МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ВГУ»)

УТВЕРЖДАЮ

Заведующий кафедрой
информационных систем
наименование кафедры, отвечающей за реализацию дисциплины
(*Борисов Д.Н.*)
подпись, расшифровка подписи
5.05.2025 г.

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ
Б1.В.ДВ.01.02 Веб-технологии и облачные сервисы

Код и наименование дисциплины в соответствии с учебным планом

1. Код и наименование направления подготовки:

09.04.02 Информационные системы и технологии

2. Профиль подготовки: *Программные технологии в инфокоммуникационных системах*

3. Квалификация выпускника: Магистр

4. Форма обучения: *заочная*

5. Кафедра, отвечающая за реализацию дисциплины: *Информационных систем*

6. Составители программы: *Ветохин В.В., кандидат технических наук.*
(ФИО, ученая степень, ученое звание)

.

7. Рекомендована: *НМС факультета компьютерных наук, протокол № 7 от 05.05.2025 г.*
(наименование рекомендующей структуры, дата, номер протокола,
отметки о продлении вносятся вручную)

8. Учебный год: *2026/2027* **Семестр/триместр(ы):** *2 курс 6 триместр*

9.**Цели и задачи учебной дисциплины**

Целью освоения учебной дисциплины *является: Целью освоения дисциплины является формирование у студентов системных знаний и навыков в области веб-разработки и облачных технологий*

Задачи учебной дисциплины: *Овладение принципами разработки веб-приложений, архитектурой веб-программирования, освоение технологий клиентской и серверной веб-разработки, изучение облачных технологий и сервисов для хостинга и масштабирования веб-приложений, освоение принципов безопасности веб-приложений и защиты данных*

10. Место учебной дисциплины в структуре ООП: *Дисциплина относится к обязательным профессиональным модулям, часть, формируемая участниками образовательных отношений блока 1.*

11. Планируемые результаты обучения по дисциплине/модулю (знания, умения, навыки), соотнесенные с планируемыми результатами освоения образовательной программы (компетенциями) и индикаторами их достижения:

Код и название компетенции	Код и название индикатора компетенции	Знания, умения, навыки
ПК-2 Способен проводить исследования на всех этапах жизненного цикла программного обеспечения, используя современные программные и вычислительные инструменты	ПК-2.2 Владеет средствами компьютерного моделирования, использует специализированное ПО для проведения виртуальных экспериментов и анализа результатов	Знать: средства компьютерного моделирования, а также основные категории ПО для веб-разработки и облачных вычислений Уметь: использовать средства компьютерного моделирования, специализированное ПО для проведения виртуальных экспериментов и анализа результатов Владеть: средствами компьютерного моделирования, использовать специализированное ПО для проведения виртуальных экспериментов и анализа результатов
ПК-3 Способен проектировать структуру программ и реализует алгоритмы с учётом требований к надёжности, модульности и масштабируемости	ПК-3.2 Использует современные языки программирования для реализации алгоритмов	Знать: основные современные языки программирования для реализации алгоритмов, принципы алгоритмизации и структуры данных, применяемые в веб-приложениях. Уметь: использовать современные языки программирования для реализации алгоритмов, реализовывать алгоритмы обработки данных в контексте веб-приложений. Владеть: современными языками программирования для реализации алгоритмов, навыками отладки и тестирования кода

12. Объем дисциплины в зачетных единицах/час. — 3/*108*.

Форма промежуточной аттестации *зачет.*

13. Трудоемкость по видам учебной работы

Вид учебной работы		Трудоемкость	
		Всего	По триместрам
			6
Аудиторные занятия		4	4
в том числе:	лекции	2	2
	практические	2	2
	лабораторные	-	-
Самостоятельная работа		100	100
Курсовая работа			
Промежуточная аттестация		4	4
Часы на контроль		4	4
Всего		108	108

13.1. Содержание дисциплины

п/п	Наименование раздела дисциплины	Содержание раздела дисциплины	Реализация раздела дисциплины с помощью онлайн-курса, ЭУМК *
	1. Лекции		
1.1	Основы веб-технологий. HTML, CSS, JavaScript	Клиент-серверная архитектура веб-приложений. Роль HTML, CSS и JavaScript в создании веб-страниц. Обзор инструментов разработчика. Структура HTML-документа. Основные HTML-теги. Семантическая верстка в HTML5. Формы и элементы ввода. Синтаксис CSS. Способы подключения стилей. Медиа-запросы и принципы адаптивного дизайна. Основы JavaScript. Функции в JavaScript. Работа с DOM. Взаимодействие HTML, CSS и JavaScript в веб-разработке. Рекомендации по дальнейшему изучению (MDN, Codecademy, практические проекты).	https://edu.vsu.ru/course/view.php?id=31764
1.2	Фреймворки фронтенд-разработки: React, Vue.	Понятие фронтенд-фреймворков и их роль в современной веб-разработке. Преимущества использования фреймворков перед нативным JavaScript. Обзор популярных фронтенд-фреймворков: React, Vue, Angular, Svelte. История создания и ключевые особенности React. Основные концепции React: компоненты, JSX, виртуальный DOM. История создания и философия Vue.js. Основные концепции Vue: реактивность, директивы, однофайловые компоненты. Работа с состоянием в Vue: data, computed, methods. Сравнительный анализ React и Vue: производительность, кривая обучения, экосистема. Оптимизация производительности в React и Vue. Критерии выбора фреймворка для проекта. Демонстрация простого приложения на React и Vue.	
1.3	Серверные технологии: Node.js, PHP, Python Flask/Django	-	https://edu.vsu.ru/course/view.php?id=317

			64
1.4	Работа с базами данных в веб-приложениях.	-	https://edu.vsu.ru/course/view.php?id=31764
1.5	API и протоколы взаимодействия (REST, GraphQL, WebSockets).	-	https://edu.vsu.ru/course/view.php?id=31764
1.6	Облачные сервисы: AWS, Google Cloud, Azure	-	https://edu.vsu.ru/course/view.php?id=31764
1.7	Контейнеризация и DevOps (Docker, Kubernetes, CI/CD)	-	https://edu.vsu.ru/course/view.php?id=31764
2. Лабораторные занятия			
2.1	Основы веб-технологий. HTML, CSS, JavaScript	-	https://edu.vsu.ru/course/view.php?id=31764
2.2	Фреймворки фронтенд-разработки: React, Vue.	-	
2.3	Серверные технологии: Node.js, PHP, Python Flask/Django	Введение в серверную разработку: клиент-серверная модель и роль бэкенда. Обзор серверных технологий: Node.js, PHP, Python (Flask/Django). Установка и настройка окружения для работы. Основы Node.js: асинхронность, модули, npm. Создание простого сервера на Node.js с помощью Express. Обработка HTTP-запросов: GET, POST, PUT, DELETE. Работа с базами данных в Node.js. Основы PHP: синтаксис, встроенный веб-сервер. Работа с MySQL в PHP. Введение во Flask. Создание API на Flask с использованием RESTful-подхода. Работа с базой данных в Flask. Основы Django: структура проекта, ORM, админ-панель. Создание CRUD-приложения на Django. Аутентификация и авторизация в Django. Сравнение производительности и удобства разработки.	
2.4	Работа с базами данных в веб-приложениях.	-	
2.5	API и протоколы взаимодействия (REST, GraphQL, WebSockets).	-	
2.6	Облачные сервисы: AWS, Google Cloud, Azure	-	
2.7	Контейнеризация и DevOps (Docker, Kubernetes, CI/CD)	Введение в DevOps и контейнеризацию. Основы Docker. Создание Docker-образов. Docker Compose. Введение в Kubernetes. Масштабирование и управление в Kubernetes. CI/CD: основы автоматизации. Инфраструктура как код (IaC). Безопасность в DevOps.	
2.8	Безопасность веб-приложений.		

13.2. Темы (разделы) дисциплины и виды занятий

№	Наименование темы	Виды занятий (количество часов)

п/п	(раздела) дисциплины	Лекции	Практические	Лабораторные	Самостоятельная работа	Всего
1.	Основы веб-технологий. HTML, CSS, JavaScript	1	-	-	10	11
2.	Фреймворки фронтенд-разработки: React, Vue.	1	-	-	20	21
3.	Серверные технологии: Node.js, PHP, Python Flask/Django	-	-	1	20	21
4.	Работа с базами данных в веб-приложениях.	-	-	-	10	10
5.	API и протоколы взаимодействия (REST, GraphQL, WebSockets).	-	-	-	10	10
6.	Облачные сервисы: AWS, Google Cloud, Azure	-	-	-	10	10
7.	Контейнеризация и DevOps (Docker, Kubernetes, CI/CD)	-	-	1	10	11
8.	Безопасность веб-приложений.				10	10
	Итого:	2	-	2	100	108

14. Методические указания для обучающихся по освоению дисциплины

Внеаудиторная самостоятельная работа студентов включает проработку материалов лекций, изучение рекомендованной литературы, подготовку к лабораторным работам и их защитам, подготовку к устному опросу и зачету.

Самостоятельная работа в аудитории выполняется под непосредственным руководством преподавателя. Во время самостоятельной работы студенты используют электронно-библиотечные системы, доступные на портале Зональной Библиотеки ВГУ по адресу www.lib.vsu.ru. Для повышения эффективности руководства при проведении лабораторных занятий, призванных обеспечить выборочное использование лекционного материала для более глубокого изучения отдельных разделов дисциплины при решении соответствующих практических задач.

К лабораторным занятиям студенты должны изучить теоретический материал предметной области.

15. Перечень основной и дополнительной литературы, ресурсов интернет, необходимых для освоения дисциплины

а) основная литература:

№ п/п	Источник
1.	*Флэнаган, Д. JavaScript: Полное руководство / Д. Флэнаган. — 7-е изд. — СПб.: Символ-Плюс, 2022. — 1080 с. — ISBN 978-5-93286-249-9.*
2.	*Таненбаум, Э., ван Стеен, М. Распределенные системы. Принципы и парадигмы / Э. Таненбаум. — 3-е изд. — СПб.: Питер, 2021. — 848 с. — ISBN 978-5-4461-1456-8.*
3.	*Дейтел, П., Дейтел, Х. Технологии веб-программирования / П. Дейтел. — М.: Бином, 2023. — 736 с. — ISBN 978-5-9518-0873-5.*

б) дополнительная литература:

№ п/п	Источник
4.	*Архитеков, А.Ю. Облачные вычисления: от основ к DevOps / А.Ю. Архитеков. — М.: ДМК Пресс, 2023. — 412 с. — ISBN 978-5-93700-178-9.*
5.	*Фаулер, М. Паттерны проектирования корпоративных приложений / М. Фаулер. — СПб.: Питер, 2021. — 544 с. — ISBN 978-5-4461-1489-6.*

в) информационные электронно-образовательные ресурсы (официальные ресурсы интернет)*:

№ п/п	Ресурс
6.	*MDN Web Docs https://developer.mozilla.org/ru/*
7.	*Microsoft Learn: Облачные сервисы и DevOps https://learn.microsoft.com/ru-ru/training/*

16. Перечень учебно-методического обеспечения для самостоятельной работы

№ п/п	Источник
1.	*Фитцпатрик, Б. Git для профессионального программиста / Б. Фитцпатрик, Б. Весков.*

	— М.: Эксмо, 2021. — 400 с. — ISBN 978-5-04-121845-9.
2.	*Лукьянов, А. Инфраструктура как код: управление конфигурациями в DevOps / А. Лукьянов. — СПб.: БХВ-Петербург, 2023. — 320 с. — ISBN 978-5-9775-4102-1.*

17. Образовательные технологии, используемые при реализации учебной дисциплины, включая дистанционные образовательные технологии (ДОТ), электронное обучение (ЭО), смешанное обучение):

1. Образовательный портал Moodle (сервер Moodle ВГУ), https://edu.vsu.ru

18. Материально-техническое обеспечение дисциплины: Лекционная аудитория, оборудованная мультимедийным проектором. Компьютерные классы факультета для проведения лабораторных занятий. Образовательный портал «Электронный университет ВГУ» https://edu.vsu.ru.

19. Оценочные средства для проведения текущей и промежуточной аттестаций.

Порядок оценки освоения обучающимися учебного материала определяется содержанием следующих разделов дисциплины:

№ п/п	Наименование раздела дисциплины (модуля)	Компетенция(и)	Индикатор(ы) достижения компетенции	Оценочные средства
1.	Основы веб-технологий. HTML, CSS, JavaScript Фреймворки фронтенд-разработки: React, Vue. Серверные технологии: Node.js, PHP, Python Flask/Django	ПК-2, ПК-3	ПК-2.2, ПК-3.2	*Тестовые задания 1*
2.	Работа с базами данных в веб-приложениях. API и протоколы взаимодействия (REST, GraphQL, WebSockets).	ПК-2, ПК-3	ПК-2.2, ПК-3.2	*Практическое задание 1*
3.	Облачные сервисы: AWS, Google Cloud, Azure Контейнеризация и DevOps (Docker, Kubernetes, CI/CD)	ПК-2, ПК-3	ПК-2.2, ПК-3.2	*Практическое задание 2*
4.	Безопасность веб-приложений.	ПК-2, ПК-3	ПК-2.2, ПК-3.2	*Тестовые задания 2*
Промежуточная аттестация форма контроля – *зачет*				*Перечень вопросов Практическое задание*

20 Типовые оценочные средства и методические материалы, определяющие процедуры оценивания

20.1 Текущий контроль успеваемости

Контроль успеваемости по дисциплине осуществляется с помощью следующих оценочных средств:

Текущая аттестация проводится в течение всего обучения в рамках освоения дисциплины и заключается в выполнении студентом лабораторных работ, успешного прохождения тестовых заданий (не менее 50 % правильных ответов), а также успешного практических заданий (получение не менее 50 баллов из 100 баллов). Тестовые задания выполняются студентами после прослушивания лекций; лабораторные работы выполняются в течении отчетной сессии, но не позже итоговой аттестации. Практические задания выполняются после предоставления отчетов по лабораторным работам.

Лабораторные работы после выполнения оцениваются преподавателем, и выставляется оценка «зачтено» по лабораторной работе при условии ответа на 80% вопросов преподавателя по предметной области лабораторной работы.

Перечень заданий

Тестовое задание 1

1. Какой тег HTML используется для создания гиперссылки?

A) <p>¶
B) <a>¶
C) <link>¶
D) <href>

2. Какой атрибут тега <img> задаёт путь к изображению?

A) href¶
B) src¶
C) alt¶
D) path

3. Какой CSS-селектор применяет стиль ко всем элементам с классом .menu?

A) #menu¶
B) menu¶
C) .menu¶
D) *menu

4. Какая команда в JavaScript выводит сообщение в консоль?

A) console.print()¶
B) log.console()
C) print()
D) console.log()

5. Какой тип данных в JavaScript используется для хранения текста?

A) int¶
B) text¶
C) string¶
D) var

6. Какой из фреймворков является фронтенд-фреймворком?

A) Flask¶
B) Django¶
C) React¶
D) Node.js

7. Что из следующего является реактивным компонентом в Vue.js?

A) v-react¶
B) ref¶
C) useEffect¶
D) state

8. Какой файл является входной точкой для большинства приложений на Node.js?

A) index.css¶
B) main.html¶
C) app.js¶
D) routes.js

9. Какой метод HTTP используется для отправки формы с изменением данных на сервере?

A) GET¶
B) POST¶
C) FETCH¶
D) PATCH

10. Какой метод массива JavaScript используется для перебора всех элементов?

A) map()¶
B) loop()¶
C) run()¶
D) iterate()

11. Как называется объект в Node.js, представляющий входящие HTTP-запросы?
A) res¶
B) http¶
C) request¶
D) req

12. Что делает директива v-if в Vue.js?
A) Повторяет элемент¶
B) Скрывает элемент навсегда¶
C) Условно отображает элемент¶
D) Применяет стиль

13. Какой фреймворк на Python используется для быстрого создания REST API?
A) Tornado¶
B) Flask¶
C) React¶
D) Laravel

14. Какой язык чаще всего используется в разработке серверной логики в Django?
A) PHP¶
B) JavaScript¶
C) Python¶
D) Java

15. Что делает npm install в проекте Node.js?
A) Удаляет проект¶
B) Запускает сервер¶
C) Устанавливает зависимости¶
D) Обновляет HTML

Тестовое задание 2

1. Что такое XSS (Cross-Site Scripting)?
A) Атака, при которой злоумышленник перехватывает файлы на сервере
B) Атака, при которой внедряется вредоносный JavaScript в веб-страницу
C) Вид DDoS-атаки
D) Инструмент защиты данных

2. Что позволяет предотвратить SQL-инъекции?
A) Использование куки
B) Парсинг JSON
C) Параметризированные запросы
D) GET-запросы

3. Что такое CSRF (Cross-Site Request Forgery)?
A) Атака, направленная на подмену JavaScript
B) Атака, которая маскируется под легитимный запрос от имени пользователя
C) Вирус, атакующий веб-сервер
D) Ошибка сервера при обработке форм

4. Какой заголовок HTTP помогает предотвратить XSS-атаки?
A) Content-Length
B) Accept-Language
C) X-Content-Type-Options
D) X-XSS-Protection

5. Какой принцип обеспечивает минимальные привилегии для пользователей?
A) Open Access

B) Full Trust
C) Least Privilege
D) Admin First

6. Какой метод аутентификации является наиболее безопасным?
A) Только по логину
B) Логин и пароль
C) Двухфакторная аутентификация (2FA)
D) Куки без шифрования

7. Что такое HTTPS?
A) Протокол баз данных
B) Безопасное расширение HTTP с шифрованием трафика
C) Язык шаблонов
D) Операционная система для веб-приложений

8. Для чего используется Content Security Policy (CSP)?
A) Для стилизации HTML
B) Для ограничения источников загрузки контента и защиты от XSS
C) Для настройки маршрутизации
D) Для оптимизации скорости загрузки

9. Что такое хэширование паролей?
A) Сжатие данных
B) Преобразование пароля в случайную строку, необратимую
C) Шифрование в режиме онлайн
D) Замена символов пароля

10. Какой инструмент используется для автоматизированного анализа уязвимостей веб-приложений?
A) Firebug
B) Postman
C) OWASP ZAP
D) Visual Studio

11. Какая организация публикует список самых опасных уязвимостей веб-приложений (Top 10)?
A) ISO
B) W3C
C) OWASP
D) NIST

12. Что может произойти при неправильной настройке CORS?
A) Увеличится скорость загрузки
B) Отключится SSL
C) Возникнет утечка данных между доменами
D) Перестанет работать JavaScript

13. Какой из следующих механизмов защиты используется для хранения пользовательских сессий?
A) GET-параметры
B) Открытые куки
C) HttpOnly cookies
D) LocalStorage

14. Какой метод защиты используется для защиты форм от CSRF?
A) CAPTCHA
B) Хеширование
C) CSRF-токен
D) Базовая аутентификация

15. Какое из утверждений верно для HTTPS-сертификата?
A) Он добавляет JavaScript-фильтрацию
B) Он используется для подписания HTML-кода
C) Он подтверждает подлинность сайта и шифрует соединение

D) Он увеличивает производительность сайта

Практическое задание 1

1. Создание REST API с подключением к базе данных

Разработайте простое REST API для управления списком задач (ToDo), используя Node.js и Express. Реализуйте операции:

- создание задачи (POST),
- получение всех задач (GET),
- обновление задачи (PUT),
- удаление задачи (DELETE).

Хранение данных — в базе данных (например, MongoDB или PostgreSQL).

2. Интеграция frontend-приложения с REST API

Создайте веб-страницу с формой для добавления задач, кнопкой для их удаления и списком уже добавленных. Реализуйте взаимодействие с REST API из предыдущего задания с использованием fetch/AJAX.

3. Реализация фильтрации и пагинации на серверной стороне

Дополните REST API функцией фильтрации задач по статусу (выполнена/не выполнена) и постраничным выводом (параметры limit, offset).

4. Создание базы данных и модели с помощью ORM

Используя Sequelize (для Node.js) или Django ORM, создайте таблицу users с полями: id, name, email, password, created_at. Настройте миграции и подключение к БД.

5. Построение GraphQL API

Разработайте GraphQL API для работы с продуктами (Products). Реализуйте:

- запрос списка продуктов,
- получение информации о продукте по ID,
- добавление нового продукта,
- обновление и удаление продукта.

Используйте Apollo Server и PostgreSQL.

6. Разработка подписок на события через WebSockets

Создайте чат-приложение, в котором сообщения сохраняются в базу данных. Используйте WebSockets (например, с библиотекой Socket.IO) для:

- получения новых сообщений в реальном времени,
- оповещения всех подключённых клиентов о новом сообщении.

7. Аутентификация пользователей и защита API

Добавьте регистрацию и вход пользователей в REST API. Реализуйте защиту маршрутов с помощью токенов (JWT). Храните пользователей в базе данных и реализуйте проверку паролей с использованием хеширования (bcrypt).

8. Интеграция GraphQL с базой данных

Свяжите GraphQL-сервер с реляционной базой данных через ORM. Реализуйте реляционные связи, например:

- User имеет много Posts,
- запрос GraphQL должен возвращать пользователя с его постами.

9. Создание CRUD-интерфейса с использованием Django и DRF

С помощью Django REST Framework создайте API для управления записями блога (BlogPost), подключите SQLite или PostgreSQL. Реализуйте сериализацию и маршрутизацию.

10. Сравнение производительности REST и GraphQL

Реализуйте REST и GraphQL API для одного и того же набора данных (например, список книг). Проведите тестирование времени отклика и количества переданных данных при выборке 1, 10 и 100 записей. Подготовьте отчёт.

Практическое задание 2

1. Развертывание виртуальной машины в облаке

Разверните виртуальную машину (EC2 в AWS / VM Instance в GCP / Azure VM).
Установите веб-сервер (например, Nginx или Apache), настройте фаерволл и откройте порт 80.

2. Хранение файлов в облаке (S3 / GCS / Azure Blob)

Создайте облачное хранилище.
Загрузите и скачайте файл через консоль и через SDK (например, Python boto3 или gsutil).
Настройте публичный доступ к отдельному файлу.

3. Контейнеризация веб-приложения с использованием Docker

Создайте Dockerfile для простого Python/Node.js/Flask-приложения.
Постройте образ и запустите контейнер локально.
Протестируйте доступность веб-сервиса по localhost.

4. Публикация Docker-образа в Docker Hub

Создайте аккаунт на Docker Hub.
Соберите и загрузите образ вашего приложения в Docker Hub.
Проверьте, что его можно загрузить и запустить на другой машине.

5. Развёртывание контейнера в облаке

Разверните контейнерное приложение на:

- AWS ECS или Elastic Beanstalk,
- Google Cloud Run или Cloud Run for Anthos,
- Azure Container Instances.

Задокументируйте шаги деплоя.

6. Создание и настройка кластера Kubernetes

Разверните Kubernetes-кластер с помощью:

- minikube (локально),
- Google Kubernetes Engine (GKE),
- Azure AKS или AWS EKS.

Разверните в нём контейнерное приложение через Deployment и Service.

7. Масштабирование приложения в Kubernetes

Настройте масштабирование реплик вручную.
Дополнительно — настройте Horizontal Pod Autoscaler на основе CPU-нагрузки.
Проверьте, как изменяется количество подов при нагрузке.

8. Настройка CI/CD с использованием GitHub Actions или GitLab CI

Создайте репозиторий с приложением.
Добавьте конфигурацию CI/CD, которая:

- собирает Docker-образ,
- проводит тестирование,
- разворачивает контейнер на удалённый сервер или в облако.

9. Использование Terraform или CloudFormation для описания инфраструктуры

Создайте инфраструктуру как код (IaC) для:

- развёртывания виртуальной машины,
- создания облачного хранилища,
- настройки сети.

Используйте Terraform (предпочтительно) или CloudFormation.

10. Мониторинг и логирование контейнеров

Подключите систему мониторинга:

- Prometheus + Grafana (локально),
- или Cloud Monitoring (в GCP, AWS, Azure).

Настройте отображение CPU/Memory метрик подов Kubernetes.
Настройте логирование в облачные системы (например, AWS CloudWatch или GCP Logging).

Приведённые ниже задания рекомендуется использовать при проведении диагностических работ для оценки остаточных знаний:

ПК-2

Задания закрытого типа (в каждом задании необходимо выбрать один или несколько ответов)

1. Какой язык используется для взаимодействия с реляционными базами данных?
   A) HTML
   B) SQL
   C) CSS
   D) XML

2. Что из следующего лучше всего описывает роль ORM в веб-приложениях?
   A) Система управления пользователями
   B) Интерфейс для работы с API
   C) Связывает объекты языка программирования с таблицами БД
   D) Шаблон для отображения HTML

3. Какой из запросов используется для получения данных из таблицы?
   A) DELETE
   B) SELECT
   C) UPDATE
   D) INSERT

4. Какая команда используется для добавления новой записи в таблицу?
   A) ADD
   B) INSERT
   C) UPDATE
   D) SELECT

5. Какой тип базы данных является реляционным?
   A) MongoDB
   B) Redis
   C) PostgreSQL
   D) Neo4j

6. Что такое «prepared statement»?
   A) Запрос, который нельзя повторно использовать
   B) Предварительно скомпилированный SQL-запрос с параметрами
   C) Хранимая процедура
   D) Команда для создания таблицы

7. Что из нижеперечисленного уменьшает риск SQL-инъекций?
   A) Использование HTML-валидаторов
   B) Прямое включение данных в строку запроса
   C) Использование параметризированных запросов
   D) Использование JavaScript

8. Что делает команда UPDATE в SQL?
   A) Удаляет таблицу
   B) Вставляет новую строку
   C) Изменяет существующие данные
   D) Создает таблицу

9. Какой формат чаще всего используется для передачи данных между клиентом и сервером в веб-приложениях?
   A) XML
   B) YAML
   C) JSON
   D) CSV

10. Какой HTTP-метод чаще всего используется для отправки формы с добавлением данных в базу?
    A) GET
    B) DELETE
    C) PUT
    D) POST

11. Что такое миграции в контексте ORM?
    A) Перенос данных между двумя базами
    B) Создание резервной копии базы данных
    C) Автоматическое обновление структуры базы
    D) Изменение значений в таблице

12. Какое расширение чаще всего используется для SQLite-файлов?
    A) .sdb
    B) .db
    C) .sql
    D) .sqlite3

13. Какой тип связи в базе данных обозначает отношение «один ко многим»?
    A) foreign key
    B) primary key
    C) unique key
    D) composite key

14. Какое из утверждений верно для транзакций?
    A) Транзакции используются только в MongoDB
    B) Транзакции — это механизм изменения структуры базы
    C) Транзакции позволяют выполнять несколько операций атомарно
    D) Транзакции применимы только к SELECT-запросам

15. Какой компонент веб-приложения обычно отвечает за выполнение SQL-запросов к базе данных?
    A) Frontend
    B) Middleware
    C) Backend
    D) CDN

Задания с открытым ответом

1. Как называется язык запросов, используемый для работы с реляционными базами данных?

2. Укажите SQL-команду, с помощью которой можно изменить значения в существующих записях таблицы.

3. Назовите структуру данных, связывающую таблицы в реляционной базе по внешнему ключу.

4. Как называется технология, позволяющая представлять таблицы базы данных в виде классов и объектов в коде?

5. Какой формат чаще всего используется для обмена данными между frontend и backend в современных веб-приложениях?

6. Укажите SQL-команду, которая используется для удаления строк из таблицы.

7. Как называется процесс, при котором несколько операций с базой данных выполняются как единое целое, с возможностью отката?

Задания с развернутым ответом

1. Опишите процесс взаимодействия веб-приложения с реляционной базой данных при авторизации пользователя. Укажите, как происходит передача данных с клиентской части на сервер, каким образом осуществляется запрос к базе данных и как сервер реагирует на результаты запроса.
2. Объясните, что такое SQL-инъекция. Приведите пример уязвимого кода и опишите, как можно защитить веб-приложение от этой уязвимости.

3. Сравните два подхода: прямое использование SQL-запросов в коде и использование ORM (например, SQLAlchemy, Django ORM, Hibernate). Укажите преимущества и недостатки каждого из подходов.
4. Опишите структуру базы данных для простого интернет-магазина (таблицы: пользователи, товары, заказы). Укажите поля, связи между таблицами и типы данных.
5. Приведите пример реализации CRUD-операций (Create, Read, Update, Delete) в веб-приложении. Распишите, какие SQL-запросы или ORM-команды будут использоваться, и как это связано с маршрутами HTTP-запросов.
6. Опишите, как можно реализовать пагинацию (постраничный вывод) данных из базы в веб-интерфейсе. Какие SQL-запросы для этого используются, и как передаются параметры на сервер?
7. Объясните, что такое миграции в контексте разработки веб-приложения с использованием фреймворков (например, Django или Laravel). Почему важно их использовать и как они помогают при командной разработке?

ПК-3

<u>Задания закрытого типа</u> (в каждом задании необходимо выбрать один или несколько ответов)

1. Какой тег HTML используется для создания гиперссылки?
    а) <link>
    б) <a> +
    в) <href>
    г) <url>

2. Что такое CSS-препроцессор?
    а) Инструмент для автоматического тестирования кода
    б) Язык, расширяющий возможности CSS (например, Sass, Less) +
    в) Программа для оптимизации изображений
    г) Библиотека JavaScript

3. Какой метод HTTP используется для отправки данных на сервер (например, формы)?
    а) GET
    б) POST +
    в) PUT
    г) DELETE

4. Что делает Virtual DOM в React?
    а) Заменяет реальный DOM для ускорения рендеринга +
    б) Создает виртуальные серверы
    в) Шифрует данные
    г) Управляет состоянием приложения

5. Какой сервис AWS предоставляет бессерверные вычисления?
    а) EC2
    б) S3
    в) Lambda +
    г) RDS

6. Для чего используется Docker?
    а) Для виртуализации операционных систем
    б) Для контейнеризации приложений +
    в) Для создания графических интерфейсов
    г) Для написания SQL-запросов

7. Что такое Kubernetes?
    а) Язык программирования
    б) Система оркестрации контейнеров +
    в) Облачное хранилище
    г) Фреймворк для фронтенда

8. Какой инструмент используется для автоматизации CI/CD?
    а) Photoshop
    б) GitHub Actions +
    в) Microsoft Word
    г) Adobe Illustrator

9. Какой язык чаще всего используется для бэкенда в связке с Django?
    а) Java
    б) Python +
    в) C#
    г) PHP

10. Что означает аббревиатура API?
    а) Automated Programming Interface
    б) Application Programming Interface +
    в) Advanced Protocol Integration
    г) Artificial Programming Intelligence

11. Какой облачный сервис предоставляет Google?
    а) AWS
    б) Azure
    в) Google Cloud Platform (GCP) +
    г) IBM Cloud

12. Что такое RESTful API?
    а) Архитектурный стиль для создания веб-сервисов +
    б) Язык разметки
    в) Библиотека для анимаций
    г) Протокол шифрования

13. Какой командой запускается контейнер Docker?
    а) docker start
    б) docker run +
    в) docker execute
    г) docker create

14. Что измеряет инструмент Prometheus?
    а) Скорость интернета
    б) Производительность и метрики приложений +
    в) Качество кода
    г) Уровень безопасности

15. Какой тег HTML5 используется для семантического обозначения «нижнего колонтитула» страницы?
    а) <bottom>
    б) <footer> +
    в) <end>
    г) <section>

Вопросы открытого типа:

1. Какой тег HTML используется для вставки изображения?
2. Назовите три основных HTTP-метода для работы с API.
3. Какой CSS-свойство изменяет цвет текста?
4. Как называется виртуальная машина в облаке AWS?
5. Какой командой в Docker можно посмотреть список запущенных контейнеров?
6. Какой инструмент используется для управления зависимостями в JavaScript?
7. Какой язык шаблонов используется во фреймворке Django?
8. Как называется минимальная единица развертывания в Kubernetes?
9. Какой облачный сервис предоставляет Microsoft?
10. Какой тип базы данных использует MongoDB?
11. Какой командой в Git можно отправить изменения в удаленный репозиторий?
12. Как называется архитектурный стиль, лежащий в основе REST API?
13. Какой инструмент используется для автоматического развертывания кода (CI/CD)?
14. Какой командой в Linux можно проверить доступность сервера по сети?
15. Какой CSS-фреймворк использует систему сетки из 12 колонок?

Задания с развернутым ответом

1. Эволюция веб-разработки: от статических страниц до современных SPA-фреймворков
2. Облачные сервисы vs локальные решения: анализ экономических и технических факторов

3. Docker и Kubernetes: как контейнеризация изменила DevOps-практики
4. REST, GraphQL и gRPC: сравнительный анализ API-архитектур
5. Будущее веб-технологий: тренды следующего десятилетия
6. Какие преимущества и недостатки имеет serverless-архитектура по сравнению с традиционным подходом?
7. Опишите жизненный цикл компонента в React.

20.2 Промежуточная аттестация

контроля - Зачет

Оценочные средства для промежуточной аттестации:

Промежуточная аттестация проводится на основании итогов выполнения студентом лабораторных работ по всем темам (100% выполненных работ), успешного прохождения двух тестов (не менее 50% правильных ответов) и успешного выполнения практических заданий (50 баллов из 100 баллов).
По итогам выполнения лабораторных работ, учета прохождения тестов, выполнения практических заданий и устного ответа (собеседование со студентом в конце триместра по вопросам из перечня вопросов к зачету и использования контрольно-измерительных материалов) студенту выставляется оценка «зачтено» или «не зачтено» по дисциплине.

Соотношение показателей, критериев и шкалы оценивания результатов обучения.

Критерии оценивания компетенций	Уровень сформированности компетенций	Шкала оценок
Обучающийся демонстрирует частичные знания языка программирования, допускает существенные ошибки в решении задач	Базовый уровень	Зачет
Обучающийся демонстрирует отрывочные, фрагментарные знания, допускает грубые ошибки, не умеет решать поставленные задачи	Пороговый уровень	Незачет

Вопросы к зачету

1. Назовите основные теги HTML для создания формы и её полей ввода.
2. Как задать стили для конкретного класса в CSS?
3. Что такое модель DOM и как JavaScript взаимодействует с ней?
4. Объясните понятие событий в JavaScript и приведите пример.
5. Что такое AJAX и для чего он используется в веб-приложениях?
6. Назовите основные различия между let, var и const в JavaScript.
7. Что такое адаптивная вёрстка и какие технологии для неё применяются?
8. Что такое компонент в React и из чего он состоит?
9. Объясните жизненный цикл компонента в React.
10. Для чего используется директива v-for во Vue.js?
11. В чём отличие props и state в React?
12. Что такое реактивность во Vue и как она реализуется?
13. Как организуется маршрутизация (routing) во frontend-фреймворках?
14. Как создать простой HTTP-сервер на Node.js?
15. В чём отличие клиентского и серверного рендеринга?
16. Что такое middleware в Express.js и как он используется?
17. Объясните принципы маршрутизации во Flask.
18. Какие функции выполняет контроллер в MVC-архитектуре веб-приложений?
19. Какие типы данных поддерживаются при работе с формами в PHP?
20. Что такое REST API и какие методы HTTP в нём используются?
21. В чём отличие REST и GraphQL?
22. Как обеспечить безопасность API (перечислите 2–3 метода)?
23. Что такое CRUD-операции и как они реализуются в веб-приложении?
24. Как реализовать подключение к базе данных в Flask или Node.js?
25. Назовите ключевые сервисы AWS/GCP/Azure для хранения данных и вычислений.
26. Что такое контейнеризация и как работает Docker?
27. Объясните процесс создания Docker-образа.
28. Что такое Kubernetes и какие сущности в нём используются?

29. Что такое CI/CD и как он применяется в разработке?
30. Какие преимущества даёт использование облака при развёртывании веб-приложений?

Пример контрольно-измерительного материала

Направление подготовки 09.04.02 Информационные системы и технологии
Дисциплина _ Веб-технологии и облачные сервисы
Курс___2___
Форма обучения ____заочная_
Вид аттестации __промежуточная_
Вид контроля ___зачет___

Контрольно-измерительный материал № 1

1. Назовите основные различия между let, var и const в JavaScript
2. Что такое компонент в React и из чего он состоит?
3. Приведите алгоритм масштабирования приложения в Kubernetes