МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

УТВЕРЖДАЮ
И.о. директора института
_А.А. Порохня
Ф.И.О.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ
Разработка Web-приложений

Направление подготовки	09.03.04 Программная инженерия
Направленность (профиль)	Инженерия сложных программных систем
Год начала обучения	2026
Форма обучения	очная
Реализуется в семестрах	3,4

Разработано
Заведующая межинститутской базовой кафедры департамента цифровых, робототехнических систем и электроники
(должность разработчика)
Новикова Елена Николаевна
Ф.И.О.

Ставрополь 2026 г.

1. Цель и задачи освоения дисциплины (модуля)

Цель дисциплины: Сформировать у студентов комплексные знания и практические навыки для самостоятельной разработки, тестирования, развертывания и сопровождения промышленных веб-приложений с использованием современных технологий (Django, React, Docker, CI/CD).

Задачи освоения дисциплины:

- Изучить архитектуру современных веб-приложений (клиент-сервер, REST, SPA).
- Освоить принципы проектирования реляционных БД и работы с PostgreSQL через ORM.
- Изучить современные паттерны проектирования (MVC/MVT, Repository, Dependency Injection).
- Освоить методы аутентификации и авторизации (Session, Token, JWT).
- Сформировать навыки разработки бэкенда на Python с использованием Django и DRF.
- Сформировать навыки разработки фронтенда на JavaScript/TypeScript с использованием React.
- Освоить создание RESTful API (сериализация, валидация, фильтрация, пагинация).
- Научиться интеграции фронтенда и бэкенда через API и настройке CORS.
- Освоить методы тестирования веб-приложений (unit, integration, e2e).
- Изучить подходы к обеспечению безопасности (XSS, CSRF, SQL-инъекции, CORS).
- **2. Место дисциплины (модуля) в структуре образовательной программы**

Дисциплина «Разработка Web-приложений» относится к части, формируемой участниками образовательных отношений.

3. Перечень планируемых результатов обучения по дисциплине (модулю), соотнесённых с планируемыми результатами освоения образовательной программы

Код, формулировка компетенции	**Код, формулировка индикатора**	**Планируемые результаты обучения по дисциплине (модулю) (с учетом «Содержание дисциплин»)**
ПК-2 Способен выполнять проектирование и разработку архитектуры программных систем среднего и крупного масштаба с применением современных паттернов и	**ИД-1пк-2**: Выделяет и анализирует функциональные и нефункциональные требования (FURPS+), выбирает архитектурный стиль (монолит, микросервисы, EDA, Clean Architecture) на основе сценариев	**Анализирует** нефункциональные требования (производительность, масштабируемость, отказоустойчивость, безопасность) с использованием метода FURPS+ и сценариев качества; **выбирает** архитектурный стиль (монолит, микросервисы, EDA, Clean Architecture) на основе требований к latency, пропускной способности и сложности разработки; **применяет** архитектурный

Код, формулировка компетенции	Код, формулировка индикатора	Планируемые результаты обучения по дисциплине (модулю) (с учетом «Содержание дисциплин»)
методов проектирования	качества	паттерн PCMEF и его адаптации для различных типов приложений (десктопных, веб-, мобильных).
	ИД-2пк-2: Документирует архитектуру с использованием нотаций C4 и UML, фиксирует архитектурные решения в формате ADR (Architecture Decision Record)	**Строит** архитектурные представления системы среднего масштаба в нотациях C4 (Context, Container, Component) и UML (диаграммы классов, последовательностей, компонентов); **фиксирует** архитектурные решения в формате ADR (Architecture Decision Record) с обоснованием выбора и альтернатив; **документирует** архитектуру с использованием Arc42, C4 model и ADR.
	ИД-3пк-2: Применяет порождающие, структурные и поведенческие паттерны GoF, а также архитектурные паттерны (CQRS, Event Sourcing, Saga, Circuit Breaker)	**Применяет** паттерны GoF (порождающие, структурные, поведенческие) для решения типовых задач при разработке ПО; **реализует** архитектурные паттерны CQRS (разделение Command и Query моделей), Event Sourcing (хранение событий и восстановление состояния через replay), Saga (хореография и оркестрация с компенсирующими транзакциями) и Circuit Breaker (состояния Closed/Open/Half-Open) для обеспечения отказоустойчивости.
	ИД-4пк-2: Реализует многослойную архитектуру (Clean Architecture, Hexagonal, PCMEF), обеспечивая независимость от фреймворков через	**Реализует** многослойную архитектуру (Clean Architecture, Hexagonal, PCMEF), обеспечивая независимость от фреймворков и внешних деталей через Dependency Inversion Principle (инверсия зависимостей); **проектирует** иерархии классов и структур данных для представления сложных предметных

Код, формулировка компетенции	Код, формулировка индикатора	Планируемые результаты обучения по дисциплине (модулю) (с учетом «Содержание дисциплин»)
	Dependency Inversion	областей; **трансформирует** существующую PCMEF-архитектуру в Clean Architecture с выделением слоев (Entities, Use Cases, Interface Adapters, Frameworks).
ПК-3 Владеет навыками разработки программного обеспечения: способен реализовывать алгоритмы, писать код на современных языках программирования, создавать веб- и desktop-приложения, работать с базами данных	**ИД-1пк-3**: Разрабатывает алгоритмы и реализует их на языках C++, C#, Python, Java, JavaScript/TypeScript, Go с использованием эффективных структур данных и стандартных библиотек	**Реализует** алгоритмы сортировки, поиска, обхода графов и деревьев на C++/Python с эффективным использованием динамической памяти и стандартных библиотек (STL); **разрабатывает** алгоритмы решения типовых вычислительных задач (поиск, сортировка, табулирование) и реализует их в виде консольных приложений; **демонстрирует** знание синтаксиса и семантики C#/C++/Python (типы данных, управляющие конструкции, принципы ООП, структуры данных).
	ИД-2пк-3: Создаёт клиент-серверные, веб- (frontend + backend) и desktop-приложения с графическим интерфейсом, применяя современные фреймворки (React, Spring, Django, WPF) и протоколы (REST, gRPC, WebSocket)	**Создает** desktop-приложения с графическим интерфейсом (Windows Forms / WPF), обрабатывает события, отображает графику и работает с файловой системой; **разрабатывает** бэкенд на Django/Spring Boot (REST API, ORM, аутентификация) и фронтенд на React (компоненты, хуки, управление состоянием через Redux Toolkit/React Query); **реализует** клиент-серверные приложения с использованием протоколов REST, gRPC (Protocol Buffers) и WebSocket (real-time уведомления).
	ИД-3пк-3: Проектирует,	**Проектирует** реляционные схемы (нормализация до 3НФ) и нереляционные

Код, формулировка компетенции	Код, формулировка индикатора	Планируемые результаты обучения по дисциплине (модулю) (с учетом «Содержание дисциплин»)
	мигрирует и оптимизирует запросы к реляционным и нереляционным БД с использованием SQL и ORM (Hibernate, Entity Framework, Django ORM)	структуры данных (документные, графовые); **пишет** оптимизированные SQL-запросы (оконные функции, CTE, сложные JOIN) и реализует CRUD-операции через ORM (Hibernate, Entity Framework, Django ORM) с устранением проблемы N+1; **разрабатывает** хранимые процедуры, триггеры и функции на стороне сервера БД; **управляет** миграциями БД (Liquibase, Flyway).
	ИД-4пк-3: Пишет модульный, документированный и тестируемый код, соблюдая стандарты оформления и принципы SOLID, использует системы сборки (Make, Gradle, npm)	**Пишет** чистый, структурированный код с соблюдением стандартов оформления (отступы, именование, комментарии) и принципов SOLID; **оформляет** программный код, делая его пригодным для чтения, сопровождения и тестирования; **использует** системы сборки (Make, Gradle, npm) и инструменты статического анализа (SonarQube, линтеры) для обеспечения качества кода.
ПК-8 Способен адаптировать и применять методы искусственного интеллекта и машинного обучения для решения профессиональных задач, включая интеграцию готовых моделей в	**ИД-1пк-8**: Выполняет предобработку и анализ данных (EDA, визуализация, кодирование, масштабирование) для подготовки датасета к обучению	**Выполняет** предобработку данных (обработка пропусков и выбросов, кодирование категориальных признаков One-Hot/Label Encoding, масштабирование StandardScaler/MinMaxScaler) с использованием Pandas и scikit-learn; **проводит** исследовательский анализ данных (EDA) с визуализацией (Matplotlib/Seaborn: гистограммы, boxplots, pair plots, корреляционные матрицы); **создает** пайплайны предобработки с использованием Pipeline и

Код, формулировка компетенции	Код, формулировка индикатора	Планируемые результаты обучения по дисциплине (модулю) (с учетом «Содержание дисциплин»)
программные продукты		ColumnTransformer для воспроизводимости.
	ИД-2пк-8: Реализует и обучает модели ML (классификация, регрессия, кластеризация) с использованием scikit-learn, XGBoost, LightGBM, а также нейросетевых архитектур (CNN, RNN, Transformer) в TensorFlow/PyTorch	**Реализует** и обучает модели машинного обучения (логистическая регрессия, Random Forest, XGBoost, LightGBM) для задач классификации, регрессии и кластеризации (k-means, DBSCAN); **применяет** нейросетевые архитектуры (многослойный перцептрон, CNN для изображений, RNN/LSTM для последовательностей, Transformer для NLP) с использованием TensorFlow/Keras или PyTorch; **проводит** гиперпараметрическую оптимизацию (GridSearch, Optuna) и кросс-валидацию для выбора лучшей модели.
	ИД-3пк-8: Интегрирует предобученные модели и API ИИ (Hugging Face, OpenAI, YOLO) в программные продукты, разворачивает их как микросервисы и оптимизирует инференс (кэширование, батчинг)	**Интегрирует** предобученные модели (Hugging Face Transformers, YOLO) и API ИИ (OpenAI API, Yandex GPT) в программные продукты через REST API; **разворачивает** модели как микросервисы с использованием FastAPI/Flask и Docker; **оптимизирует** инференс с помощью кэширования (exact match и similarity cache), динамического батчирования запросов и асинхронной обработки через очереди (Celery, Kafka); **выполняет** дообучение (fine-tuning) предобученных моделей под конкретную предметную область.

4. Объем учебной дисциплины и формы контроля *

Объем занятий: всего: 4 з.е. 144 ак.ч.	ОФО, в астр. часах
Контактная работа:	144
Лекции/из них практическая подготовка	72
Лабораторных работ/из них практическая подготовка	72ф
Практических занятий/из них практическая подготовка	
Самостоятельная работа	58
Формы контроля	
Экзамен	54
Зачет	
Зачет с оценкой	
Расчетно-графические работы	
Курсовой проект	+
Контрольные работы	

* Дисциплина (модуль) предусматривает применение электронного обучения, дистанционных образовательных технологий

5. Содержание дисциплины (модуля), структурированное по темам (разделам) с указанием количества часов и видов занятий

№	Раздел (тема) дисциплины и краткое содержание	Формируемые компетенции, индикаторы	очная форма			
			Контактная работа обучающихся с преподавателем /из них в форме практической подготовки, часов			Самостоятельная работа, часов
			Лекции	Практические занятия	Лабораторные работы	
Модуль 1: Бэкенд-разработка на Django						
1	Введение в архитектуру современных веб-приложений: Эволюция веб-архитектур (монолит, SPA, микросервисы). Клиент-серверная модель. Жизненный цикл HTTP-запроса в SPA. Роль бэкенда в современном приложении. Настройка окружения: Python, виртуальное окружение, создание Django-проекта. Структура проекта Django. Запуск сервера разработки.	ИД-1пк-2	2		2	4
2	Модели и миграции в Django ORM: Теория реляционных БД: таблицы, связи (1-1, 1-M, M-M), нормализация. Проектирование моделей. Типы полей, связи. Миграции: создание, применение, откат. Настройка PostgreSQL. Работа с данными через Django shell	ИД-3пк-3	2		2	4
3	Продвинутые возможности Django ORM: Сложные QuerySet: агрегация (aggregate, annotate). Оптимизация запросов: select_related, prefetch_related, only, defer. Работа с Q- и F-объектами. Кастомные менеджеры моделей. Транзакции и управление целостностью данных. Сигналы (Signals) и их применение	ИД-3пк-3	2		2	4

4	Представления (Views) и система аутентификации Django: Функциональные (FBV) и классовые представления (CBV). Миксины для проверки прав доступа. Встроенная система аутентификации Django. Кастомизация модели пользователя (AbstractUser, AbstractBaseUser). Система шаблонов Django. Статические файлы.	ИД-4пк-3	2		2	4
5	Введение в Django REST Framework: Принципы REST. Сериализаторы (Serializer, ModelSerializer). ViewSets и Routers для быстрого построения CRUD API. Базовые настройки permission (IsAuthenticated, IsAdminUser). Пагинация (PageNumberPagination, LimitOffsetPagination, CursorPagination). Простая фильтрация через get_queryset().	ИД-2пк-3 ИД-4пк-3	2		2	4
6	Реализация JWT с библиотекой simplejwt (access/refresh токены). Кастомные permissions для сложной бизнес-логики. Throttling (ограничение частоты запросов) для защиты от DoS. ДОПОЛНЕНО: продвинутая настройка JWT: AUTH_TOKEN_CLASSES (кастомные классы токенов), TOKEN_TYPE_CLAIM (идентификация типа токена).	ИД-2пк-3	2		2	4
7	Продвинутые сериализаторы и валидация: Вложенные сериализаторы. Динамические поля. Кастомная валидация на уровне поля и объекта. Переопределение методов create() и update(). Оптимизация запросов N+1 на уровне сериализатора. ДОПОЛНЕНО: связь с кастомными токенами - создание сериализатора для кастомного payload в JWT через AUTH_TOKEN_CLASSES.	ИД-4пк-2	2		2	4
8	Фильтрация, поиск и сортировка в API: Библиотека django-filter для сложной фильтрации. Полнотекстовый поиск (PostgreSQL SearchVector, django.contrib.postgres). Сортировка по множественным полям. Кастомная пагинация с метаданными.	ИД-3пк-3	2		2	4

9	Документирование API: Swagger (OpenAPI), Redoc. Автогенерация схемы с drf-yasg или drf-spectacular. Описание эндпоинтов, параметров и ответов. Интерактивная документация для тестирования API. ДОПОЛНЕНО: документирование JWT-аутентификации, описание кастомных токенов (AUTH_TOKEN_CLASSES, TOKEN_TYPE_CLAIM) в документации.	ИД-2пк-2	2		2	4
10	Кэширование на бэкенде: Стратегии кэширования: на уровне представления, запроса, фрагмента шаблона. Настройка Redis/Memcached. Инвалидация кэша. Кэширование в ORM с django-cachalot. Практическое применение для ускорения API.	ИД-1пк-2	2		2	4
11	Фоновые задачи и Celery: Введение в асинхронные задачи. Установка и настройка Celery + Redis. Создание простых и периодических (celery beat) задач. Обработка длительных операций (отправка email, генерация отчетов). Мониторинг задач с Flower.	ИД-3пк-2	2		2	4
12	Загрузка и хранение файлов: Обработка FileField и ImageField в Django и DRF. Загрузка файлов на клиенте и отправка через API. Интеграция с облачными хранилищами (AWS S3, Yandex Cloud). Генерация превью изображений (django-imagekit).	ИД-3пк-3	2		2	4
13	Уведомления (WebSocket) с Channels: Введение в WebSocket и его отличие от HTTP. Настройка Django Channels. Создание потребителей (Consumers) для чатов или live-уведомлений. Интеграция с ASGI-сервером (Daphne, Uvicorn).	ИД-2пк-3 ИД-3пк-2	2		2	4
14	Производительность и мониторинг: Профилирование запросов с django-debug-toolbar. Логирование: настройка LOGGING, уровни логирования, форматтеры, обработчики. Структурированные логи (JSON). Мониторинг метрик (Prometheus, Grafana) и здоровья приложения. Принципы оптимизации Django-приложений. ДОПОЛНЕНО: мониторинг и аудит JWT-токенов, отслеживание использования AUTH_TOKEN_CLASSES, метрики для JWT-эндпоинтов.	ИД-1пк-2	2		2	4
Модуль 2: Фронтенд-разработка с React						

15	Настройка React-проекта и архитектура SPA: Создание проекта с Vite. Структура проекта. Компонентный дизайн. Настройка роутинга с React Router v6 (BrowserRouter, Routes, Route, Link). Связь с бэкендом: настройка базового URL для API.	ИД-1пк-3 ИД-2пк-2	2		2	4
16	Управление состоянием – основы и Context API: Проблема проброса пропсов. Глобальное состояние. React Context API: создание провайдера и кастомного хука useContext. Паттерны использования. Ограничения Context API.	ИД-1пк-3	2		2	4
17	Управление состоянием с Redux Toolkit (RTK): Принципы Redux: Store, Actions, Reducers. Redux Toolkit: createSlice, configureStore. Интеграция с React: Provider, useSelector, useDispatch. Асинхронная логика с createAsyncThunk.	ИД-1пк-3	2		2	4
18	Работа с API на клиенте (React Query / RTK Query): Проблемы нативного fetch/axios. TanStack Query (React Query): useQuery, useMutation. Стратегии кэширования, фоновая синхронизация, оптимистичные обновления. RTK Query как альтернатива.	ИД-1пк-3	2		2	4
	ИТОГО за 1 семестр		36		36	72
19	Аутентификация и защита роутов на фронтенде: Хранение JWT-токенов (HTTP-only cookies vs localStorage). Создание контекста/слайса Redux для данных пользователя. Защищенные (PrivateRoute) и публичные маршруты. Автоматическое обновление access-токена.	ИД-2пк-3	2		2	3
20	Формы в React – управляемые компоненты и Formik: Сложности управления состоянием форм. Formik: управление значениями, валидация, обработка отправки. Интеграция с Yup. Отображение ошибок и состояние загрузки.	ИД-2пк-3	2		2	3
21	UI-библиотеки и компонентный дизайн: Обзор библиотек: Material-UI (MUI), Ant Design, Chakra UI. Стилизация: CSS-in-JS (Emotion, Styled Components) vs CSS-модули. Создание библиотеки переиспользуемых компонентов. Storybook для разработки и документирования компонентов.	ИД-2пк-3	2		2	3

22	Тестирование фронтенда: Виды тестов: unit, integration, e2e. Jest. React Testing Library (RTL) для тестирования компонентов. Мокирование API-запросов (MSW - Mock Service Worker).	ИД-4пк-3	2		2	3
23	Оптимизация производительности React-приложения: React DevTools. Методы оптимизации: React.memo, useMemo, useCallback. Ленивая загрузка компонентов с React.lazy и Suspense. Код-сплиттинг и анализ бандла.	ИД-1пк-3	2		2	3
Модуль 3: Интеграция, Тестирование и DevOps						
24	Деплой статического фронтенда: Сборка проекта (npm run build). Размещение на статических хостингах: Vercel, Netlify, S3 + CloudFront. Настройка роутинга и перенаправлений. Интеграция с CI/CD.	ИД-4пк-3	2		2	3
25	Тестирование бэкенда (Django, DRF): Пирамида тестирования. unittest и pytest. Тестирование моделей, форм, представлений. Тестирование API с APITestCase. Мокирование внешних сервисов и использование фикстур (factory_boy). ДОПОЛНЕНО: тестирование кастомных AUTH_TOKEN_CLASSES и TOKEN_TYPE_CLAIM.	ИД-2пк-3	2		2	3
26	E2E-тестирование с Cypress / Playwright: Роль end-to-end тестов. Выбор инструмента. Написание сценариев: авторизация, CRUD-операции. Best practices: селекторы, ожидания, отладка. Интеграция в пайплайн CI/CD.	ИД-4пк-3	2		2	3
27	Контейнеризация приложения (Docker): Введение в Docker: образы, контейнеры, Dockerfile. Создание Dockerfile для Django-бэкенда и React-фронтенда. Управление переменными окружения. Оптимизация размера образов (мульти-стадия сборки).	ИД-4пк-3	2		2	3
28	Оркестрация контейнеров (Docker Compose): Введение в Docker Compose. Создание docker-compose.yml: Django, React, PostgreSQL, Redis, Nginx. Настройка сетей и volumes. Запуск многоконтейнерного приложения.	ИД-4пк-2, ИД-4пк-3	2		2	3

29	Веб-сервер и развертывание (Nginx, Gunicorn): Роль веб-сервера (Nginx) и WSGI/ASGI-сервера (Gunicorn, Uvicorn). Настройка Gunicorn (workers, threads). Конфигурация Nginx как reverse proxy, раздача статики/медиа. Настройка SSL/TLS (Let's Encrypt).	ИД-4пк-2, ИД-4пк-3	2		2	3
30	Развертывание на VPS (инфраструктура как код): Выбор и настройка VPS (Ubuntu). Базовый фаервол (UFW). Ручное развертывание: клонирование кода, настройка systemd. Автоматизация с bash-скриптами. Мониторинг базовых метрик сервера	ИД-4пк-2	2		2	3
31	CI/CD – непрерывная интеграция и доставка: Принципы CI/CD. Обзор платформ: GitLab CI, GitHub Actions, Jenkins. Написание пайплайна: линтинг, тесты, сборка Docker-образов. Автоматический деплой на тестовый и production-сервер. Уведомления о статусе сборки.	ИД-4пк-2	2		2	3
32	Инфраструктура как код (IaC) с Ansible: Введение в Ansible: инвентари, плейбуки, роли. Автоматизация настройки VPS: установка Docker, настройка пользователей. Плейбук для развертывания приложения. Комбинирование CI/CD и Ansible	ИД-4пк-3	2		2	3
33	Мониторинг и логирование в production: Централизованный сбор логов (ELK Stack, Loki + Promtail + Grafana). Настройка LOGGING для production: структурированные логи (JSON), ротация. Мониторинг приложения и БД в Grafana. Трекинг ошибок (Sentry). Системы алертинга. ДОПОЛНЕНО: мониторинг JWT-аутентификации в production, логирование событий AUTH_TOKEN_CLASSES, алерты при аномалиях.	ИД-4пк-2	2		2	3
34	Резервное копирование и отказоустойчивость: Стратегии бэкапа базы данных и медиафайлов. Автоматизация бэкапов (скрипты, pg_dump, S3). Понятия High Availability (HA). Балансировщики нагрузки. План аварийного восстановления (DRP).	ИД-1пк-2	2		2	3

35	Итоговая архитектура и best practices: Повторение полной архитектуры приложения (от кода до инфраструктуры). Обзор best practices: безопасность, производительность, поддерживаемость кода. Дальнейшие пути развития: GraphQL, микросервисы, Kubernetes. Защита итогового проекта. ДОПОЛНЕНО: продвинутые практики JWT: когда использовать AUTH_TOKEN_CLASSES, роль TOKEN_TYPE_CLAIM, асимметричное шифрование (RS256) и VERIFYING_KEY, сравнение HS256 и RS256, настройка для микросервисной архитектуры.	2ИД-1пк-	2		2	3

36	Тема 36: Интеграция моделей искусственного интеллекта в полнофункциональные веб-приложения Архитектурные подходы к интеграции ML-моделей – место модели в веб-приложении (синхронный vs асинхронный инференс), схема "бэкенд → ML-сервис", микросервисная vs монолитная интеграция. Выбор и подготовка модели для интеграции – критерии выбора (размер, latency, точность), форматы сохранения (ONNX, TensorFlow SavedModel, pickle), обзор источников (Hugging Face, OpenAI API, YOLO, пользовательские модели). Создание API-обертки на Django REST Framework – проектирование эндпоинтов /predict, сериализаторы для входных/выходных данных, валидация входных параметров, обработка ошибок, версионирование API. Асинхронная обработка запросов с Celery – проблема блокировки при длительном инференсе, архитектура: DRF → Celery task → Redis/RabbitMQ → worker → результат, получение статуса задачи через polling или WebSocket. Кэширование результатов инференса – стратегии: exact match кэш (Redis), similarity cache (FAISS для эмбеддингов), TTL и инвалидация кэша, влияние на latency и стоимость вычислений. Интеграция с React-фронтендом – создание сервиса для запросов к API, обработка состояний загрузки/ошибки/результата, отображение прогресса асинхронных задач, пример UI для классификации или генерации текста. Мониторинг и логирование ML-компонента – отслеживание времени инференса, logging входных/выходных данных (с учетом конфиденциальности), алерты при падении модели или превышении latency, метрики качества предсказаний. Документирование ML-API – спецификация OpenAPI для эндпоинтов /predict, описание форматов входных/выходных данных, примеры запросов, требования к модели.	ИД-1пк-2, ИД-2пк-2, ИД-3пк-2 ИД-1пк-8 ИД-2пк-8 ИД-3пк-8	2		2	3
	ИТОГО за 2 семестр		36		36	54
	ИТОГО		72		72	126

6. Фонд оценочных средств по дисциплине (модулю)

Фонд оценочных средств (ФОС) по дисциплине (модулю) базируется на перечне осваиваемых компетенций с указанием индикаторов. ФОС обеспечивает объективный контроль достижения запланированных результатов обучения. ФОС включает в себя:

- описание показателей и критериев оценивания компетенций на различных этапах их формирования, описание шкал оценивания;

- методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций (включаются в методические указания по тем видам работ, которые предусмотрены учебным планом и предусматривают оценку сформированности компетенций);

- типовые оценочные средства, необходимые для оценки знаний, умений и уровня сформированности компетенций.

ФОС является приложением к данной программе дисциплины.

7. Методические указания для обучающихся по освоению дисциплины

Приступая к работе, каждый студент должен принимать во внимание следующие положения.

Дисциплина построена по тематическому принципу, каждая тема представляет собой логически завершенный раздел.

Практические занятия проводятся с целью закрепления усвоенной информации, приобретения навыков ее применения при решении практических задач в соответствующей предметной области.

Самостоятельная работа студентов направлена на самостоятельное изучение дополнительного материала, подготовку к лабораторным занятиям, а также выполнения всех видов самостоятельной работы.

Для успешного освоения дисциплины, необходимо выполнить все виды самостоятельной работы, используя рекомендуемые источники информации.

8. Учебно-методическое и информационное обеспечение дисциплины

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

8.1.1. Перечень основной литературы:

1. Документация Django (djbook.ru) - Официальная документация Django на русском языке, версия 4.1/4.2. URL: https://djbook.ru
2. Документация React (ru.reactjs.org) - Официальная документация React на русском языке. URL: https://ru.react.js.org
3. Документация Django REST Framework (русский перевод) - Перевод официальной документации DRF. URL: https://github.com/ilyachch/django-rest-framework-rusdoc
4. Документация PostgreSQL (русский перевод) - Официальная документация PostgreSQL на русском языке (версия 15 и выше). Доступна на сайте: https://postgrespro.ru/docs/postgresql
5. Документация Docker - Официальная документация Docker на английском языке. URL: https://docs.docker.com
6. Документация Docker Compose - Официальная документация Docker Compose. URL: https://docs.docker.com/compose
7. Документация GitHub Actions - Официальная документация GitHub Actions. URL: https://docs.github.com/actions
8. Документация Ansible - Документация на официальном сайте. URL: https://docs.ansible.com

9. Документация Celery - Официальная документация Celery. URL: https://docs.celeryq.dev
10. Документация Redis - Официальная документация Redis. URL: https://redis.io/documentation
11. Документация Prometheus - Официальная документация Prometheus. URL: https://prometheus.io/docs
12. Документация Grafana - Официальная документация Grafana. URL: https://grafana.com/docs

8.1.2. Перечень дополнительной литературы:

1. Django 4 by Example (4th Edition) - Antonio Melé. Packt Publishing, 2022. Примеры разработки веб-приложений на Django (доступно в электронном виде).
2. The Road to React - Robin Wieruch. Документация по React с практическими примерами. URL: https://www.roadtoreact.com
3. Django REST Framework official documentation - Официальная документация DRF на английском языке. URL: https://www.django-rest-framework.org
4. Python Celery: руководство - Введение в Celery на русском языке. URL: https://pythonturbo.ru/python-celery
5. Docker Deep Dive - Nigel Poulton. Книга по контейнеризации и Docker.
6. Ansible для администраторов - Документация на русском языке в составе Astra Linux. URL: https://wiki.astralinux.ru/pages/viewpage.action?pageId=153487981
7. Официальная документация Cypress - Документация по E2E-тестированию. URL: https://docs.cypress.io
8. React Testing Library - Документация по тестированию React-компонентов. URL: https://testing-library.com/docs/react-testing-library/intro
9. Formik documentation - Документация по работе с формами в React. URL: https://formik.org/docs/overview
10. Yup validation library - Документация по валидации схем. URL: https://github.com/jquense/yup

8.2. Перечень учебно-методического обеспечения самостоятельной работы обучающихся по дисциплине

8.3. Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины

Официальная документация (доступна на территории РФ)

1. Django на русском - https://djbook.ru
2. React на русском - https://ru.reactjs.org
3. PostgreSQL на русском - https://postgrespro.ru/docs/postgresql
4. Docker официальный - https://docs.docker.com
5. GitHub Actions - https://docs.github.com/actions
6. Celery официальный - https://docs.celeryq.dev
7. Redis официальный - https://redis.io/documentation
8. Prometheus - https://prometheus.io/docs
9. Grafana - https://grafana.com/docs
10. Ansible - https://docs.ansible.com
11. Cypress - https://docs.cypress.io
12. React Testing Library - https://testing-library.com/docs/react-testing-library
13. Formik - https://formik.org/docs

14. Redux Toolkit - https://redux-toolkit.js.org
15. React Query (TanStack Query) - https://tanstack.com/query/latest

Русскоязычные сообщества и форумы

1. Stack Overflow на русском - https://ru.stackoverflow.com
2. Habr - https://habr.com
3. Django на русском (Telegram-сообщество)
4. Python Telegram сообщество

Интерактивные обучающие платформы

1. Яндекс.Практикум - https://practicum.yandex.ru
2. Stepik - https://stepik.org
3. Codecademy - https://www.codecademy.com
4. LeetCode - https://leetcode.com

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

При чтении лекций используется компьютерная техника, демонстрации презентационных мультимедийных материалов. На семинарских и практических занятиях студенты представляют презентации, подготовленные ими в часы самостоятельной работы.

Информационные справочные системы:

1. GitHub — https://github.com
2. GitLab — https://gitlab.com
3. Docker Hub — https://hub.docker.com
4. PyPI — https://pypi.org
5. npm — https://www.npmjs.com
6. Stack Overflow — https://stackoverflow.com
7. MDN Web Docs — https://developer.mozilla.org
8. DevOps.stackexchange — https://devops.stackexchange.com

Информационно-справочные и информационно-правовые системы, используемые при изучении дисциплины:

Программное обеспечение:

1. Python 3.11+ (Open Source, PSF License) - язык программирования для бэкенда
2. Django 4.2+ (Open Source, BSD License) - веб-фреймворк
3. Django REST Framework (Open Source, BSD License) - создание API
4. PostgreSQL 15+ (Open Source, PostgreSQL License) - реляционная база данных
5. Redis (Open Source, BSD License) - кэширование, брокер задач
6. Celery (Open Source, BSD License) - фоновые задачи
7. Node.js 20+ (Open Source, MIT License) - среда выполнения JavaScript
8. React (Open Source, MIT License) - библиотека для фронтенда
9. Vite (Open Source, MIT License) - сборщик проектов
10. Redux Toolkit (Open Source, MIT License) - управление состоянием
11. Docker (Freemium, Docker Engine - Apache 2.0) - контейнеризация
12. Docker Compose (Open Source, Apache 2.0) - оркестрация контейнеров
13. Nginx (Open Source, BSD License) - веб-сервер, reverse proxy
14. Gunicorn (Open Source, MIT License) - WSGI-сервер для Django
15. Git (Open Source, GPL License) - система контроля версий
16. GitHub Actions (Freemium) - CI/CD платформа
17. Ansible (Open Source, GPL License) - инфраструктура как код

18. Prometheus (Open Source, Apache 2.0) - сбор метрик
19. Grafana (Open Source, AGPL License) - визуализация метрик
20. Loki (Open Source, AGPL License) - централизованный сбор логов
21. Promtail (Open Source, AGPL License) - агент для сбора логов
22. Cypress (Open Source, MIT License) - E2E-тестирование
23. pytest (Open Source, MIT License) - тестирование Python
24. Jest (Open Source, MIT License) - тестирование JavaScript/React
25. React Testing Library (Open Source, MIT License) - тестирование React-компонентов
26. Postman (Freemium) - тестирование API
27. VS Code (Freeware, Microsoft) - редактор кода
28. PyCharm Community (Free, Apache 2.0) - IDE для Python
29. SQLite Studio (Open Source, GPL) - визуальное управление SQLite
30. PgAdmin (Open Source, PostgreSQL License) - управление PostgreSQL
31. DBeaver (Open Source, Apache 2.0) - универсальный клиент баз данных

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

Лекционные занятия	Учебная аудитория для проведения учебных занятий, оснащенная мультимедийным оборудованием и техническими средствами обучения.
Лабораторные работы	Учебная аудитория для проведения учебных занятий, оснащенная мультимедийным оборудованием и техническими средствами обучения.
Самостоятельная работа	Помещение для самостоятельной работы обучающихся оснащенное компьютерной техникой с возможностью подключения к сети "Интернет" и возможностью доступа к электронной информационно-образовательной среде университета

11. Особенности освоения дисциплины (модуля) лицами с ограниченными возможностями здоровья

Обучающимся с ограниченными возможностями здоровья предоставляются специальные учебники, учебные пособия и дидактические материалы, специальные технические средства обучения коллективного и индивидуального пользования, услуги ассистента (помощника), оказывающего обучающимся необходимую техническую помощь, а также услуги сурдопереводчиков и тифлосурдопереводчиков.

Освоение дисциплины (модуля) обучающимися с ограниченными возможностями здоровья может быть организовано совместно с другими обучающимися, а также в отдельных группах.

Освоение дисциплины (модуля) обучающимися с ограниченными возможностями здоровья осуществляется с учетом особенностей психофизического развития, индивидуальных возможностей и состояния здоровья.

В целях доступности получения высшего образования по образовательной программе лицами с ограниченными возможностями здоровья при освоении дисциплины (модуля) обеспечивается:

1) для лиц с ограниченными возможностями здоровья по зрению:

- присутствие ассистента, оказывающий студенту необходимую техническую помощь с учетом индивидуальных особенностей (помогает занять рабочее место, передвигаться, прочитать и оформить задание, в том числе, записывая под диктовку),

- письменные задания, а также инструкции о порядке их выполнения оформляются увеличенным шрифтом,

- специальные учебники, учебные пособия и дидактические материалы (имеющие крупный шрифт или аудиофайлы),

- индивидуальное равномерное освещение не менее 300 люкс,

- при необходимости студенту для выполнения задания предоставляется увеличивающее устройство;

2) для лиц с ограниченными возможностями здоровья по слуху:

- присутствие ассистента, оказывающий студенту необходимую техническую помощь с учетом индивидуальных особенностей (помогает занять рабочее место, передвигаться, прочитать и оформить задание, в том числе, записывая под диктовку),

- обеспечивается наличие звукоусиливающей аппаратуры коллективного пользования, при необходимости обучающемуся предоставляется звукоусиливающая аппаратура индивидуального пользования;

- обеспечивается надлежащими звуковыми средствами воспроизведения информации;

3) для лиц с ограниченными возможностями здоровья, имеющих нарушения опорно-двигательного аппарата (в том числе с тяжелыми нарушениями двигательных функций верхних конечностей или отсутствием верхних конечностей):

- письменные задания выполняются на компьютере со специализированным программным обеспечением или надиктовываются ассистенту;

- по желанию студента задания могут выполняться в устной форме.

12. Особенности реализации дисциплины с применением дистанционных образовательных технологий и электронного обучения

Согласно части 1 статьи 16 Федерального закона от 29 декабря 2012 г. № 273-ФЗ «Об образовании в Российской Федерации» под *электронным обучением* понимается организация образовательной деятельности с применением содержащейся в базах данных и используемой при реализации образовательных программ информации и обеспечивающих ее обработку информационных технологий, технических средств, а также информационно-телекоммуникационных сетей, обеспечивающих передачу по линиям связи указанной информации, взаимодействие обучающихся и педагогических работников. Под *дистанционными образовательными технологиями* понимаются образовательные технологии, реализуемые в основном с применением информационно-телекоммуникационных сетей при опосредованном (на расстоянии) взаимодействии обучающихся и педагогических работников.

Реализация дисциплины может быть осуществлена с применением дистанционных образовательных технологий и электронного обучения полностью или частично. Компоненты УМК дисциплины (рабочая программа дисциплины, оценочные и методические материалы, формы аттестации), реализуемой с применением дистанционных образовательных технологий и электронного обучения, содержат указание на их использование.

При организации образовательной деятельности с применением дистанционных образовательных технологий и электронного обучения могут предусматриваться асинхронный и синхронный способы осуществления взаимодействия участников образовательных отношений посредством информационно-телекоммуникационной сети «Интернет».

При применении дистанционных образовательных технологий и электронного обучения в расписании по дисциплине указываются: способы осуществления взаимодействия участников образовательных отношений посредством информационно-телекоммуникационной сети «Интернет» (ВКС-видеоконференцсвязь, ЭТ – электронное тестирование); ссылки на электронную информационно-образовательную среду СКФУ, на образовательные платформы и ресурсы иных организаций, к которым предоставляется открытый доступ через информационно-телекоммуникационную сеть «Интернет»; для

синхронного обучения - время проведения онлайн-занятий и преподаватели; для асинхронного обучения - авторы онлайн-курсов.

При организации промежуточной аттестации с применением дистанционных образовательных технологий и электронного обучения используются Методические рекомендации по применению технических средств, обеспечивающих объективность результатов при проведении промежуточной и государственной итоговой аттестации по образовательным программам высшего образования - программам бакалавриата, программам специалитета и программам магистратуры с применением дистанционных образовательных технологий (Письмо Минобрнаки России от 07.12.2020 г. № МН-19/1573-АН "О направлении методических рекомендаций").

Реализация дисциплины с применением электронного обучения и дистанционных образовательных технологий осуществляется с использованием электронной информационно-образовательной среды СКФУ, к которой обеспечен доступ обучающихся через информационно-телекоммуникационную сеть «Интернет», или с использованием ресурсов иных организаций, в том числе платформ, предоставляющих сервисы для проведения видеоконференций, онлайн-встреч и дистанционного обучения (МТС-Линк), а также с использованием возможностей социальных сетей для осуществления коммуникации обучающихся и преподавателей.

Учебно-методическое обеспечение дисциплины, реализуемой с применением электронного обучения и дистанционных образовательных технологий, включает представленные в электронном виде рабочую программу, учебно-методические пособия или курс лекций, методические указания к выполнению различных видов учебной деятельности обучающихся, предусмотренных дисциплиной, и прочие учебно-методические материалы, размещенные в информационно-образовательной среде СКФУ.