработка

1. Какие основные этапы работы с Git (clone, commit, push, pull, merge)?
2. В чем разница между git merge и git rebase?
3. Как создать и переключиться на новую ветку в Git?
4. Что такое GitFlow и GitLab Flow? В чем их различия?
5. Как отменить последний коммит в Git?
6. Что такое .gitignore и зачем он нужен?
7. Как разрешить конфликт слияния (merge conflict) в Git?
8. Какие бывают стратегии слияния веток?
9. Как просмотреть историю коммитов (git log с фильтрами)?
10. Что такое git stash и в каких случаях он используется?
11. Зачем нужны теги (tags) в Git и как их создать?
12. Как работает git cherry-pick?
13. Что такое **unit-тесты** и зачем они нужны?
14. Как запустить unit-тесты в Python/Java/JavaScript?
15. Как настроить CI/CD в GitLab для автоматического запуска тестов?

Проектирование ПО

1. Какие **4 принципа ООП**? Кратко объясните каждый.

2. Что такое **инкапсуляция** и как она реализуется в коде?
3. В чем разница между **наследованием** и **композицией**?
4. Что такое **полиморфизм**? Приведите пример.
5. Какие **типы исключений** бывают (checked/unchecked в Java, Exception/Error в Python)?
6. Как правильно обрабатывать исключения (try-catch-finally)?
7. Что такое **прокси-объекты** и **mock-объекты**?
8. Зачем нужны **моки** в тестировании?
9. Как протестировать метод, который зависит от внешнего API?
10. Какие **аннотации/декораторы** используются для тестов (JUnit, pytest, Jest)?
11. Что такое **TDD (Test-Driven Development)**?
12. Какие **5 принципов SOLID**? Кратко объясните каждый.
13. Как **Dependency Injection** помогает в тестировании?
14. В чем разница между **статическим и динамическим полиморфизмом**?
15. Как **рефакторинг** улучшает код? Приведите пример.

Разработка веб-API

1. Какие **HTTP-методы** чаще всего используются в REST API?
2. В чем разница между **REST и GraphQL**?
3. Как структурировать **URL эндпоинтов** в REST API?
4. Что такое **DTO (Data Transfer Object)** и зачем он нужен?
5. Как работает **JWT-аутентификация** в API?
6. Какие **коды состояния HTTP** вы знаете? (200, 201, 400, 401, 404, 500)
7. Как **ORM** упрощает работу с БД? (SQLAlchemy, Hibernate, TypeORM)
8. Что такое **миграции БД** и зачем они нужны?
9. Как **документировать API** (Swagger/OpenAPI)?
10. Что делает **Docker** и зачем он нужен?
11. Как написать Dockerfile для веб-приложения?
12. Что такое **docker-compose** и чем он отличается от Dockerfile?
13. Как работает **CI/CD** в веб-разработке?
14. Что такое **интеграционное тестирование** и как его проводить?
15. Как **развернуть API** на облачном сервере (AWS, Heroku, VPS)?

Frontend-разработка

1. Какие **основные теги HTML** вы знаете? (, , <a>, <input>)
2. Как **подключить CSS** к HTML (inline, internal, external)?
3. Что такое **CSS-селекторы** (класс, id, тег, атрибут)?
4. Как работает **Flexbox** и **Grid** в CSS?
5. Какие **типы данных** есть в JavaScript (number, string, boolean, object)?
6. Что такое **DOM** и как с ним работать?
7. Как **открыть DevTools** в браузере и для чего они нужны?
8. Что такое **SPA (Single Page Application)** и как оно работает?
9. В чем разница между **React, Angular и Vue**?
10. Как создать **компонент в React** (функциональный и классовый)?
11. Что такое **JSX** и зачем он нужен?
12. Как **передавать props** между компонентами в React?
13. Что такое **хуки (hooks) в React** (useState, useEffect)?
14. Как **отправить HTTP-запрос** из React (fetch, axios)?
15. Как **развернуть фронтенд-приложение** (Netlify, Vercel, GitHub Pages)?

Примерные задания по контрольным работам

Контрольная работа 1

1. Командная разработка (Git, Unit-тесты, CI/CD)

1. **Git:** Создайте новый репозиторий, сделайте 3 коммита в ветке feature/login, затем смержите её в main через Merge Request.
2. **Git:** Разрешите конфликт слияния между двумя ветками (имитируйте изменение одного файла в обеих ветках).
3. **Unit-тесты:** Напишите 2 теста для функции, проверяющей валидность email-адреса (на языке Python/Java/JS).
4. **CI/CD:** Настройте .gitlab-ci.yml, чтобы тесты запускались автоматически при push в репозиторий.
5. **GitFlow:** Опишите, в каких случаях создаются ветки feature/, release/, hotfix/ в GitFlow.

2. Проектирование ПО (ООП, Исключения, Mock-тесты)

6. **ООП:** Создайте класс BankAccount с методами deposit(), withdraw(), getBalance(), соблюдая инкапсуляцию.
7. **ООП:** Реализуйте наследование: класс SavingsAccount расширяет BankAccount с добавлением метода addInterest().
8. **Исключения:** Добавьте в BankAccount обработку InsufficientFundsException при попытке снять больше, чем есть на счету.
9. **Mock-тесты:** Напишите тест для BankAccount, используя mock для внешнего сервиса проверки лимита снятия.
10. **SOLID:** Проведите рефакторинг класса BankAccount, применив **принцип единственной ответственности**.

3. Разработка веб-API (REST, Docker, ORM, CI)

11. **REST API:** Создайте эндпоинт GET /api/users, возвращающий JSON-список пользователей (можно mock-данные).
12. **ORM:** Напишите модель Product (поля: id, name, price) и 2 SQL-запроса через ORM: выбор всех товаров и фильтр по цене.
13. **Docker:** Напишите Dockerfile для развертывания API и docker-compose.yml с подключением PostgreSQL.
14. **Swagger:** Задокументируйте ваш API в OpenAPI (пример для /api/users).
15. **Интеграционные тесты:** Напишите тест, проверяющий статус-код и структуру JSON для GET /api/users.

4. Frontend-разработка (HTML/CSS, React, DevTools)

16. **HTML/CSS:** Сверстайте кнопку с hover-эффектом, меняющую цвет и добавляющую тень.
17. **JavaScript:** Напишите функцию, которая фильтрует массив чисел, оставляя только чётные.
18. **DevTools:** Используя браузерные инструменты, исправьте ошибку в верстке (например, перекрытие элементов).
19. **React:** Создайте компонент Counter с кнопками "+" и "-", изменяющими состояние.
20. **React:** Напишите код для загрузки данных с API (fetch/axios) и отображения их в списке.

Примерное описание и критерии оценивания к проекту
Проектное задание: Разработка веб-API для управления задачами (Task

Manager API)

Цель проекта

Разработать RESTful API для управления задачами (CRUD) с использованием Python/Java/Node.js, Docker, PostgreSQL и автоматизированным CI/CD-пайплайном.

Этапы выполнения

1. Подготовка и проектирование

- Создать репозиторий в GitLab/GitHub.
- Определить структуру API (эндпоинты, модели данных).
- Написать OpenAPI-документацию (Swagger/Postman).

2. Разработка API

- Реализовать CRUD-эндпоинты для задач (GET /tasks, POST /tasks, PUT /tasks/{id}, DELETE /tasks/{id}).
- Добавить фильтрацию задач по статусу (GET /tasks?status=completed).
- Подключить PostgreSQL через ORM (SQLAlchemy/Hibernate/TypeORM).

3. Тестирование

- Написать unit-тесты для сервисов (например, TaskService).
- Провести интеграционные тесты (например, проверка статуса 404 при удалении несуществующей задачи).
- Настроить mock-объекты для тестирования внешних зависимостей.

4. Деплой и CI/CD

- Создать Dockerfile и docker-compose.yml для развертывания API и БД.
- Настроить GitLab CI/CD для автоматического запуска тестов и деплоя на сервер.

5. Документирование

- Обновить Swagger-документацию.
- Написать README с инструкцией по запуску.

Критерии оценивания

Категория	Критерий	Баллы
Работа с Git	Корректное использование веток, коммитов, MR/PR.	1
ООП и код	Чистая архитектура, SOLID, обработка исключений.	1
API	Работоспособность эндпоинтов, валидация данных, документация.	2
Тестирование	Unit-тесты (≥80% покрытия), интеграционные тесты.	2
Docker & CI/CD	Корректная настройка контейнеров и пайплайна.	2

Электронный документ

Категория	Критерий	Баллы
Документация	Четкий README, Swagger, комментарии в коде.	2
Дополнительно	Аутентификация (JWT), пагинация, кеширование (Redis).	2

Максимальный балл: 12.

Критерии защиты проекта

1. **Демонстрация работы API**
   - — Показ всех эндпоинтов через Postman/Swagger.
   - — Примеры запросов и ответов.
2. **Объяснение архитектуры**
   - — Почему выбраны такие модели данных?
   - — Как организованы слои приложения (контроллеры, сервисы, репозитории)?
3. **Анализ тестов**
   - — Какие сценарии покрыты unit- и интеграционными тестами?
   - — Как использовались mock-объекты?
4. **CI/CD и Docker**
   - — Как настроен пайплайн? Какие этапы в нем есть?
   - — Как запустить проект локально и в продакшене?
5. **Ответы на вопросы**
   - — Как можно улучшить API (например, добавить авторизацию)?
   - — Какие были сложности и как их решили?

Примеры технологий

- **Backend:** Python (FastAPI/Flask), Java (Spring Boot), Node.js (Express/NestJS).
- **База данных:** PostgreSQL, MongoDB.
- **Тестирование:** pytest/JUnit/Jest, Postman/Newman.
- **CI/CD:** GitLab CI, GitHub Actions.

Задания для промежуточной аттестации по дисциплине (модулю)

№ п/п	Задание	Ответ	Компетенция
1.	Как называется ветка, в которую сливаются готовые фичи перед релизом?	main / master	УК-6
2.	Какой инструмент подменяет реальные зависимости в тестах?	Mock-объект	ПК-2
3.	Какой принцип ООП запрещает прямой доступ к данным класса извне?	Инкапсуляция	ПК-8
4.	Какой блок в Python/Java обрабатывает ошибки без прерывания программы?	try-catch / try-except	УК-6
5.	Какой HTTP-метод изменяет существующий ресурс?	PUT / PATCH	ОПК-2
6.	Какой файл описывает multi-контейнерные приложения?	docker-compose.yml	ПК-3
7.	Какой хук React управляет состоянием компонента?	useState	ПК-2
8.	В какой вкладке браузера можно отлаживать CSS?	Elements / Inspector	ОПК-2
9.	Какой фреймворк Google использует концепцию SPA?	Angular	УК-6

Электронный документ