МИНИСТЕРСТВО ОБРАЗОВАНИЯ ОРЕНБУРГСКОЙ ОБЛАСТИ
ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ПРОФЕССИОНАЛЬНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
«ОРЕНБУРГСКИЙ КОЛЛЕДЖ ЭКОНОМИКИ И ИНФОРМАТИКИ»
(ГАПОУ «ОКЭИ»)

РАССМОТРЕНО
на заседании ПЦК
информационных технологий
протокол №__ от « __»_____2025 г
Колотвина М.Г. - председатель ПЦК
информационных технологий_____

СОГЛАСОВАНО
И.о заместителя директора по УР
_____А.А.Илюсизова
«_____»_____2025 г.

КОНТРОЛЬНО-ОЦЕНОЧНЫЕ СРЕДСТВА
ЭКЗАМЕН

Учебная дисциплина/МДК/ПМ: МДК 09.03 Обеспечение безопасности веб-приложений
Специальность, квалификация: 09.02.07 Разработчик «веб и мультимедийных приложений»
Группа(ы): 4вб1, 4вб2, 4вб3
Форма проведения: комбинированная

ПЕРЕЧЕНЬ ВОПРОСОВ

1. Определите, что такое кибербезопасность и в чем заключается её основная цель. Основные типы угроз и примеры. Последствия отсутствия защиты.
2. Суперпользователь и чем он отличается от обычного пользователя. Система прав доступа в Linux: rwx-права для пользователя, группы и других.
3. SQL-инъекция.Принцип работы. Методы предотвращения SQL-инъекций
4. XSS-атака и её основные типы. Способ внедрения. Механизмы защиты от XSS существуют на уровне сервера и клиента
5. ORM. Роль в разработке. Преимущества перед простыми SQL-запросами. Недостатки
6. CSRF-атака. Принцип работы. Механизмы защиты
7. API. REST API
8. Reverse Proxy. Примеры приложений. Функциональные возможности
9. Шифрование. Примеры чувствительных данных. Брутфорс
10. Хеширование. Примеры чувствительных данных. Отличие от шифрования
11. Капча. Rate limit. Примеры решаемых проблем
12. OAuth. Назначение. Примеры интеграции
13. Метрики. Метрики производительности используемые для оценки качества загрузки страницы
14. Способы оптимизации загрузки страницы. Форматы изображений. Lazy loading
15. Асинхронность. Реализация в Java Script
16. Этапы загрузки страницы.
17. Инструменты для анализа производительности сайта.
18. Кэширование. Типы кешей. Предназначение

19. Принципы ООП: инкапсуляция, наследование, полиморфизм.
20. Методы передачи данных (GET, POST). Предназначение
21. Cookie: назначение, установка и чтение.
22. Типы баз данных: реляционные и нереляционные.
23. Реляционные базы данных: основные концепции.
24. Пакеты. Пакетный менеджер npm, и его предназначение
25. JSON. Применение в веб-разработке. Примеры
26. Определение простых классов и методов
27. Middleware. Пример использования
28. jQuery. Преимущества по сравнению с нативным JavaScript
29. Модели в orm. Что это, для чего используются? Примеры.
30. Связи между таблицами в SQL. Виды, предназначение, правила, примеры, ограничения. Обязательные и необязательные данные в бд
31. AJAX, и как эта технология позволяет обновлять веб-страницу без перезагрузки?
32. Статический контент. Раздача статических файлов в Express
33. Сессии. Назначение. Отличие от кук.
34. Отличие https от http
35. SEO. Виды SEO
36. Порт. Назначение. Порт по умолчанию для http и https
37. Модификаторы доступа (public, private, protected). Применение
38. Аутентификация. Авторизация
39. Способы авторизации. Примеры
40. JWT (JSON Web Token) . Составные части. Преимущества перед куки и сессиями
41. Определение базы данных. СУБД. Примеры
42. Проблемы авторизации через куки
43. Первичный и внешний ключ. Индексы. Примеры
44. Хранилища файлов на сервере
45. Данные, которые можно хранить в куки, сессиях и токенах. Примеры
46. Access и refresh токен. Предназначение
47. Клиент-серверная архитектура. Принцип работы. Используемые протоколы.
48. SSL/TLS-сертификаты. Предназначение
49. Транзакция в база данных. Предназначение. Пример.
50. Кэширование. Примеры кэшируемых данных
51. Работа с пакетами в node.js. Файл package.json.
52. Что такое объектно-ориентированное программирование, и какие задачи оно решает

ПЕРЕЧЕНЬ ПРАКТИЧЕСКИХ ЗАДАНИЙ

1. Создайте веб-страницу с формой регистрации (поля: имя пользователя, email, пароль), отправляющей данные на сервер, где данные сохраняются в базе данных. Убедитесь, что на сервере нет валидации данных, и попробуйте ввести вредоносный скрипт), чтобы продемонстрировать XSS-уязвимость. Затем исправьте уязвимость, реализовав серверную валидацию данных, экранирование HTML-символов.

2. Реализуйте форму авторизации (email и пароль), которая отправляет данные на сервер. На сервере создайте уязвимый SQL-запрос, например: SELECT * FROM users WHERE email = '$email' AND password = '$password'. Попробуйте выполнить SQL-инъекцию, введя в поле email: ' OR '1'='1. Затем исправьте уязвимость.

3. Создайте страницу с формой добавления комментариев, которые сохраняются в базе данных и отображаются на странице. Введите в поле комментария вредоносный код и убедитесь, что он срабатывает. После этого защитите приложение, экранируя специальные символы

4. Реализуйте API для управления пользователями с использованием ORM. Создайте модель User с полями id, name, email, password. Реализуйте маршруты для создания и получения пользователей.

5. Создайте форму для изменения email-адреса пользователя, которая отправляет POST-запрос на сервер. Смоделируйте CSRF-атаку, создав внешнюю страницу, которая автоматически отправляет запрос на ваш сервер от имени пользователя. Затем защитите приложение.

6. Создайте REST API для управления пользователями с маршрутами GET /users (получение списка), POST /users (создание пользователя) и DELETE /users/:id (удаление пользователя). На клиентской стороне реализуйте форму для добавления пользователя и таблицу для отображения списка пользователей.

7. Создайте форму регистрации, в которой пароли пользователей шифруются перед сохранением в базу данных. Для сравнения реализуйте хеширование паролей с использованием bcrypt и объясните разницу между шифрованием и хешированием, а также преимущества использования хеширования для хранения паролей.

8. Реализуйте регистрацию и авторизацию пользователей, при этом пароли должны хешироваться с использованием bcrypt перед сохранением в базу данных. При авторизации сравнивайте введённый пароль с хешем, хранящимся в базе данных, для проверки подлинности пользователя.

9. Создайте страницу с формой регистрации и измерьте метрики производительности загрузки страницы. Определите узкие места и предложите способы их устранения.

10. Реализуйте форму авторизации, после успешного входа сохраните данные о пользователе (например, session_id или auth_token) в cookie с настройками HttpOnly и Secure, чтобы минимизировать риски XSS-атак, и используйте эти cookie для проверки авторизации при последующих запросах на сервер.

11. Проектируйте базу данных с таблицами users и orders, где один пользователь может иметь несколько заказов (отношение «один ко многим»), добавьте ограничения целостности данных и реализуйте API для работы с этими данными.

12. Создайте форму регистрации на клиенте и отправьте данные в формате JSON на сервер с использованием fetch API. На сервере сохраните данные в базу данных.

13. Определите класс User в JavaScript с конструктором для инициализации имени и email, а также методами register() и login(), которые будут выводить сообщения о регистрации и входе пользователя.

14. В Express-приложении создайте middleware для логирования всех входящих запросов (метод, URL, время запроса) и подключите его к серверу, чтобы отслеживать активность пользователей.

15. Создайте модель User с полями id, name, email, password, настройте валидацию данных и реализуйте CRUD-операции для работы с моделью через API.

16. Создайте таблицы authors и books в SQL, реализовав связь «один ко многим» и напишите SQL-запрос для получения списка всех книг вместе с именами их авторов.

17. Реализуйте механизм сессий с использованием express-session: после авторизации создайте сессию для пользователя, храните данные о сессии на сервере и используйте их для проверки авторизации на защищённых маршрутах.

18. Реализуйте механизм создания и валидации JWT-токенов для пользователей, включая в токен информацию о ролях и правах доступа, чтобы ограничить доступ к определённым маршрутам.

19. Создайте таблицы departments и employees в базе данных, установите первичные и внешние ключи для связи данных и выполните запрос для получения всех сотрудников с указанием их отдела.

20. Разработайте клиент-серверное приложение, где клиент и сервер находятся на разных доменах, продемонстрируйте проблему CORS и решите её, настроив соответствующие заголовки на сервере с помощью middleware cors для Express.

21. Разработайте сайт, на котором будет отображаться список всех пользователей, которые хранятся в базе данных

22. Реализовать простой TODO список, который будет отображать список задач, дату начала и статус задачи

23. Реализуйте страницу, на которой будет отображаться количество раз, которое пользователь посетил страницу /visit. Количество посещений хранить в сессии

24. Реализуйте сайт, на котором будет отображаться список постов. Реализуйте возможность оценки каждого поста. Количество «Понравилось» должно сохраняться в базе данных.

25. Реализуйте сайт, на котором пользователь сможет записывать свои доходы и расходы, результат должен отображаться на странице. Все транзакции записываются в базу данных

26. Реализуйте регистрацию и авторизацию по почте и паролю. Данные о пользователе хранятся в куки. Профиль пользователя получается по пути /profile

Результаты обучения (освоенные умения, усвоенные знания)	Перечень вопросов и заданий
Уметь: – выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам; – работать с современными case-средствами проектирования баз данных; – проектировать логическую и физическую схемы баз данных; – проектировать серверную часть; – применять современные методы проектирования	Задачи № 1-30 Задачи № 4, 5, 16, 20, 28, 29

приложений% – работать с объектами базы данных, включая загрузку и отображение изображений; – применять стандартные методы для защиты объектов базы данных; – разрабатывать сложные sql-запросы для анализа данных; – уметь находить уязвимые места в приложении; – понимание устройства современных сайтов; – разрабатывать интерфейсы для авторизации, фильтрации и ввода; – обеспечивать информационную безопасность на уровне базы данных. ***Знать:*** – основные виды используемых архитектур сайтов; – устройство сети; – устройство сетевых протоколов; – принцип работы шифрования; – принцип работы хеширования; – принцип работы брутфорса – методы описания схем баз данных в современных системах управления базами данных (СУБД); – структуры данных в СУБД: организация представлений, таблиц, индексов и кластеров; – методы организации целостности данных; – способы контроля доступа к данным и управления привилегиями; – основные методы и средства защиты данных в базах данных; – импорт и экспорт данных.	Задачи № 6, 8, 14, 19, 25, 26 Задачи № 3, 10, 12, 24 Задачи № 2, 8, 11, 18, 22, 26 Задачи № 9, 15, 17, 22, 27 Задачи № 7, 11, 20, 22 Задачи № 13, 18, 28, 30 Задачи № 20, 22, 27 Вопросы № 1, 2, 7 Вопросы № 8, 9 Вопросы № 3-5, 28 Вопросы № 10, 13, 12, 29 Вопросы № 6, 25-26 Вопросы № 11, 22, 8, 32 Вопросы № 31, 34 Вопросы № 35- 36 Вопросы № 14, 15 Вопросы № 17, 18 Вопросы № 19-21, 23 Вопросы № 27, 30 Вопросы № 28, 29 Вопросы № 32, 33

Критерии оценивания:

1. **Оценка «5» (26-30 баллов) ставится, если обучающийся:**

- дает правильный и полный ответ на поставленный вопрос;
- способен привести необходимые примеры;
- четко и коротко формулирует ответы на дополнительные вопросы;
- свободно оперирует техническими терминами;
- безошибочно выполняет практическое задание;

2. **Оценка «4» (21-25 баллов) ставится, если обучающийся**:

- дает правильный ответ на поставленный вопрос, но допускает неточности;
- способен привести необходимые примеры с помощью преподавателя;
- допускает ошибки в терминологии;
- самостоятельно или с помощью преподавателя выполнено практическое задание, не делая при этом ошибок в вычислениях; допускаются лишь ошибки второстепенного характера;

3. **Оценка «3» (15 20 баллов) ставится, если обучающийся**:

- допускает ошибки при ответе на поставленный вопрос;
- допускает ошибки в терминологии;
- не способен привести необходимые примеры;
- практическое задание выполняет не полностью, либо с ошибками, которые может исправить только с помощью преподавателя;

4. **Оценка «2» (менее 15 баллов) ставится, если обучающийся**:

- имеет поверхностные знания по дисциплине;
- не способен привести необходимые примеры;
- не выполнено практическое задание.

Перевод оценок из 100-балльной системы в пятибалльную

Форма промежуточной аттестации	**Результат**	**Положительный результат**		
экзамен	неудовлетворительно	удовлетворительно	хорошо	отлично
	(менее 50 баллов)	50-64 баллов	65-84 баллов	85-100 баллов

Преподаватель _____ Левченко И.А.