федеральное государственное автономное образовательное учреждение высшего образования
Первый Московский государственный медицинский университет им. И.М. Сеченова Министерства здравоохранения Российской Федерации (Сеченовский Университет)

Институт цифрового биодизайна и искусственного интеллекта в медицине
Кафедра информационных технологий и обработки медицинских данных

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

Веб-программирование

основная профессиональная образовательная программа высшего профессионального образования - программа бакалавриата

45.03.04 Интеллектуальные системы в гуманитарной сфере

Перечень вопросов по курсу для проведения централизованного тестирования.
Правильные варианты ответов обозначены буквой «а».

1. Какой статус-код HTTP указывает на успешный запрос?
a) 200
b) 404
c) 500
d) 301

2. Какой метод HTTP используется для создания ресурса?
a) POST
b) GET
c) PUT
d) DELETE

3. Какой из следующих методов предназначен для обновления существующего ресурса?
a) PUT
b) GET
c) POST
d) PATCH

4. Что обозначает принцип REST "статусность"?
a) Каждый запрос должен содержать всю необходимую информацию для его обработки
b) Сервер хранит состояние клиента
c) Клиент хранит состояние сервера
d) Сервер обрабатывает все запросы одинаково

5. Какой метод используется для удаление ресурса в REST API?
a) DELETE
b) GET
c) POST
d) PUT

6. Какой из следующих протоколов обеспечивает создание защищенного соединения?
a) HTTPS
b) HTTP
c) FTP
d) SMTP

7. Для чего используется CORS?
a) Для разрешения запросов к ресурсам с других доменов
b) Для управления доступом к API
c) Для шифрования данных
d) Для кэширования данных

8. Какой из нижеперечисленных стеков чаще всего используется для разработки

одностраничных приложений?
a) MERN
b) LAMP
c) MEAN
d) Django

9. В каком случае будет возвращен статус-код 404?
a) Когда ресурс не найден
b) Когда запрос успешно обработан
c) Когда сервер внутренне ошибается
d) Когда доступ запрещен

10. Какой из следующих методов используется для получения данных в REST API?
a) GET
b) POST
c) PATCH
d) DELETE

11. Какой из следующих компонентов не является частью архитектуры REST?
a) База данных
b) Клиент
c) Сервер
d) Программный интерфейс

12. Какой метод в React используется для отрисовки компонентов?
a) render()
b) draw()
c) output()
d) display()

13. Какой из следующих инструментов является фреймворком для создания RESTful API на Python?
a) FastAPI
b) Django
c) Flask
d) Node.js

14. Какой компонент управляет состоянием в React?
a) useState
b) useEffect
c) componentDidMount
d) render()

15. Какой из следующих методов используется для получения данных из базы данных в Django REST Framework?
a) get_queryset()
b) fetch_data()
c) retrieve()
d) list_items()

16. Что такое CDN?
a) Сеть доставки контента
b) Центральная база данных
c) Программная платформа
d) Пользовательская точка доступа

17. Какой метод для инициализации проекта в Django?
a) django-admin startproject
b) react init
c) django create project
d) flask init

18. Какой из следующих вариантов является основным принципом клиент-серверной модели?
a) Сервер предоставляет ресурсы клиентам
b) Клиенты могут только запрашивать данные
c) Клиенты и серверы хранят одинаковую информацию
d) Серверы не могут общаться между собой

19. Какой статус-код указывает на ошибку сервера?
a) 500
b) 400
c) 401
d) 403

20. Какой из следующих инструментов используется для тестирования API?
a) Postman
b) SQL
c) React
d) Git

21. Какой класс в DRF обычно используется для сериализации связанных моделей с возможностью чтения и записи полей связанных объектов?
a) serializers.ModelSerializer
b) serializers.Serializer
c) serializers.StringRelatedField
d) serializers.HyperlinkedIdentityField

22. Какой ViewSet обеспечивает реализацию стандартных действий list, retrieve, create, update, partial_update, destroy без ручного описания методов?
a) ModelViewSet
b) ReadOnlyModelViewSet
c) GenericViewSet
d) mixins.CreateModelMixin

23. Какой роутер автоматически генерирует маршруты для стандартных действий ViewSet и регистрирует их по basename?
a) DefaultRouter
b) SimpleRouter
c) RouteSet

d) ManualRouter

24. Какая настройка в сериализаторе ModelSerializer указывает явный список полей, включаемых в сериализацию?
a) fields
b) read_only_fields
c) extra_kwargs
d) depth

25. Какой класс аутентификации в DRF использует заголовок Authorization с токеном формата "Token <key>"?
a) TokenAuthentication
b) SessionAuthentication
c) JWTAuthentication
d) BasicAuthentication

26. Какой permission-класс разрешает доступ только аутентифицированным пользователям (HTTP 401 для анонимов)?
a) IsAuthenticated
b) AllowAny
c) IsAdminUser
d) IsAuthenticatedOrReadOnly

27. Если нужно, чтобы список объектов был доступен для всех, а создание — только для админов, какое сочетание permission и метода предпочтительно?
a) Использовать IsAuthenticatedOrReadOnly и добавить IsAdminUser в perform_create/permission_classes_for_action
b) Указать IsAdminUser глобально
c) Прописать AllowAny глобально
d) Использовать только IsAuthenticated

28. Какой подход в DRF позволяет версионировать API в URL (например /v1/...)?
a) URLPathVersioning
b) Versioning по заголовкам
c) NamespaceVersioning
d) HostNameVersioning

29. Какая пагинация возвращает номера страниц и ссылки на next/previous в ответе (по умолчанию в DRF)?
a) PageNumberPagination
b) LimitOffsetPagination
c) CursorPagination
d) NoPagination

30. Какое поле сериализатора удобно для представления отношений по URL вместо PK (например /users/1/)?
a) HyperlinkedRelatedField
b) PrimaryKeyRelatedField
c) SlugRelatedField
d) StringRelatedField

31. Какой метод ViewSet отвечает за частичное обновление ресурса (PATCH)?
a) partial_update()
b) update()
c) patch_update()
d) modify()

32. Где чаще всего реализуют логику фильтрации queryset на уровне ViewSet?
a) Переопределяют get_queryset()
b) В serializer.validate()
c) В router
d) В urls.py

33. Какой параметр в сериализаторе ModelSerializer позволяет пометить поле только для чтения (не разрешать запись)?
a) read_only=True
b) required=False
c) write_only=True
d) allow_null=True

34. Если нужно вернуть вложенные данные связанной модели при чтении, но принимать только PK при записи, какое решение подходящее?
a) Создать отдельные поля: nested read-only serializer для чтения и PrimaryKeyRelatedField для записи
b) Использовать только PrimaryKeyRelatedField
c) Делать deepcopy в представлении
d) Запрещать обновление связанных полей

35. Какой рекомендуемый способ регистрации ViewSet в urls.py при использовании DefaultRouter?
a) Создать router = DefaultRouter(); router.register('items', ItemViewSet); urlpatterns = router.urls
b) router.register('path', view, basename='x') и вручную добавлять urlpatterns
c) Использовать include('rest_framework.urls')
d) Регистрировать классы через path(..., view.as_view())

36. Что делает метод serializer.is_valid(raise_exception=True)?
a) Выполняет валидацию и бросает ValidationError при ошибках
b) Валидирует и возвращает ошибки в поле errors
c) Сохраняет объект в БД
d) Ничего, просто проверяет типы

37. При использовании CursorPagination, какое преимущество перед LimitOffsetPagination важно учитывать?
a) Гарантирует стабильную пагинацию при изменении набора данных (эффективность и корректность курсора)
b) Более простая реализация
c) Поддерживает только отрицательные оффсеты
d) Возвращает все объекты сразу

38. Какой декоратор или класс используют для ограничения доступа к отдельному действию ViewSet (например только создателям ресурса)?
a) permission_classes или get_permissions с кастомной логикой
b) @api_view
c) throttle_classes
d) renderer_classes

39. Какой метод сериализатора используется для создания объекта из validated_data при вызове serializer.save()?
a) create()
b) validate()
c) to_representation()
d) update()

40. Как правильно включить JWT-аутентификацию в DRF проекте (кратно)?
a) Установить пакет (напр., djangorestframework-simplejwt), добавить его аутентификационный класс в REST_FRAMEWORK['DEFAULT_AUTHENTICATION_CLASSES'] и настроить endpoints/token
b) Добавить middleware
c) Установить DRF-JWT и прописать AUTH_USER_MODEL
d) Использовать TokenAuthentication без изменений

41. Если нужно экспонировать несколько версий API одновременно и переадресовывать v1 к v2 по умолчанию, какое решение подходит?
a) Разместить маршруты в разных namespace (/v1/, /v2/) и настроить редирект с корня на /v2/ или использовать URLPathVersioning с настройкой default_version
b) Удалить v1
c) Версионировать через заголовки и ничего не делать в URL
d) Менять поведение в serializer на основе settings.DEBUG

42. Где лучше всего настраивать глобальные классы permission и authentication для всего API?
a) В settings.py в REST_FRAMEWORK (DEFAULT_PERMISSION_CLASSES, DEFAULT_AUTHENTICATION_CLASSES)
b) В каждом ViewSet отдельно
c) В routers.py
d) В models.py

43. Что такое нормализация данных в реляционных базах и зачем она нужна?
a) Разделение данных на таблицы с целью устранения избыточности и аномалий обновления
b) Процесс денормализации для ускорения чтения
c) Уменьшение размера индексов
d) Автоматическое создание резервных копий

44. Какая операция в SQL наиболее эффективно использует индекс для выборки по диапазону значений?
a) WHERE column BETWEEN x AND y
b) JOIN

c) INSERT
d) UPDATE

45. Какой тип индекса в PostgreSQL эффективен для полнотекстового поиска?
a) GIN (или GiST для некоторых случаев)
b) B-tree
c) Hash
d) BRIN

46. При проектировании схемы для "постов" и "комментариев" где комментарии имеют FK на пост, какое отношение выбирается?
a) One-to-Many (один пост — много комментариев)
b) One-to-One
c) Many-to-Many
d) Polymorphic

47. Какой подход предпочтителен при работе с большими аналитическими запросами, которые не должны нагружать OLTP базу?
a) Использовать репликацию и выполнять аналитические запросы на read replica или ETL в хранилище данных
b) Делать их в транзакциях с автокоммитом
c) Выполнять прямо в основном приложении в фоне
d) Уменьшить индексы, чтобы ускорить вставки

48. Какой механизм миграций в Django отвечает за создание и применение изменений схемы?
a) Django migrations (makemigrations / migrate)
b) South
c) Alembic
d) syncdb

49. При использовании SQLAlchemy, какой объект описывает соответствие между классом и таблицей?
a) Declarative base / mapped class с таблицей (ORM mapping)
b) Session
c) Engine
d) Query

50. Что такое "N+1 проблема" в ORM и как её обычно решают?
a) Множество дополнительных запросов при загрузке связанных объектов; решают через eager loading (select_related/prefetch_related или joinedload)
b) Ошибка синтаксиса SQL
c) Проблема с N-пользователями одновременно
d) Скопление миграций

51. Какой тип NoSQL базы лучше подходит для хранения документов с гибкой схемой (JSON)?
a) Документо-ориентированная (например, MongoDB)
b) Колонко-ориентированная
c) Графовая

d) Ключ-значение

52. Когда стоит использовать денормализацию в реляционной БД?
a) Когда требуется значительно ускорить чтение и допустима некоторая избыточность и сложность при записи/обновлении
b) Никогда
c) Только при использовании NoSQL
d) Чтобы уменьшить потребление памяти на сервере

53. Какой инструмент обычно используют для миграций в проектах на SQLAlchemy?
a) Alembic
b) Django migrations
c) Alembic
d) Liquibase

54. Какой тип кеширования лучше подходит для быстрого хранения сессий и данных с ограниченным TTL?
a) Redis (in-memory, поддерживает TTL и структуры данных)
b) Файловое кеширование
c) Memcached
d) SQLite

55. Что делает индекс covering (покрывающий индекс)?
a) Содержит все поля, необходимые для запроса, позволяя избежать обращения к таблице
b) Индекс только для текстовых полей
c) Индекс сжатия данных
d) Индекс для ускорения вставок

56. Какой способ хранения больших бинарных файлов (изображений, видео) чаще рекомендуют в веб-проектах?
a) Хранить в объектном хранении (S3, MinIO) и в БД — только ссылку/метаданные
b) Хранить в базе данных BLOB
c) Хранить в таблице как base64
d) Хранить в куках

57. При оптимизации запросов, что обычно полезно сделать первым шагом?
a) Посмотреть план выполнения (EXPLAIN / EXPLAIN ANALYZE) и найти узкие места
b) Переустановить сервер БД
c) Удалить все индексы
d) Мигрировать на NoSQL

58. Какой подход в проектировании схемы удобен для хранения временных рядов (metrics, события)?
a) Партиционирование таблиц по времени или использование специализированных TSDB (InfluxDB, TimescaleDB)
b) Обычная реляционная таблица без партиционирования
c) Использовать Many-to-Many для каждых метрик

d) Хранить в JSONB без индексов

59. Что делает инструмент VACUUM в PostgreSQL?
a) Освобождает место от dead tuples, поддерживает статистику и предотвращает раздувание таблиц
b) Делает бэкап базы
c) Рестартует сервер
d) Оптимизирует индексы автоматически

60. Какой тип индекса не подходит для поиска по префиксу строк (например WHERE name LIKE 'abc%')?
a) Hash (ограничен равенствами, не поддерживает диапазоны/префиксы)
b) B-tree
c) Trigram/Gin для более гибкого поиска
d) BRIN для больших таблиц

61. Если нужно кэшировать результат тяжёлого запроса на 5 минут в Django, какое простое решение?
a) Использовать Redis и декоратор cache_page(300) или low-level cache.set(key, value, timeout=300)
b) Писать результат в отдельную таблицу без TTL
c) Записывать в файл и читать синхронно
d) Повторно выполнять запрос в каждый запрос пользователя

62. Какой поток аутентификации чаще всего используется в OAuth2 для серверных приложений (без публичного клиента)?
a) Client Credentials
b) Authorization Code
c) Implicit
d) Resource Owner Password Credentials

63. Какой заголовок помогает предотвратить клики через iframe (clickjacking)?
a) X-Frame-Options
b) Content-Security-Policy
c) X-Content-Type-Options
d) Referrer-Policy

64. Какой тип токена обычно возвращает сервер при аутентификации через JWT?
a) JSON Web Token (JWT) — самодостаточный подписанный токен
b) Сессионный идентификатор в куке
c) OAuth1 token
d) Plain text password

65. Какой механизм CSRF-защиты в Django наиболее распространён?
a) CSRF-токен, вставляемый в формы и проверяемый на сервере
b) Использование SameSite=None
c) BasicAuth
d) Проверка Referer заголовка только

66. Какая настройка cookie рекомендуется для защиты от доступа JavaScript (XSS)?

a) HttpOnly=True
b) SameSite=Strict
c) Secure=True
d) Path=/

67. Какой подход эффективен против SQL-инъекций при работе с базой данных?
a) Использовать параметризованные запросы / подготовленные выражения (ORM или placeholders)
b) Экранировать кавычки вручную
c) Использовать string interpolation для SQL
d) Удалить все индексы

68. Что делает заголовок Content-Security-Policy (CSP)?
a) Ограничивает источники загружаемых скриптов, стилей и ресурсов для предотвращения XSS
b) Шифрует трафик
c) Блокирует все куки
d) Автоматически валидирует ввод

69. При хранении паролей, какой метод считается безопасным?
a) Использовать адаптивный алгоритм хеширования (bcrypt, Argon2, PBKDF2) с солью
b) Хранить в plaintext
c) Хранить MD5-хеш
d) Сохранять только первые 8 символов пароля

70. Какой атрибут cookie нужен, чтобы браузер отправлял cookie только по HTTPS?
a) Secure
b) HttpOnly
c) SameSite=Lax
d) Domain

71. Что такое ревокация JWT и в чём сложность её реализации?
a) Невозможность немедленно отозвать уже выданный подписанный JWT без дополнительной инфраструктуры (blacklist/whitelist, короткий срок жизни + refresh tokens)
b) Удаление токена из БД, простая реализация
c) Отдельный тип токена, который истекает мгновенно
d) Автоматическая при истечении сессии сервера

72. Как безопасно обрабатывать загружаемые пользователями файлы на сервере?
a) Проверять тип и размер, давать уникальные имена, хранить в безопасном объектном хранилище/папке без выполнения и сканировать на вирусы
b) Сохранять файлы в директории с execute-разрешениями
c) Помещать файлы прямо в статические ресурсы без проверки
d) Выполнять все загруженные файлы на сервере для валидации

73. Какой режим SameSite куки помогает уменьшить риск CSRF при сторонних запросах?
a) SameSite=Lax
b) SameSite=None

c) HttpOnly
d) Secure

74. Что делает механизм rate limiting (ограничение запросов) в контексте безопасности?
a) Предотвращает перебор паролей и DoS-атаки путём ограничения частоты запросов от клиента
b) Усиливает шифрование
c) Скрывает ошибки сервера
d) Автоматически блокирует всех анонимных пользователей

75. Какой метод хранения JWT безопаснее для браузерного клиента: localStorage или cookie с HttpOnly и Secure?
a) HttpOnly + Secure cookie (с корректным SameSite), т.к. недоступна JS и менее уязвима к XSS
b) localStorage
c) Оба одинаково
d) В URL параметре

76. Какой заголовок помогает предотвратить MIME-sniffing атакующие загрузку контента неправильного типа?
a) X-Content-Type-Options: nosniff
b) X-Frame-Options
c) Content-Security-Policy
d) Referrer-Policy

77. Какая практика по обновлению зависимостей важна для безопасности приложения?
a) Регулярно отслеживать и применять патчи безопасности (dependabot, регулярные обновления), тестировать изменения
b) Обновлять только major версии
c) Никогда обновлять, чтобы избежать регрессий
d) Обновлять только dev-зависимости

78. При использовании OAuth2 с фронтендом SPA, какой поток сейчас рекомендуют для повышения безопасности?
a) Authorization Code Flow с PKCE
b) Implicit flow
c) Resource Owner Password Credentials
d) Client Credentials

79. Какая практика помогает защититься от утечки чувствительных данных в логах?
a) Redact/маскировать секреты и личные данные в логах, ограничивать доступ и ротацию логов
b) Логировать все поля формы
c) Хранить логи бессрочно
d) Хранить логи в публичных репозиториях

80. Какую роль выполняет HSTS (HTTP Strict Transport Security)?
a) Принуждает браузер обращаться к сайту только по HTTPS в течение заданного

времени
b) Позволяет отдавать HTTP контент быстрее
c) Шифрует содержимое страницы
d) Управляет политикой SameSite куки

81. Какой механизм помогает предотвратить внедрение скриптов через пользовательский ввод при выводе в HTML?
a) Экранирование/санитизация вывода (output encoding) и использование безопасных шаблонизаторов
b) Отключение cookies
c) HTTP Basic Auth
d) Увеличение таймаута сессии

82. Какой принцип следует применять при выдаче прав доступа (authorization) в приложении?
a) Принцип наименьших привилегий — минимальные необходимые права для задачи
b) Давать все права всем пользователям по умолчанию
c) Давать права на основе совпадения IP
d) Отключать авторизацию для внутренних сервисов

83. Какой синтаксис объявляет переменную, значение которой можно переназначать, но имеет блочную область видимости?
a) let
b) var
c) const
d) function

84. Какой метод массива возвращает новый массив, содержащий результаты вызова функции для каждого элемента?
a) map()
b) forEach()
c) find()
d) filter()

85. Что делает оператор "??" (nullish coalescing) в JavaScript?
a) Возвращает левый операнд, если он не равен null/undefined; иначе правый
b) Возвращает левый операнд, если он truthy; иначе правый
c) Вызывает оператор побитового И
d) То же самое, что ||

86. Какой способ правильно объявить асинхронную функцию и внутри выполнить fetch с ожиданием ответа?
a) async function foo() { const res = await fetch(...); }
b) function foo() { fetch(...) }
c) async function foo() { fetch(...) }
d) function* foo() { yield fetch(...) }

87. Какой метод DOM используется для безопасного добавления HTML-элемента в документ?
a) createElement + appendChild (или append)

b) innerHTML = html
c) insertAdjacentHTML('beforeend', html)
d) document.write(html)

88. Как предотвратить стандартное поведение события (например сабмита формы) в обработчике?
a) event.preventDefault()
b) return false
c) stopPropagation()
d) stopImmediatePropagation()

89. Какой подход уменьшает вероятность блокировки основного потока при выполнении тяжёлой работы в фронтенде?
a) Вынести работу в Web Worker
b) Использовать sync XHR
c) Поместить всё в setTimeout(fn, 0)
d) Делать больше DOM-манипуляций синхронно

90. Что делает метод fetch() по умолчанию при получении ответа с ошибочным HTTP-статусом (например 404)?
a) Возвращает resolved Promise с объектом Response; нужно проверять response.ok
b) Бросает исключение
c) Возвращает null
d) Переподключается автоматически

91. Какая конструкция модуля ES экспортирует по умолчанию одно значение?
a) export default function() {...}
b) export const x = ...
c) module.exports = ...
d) require('...')

92. При использовании async/await, как правильно обрабатывать ошибки асинхронного кода?
a) Оборачивать await в try { await ... } catch (err) { ... }
b) Ничего — ошибки не происходят
c) Использовать .catch после await
d) Ловить ошибки только глобальным обработчиком onerror

93. Какой метод позволяет слушать событие клика на множестве динамически создаваемых элементов эффективно (event delegation)?
a) (вариант 3) — делегировать событие на общего родителя и фильтровать по селектору
b) Назначать обработчик на каждый элемент при создании
c) Использовать setInterval для проверки кликов
d) addEventListener('click', handler) на document и проверять event.target с селектором

94. Что делает метод Promise.allSettled(iterable)?
a) Ждёт завершения всех промисов и возвращает их статусы и результаты независимо от успеха/ошибки
b) Возвращает первый успешный промис

c) Останавливает выполнение, если один промис отклонён
d) То же что Promise.all

95. Какой инструмент обычно используют для транспиляции современного JavaScript (ES6+) в совместимый код для старых браузеров?
a) Babel
b) ESLint
c) Webpack dev server
d) PostCSS

96. Что такое tree-shaking в контексте сборщиков модулей?
a) Удаление неиспользуемого экспортированного кода при сборке
b) Минификация CSS
c) Удаление синтаксических ошибок
d) Запуск тестов при сборке

97. Какой заголовок нужно поставить при fetch, чтобы отправлять JSON-данные на сервер?
a) Content-Type: application/json
b) Accept: text/html
c) Authorization: Bearer ...
d) X-Requested-With: XMLHttpRequest

98. Какой метод массива используется для последовательного уменьшения массива к одному значению (сумма, конкатенация)?
a) reduce()
b) map()
c) filter()
d) find()

99. При работе с большим числом DOM-обновлений, какой приём лучше уменьшит количество перерисовок?
a) Изменять DocumentFragment или использовать виртуальный DOM и затем вставлять единожды
b) Изменять DOM по одному элементу
c) Использовать document.write()
d) Вызывать getComputedStyle часто

100. Какой синтаксис импорта модулей ES корректен для имени экспорта "foo"?
a) import { foo } from 'mod';
b) import foo from 'mod';
c) require('mod').foo
d) export foo from 'mod';

101. Что делает оператор optional chaining (?.) в выражении obj?.prop()?
a) Если obj или prop отсутствуют, выражение возвращает undefined вместо ошибки; иначе вызывает метод
b) Вызывает prop независимо от obj
c) Бросает ошибку, если obj undefined
d) Преобразует значение в булево

102. Какой подход рекомендуется для безопасной работы с REST API из фронтенда при необходимости аутентификации по токену?
a) Хранить токен в защищённом HttpOnly cookie или, при использовании токена в JS, минимизировать TTL и защищать приложение от XSS; добавлять Authorization заголовок при запросах
b) Хранить токен в localStorage и прикреплять вручную ко всем запросам
c) Отправлять токен в URL параметре
d) Никогда использовать токены; только Basic auth

103. Какой класс в FastAPI обычно используют для объявления входных/выходных схем и валидации данных?
a) pydantic.BaseModel
b) BaseModel из dataclasses
c) typing.NamedTuple
d) sqlalchemy.Model

104. Какой способ определения маршрута в FastAPI соответствует HTTP GET по пути /items/{item_id}?
a) @app.get('/items/{item_id}') def read_item(item_id: int): ...
b) app.route('/items/{item_id}', methods=['GET'])
c) @app.get('/items') def get_item(item_id): ...
d) @app.post('/items/{item_id}')

105. Как пометить в Pydantic поле как опциональное (могут быть None)?
a) from typing import Optional; field: Optional[str] = None
b) field: Optional = None
c) field: str | None
d) field?: str

106. Какой параметр функции-обработчика позволяет FastAPI автоматически валидировать и преобразовывать параметры пути и query?
a) аннотации типов аргументов функции (type hints)
b) request.args
c) raw body parsing
d) json.loads вручную

107. Как объявить асинхронный маршрут в FastAPI?
a) @app.get('/') async def index(): await ...
b) def route(): ...
c) @app.get('/') def index(): ...
d) @app.get('/') generator function

108. Что делает Depends в FastAPI?
a) Объявляет зависимость (dependency injection) для переиспользуемой логики/валидации/аутентификации
b) Импортирует зависимости из pip
c) Выполняет middleware
d) Создаёт сессию в базе данных автоматически

109. Какой ответ FastAPI автоматически генерирует на основе Pydantic-схемы для документации?
a) OpenAPI schema и JSON Schema для моделей, используемых в запросах/ответах
b) Только список эндпоинтов
c) Только примеры запросов
d) SQL-модель для БД

110. Как в FastAPI вернуть кастомный HTTP-статус (например 201) из обработчика?
a) from fastapi import Response; return Response(content=..., status_code=201) или return JSONResponse(status_code=201, content=...)
b) return {'status': 201}
c) raise HTTPException(status_code=201)
d) Нельзя менять статус — только 200

111. Где логично регистрировать middleware в приложении FastAPI?
a) При создании приложения: app.add_middleware(...) или с помощью @app.middleware('http')
b) Внутри обработчика каждого маршрута
c) Перед импортом fastapi
d) В Pydantic моделях

112. Что делает BackgroundTasks в FastAPI?
a) Позволяет зарегистрировать задачу, которая выполнится после ответа клиенту в том же процессе
b) Запускает отдельный процесс на сервере
c) Создаёт cron job
d) Асинхронно обновляет OpenAPI схему

113. Какой тип в Pydantic удобно использовать для валидации e-mail?
a) pydantic.EmailStr
b) str
c) Any
d) typing.Email

114. Какой инструмент встроен в FastAPI для автогенерации интерактивной документации с UI?
a) Swagger UI и ReDoc (по умолчанию доступные по /docs и /redoc)
b) Swagger только
c) ReDoc только
d) Sphinx

115. Как получить тело запроса как сырой bytes внутри обработчика FastAPI?
a) await request.body()
b) request.body()
c) request.json()
d) request.data

116. Как правильно управлять подключением к базе данных через зависимости, чтобы закрыть соединение после запроса?
a) Использовать зависимость с yield: setup -> yield resource -> teardown (закрытие после

yield)
b) Открывать в глобальной переменной
c) Оставлять открытым навсегда
d) Делать commit в каждом маршруте вручную

117. Какой ответ/исключение в FastAPI используют для возврата ошибки с определённым статусом и сообщением?
a) fastapi.HTTPException(status_code=400, detail='msg')
b) ValueError
c) return {'error': 'msg', 'code': 400}
d) raise Exception('HTTP 400')

118. Как валидация Pydantic обрабатывает дополнительные неизвестные поля по умолчанию?
a) По умолчанию ignore/forbid/allow можно настроить через Config; стандартно extra = ignore (в новых версиях — ignore)
b) Всегда выбрасывает ошибку
c) Удаляет все поля
d) Сохраняет их в отдельное поле meta

119. Какой тип ответа удобно использовать для стриминга больших файлов в FastAPI?
a) StreamingResponse или FileResponse
b) return file path string
c) return bytes
d) Сохранять файл в логах

120. Как предотвратить блокировку при вызове синхронной тяжёлой функции внутри async-обработчика?
a) Запустить в пуле исполнителей: await run_in_threadpool(sync_func, ...) или use asyncio.to_thread
b) Ничего не делать
c) Делать await sync_func()
d) Переписать весь код синхронно

121. Как указать в маршруте, что ответ должен соответствовать Pydantic-модели и автоматически документироваться?
a) Указать response_model=MyModel в декораторе маршрута
b) Только в описании документации
c) return model instance без аннотаций
d) Сериализовать вручную в JSON

122. Как настроить версионирование API в FastAPI без сторонних библиотек при сохранении OpenAPI для каждой версии?
a) Создать отдельные FastAPI / APIRouter экземпляры для каждой версии и монтировать их (app.mount or include_router с префиксом /v1, /v2)
b) Менять Pydantic модели по версии
c) Использовать один большой router с условием версии
d) Версионировать только через заголовки без изменений роутинга

123. Что делает команда docker run -d --name web -p 8080:80 myapp?
a) Запускает контейнер в фоновом режиме (-d), даёт имя web и пробрасывает порт 8080 хоста на 80 контейнера
b) Запускает контейнер в интерактивном режиме
c) Запускает контейнер и удаляет его после остановки
d) Собирает образ myapp из Dockerfile

124. Какой файл обычно используют для описания мультиконтейнерной конфигурации локально (dev)?
a) docker-compose.yml
b) Dockerfile
c) .env
d) kubernetes.yaml

125. Что такое container image layer и почему это важно при оптимизации образа?
a) Слойная файловая система образа; уменьшение количества/размера слоёв и порядок инструкций уменьшают размер и ускоряют кеширование сборки
b) Способ шифрования образа
c) Облако для хранения образов
d) Раздел в docker-compose для volumes

126. Какой подход уменьшает размер итогового Docker-образа в многоступенчатой сборке?
a) Использовать multi-stage build: собирать в одном этапе и копировать только артефакты в минимальный runtime-образ
b) Использовать больше RUN команд
c) Копировать весь контент проекта на этапе финального образа
d) Увеличивать количество пакетов в образе

127. Что даёт использование read-only файловой системы в контейнере (readonly rootfs)?
a) Повышает безопасность, предотвращая запись/изменение в рантайме; нужно выделять writable тома для данных
b) Ускоряет запись логов
c) Позволяет контейнеру устанавливать пакеты при старте
d) Автоматически обновляет образ при перезапуске

128. В CI/CD пайплайне, где лучше запускать интеграционные тесты, которые требуют поднятия сервисов (БД, очередь)?
a) В изолированном этапе пайплайна (staging/test environment или using ephemeral environments/containers) до деплоя
b) На продакшен-сервере после деплоя
c) Никогда их не запускать
d) Локально на машине разработчика

129. Какой подход для управления конфигурацией приложения в разных окружениях (dev/stage/prod) соответствует 12-factor app?
a) Использовать переменные окружения: конфигурация через ENV, не через закоммиченные файлы
b) Хранить конфиг в коде

c) Создавать отдельные ветки репозитория для каждого окружения
d) Хранить файлы конфигурации в публичном S3

130. Что такое health check readiness probe в Kubernetes и зачем он нужен?
a) Проверяет, готов ли контейнер принимать трафик; если не готов — сервис не направляет на него запросы
b) Тестирует производительность кластера
c) Перенастраивает контейнеры при старте
d) Делает бэкап контейнера

131. При настройке балансировщика нагрузки, зачем настраивать проверки состояния бекендов (health checks)?
a) Чтобы не направлять трафик на упавшие или некорректно работающие экземпляры, обеспечить устойчивость
b) Чтобы увеличить стоимость инфраструктуры
c) Для логирования всех запросов
d) Для автоматического масштабирования базы данных

132. Какой инструмент часто используют для централизованного сбора логов из контейнеров и их обработки?
a) Fluentd/Fluent Bit + Elasticsearch + Kibana (часть EFK/ELK стека)
b) systemd journal только локально
c) GitHub Actions
d) Rsync

133. Что такое immutable tags vs mutable tags для Docker-образов и почему immutable предпочтительнее в деплое?
a) Immutable (фиксированные версии, напр. myapp:1.0) гарантируют воспроизводимость релиза; mutable (latest) может привести к неопределённому поведению при перезапуске
b) Immutable — изменяемые теги, mutable — постоянные
c) Immutable теги удаляются автоматически
d) Только immutable можно хранить в реестре

134. Какой показатель в мониторинге лучше использовать для триггера автоскейлинга?
a) Метрики нагрузки: CPU, latency, очередь задач или специфические бизнес-метрики (например длина очереди)
b) Количество строк в логах
c) Количество ошибок 404
d) Время последнего релиза

135. Что такое secret rotation и зачем её выполнять в инфраструктуре?
a) Периодическая замена секретов (ключей, паролей, токенов) чтобы уменьшить риск компрометации и соответствовать политикам безопасности
b) Смена логов по расписанию
c) Переустановка сервисов
d) Репликация секретов между окружениями

136. При деплое базы данных в продакшен, почему стоит использовать бэкапы и

проверять их восстановление на регулярной основе?
a) Чтобы гарантировать возможность восстановления после отказа или повреждения данных — тестировать резервные копии регулярно
b) Чтобы занимать место в хранилище
c) Для ускорения запросов
d) Потому что CI требует наличия бэкапов

137. Какой способ передачи конфигурации в контейнеры безопаснее для секретов?
a) Использовать secret-провайдеры/secret менеджеры и монтировать секреты динамически (Kubernetes Secrets, Vault)
b) Хранить секреты в ENV в репозитории
c) Встраивать секреты в образ на этапе сборки
d) Отправлять секреты по email при старте контейнера

138. Что такое canary deployment и в чём его преимущество по сравнению с blue-green?
a) Постепенный выпуск новой версии на небольшой процент трафика для наблюдения поведения; позволяет минимизировать риск и быстро откатить при проблемах
b) Переименование окружений
c) Полный откат продакшена
d) Использование только одной среды

139. Как уменьшить время доставки артефактов в CI/CD между пайплайн-стадиями?
a) Использовать артефакт-репозитории/кеширование (например Nexus, Artifactory, Docker registry) и кэш стадий сборки
b) Копировать артефакты по FTP вручную
c) Хранить артефакты в Git-репозитории
d) Пересобирать всё с нуля в каждой стадии

140. Какой подход нужен для разграничения прав доступа к инфраструктуре между командами?
a) RBAC (role-based access control) и принцип наименьших привилегий; отдельные сервисные аккаунты и аудит доступа
b) Делиться общими учетками
c) Никому не давать доступ
d) Хранить пароли в общих заметках

141. Какой метод уменьшает риск утечки логов с чувствительными данными?
a) Маскировать/редактировать чувствительные поля в логах, фильтровать перед отправкой в централизованную систему и ограничивать доступ
b) Логировать всё подряд
c) Хранить логи в публичных хранилищах
d) Отправлять логи по email всем разработчикам

142. Что рекомендуется для управления конфигурациями инфраструктуры как кодом (IaC)?
a) Использовать инструменты IaC (Terraform, CloudFormation, Ansible) с версионированием, review и тестированием изменений
b) Ручные команды на серверах
c) Хранить сценарии в приватных текстовых файлах

d) Редактировать конфигурацию через веб-интерфейс без аудита

143. Что делает fixture в pytest и зачем она нужна?
a) Позволяет подготовить и повторно использовать окружение/ресурсы для тестов (setup/teardown)
b) Запускает тесты параллельно
c) Предоставляет параметризованный набор данных автоматически
d) Генерирует отчёты о покрытии

144. Какой командой запускают pytest с подробным выводом и показом пропущенных тестов?
a) pytest -v
b) pytest --short
c) pytest -q
d) pytest --quiet

145. Что такое мок (mock) в контексте юнит-тестирования?
a) Объект-заглушка, имитирующий поведение внешних зависимостей для изоляции тестируемого кода
b) Дополнительный тестовый фреймворк
c) Запуск тестов в контейнере
d) Инструмент для генерации данных

146. Какой подход предпочтителен для тестирования HTTP-эндпоинтов в FastAPI/Django?
a) Использовать клиент тестовой библиотеки (TestClient) и мокировать внешние зависимости
b) Тестировать только через UI
c) Использовать реальные внешние сервисы в тестах
d) Не тестировать эндпоинты вовсе

147. В pytest, как указать, что тест должен ожидать выброса исключения ValueError?
a) with pytest.raises(ValueError): функция()
b) try/except внутри теста
c) assert raises(ValueError)
d) pytest.expect(ValueError)

148. Что такое интеграционный тест по сравнению с юнит-тестом?
a) Тест, проверяющий взаимодействие между модулями/сервисами (включая БД, сети) в отличие от изолированных юнит-тестов
b) Тот же самый тест с другой библиотекой
c) Тест, который выполняется медленнее по времени
d) Тест только для UI

149. Какой инструмент в JS экосистеме часто используют для тестирования React-компонентов?
a) Jest + React Testing Library (или Enzyme)
b) Jest только
c) Cypress только
d) Mocha без assertion

150. Что такое code coverage и почему важно смотреть не только процент покрытия?
a) Покрытие показывает, какие участки кода выполнялись тестами; важно качество тестов и проверка логики, а не лишь высокий процент
b) Количество строк в проекте
c) Количество тестов
d) Метрика производительности тестов

151. Как правильно документировать REST API, чтобы клиенты могли автоматически генерировать SDK?
a) Предоставить полноценную OpenAPI/Swagger схему с описаниями, типами и примерами
b) Написать README вручную
c) Создать endpoint /docs только
d) Описать API в виде таблицы в Word

152. Что даёт использование contract tests (потребитель-поставщик) в микросервисной архитектуре?
a) Гарантирует, что интерфейсы между сервисами соответствуют ожиданиям сторон (consumer-driven contracts), снижая регрессии при изменениях
b) Ускоряет деплой на прод
c) Полностью заменяет интеграционные тесты
d) Обеспечивает безопасность API

153. Какой формат чаще используют для машинно-читаемой API-документации?
a) OpenAPI (JSON/YAML)
b) Markdown только
c) PDF
d) Plain text

154. Что рекомендуется включать в техническую документацию для разработчиков?
a) Архитектуру, инструкции по локальному запуску, схемы данных, API спецификации, примеры использования и правила деплоя
b) Только архитектурные диаграммы
c) Только список зависимостей
d) Только контакты авторов

155. Какой практикой предотвращают регрессии при изменении API?
a) Поддерживать автоматические тесты (unit/integration/contract) и версионирование API
b) Не менять API никогда
c) Писать только ручные тесты
d) Увеличивать таймауты в тестах

156. В CI, какой шаг обеспечивает качество кода до слияния в main?
a) Пайплайн pre-merge: линтинг, unit-тесты, статический анализ и, при необходимости, интеграционные тесты
b) Деплой на продакшен
c) Запуск только линтера
d) Ручная проверка без автоматизации

157. Что такое flaky test и как с ними справляться?
a) Нестабильные тесты с непредсказуемым результатом; решают чрез детерминизацию окружения, убирают сетевые зависимости/тайминги или помечают/исправляют их
b) Тесты, которые всегда проходят
c) Тесты, написанные на стороннем языке
d) Тесты с высокой производительностью

158. Какой инструмент используют для автоматической генерации документации на основе OpenAPI в большинстве фреймворков?
a) Swagger UI / ReDoc
b) Sphinx
c) JSDoc
d) Doxygen

159. Почему полезно включать примеры запросов/ответов в API документацию?
a) Потому что примеры ускоряют интеграцию клиентов и уменьшают ошибки при использовании API
b) Для украшения документации
c) Чтобы увеличить объём документа
d) Это обязательное требование всех стандартов

160. Какую роль играет статический анализатор (linters, type checkers) в процессе обеспечения качества кода?
a) Находит стильные и потенциальные ошибочные участки кода до выполнения, улучшает читаемость и предотвращает баги
b) Запускает unit-тесты
c) Проводит нагрузочное тестирование
d) Заменяет код-ревью

161. Как организовать тестовые данные для повторяемых тестов?
a) Применять фикстуры/фабрики или изолированные тестовые БД и откатывать изменения между тестами
b) Использовать живую продакшен базу
c) Редактировать данные вручную перед каждым тестом
d) Хранить данные в локальном файле без контроля версий

162. Что такое Definition of Done (DoD) в контексте стандартов разработки и документации?
a) Набор критериев, которые должны быть выполнены, чтобы функциональность считалась завершённой (тесты, документация, обзоры, CI)
b) Список задач в багтрекере
c) Политика деплоя
d) Только завершение кода без тестов

163. Какой формат отчёта о тестах удобно интегрировать в CI для аналитики и отображения результатов?
a) JUnit XML (или другие стандартизованные форматы), которые CI-системы могут парсить и визуализировать
b) HTML только локальный

c) Plain text в логах
d) Скриншоты тестов

Принята на заседании кафедры Информационных технологий и обработки медицинских данных ЦЦМиИИМ ИЦБиИИМ

от «23» ноября 2024 г., протокол № 5

Заведующий кафедрой		Лебедев Г.С.
Информационных технологий и обработки медицинских данных ЦЦМиИИМ ИЦБиИИМ	(подпись)	(фамилия, инициалы)