Автономная некоммерческая организация высшего образования
«МОСКОВСКИЙ МЕЖДУНАРОДНЫЙ УНИВЕРСИТЕТ»

Рабочая программа дисциплины

Разработка клиент-серверных приложений

Направление подготовки	Информатика и вычислительная техника
Код	09.03.01
Направленность (профиль)	Автоматизированные системы обработки информации и управления
Квалификация выпускника	бакалавр

Москва
2026

1. Перечень кодов компетенций, формируемых дисциплиной в процессе освоения образовательной программы

Группа компетенций	Категория компетенций	Код
Профессиональные	-	ПК-1
Профессиональные	-	ПК-2
Профессиональные	-	ПК-3

2. Компетенции и индикаторы их достижения

Код компетенции	Формулировка компетенции	Индикаторы достижения компетенции
ПК-1	**ПК-1.** Способен к разработке требований к программному обеспечению и комплекта проектной документации информационных систем в соответствии с требованиями нормативно-технической документации	**ПК-1.1.** Знает методы представления данных и разработки основного алгоритма и комплекса программ; тестирования и верификации процедур обработки данных **ПК-1.2.** Умеет составлять комплект проектной документации информационных систем: технического задания, технического предложения, эскизного проекта, технического и рабочего проектов **ПК-1.3.** Владеет навыками выполнения мероприятий контроля соответствия технологических требований и инновационных предложений при разработке программного обеспечения
ПК-2	**ПК-2.** Способен анализировать информационное обеспечение систем среднего и крупного масштаба сложности, реализовать инновационные решения по их интеграции, комплексному развитию и функциональной направленности	**ПК-2.1.** Знает методы подбора, обработки и анализа научно-технической и патентной информации по тематике исследования с использованием специализированных баз данных и информационных технологий **ПК-2.2.** Умеет осуществлять выбор эффективных средств и способов обеспечения качества разработки программного обеспечения, его нагрузочное, конфигурационное и интеграционное тестирование **ПК-2.3.** Владеет навыками создания программных продуктов, сетевых решений, в результате экспериментального исследования информационных систем, их математического описания, цифровых технологий
ПК-3	**ПК-3.** Способен управлять проектами в	**ПК-3.1.** Знает первоначальные требования к ИС, перечень и

	области информационных технологий на основе полученных планов проектов, поддерживать процессы разработки и модернизации	последовательность работ, план мероприятий по управлению разработкой программного обеспечения, определение ресурсов, объемов работ по продвижению IT-продуктов **ПК-3.2.** Умеет проводить разработку архитектуры и прототипов информационных систем, их проектирование и дизайн, обеспечение кодирования на языках программирования, создавать пользовательскую документацию **ПК-3.3.** Владеет навыками проведения валидации программных продуктов и информационных систем, их интеграция с использованием инновационных аналитических методик

3. Описание планируемых результатов обучения по дисциплине

3.1. Описание планируемых результатов обучения по дисциплине

Планируемые результаты обучения по дисциплине представлены дескрипторами (знания, умения, навыки).

Дескрипторы по дисциплине	**Знать**	**Уметь**	**Владеть**
Код компетенции	**ПК-1**		
	- математический аппарат, методологию программирования и современные компьютерные технологии для решения практических задач получения, хранения, обработки и передачи информации; - основные идеи, понятия и методы, определяющие стиль написания, отладки и сопровождения программ; - характеристики основных парадигм программирования;	- использовать математический аппарат, методологию программирования и современные компьютерные технологии для решения практических задач получения, хранения, обработки и передачи информации; - применять современные компьютерные технологии для решения практических задач; - делать обоснованный выбор инструментария для решения прикладных задач;	- навыками использования математического аппарата, методологии программирования и современных компьютерных технологий для решения практических задач получения, хранения, обработки и передачи информации; - математическим аппаратом для построения вычислительных моделей

			практических задач; - навыками использования стандартных алгоритмических моделей для решения задач хранения и обработки информации
Код компетенции	**ПК-2**		
	Основы операционных систем: Знать архитектуру и основные компоненты операционных систем (например, Windows, Linux). Знать принципы работы файловых систем и управления процессами. Сетевые протоколы: Знать основные сетевые протоколы (TCP/IP, HTTP, HTTPS, FTP) и их функции. Знать принципы работы сетевых устройств (маршрутизаторов, коммутаторов, брандмауэров). Инструменты конфигурирования: Знать инструменты и утилиты для конфигурирования операционных систем и сетевых устройств (например, командная строка, PowerShell, SSH). Безопасность: Знать основные принципы и методы обеспечения	Конфигурировать операционные системы: Уметь устанавливать и настраивать операционные системы для веб-разработки. Уметь управлять пользователями и правами доступа в операционных системах. Настраивать сетевые устройства: Уметь конфигурировать маршрутизаторы и коммутаторы для обеспечения сетевой связности. Уметь настраивать брандмауэры для защиты веб-приложений. Использовать инструменты командной строки: Уметь использовать командную строку для выполнения задач администрирования и конфигурирования. Уметь применять скрипты для автоматизации процессов конфигурирования. Обеспечивать безопасность систем: Уметь применять методы	Практическими навыками конфигурирования: Владеть навыками установки и настройки операционных систем в средах веб-разработки. Владеть навыками конфигурирования сетевых устройств для оптимизации работы веб-приложений. Работой с документацией: Владеть умением читать и интерпретировать техническую документацию по операционным системам и сетевым устройствам. Решением проблем: Владеть навыками диагностики и устранения неполадок в операционных

	безопасности операционных систем и сетевых устройств.	шифрования и аутентификации для защиты данных. Уметь проводить аудит безопасности и выявлять уязвимости в системах.	системах и сетевых устройствах. Работой в команде: Владеть навыками взаимодействия с другими специалистами (разработчиками, системными администраторам и) для эффективного конфигурировани я и поддержки веб-приложений.
Код компетенции	**ПК-3**		
	- Знать основы математического аппарата Понимать ключевые математические концепции и методы, используемые для решения задач в области обработки информации, такие как алгебра, статистика и дискретная математика. - Знать методологию программирования Осознавать основные принципы и парадигмы программирования, включая объектно-ориентированное, функциональное и процедурное программирование. - Знать современные компьютерные технологии Иметь представление о современных технологиях и инструментах для	- Уметь применять математические методы Способность использовать математические модели и алгоритмы для решения практических задач, связанных с получением и обработкой информации. - Уметь разрабатывать программные решения Умение создавать программное обеспечение для автоматизации процессов получения, хранения и обработки данных с использованием различных языков программирования. - Уметь работать с компьютерными технологиями Способность эффективно использовать современные инструменты и технологии для управления данными, включая системы управления базами данных (СУБД) и средства визуализации данных.	- Владеть навыками программировани я Умение писать код на нескольких языках программировани я (например, Python, Java, C++) для решения задач обработки информации. - Владеть инструментами анализа данных Освоение специализированн ых программных средств для анализа, обработки и визуализации данных, таких как Excel, R или Tableau. - Владеть навыками проектирования

	работы с данными, таких как базы данных, облачные решения и языки программирования.		систем Умение разрабатывать архитектуру информационных систем, учитывая требования к получению, хранению и передаче информации.

4. Место дисциплины (модуля) в структуре образовательной программы

Дисциплина «Разработка клиент-серверных приложений» относится к части, формируемой участниками образовательных отношений учебного плана ОПОП.

Данная дисциплина взаимосвязана с другими дисциплинами, такими как: «Объектно-ориентированное программирование», «Введение в инженерную деятельность», «Операционные системы и среды», «Базы данных», «Основы использования и конфигурирования 1С: Предприятие».

В рамках освоения программы бакалавриата выпускники готовятся к решению задач профессиональной деятельности следующих типов:

Профиль (направленность) программы установлена путем ее ориентации на сферу профессиональной деятельности выпускников: Автоматизированные системы обработки информации и управления

5. Объем дисциплины

Виды учебной работы	***Формы обучения***
	Очно-заочная
Общая трудоемкость: зачетные единицы/часы	5/180
Контактная работа:	
Занятия лекционного типа	12
Занятия семинарского типа	20
Промежуточная аттестация: **экзамен**	9
Самостоятельная работа (СРС)	139

6. Содержание дисциплины (модуля), структурированное по темам / разделам с указанием отведенного на них количества академических часов и видов учебных занятий

6.1. Распределение часов по разделам/темам и видам работы

6.1.1. Очно-заочная форма обучения

№ п/п	**Раздел/тема**	**Виды учебной работы (в часах)**		
		Контактная работа		**Самостоятельная работа**
		Занятия лекционного	**Занятия семинарского типа**	

		типа						
		Лекции	*Иные учебные занятия*	*Практические занятия*	*Семинары*	*Лабораторные работы*	*Иные*	
1.	Основы веб-разработки и архитектуры клиент-сервер	1			2			15
2.	Клиентская разработка: HTML, CSS, JavaScript	1			2			15
3.	Фреймворки и библиотеки для клиентской разработки	1			2			15
4.	Серверная разработка: языки и платформы	2			2			15
5.	Работа с базами данных	1			2			15
6.	API и взаимодействие клиент-сервер	1			2			16
7.	Безопасность веб-приложений	1			2			16
8.	Развёртывание и эксплуатация приложений	2			3			16
9.	Оптимизация и тестирование приложений	2			3			16
	Промежуточная аттестация	**9**						
	Итого	**12**			**20**			**139**

6.2 Программа дисциплины, структурированная по темам / разделам

6.2.1 Содержание лекционного курса

№ п/п	**Наименование раздела дисциплины**	**Содержание раздела дисциплины**
1.	Основы веб-разработки и архитектуры клиент-сервер	Принципы работы клиент-серверной архитектуры. Разделение ответственности: клиентская и серверная части. Протоколы взаимодействия (HTTP/HTTPS, WebSocket). REST и SOAP — сравнение подходов. Одностраничные приложения (SPA) и многостраничные приложения (MPA). Микросервисная и монолитная архитектуры.

2.	Клиентская разработка: HTML, CSS, JavaScript	Основы HTML5: семантическая разметка, формы, мультимедиа. CSS3: стилизация, Flexbox, Grid, адаптивный дизайн. JavaScript: синтаксис, переменные, функции, объекты, массивы. DOM-манипуляции и обработка событий. ES6+: стрелочные функции, модули, промисы, async/await. БЭМ-методология для структурирования CSS.
3.	Фреймворки и библиотеки для клиентской разработки	React: компоненты, состояние, props, хуки. Vue.js: реактивность, компоненты, маршрутизация. Angular: модули, сервисы, директивы. Управление состоянием: Redux, Vuex. Маршрутизация: React Router, Vue Router. Сборщики проектов: Webpack, Vite.
4.	Серверная разработка: языки и платформы	Обзор языков: Node.js, Python (Django/Flask), Java (Spring), PHP (Laravel), C# (ASP.NET). Выбор технологии под задачу. Структура серверного приложения. Middleware и обработка запросов. Работа с HTTP-серверами (Express.js, Koa).
5.	Работа с базами данных	Реляционные БД: PostgreSQL, MySQL. Нереляционные БД: MongoDB, Redis. ORM и ODM: Sequelize, Mongoose, SQLAlchemy. Проектирование схемы данных. Индексы, транзакции, миграции. Кэширование данных (Redis).
6.	API и взаимодействие клиент-сервер	Проектирование RESTful API. Методы HTTP (GET, POST, PUT, DELETE, PATCH). Формат данных: JSON, XML. Аутентификация и авторизация: JWT, OAuth2, API-ключи. Документация API: Swagger/OpenAPI. GraphQL: запросы, мутации, резолверы.
7.	Безопасность веб-приложений	Основные угрозы: XSS, CSRF, SQL-инъекции, DDoS. Защита от атак: валидация ввода, санитаризация, CSP. HTTPS и SSL/TLS. Хранение паролей: хеширование (bcrypt). CORS и политика безопасности. Аудит безопасности и пентест.
8.	Развёртывание и эксплуатация приложений	Окружения: разработка, тестирование, продакшн. Контейнеризация: Docker, Docker Compose. Оркестрация: Kubernetes. CI/CD: GitLab CI, GitHub Actions, Jenkins. Облачные платформы: AWS, Azure, GCP, Heroku. Мониторинг и логирование (Prometheus, Grafana, ELK).
9.	Оптимизация и	Виды тестирования: юнит, интеграционное, E2E.

	тестирование приложений	Инструменты: Jest, Mocha, Selenium, Cypress. Оптимизация производительности: минификация, бандлинг, ленивая загрузка. SEO-оптимизация клиентских приложений. Прогрессивные веб-приложения (PWA). Оценка производительности (Lighthouse, WebPageTest).

6.2.2 Содержание практических занятий

№ п/п	Наименование раздела дисциплины	Содержание практических занятий
1.	Основы веб-разработки и архитектуры клиент-сервер	Принципы работы клиент-серверной архитектуры. Разделение ответственности: клиентская и серверная части. Протоколы взаимодействия (HTTP/HTTPS, WebSocket). REST и SOAP — сравнение подходов. Одностраничные приложения (SPA) и многостраничные приложения (MPA). Микросервисная и монолитная архитектуры.
2.	Клиентская разработка: HTML, CSS, JavaScript	Основы HTML5: семантическая разметка, формы, мультимедиа. CSS3: стилизация, Flexbox, Grid, адаптивный дизайн. JavaScript: синтаксис, переменные, функции, объекты, массивы. DOM-манипуляции и обработка событий. ES6+: стрелочные функции, модули, промисы, async/await. БЭМ-методология для структурирования CSS.
3.	Фреймворки и библиотеки для клиентской разработки	React: компоненты, состояние, props, хуки. Vue.js: реактивность, компоненты, маршрутизация. Angular: модули, сервисы, директивы. Управление состоянием: Redux, Vuex. Маршрутизация: React Router, Vue Router. Сборщики проектов: Webpack, Vite.
4.	Серверная разработка: языки и платформы	Обзор языков: Node.js, Python (Django/Flask), Java (Spring), PHP (Laravel), C# (ASP.NET). Выбор технологии под задачу. Структура серверного приложения. Middleware и обработка запросов. Работа с HTTP-серверами (Express.js, Koa).
5.	Работа с базами данных	Реляционные БД: PostgreSQL, MySQL. Нереляционные БД: MongoDB, Redis. ORM и ODM: Sequelize, Mongoose, SQLAlchemy. Проектирование схемы данных. Индексы, транзакции, миграции. Кэширование данных (Redis).
6.	API и взаимодействие клиент-сервер	Проектирование RESTful API.

		Методы HTTP (GET, POST, PUT, DELETE, PATCH). Формат данных: JSON, XML. Аутентификация и авторизация: JWT, OAuth2, API-ключи. Документация API: Swagger/OpenAPI. GraphQL: запросы, мутации, резолверы.
7.	Безопасность веб-приложений	Основные угрозы: XSS, CSRF, SQL-инъекции, DDoS. Защита от атак: валидация ввода, санитаризация, CSP. HTTPS и SSL/TLS. Хранение паролей: хеширование (bcrypt). CORS и политика безопасности. Аудит безопасности и пентест.
8.	Развёртывание и эксплуатация приложений	Окружения: разработка, тестирование, продакшн. Контейнеризация: Docker, Docker Compose. Оркестрация: Kubernetes. CI/CD: GitLab CI, GitHub Actions, Jenkins. Облачные платформы: AWS, Azure, GCP, Heroku. Мониторинг и логирование (Prometheus, Grafana, ELK).
9.	Оптимизация и тестирование приложений	Виды тестирования: юнит, интеграционное, E2E. Инструменты: Jest, Mocha, Selenium, Cypress. Оптимизация производительности: минификация, бандлинг, ленивая загрузка. SEO-оптимизация клиентских приложений. Прогрессивные веб-приложения (PWA). Оценка производительности (Lighthouse, WebPageTest).

6.2.3 Содержание самостоятельной работы

№ п/п	**Наименование раздела дисциплины**	**Содержание самостоятельной работы**
1.	Основы веб-разработки и архитектуры клиент-сервер	Принципы работы клиент-серверной архитектуры. Разделение ответственности: клиентская и серверная части. Протоколы взаимодействия (HTTP/HTTPS, WebSocket). REST и SOAP — сравнение подходов. Одностраничные приложения (SPA) и многостраничные приложения (MPA). Микросервисная и монолитная архитектуры.
2.	Клиентская разработка: HTML, CSS, JavaScript	Основы HTML5: семантическая разметка, формы, мультимедиа. CSS3: стилизация, Flexbox, Grid, адаптивный дизайн. JavaScript: синтаксис, переменные, функции, объекты, массивы. DOM-манипуляции и обработка событий.

		ES6+: стрелочные функции, модули, промисы, async/await. БЭМ-методология для структурирования CSS.
3.	Фреймворки и библиотеки для клиентской разработки	React: компоненты, состояние, props, хуки. Vue.js: реактивность, компоненты, маршрутизация. Angular: модули, сервисы, директивы. Управление состоянием: Redux, Vuex. Маршрутизация: React Router, Vue Router. Сборщики проектов: Webpack, Vite.
4.	Серверная разработка: языки и платформы	Обзор языков: Node.js, Python (Django/Flask), Java (Spring), PHP (Laravel), C# (ASP.NET). Выбор технологии под задачу. Структура серверного приложения. Middleware и обработка запросов. Работа с HTTP-серверами (Express.js, Koa).
5.	Работа с базами данных	Реляционные БД: PostgreSQL, MySQL. Нереляционные БД: MongoDB, Redis. ORM и ODM: Sequelize, Mongoose, SQLAlchemy. Проектирование схемы данных. Индексы, транзакции, миграции. Кэширование данных (Redis).
6.	API и взаимодействие клиент-сервер	Проектирование RESTful API. Методы HTTP (GET, POST, PUT, DELETE, PATCH). Формат данных: JSON, XML. Аутентификация и авторизация: JWT, OAuth2, API-ключи. Документация API: Swagger/OpenAPI. GraphQL: запросы, мутации, резолверы.
7.	Безопасность веб-приложений	Основные угрозы: XSS, CSRF, SQL-инъекции, DDoS. Защита от атак: валидация ввода, санитаризация, CSP. HTTPS и SSL/TLS. Хранение паролей: хеширование (bcrypt). CORS и политика безопасности. Аудит безопасности и пентест.
8.	Развёртывание и эксплуатация приложений	Окружения: разработка, тестирование, продакшн. Контейнеризация: Docker, Docker Compose. Оркестрация: Kubernetes. CI/CD: GitLab CI, GitHub Actions, Jenkins. Облачные платформы: AWS, Azure, GCP, Heroku. Мониторинг и логирование (Prometheus, Grafana, ELK).
9.	Оптимизация и тестирование приложений	Виды тестирования: юнит, интеграционное, E2E. Инструменты: Jest, Mocha, Selenium, Cypress. Оптимизация производительности: минификация, бандлинг, ленивая загрузка. SEO-оптимизация клиентских приложений. Прогрессивные веб-приложения (PWA). Оценка производительности (Lighthouse, WebPageTest).

7. Текущий контроль по дисциплине (модулю) в рамках учебных занятий

В рамках текущего контроля преподаватель самостоятельно может проводить следующие мероприятия:

№ п/п	Контролируемые разделы (темы)	Наименование оценочного средства
1.	Основы веб-разработки и архитектуры клиент-сервер	Устный опрос, проблемно-аналитические задания, дискуссия, реферат
2.	Клиентская разработка: HTML, CSS, JavaScript	Устный опрос, проблемно-аналитические задания
3.	Фреймворки и библиотеки для клиентской разработки	Устный опрос, проблемно-аналитические задания, тестирование
4.	Серверная разработка: языки и платформы	Устный опрос, Проблемно-аналитические задания, презентация
5.	Работа с базами данных	Устный опрос, проблемно-аналитические задания, тестирование
6.	API и взаимодействие клиент-сервер	Устный опрос, Проблемно-аналитические задания
7.	Безопасность веб-приложений	Устный опрос, Проблемно-аналитические задания
8.	Развёртывание и эксплуатация приложений	Устный опрос, Проблемно-аналитические задания
9.	Оптимизация и тестирование приложений	Устный опрос, проблемно-аналитические задания, тестирование

8. Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины (модуля)

8.1. Основная литература:

1. Позевалкин, В. В. Разработка веб-приложений на основе клиентских каркасов и библиотек : учебное пособие / В. В. Позевалкин, Н. Ф. Панова. — Оренбург : Оренбургский государственный университет, ЭБС АСВ, 2025. — 191 с. — ISBN 978-5-7410-3380-7. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/153228.html

2. Щенёва, Ю. Б. Проектирование и разработка клиент-серверных приложений. Ч.1 : учебное пособие / Ю. Б. Щенёва. — Рязань : Рязанский государственный радиотехнический университет, 2024. — 124 с. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/150308.html

8.2. Дополнительная литература:

1. Хабаров, А. Н. Разработка программных приложений : учебник / А. Н. Хабаров, А. Н. Ермакова. — Ставрополь : АГРУС, 2025. — 208 с. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/156614.html

2. Сычев, А. В. Теория и практика разработки современных клиентских веб-

приложений : учебное пособие / А. В. Сычев. — 4-е изд. — Москва : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2025. — 482 с. — ISBN 978-5-4497-0943-1. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/146402.html

8.3. Периодические издания

1. Вестник Московского государственного технического университета имени Н.Э. Баумана. Серия Естественные науки. ISSN 1812-3368. https://www.iprbookshop.ru/23124.html
2. Открытые Системы. СУБД. ISSN 1028-7493. https://www.iprbookshop.ru/76383.html
3. Информационные технологии моделирования и управления. ISSN 1813-9744. https://www.iprbookshop.ru/43350.html

9. Перечень ресурсов информационно-телекоммуникационной сети "Интернет" (далее - сеть "Интернет"), необходимых для освоения дисциплины (модуля)

1. Федеральный портал «Российское образование» http://www.edu.ru
2. Электронно-библиотечная система «Научная электронная библиотека eLIBRARY.RU» https://www.elibrary.ru
3. Электронно-библиотечная система IPR BOOKS https://www.iprbookshop.ru
4. Электронная библиотечная система https://biblioclub.ru

10. Методические указания для обучающихся по освоению дисциплины (модуля)

Успешное освоение данного курса базируется на рациональном сочетании нескольких видов учебной деятельности – лекций, семинарских занятий, самостоятельной работы. При этом самостоятельную работу следует рассматривать одним из главных звеньев полноценного высшего образования, на которую отводится значительная часть учебного времени.

Самостоятельная работа студентов складывается из следующих составляющих:

1. Работа с основной и дополнительной литературой, с материалами интернета и конспектами лекций;
2. Внеаудиторная подготовка к контрольным работам, выполнение докладов, рефератов и курсовых работ;
3. Выполнение самостоятельных практических работ;
4. Подготовка к экзаменам (зачетам) непосредственно перед ними.

Для правильной организации работы необходимо учитывать порядок изучения разделов курса, находящихся в строгой логической последовательности. Поэтому хорошее усвоение одной части дисциплины является предпосылкой для успешного перехода к следующей. Задания, проблемные вопросы, предложенные для изучения дисциплины, в том числе и для самостоятельного выполнения, носят междисциплинарный характер и базируются, прежде всего, на причинно-следственных связях между компонентами окружающего нас мира. Важным составляющим в изучении данного курса является решение ситуационных задач и работа над проблемно-аналитическими заданиями, что предполагает знание соответствующей научной терминологии и т.д.

Для лучшего запоминания материала целесообразно использовать индивидуальные особенности и разные виды памяти: зрительную, слуховую, ассоциативную. Успешному запоминанию также способствует приведение ярких свидетельств и наглядных примеров. Учебный материал должен постоянно повторяться и закрепляться.

При выполнении докладов, творческих, информационных, исследовательских проектов особое внимание следует обращать на подбор источников информации и методику работы с ними.

Для успешной сдачи экзамена (зачета) рекомендуется соблюдать следующие правила:

1. Подготовка к экзамену (зачету) должна проводиться систематически, в течение всего семестра.
2. Интенсивная подготовка должна начаться не позднее, чем за месяц до экзамена.
3. Время непосредственно перед экзаменом (зачетом) лучше использовать таким образом, чтобы оставить последний день свободным для повторения курса в целом, для систематизации материала и доработки отдельных вопросов.

На экзамене высокую оценку получают студенты, использующие данные, полученные в процессе выполнения самостоятельных работ, а также использующие собственные выводы на основе изученного материала.

Учитывая значительный объем теоретического материала, студентам рекомендуется регулярное посещение и подробное конспектирование лекций.

11. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень программного обеспечения и информационных справочных систем (при необходимости)

1. Microsoft Windows Server;
2. Семейство ОС Microsoft Windows;
3. Libre Office свободно распространяемый офисный пакет с открытым исходным кодом;
4. Информационно-справочная система: Система КонсультантПлюс (КонсультантПлюс);
5. Информационно-правовое обеспечение Гарант: Электронный периодический справочник «Система ГАРАНТ» (Система ГАРАНТ);

Перечень используемого программного обеспечения указан в п.12 данной рабочей программы дисциплины.

12. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю)

12.1. Учебная аудитория для проведения учебных занятий, предусмотренных образовательной программой, оснащенная оборудованием и техническими средствами обучения.

Специализированная мебель:

Комплект учебной мебели (стол, стул) по количеству обучающихся; комплект мебели для преподавателя; доска (маркерная).

Технические средства обучения:

Компьютер в сборе для преподавателя, проектор, экран, колонки

Перечень лицензионного программного обеспечения, в том числе отечественного производства:

Windows 10, КонсультантПлюс, Kaspersky Endpoint Security.

Перечень свободно распространяемого программного обеспечения:

Яндекс Браузер, LibreOffice, МТС Линк.

Подключение к сети «Интернет» и обеспечение доступа в электронную информационно-образовательную среду ММУ.

12.2. Помещение для самостоятельной работы обучающихся.

Специализированная мебель:

Комплект учебной мебели (стол, стул) по количеству обучающихся; комплект мебели для преподавателя; доска (маркерная).

Технические средства обучения:

Компьютер в сборе для преподавателя; компьютеры в сборе для обучающихся; колонки; проектор, экран.

Перечень лицензионного программного обеспечения, в том числе отечественного производства:

Windows Server 2016, Windows 10, Microsoft Office, КонсультантПлюс, Kaspersky Endpoint Security.

Перечень свободно распространяемого программного обеспечения:

Яндекс Браузер, LibreOffice, МТС Линк, Gimp, Paint.net, AnyLogic, Inkscape.

Помещение для самостоятельной работы обучающихся оснащено компьютерной техникой с возможностью подключения к сети "Интернет" и обеспечением доступа в электронную информационно-образовательную среду ММУ.

13.Образовательные технологии, используемые при освоении дисциплины

Для освоения дисциплины используются как традиционные формы занятий – лекции (типы лекций – установочная, вводная, текущая, заключительная, обзорная; виды лекций – проблемная, визуальная, лекция конференция, лекция консультация); и семинарские (практические) занятия, так и активные и интерактивные формы занятий - деловые и ролевые игры, решение ситуационных задач и разбор конкретных ситуаций.

На учебных занятиях используются технические средства обучения мультимедийной аудитории: компьютер, монитор, колонки, настенный экран, проектор, микрофон, пакет программ Microsoft Office для демонстрации презентаций и медиафайлов, видеопроектор для демонстрации слайдов, видеосюжетов и др. Тестирование обучаемых может осуществляться с использованием компьютерного оборудования университета.

13.1. В освоении учебной дисциплины используются следующие традиционные образовательные технологии:

- чтение проблемно-информационных лекций с использованием доски и видеоматериалов;
- семинарские занятия для обсуждения, дискуссий и обмена мнениями;
- контрольные опросы;
- консультации;
- самостоятельная работа студентов с учебной литературой и первоисточниками;
- подготовка и обсуждение рефератов (проектов), презентаций (научно-исследовательская работа);
- тестирование по основным темам дисциплины.

13.2. Активные и интерактивные методы и формы обучения

Из перечня видов: («*мозговой штурм», анализ НПА, анализ проблемных ситуаций, анализ конкретных ситуаций, инциденты, имитация коллективной профессиональной деятельности, разыгрывание ролей, творческая работа, связанная с освоением дисциплины, ролевая игра, круглый стол, диспут, беседа, дискуссия, мини-конференция и др.*) используются следующие:

- *диспут;*
- *анализ проблемных, творческих заданий, ситуационных задач;*
- *ролевая игра;*
- *круглый стол;*
- *мини-конференция;*

- *дискуссия;*
- *беседа.*

13.3. Особенности обучения инвалидов и лиц с ограниченными возможностями здоровья (ОВЗ)

При организации обучения по дисциплине учитываются особенности организации взаимодействия с инвалидами и лицами с ограниченными возможностями здоровья (далее – инвалиды и лица с ОВЗ) с целью обеспечения их прав. При обучении учитываются особенности их психофизического развития, индивидуальные возможности и при необходимости обеспечивается коррекция нарушений развития и социальная адаптация указанных лиц.

Выбор методов обучения определяется содержанием обучения, уровнем методического и материально-технического обеспечения, особенностями восприятия учебной информации студентов с инвалидностью и студентов с ограниченными возможностями здоровья и т.д. В образовательном процессе используются социально-активные и рефлексивные методы обучения, технологии социокультурной реабилитации с целью оказания помощи в установлении полноценных межличностных отношений с другими студентами, создании комфортного психологического климата в студенческой группе.

При обучении лиц с ограниченными возможностями здоровья электронное обучение и дистанционные образовательные технологии предусматривают возможность приема-передачи информации в доступных для них формах.

Обучающиеся из числа лиц с ограниченными возможностями здоровья обеспечены печатными и электронными образовательными ресурсами в формах, адаптированных к ограничениям их здоровья.

Приложение

Автономная некоммерческая организация высшего образования
«МОСКОВСКИЙ МЕЖДУНАРОДНЫЙ УНИВЕРСИТЕТ»

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДЛЯ ПРОВЕДЕНИЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ПО ДИСЦИПЛИНЕ

Разработка клиент-серверных приложений

Направление подготовки	Информатика и вычислительная техника
Код	09.03.01
Направленность (профиль)	Автоматизированные системы обработки информации и управления
Квалификация выпускника	бакалавр

Москва
2026

1. **Перечень кодов компетенций, формируемых дисциплиной в процессе освоения образовательной программы**

Группа компетенций	Категория компетенций	Код
Профессиональные	-	ПК-1
Профессиональные	-	ПК-2
Профессиональные	-	ПК-3

2. **Компетенции и индикаторы их достижения**

Код компетенции	Формулировка компетенции	Индикаторы достижения компетенции
ПК-1	**ПК-1.** Способен к разработке требований к программному обеспечению и комплекта проектной документации информационных систем в соответствии с требованиями нормативно-технической документации	**ПК-1.1.** Знает методы представления данных и разработки основного алгоритма и комплекса программ; тестирования и верификации процедур обработки данных **ПК-1.2.** Умеет составлять комплект проектной документации информационных систем: технического задания, технического предложения, эскизного проекта, технического и рабочего проектов **ПК-1.3.** Владеет навыками выполнения мероприятий контроля соответствия технологических требований и инновационных предложений при разработке программного обеспечения
ПК-2	**ПК-2.** Способен анализировать информационное обеспечение систем среднего и крупного масштаба сложности, реализовать инновационные решения по их интеграции, комплексному развитию и функциональной направленности	**ПК-2.1.** Знает методы подбора, обработки и анализа научно-технической и патентной информации по тематике исследования с использованием специализированных баз данных и информационных технологий **ПК-2.2.** Умеет осуществлять выбор эффективных средств и способов обеспечения качества разработки программного обеспечения, его нагрузочное, конфигурационное и интеграционное тестирование **ПК-2.3.** Владеет навыками создания программных продуктов, сетевых решений, в результате экспериментального исследования информационных систем, их математического описания, цифровых технологий
ПК-3	**ПК-3.** Способен управлять проектами в	**ПК-3.1.** Знает первоначальные требования к ИС, перечень и

	области информационных технологий на основе полученных планов проектов, поддерживать процессы разработки и модернизации	последовательность работ, план мероприятий по управлению разработкой программного обеспечения, определение ресурсов, объемов работ по продвижению IT-продуктов **ПК-3.2.** Умеет проводить разработку архитектуры и прототипов информационных систем, их проектирование и дизайн, обеспечение кодирования на языках программирования, создавать пользовательскую документацию **ПК-3.3.** Владеет навыками проведения валидации программных продуктов и информационных систем, их интеграция с использованием инновационных аналитических методик

3. Описание планируемых результатов обучения по дисциплине

3.1. Описание планируемых результатов обучения по дисциплине

Планируемые результаты обучения по дисциплине представлены дескрипторами (знания, умения, навыки).

Дескрипторы по дисциплине	**Знать**	**Уметь**	**Владеть**
Код компетенции	**ПК-1**		
	- математический аппарат, методологию программирования и современные компьютерные технологии для решения практических задач получения, хранения, обработки и передачи информации; - основные идеи, понятия и методы, определяющие стиль написания, отладки и сопровождения программ; - характеристики основных парадигм программирования;	- использовать математический аппарат, методологию программирования и современные компьютерные технологии для решения практических задач получения, хранения, обработки и передачи информации; - применять современные компьютерные технологии для решения практических задач; - делать обоснованный выбор инструментария для решения прикладных задач;	- навыками использования математического аппарата, методологии программирования и современных компьютерных технологий для решения практических задач получения, хранения, обработки и передачи информации; - математическим аппаратом для построения вычислительных моделей практических

			задач; - навыками использования стандартных алгоритмических моделей для решения задач хранения и обработки информации
Код компетенции	**ПК-2**		
	Основы операционных систем: Знать архитектуру и основные компоненты операционных систем (например, Windows, Linux). Знать принципы работы файловых систем и управления процессами. Сетевые протоколы: Знать основные сетевые протоколы (TCP/IP, HTTP, HTTPS, FTP) и их функции. Знать принципы работы сетевых устройств (маршрутизаторов, коммутаторов, брандмауэров). Инструменты конфигурирования: Знать инструменты и утилиты для конфигурирования операционных систем и сетевых устройств (например, командная строка, PowerShell, SSH). Безопасность: Знать основные принципы и методы обеспечения безопасности	Конфигурировать операционные системы: Уметь устанавливать и настраивать операционные системы для веб-разработки. Уметь управлять пользователями и правами доступа в операционных системах. Настраивать сетевые устройства: Уметь конфигурировать маршрутизаторы и коммутаторы для обеспечения сетевой связности. Уметь настраивать брандмауэры для защиты веб-приложений. Использовать инструменты командной строки: Уметь использовать командную строку для выполнения задач администрирования и конфигурирования. Уметь применять скрипты для автоматизации процессов конфигурирования. Обеспечивать безопасность систем: Уметь применять методы шифрования и	Практическими навыками конфигурирования: Владеть навыками установки и настройки операционных систем в средах веб-разработки. Владеть навыками конфигурирования сетевых устройств для оптимизации работы веб-приложений. Работой с документацией: Владеть умением читать и интерпретировать техническую документацию по операционным системам и сетевым устройствам. Решением проблем: Владеть навыками диагностики и устранения неполадок в операционных системах и

	операционных систем и сетевых устройств.	аутентификации для защиты данных. Уметь проводить аудит безопасности и выявлять уязвимости в системах.	сетевых устройствах. Работой в команде: Владеть навыками взаимодействия с другими специалистами (разработчиками, системными администраторами) для эффективного конфигурирования и поддержки веб-приложений.
Код компетенции	**ПК-3**		
	- Знать основы математического аппарата Понимать ключевые математические концепции и методы, используемые для решения задач в области обработки информации, такие как алгебра, статистика и дискретная математика. - Знать методологию программирования Осознавать основные принципы и парадигмы программирования, включая объектно-ориентированное, функциональное и процедурное программирование. - Знать современные компьютерные технологии Иметь представление о современных технологиях и инструментах для работы с данными,	- Уметь применять математические методы Способность использовать математические модели и алгоритмы для решения практических задач, связанных с получением и обработкой информации. - Уметь разрабатывать программные решения Умение создавать программное обеспечение для автоматизации процессов получения, хранения и обработки данных с использованием различных языков программирования. - Уметь работать с компьютерными технологиями Способность эффективно использовать современные инструменты и технологии для управления данными, включая системы управления базами данных (СУБД) и средства визуализации данных.	- Владеть навыками программирования Умение писать код на нескольких языках программирования (например, Python, Java, C++) для решения задач обработки информации. - Владеть инструментами анализа данных Освоение специализированных программных средств для анализа, обработки и визуализации данных, таких как Excel, R или Tableau. - Владеть навыками проектирования систем

	таких как базы данных, облачные решения и языки программирования.		Умение разрабатывать архитектуру информационных систем, учитывая требования к получению, хранению и передаче информации.

3.2. Критерии оценки знаний студентов

Шкала оценивания	**Индикаторы достижения**	**Показатели оценивания результатов обучения**
ОТЛИЧНО/Зачтено	Знает:	- студент глубоко и всесторонне усвоил материал, уверенно, логично, последовательно и грамотно его излагает, опираясь на знания основной и дополнительной литературы, - на основе системных научных знаний делает квалифицированные выводы и обобщения, свободно оперирует категориями и понятиями.
	Умеет:	- студент умеет самостоятельно и правильно решать учебно-профессиональные задачи или задания, уверенно, логично, последовательно и аргументировано излагать свое решение, используя научные понятия, ссылаясь на нормативную базу.
	Владеет:	- студент владеет рациональными методами (с использованием рациональных методик) решения сложных профессиональных задач, представленных деловыми играми, кейсами и т.д.; При решении продемонстрировал навыки - выделения главного, - связкой теоретических положений с требованиями руководящих документов, - изложения мыслей в логической последовательности, - самостоятельного анализа факты, событий, явлений, процессов в их взаимосвязи и диалектическом развитии.
ХОРОШО/Зачтено	Знает:	- студент твердо усвоил материал, достаточно грамотно его излагает, опираясь на знания основной и дополнительной литературы, - затрудняется в формулировании квалифицированных выводов и обобщений, оперирует категориями и понятиями, но не всегда правильно их верифицирует.
	Умеет:	- студент умеет самостоятельно и в основном правильно решать учебно-профессиональные задачи или задания, уверенно, логично, последовательно и аргументировано излагать свое решение, не в полной мере используя научные понятия и ссылки на нормативную базу.
	Владеет:	- студент в целом владеет рациональными методами

		решения сложных профессиональных задач, представленных деловыми играми, кейсами и т.д.; При решении смог продемонстрировать достаточность, но не глубинность навыков - выделения главного, - изложения мыслей в логической последовательности. - связки теоретических положений с требованиями руководящих документов, - самостоятельного анализа факты, событий, явлений, процессов в их взаимосвязи и диалектическом развитии.
УДОВЛЕТВОРИТЕЛЬНО/Зачтено	Знает:	- студент ориентируется в материале, однако затрудняется в его изложении; - показывает недостаточность знаний основной и дополнительной литературы; - слабо аргументирует научные положения; - практически не способен сформулировать выводы и обобщения; - частично владеет системой понятий.
	Умеет:	- студент в основном умеет решить учебно-профессиональную задачу или задание, но допускает ошибки, слабо аргументирует свое решение, недостаточно использует научные понятия и руководящие документы.
	Владеет:	- студент владеет некоторыми рациональными методами решения сложных профессиональных задач, представленных деловыми играми, кейсами и т.д.; При решении продемонстрировал недостаточность навыков - выделения главного, - изложения мыслей в логической последовательности. - связки теоретических положений с требованиями руководящих документов, - самостоятельного анализа факты, событий, явлений, процессов в их взаимосвязи и диалектическом развитии.
НЕУДОВЛЕТВОРИТЕЛЬНО/Не зачтено	Знает:	- студент не усвоил значительной части материала; - не может аргументировать научные положения; - не формулирует квалифицированных выводов и обобщений; - не владеет системой понятий.
	Умеет:	студент не показал умение решать учебно-профессиональную задачу или задание.
	Владеет:	не выполнены требования, предъявляемые к навыкам, оцениваемым “удовлетворительно”.

При ответе на вопросы в рамках прохождения промежуточной аттестации (зачет/ зачет с оценкой/ экзамен) допускается вольная формулировка ответа, по смыслу раскрывающая содержание ответа, указанного в фонде оценочных средств, в качестве верного ответа.

4. Типовые контрольные задания (закрытого, открытого и иного типа) для проведения промежуточной аттестации, необходимые для оценки достижения компетенции, соотнесенной с результатами обучения по дисциплине

ТЕСТИРОВАНИЕ

8 СЕМЕСТР
ПК-1

1. Какой архитектурный стиль предполагает разделение системы на две логические части — поставщика услуг (сервер) и потребителя услуг (клиент)?
а) Многоуровневая архитектура
б) Клиент-серверная архитектура
в) Компонентная архитектура
г) Монолитная архитектура
Правильный ответ: б

2. Какой протокол является основным для веб-сервисов и обеспечивает передачу гипертекстовых документов?
а) FTP
б) SMTP
в) HTTP/HTTPS
г) DNS
Правильный ответ: в

3. Что из перечисленного является основной характеристикой «тонкого клиента»?
а) Выполняет всю логику обработки данных на своей стороне
б) Основная логика приложения выполняется на сервере, клиент отвечает только за отображение
в) Имеет собственное хранилище данных
г) Требует установки тяжелого программного обеспечения
Правильный ответ: б

4. Какой протокол обеспечивает надежную передачу данных с установлением соединения и контролем доставки?
а) UDP
б) TCP
в) IP
г) ARP
Правильный ответ: б

5. Как называется стандартный формат обмена данными в RESTful API, использующий структуру «ключ-значение»?
а) XML
б) YAML
в) JSON

г) CSV
Правильный ответ: в

6. Какой HTTP-метод согласно REST-конвенциям обычно используется для получения данных без изменения состояния сервера?
а) POST
б) PUT
в) DELETE
г) GET
Правильный ответ: г

7. Какой HTTP-статус код означает успешное выполнение запроса (OK)?
а) 200
б) 404
в) 500
г) 403
Правильный ответ: а

8. Какой паттерн проектирования предполагает наличие промежуточного слоя между клиентом и сервером, агрегирующего данные из нескольких источников?
а) Singleton
б) Factory
в) BFF (Backend for Frontend)
г) Observer
Правильный ответ: в

9. Что из перечисленного НЕ относится к протоколам транспортного уровня модели OSI?
а) TCP
б) UDP
в) HTTP
г) SCTP
Правильный ответ: в

10. Как в клиент-серверном приложении называется уникальный идентификатор, позволяющий серверу различать подключения от разных клиентов?
а) IP-адрес
б) MAC-адрес
в) Сокет (Socket)
г) Порт
Правильный ответ: в

11. Какое требование предъявляется к серверу при реализации многопользовательского режима для обработки одновременных запросов нескольких клиентов?
а) Наличие графического интерфейса
б) Поддержка многопоточности или асинхронности
в) Обязательное использование протокола UDP
г) Наличие отдельного монитора для каждого клиента
Правильный ответ: б

12. Какой архитектурный подход предполагает разбиение приложения на небольшие независимые сервисы, взаимодействующие через сеть?

а) Монолитная архитектура
б) Микросервисная архитектура
в) Клиент-серверная архитектура
г) Модульная архитектура
Правильный ответ: б

13. Для чего используется протокол WebSocket в клиент-серверных приложениях?
а) Для однократной передачи большого объема данных
б) Для установления постоянного двунаправленного канала связи в реальном времени
в) Для шифрования трафика
г) Для маршрутизации пакетов
Правильный ответ: б

14. Что такое «сокет» (socket) в контексте сетевого программирования?
а) Тип процессора
б) Комбинация IP-адреса и номера порта, являющаяся конечной точкой соединения
в) Сетевой кабель
г) Операционная система
Правильный ответ: б

15. Какой код состояния HTTP указывает на отсутствие запрашиваемого ресурса (Not Found)?
а) 200
б) 302
в) 404
г) 500
Правильный ответ: в

8 СЕМЕСТР
ПК-2

1. Какой HTTP-метод используется для частичного обновления ресурса?
а) PUT
б) POST
в) PATCH
г) DELETE
Правильный ответ: в

2. Что означает принцип «Stateless» (отсутствие состояния) в REST архитектуре?
а) Сервер не хранит данные о клиенте между запросами
б) Клиент не может отправлять данные на сервер
в) Сервер всегда возвращает один и тот же ответ
г) Приложение не использует базы данных
Правильный ответ: а

3. Какой HTTP-метод используется для полной замены существующего ресурса или создания нового по указанному URL?
а) POST
б) PATCH
в) PUT
г) GET
Правильный ответ: в

4. Какая библиотека на языке Python чаще всего используется для отправки HTTP-запросов?
а) Django
б) Flask
в) Requests
г) NumPy
Правильный ответ: в

5. Какой код состояния HTTP возвращается при успешном создании нового ресурса (обычно в ответ на POST-запрос)?
а) 200
б) 201
в) 202
г) 204
Правильный ответ: б

6. Что из перечисленного является обязательным компонентом REST-запроса к API?
а) Тело запроса
б) HTTP-метод и URL
в) Заголовок Authorization
г) XML-документ
Правильный ответ: б

7. Для чего используется заголовок «Content-Type» в HTTP-запросе?
а) Для указания длины сообщения
б) Для указания формата данных в теле запроса (например, application/json)
в) Для аутентификации пользователя
г) Для указания кодировки символов
Правильный ответ: б

8. Какой метод библиотеки Requests на Python выполняет DELETE-запрос?
а) requests.get()
б) requests.post()
в) requests.delete()
г) requests.remove()
Правильный ответ: в

9. Какая аннотация в Spring Framework используется для создания REST-контроллера?
а) @Controller
б) @RestController
в) @Service
г) @Component
Правильный ответ: б

10. Какой код состояния HTTP возвращается при отсутствии прав доступа к ресурсу (Forbidden)?
а) 401
б) 403
в) 404
г) 405
Правильный ответ: б

11. Что означает код состояния HTTP 500?
а) Успешное выполнение запроса
б) Ресурс не найден
в) Внутренняя ошибка сервера
г) Требуется аутентификация
Правильный ответ: в

12. Как в REST API обычно передаются параметры фильтрации (например, «показать пользователей старше 25 лет»)?
а) В теле запроса
б) В заголовках запроса
в) В строке запроса (query parameters)
г) В методе запроса
Правильный ответ: в

13. Какая аннотация в Spring Framework используется для маппинга HTTP-запросов на методы контроллера?
а) @GetMapping
б) @RequestMapping
в) @PostMapping
г) Все перечисленные
Правильный ответ: г

14. Какой тип аутентификации предполагает передачу логина и пароля в открытом виде (кодированных в Base64) в заголовке Authorization?
а) OAuth 2.0
б) Bearer Token
в) Basic Authentication
г) API Key
Правильный ответ: в

15. Для чего используется аннотация @RequestBody в Spring Framework?
а) Для привязки параметра метода к переменной пути URL
б) Для привязки параметра метода к телу HTTP-запроса
в) Для привязки параметра метода к query-параметру
г) Для внедрения зависимости
Правильный ответ: б

8 СЕМЕСТР
ПК-3

1. Какой протокол НЕ имеет встроенного механизма контроля доставки и не требует установления соединения?
а) TCP
б) HTTP
в) UDP
г) HTTPS
Правильный ответ: в

2. В чем основное преимущество использования асинхронных вызовов при разработке клиент-серверных приложений?
а) Упрощение кода

б) Отсутствие необходимости в обработке ошибок
в) Клиент не блокируется в ожидании ответа сервера
г) Автоматическая аутентификация
Правильный ответ: в

3. Какая библиотека на JavaScript/Python чаще всего используется для выполнения асинхронных HTTP-запросов?
а) NumPy
б) Pandas
в) Axios (JS) / asyncio + aiohttp (Python)
г) Matplotlib
Правильный ответ: в

4. Что характерно для синхронного взаимодействия клиента и сервера?
а) Клиент продолжает работу, не дожидаясь ответа
б) Клиент блокируется до получения ответа от сервера
в) Сервер отправляет ответ асинхронно через callback
г) Клиент и сервер обмениваются сообщениями через шину
Правильный ответ: б

5. Какой инструмент используется для тестирования REST API без написания кода?
а) PyCharm
б) Postman
в) Git
г) Docker
Правильный ответ: б

6. Какая модель согласованности гарантирует, что после завершения записи все последующие чтения будут видеть эту запись?
а) Eventual consistency (конечная согласованность)
б) Strong consistency (строгая согласованность)
в) Weak consistency (слабая согласованность)
г) Read-your-writes consistency
Правильный ответ: б

7. Как называется способность системы обрабатывать возрастающее количество запросов путем добавления ресурсов?
а) Отказоустойчивость
б) Масштабируемость
в) Безопасность
г) Согласованность
Правильный ответ: б

8. Что такое «нагрузочное тестирование» клиент-серверного приложения?
а) Проверка корректности бизнес-логики
б) Проверка поведения системы под ожидаемой и пиковой нагрузкой
в) Проверка интерфейса пользователя
г) Проверка документации
Правильный ответ: б

9. Какой протокол чаще всего используется для потоковой передачи данных и чат-приложений благодаря постоянному соединению?

а) HTTP
б) FTP
в) WebSocket
г) SMTP
Правильный ответ: в

10. Как в контексте качества ПО называется способность системы оставаться работоспособной при отказе отдельных компонентов?
а) Масштабируемость
б) Производительность
в) Отказоустойчивость
г) Безопасность
Правильный ответ: в

11. Какая архитектура развертывания подразумевает, что один сервер обслуживает запросы всех клиентов?
а) Кластерная архитектура
б) Централизованная архитектура (один сервер)
в) Распределенная архитектура
г) Децентрализованная архитектура
Правильный ответ: б

12. Для чего используется формат OpenAPI (Swagger) при разработке клиент-серверных приложений?
а) Для хранения данных
б) Для описания спецификации REST API в машиночитаемом формате
в) Для шифрования трафика
г) Для сборки проекта
Правильный ответ: б

13. Что такое "сквозное тестирование" (end-to-end testing) клиент-серверного приложения?
а) Тестирование только серверной части
б) Тестирование только клиентской части
в) Тестирование полного пути пользователя через все компоненты системы
г) Тестирование отдельных функций
Правильный ответ: в

14. Какой механизм позволяет распределять входящий трафик между несколькими серверными экземплярами?
а) Кэширование
б) Балансировка нагрузки (load balancing)
в) Резервное копирование
г) Сжатие данных
Правильный ответ: б

15. Какая из перечисленных метрик является ключевой для оценки производительности сервера?
а) Количество строк кода
б) Время ответа (response time) и количество запросов в секунду (RPS)
в) Размер базы данных
г) Версия операционной системы
Правильный ответ: б