Компонент ОПОП

09.03.01 Информатика и вычислительная техника

наименование ОПОП

направленность (профиль) «Технологии разработки программного обеспечения»

Б1.В.ДВ.01.01

шифр дисциплины

РАБОЧАЯ ПРОГРАММА

Дисциплины (модуля) **Разработка приложений на платформе node.js**

Разработчик (и):

Ляш О.И.

ФИО

зав.кафедрой

должность

канд.пед.наук, доцент

ученая степень, звание

Утверждено на заседании кафедры информационных технологий

наименование кафедры

протокол № 6 от 24.02.2026

Заведующий кафедрой ИТ

подпись

Ляш О.И.

ФИО

Мурманск
2026

Пояснительная записка

Объем дисциплины 4 з.е.

1. Результаты обучения по дисциплине (модулю), соотнесенные с индикаторами достижения компетенций, установленными образовательной программой

Компетенции	Индикаторы достижения компетенций	Результаты обучения по дисциплине (модулю)
ПК-1 Способен разрабатывать технические спецификации на программные компоненты и их взаимодействие	ИД-1$_{ПК-1}$ Анализирует существующую документацию (техническое задание, спецификации) и требования к программному обеспечению для выделения перечня подлежащих разработке программных компонентов. ИД-2$_{ПК-1}$ Разрабатывает, согласовывает и утверждает технические спецификации на программные компоненты, описывающие их структуру, функции, логику работы и атрибуты качества. ИД-3$_{ПК-1}$ Проектирует интерфейсы взаимодействия программных компонентов (API) и протоколы обмена данными между ними, фиксируя результаты в спецификации. ИД-4$_{ПК-1}$ Оформляет техническую документацию на компоненты и их взаимодействие в соответствии с требованиями стандартов.	**Знать:** – Принципы работы платформы Node.js: событийно-ориентированная архитектура, цикл событий (event loop), асинхронное программирование (callbacks, promises, async/await). – Типовые архитектурные паттерны серверных приложений (MVC, слоистая архитектура, микросервисы) и способы взаимодействия компонентов (REST, GraphQL, WebSocket). – Методы проектирования баз данных (реляционные и NoSQL) и стандарты оформления технической документации (OpenAPI, Swagger, ЕСПД). – Основные метрики качества ПО (производительность, масштабируемость, безопасность) и подходы к анализу рисков в среде Node.js. **Уметь:** – Анализировать техническое задание и выделять перечень программных компонентов для реализации на Node.js. – Проектировать интерфейсы взаимодействия компонентов (REST API, GraphQL-схемы) и фиксировать их в спецификациях с использованием OpenAPI/Swagger. – Выбирать архитектуру приложения (монолит, микросервисы) и выполнять декомпозицию на модули с учётом требований к качеству. – Разрабатывать логическую и физическую схемы баз данных, интегрируя их с бизнес-логикой на Node.js, а также создавать проектную документацию для последующего кодирования и развёртывания. **Владеть:** – Навыками разработки серверных приложений на
ПК-2 Способен проектировать компьютерное программное обеспечение	ИД-1$_{ПК-2}$ Выбирает архитектуру программного обеспечения и декомпозирует его на модули и компоненты. ИД-2$_{ПК-2}$ Применяет объектно-ориентированные или иные методологии для разработки моделей предметной области, структур данных и бизнес-логики будущего ПО. ИД-3$_{ПК-2}$ Разрабатывает логическую и физическую схемы баз данных, а также осуществляет проектирование пользовательских интерфейсов с учетом требований эргономики. ИД-4$_{ПК-2}$ Создает проектную документацию на программное обеспечение, позволяющую	

	осуществить кодирование и дальнейшее развертывание продукта. ИД-5$_{ПК-2}$ Оценивает соответствие разработанного проекта исходным требованиям и техническому заданию, а также анализирует возможные риски реализации.	Node.js с использованием фреймворков (Express.js, NestJS, Fastify) и встроенных модулей (http, fs, stream). – Методами асинхронного программирования и управления потоком событий (промисы, async/await, паттерны «Pub/Sub»). – Приёмами написания технической документации (спецификации API в Swagger, описание компонентов, инструкции по сборке). – Техниками модульной декомпозиции, проектирования архитектуры и оценки соответствия проекта требованиям (юнит-тестирование, профилирование производительности).

2. Содержание дисциплины (модуля)

Тема 1. Основы серверной разработки на платформе Node.js. *Установка и настройка окружения. Модульная система (CommonJS, ES Modules). Цикл событий (event loop) и асинхронное программирование (callbacks, promises, async/await). Создание HTTP-сервера с использованием встроенного модуля http. Обзор фреймворков Express.js, Fastify, NestJS. Маршрутизация, обработка запросов и ответов. Middleware. Работа со статическими файлами. Обработка ошибок и логирование. Основы работы с файловой системой (fs, stream). Управление процессами и кластеризация.*

Тема 2. Проектирование интерфейсов и архитектуры приложений. *Разработка технических спецификаций на программные компоненты. Проектирование REST API: ресурсы, методы, коды ответов. Документирование API с помощью OpenAPI (Swagger). Согласование и утверждение спецификаций. Архитектурные паттерны: MVC, слоистая архитектура, внедрение зависимостей. Микросервисный подход в Node.js (обмен через HTTP, gRPC, сообщения). Проектирование интерфейсов взаимодействия компонентов (API Gateway, очереди). Протоколы обмена: JSON, Protobuf. Оформление технической документации в соответствии со стандартами.*

Тема 3. Работа с базами данных и бизнес-логика. *Проектирование логической и физической схем баз данных. Реляционные БД: PostgreSQL, MySQL, работа через ORM (Sequelize, TypeORM, Prisma). NoSQL базы данных: MongoDB, Mongoose. Выполнение миграций и заполнение данными. Разработка бизнес-логики приложения: сервисы, репозитории, DTO. Валидация данных и обработка исключений. Аутентификация и авторизация (JWT, сессии, роли). Тестирование компонентов (unit-тесты, интеграционные тесты с использованием Jest, Supertest). Оценка соответствия проекта исходным требованиям и техническому заданию.*

Тема 4. Развертывание, оптимизация и анализ рисков. *Контейнеризация приложений с Docker. Создание Dockerfile, docker-compose для Node.js. Непрерывная интеграция и доставка (CI/CD). Развертывание на облачных платформах (Heroku, Render, Yandex Cloud, AWS). Управление процессами с помощью PM2. Профилирование производительности (clinic.js, Node.js Inspector). Обеспечение безопасности: защита от распространённых уязвимостей (инъекции, XSS, CSRF, DoS). Анализ рисков производительности, масштабируемости и отказоустойчивости. Создание проектной документации для кодирования и развертывания. Финальная сборка и демонстрация готового приложения.*

3. Перечень учебно-методического обеспечения дисциплины (модуля)

– мультимедийные презентационные материалы по дисциплине (модулю) представлены в электронном курсе в ЭИОС МАУ;

– методические указания к выполнению лабораторных/практических/контрольных работ (выбрать) представлены в электронном курсе в ЭИОС МАУ;

– методические материалы для обучающихся по освоению дисциплины (модуля) представлены на официальном сайте МАУ в разделе «Информация по образовательным программам, в том числе адаптированным».

4. Фонд оценочных средств по дисциплине (модулю)

Является отдельным компонентом образовательной программы, разработан в форме отдельного документа, включает в себя:

– перечень компетенций с указанием этапов их формирования в процессе освоения дисциплины (модуля);

– задания текущего контроля;

– задания промежуточной аттестации;

– задания внутренней оценки качества образования.

5. Перечень основной и дополнительной учебной литературы (печатные издания, электронные учебные издания и (или) ресурсы электронно-библиотечных систем)

Основная литература:

1. Заяц, А. М. Проектирование и разработка WEB-приложений. Введение в frontend и backend разработку на JavaScript и node.js : учебное пособие для вузов / А. М. Заяц, Н. П. Васильев. — 4-е изд., стер. — Санкт-Петербург : Лань, 2025. — 120 с. — ISBN 978-5-507-51011-5. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/499418.

2. Баланов, А. Н. Бэкенд-разработка веб-приложений: архитектура, проектирование и управление проектами : учебное пособие для вузов / А. Н. Баланов. — 2-е изд., стер. — Санкт-Петербург : Лань, 2025. — 312 с. — ISBN 978-5-507-52472-3. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/451820.

3. Солодушкин, С. И. Разработка программных комплексов на языке JavaScript : учебное пособие / С. И. Солодушкин, И. Ф. Юманова ; под общ. ред. В. Г. Пименова ; Уральский федеральный университет им. первого Президента России Б. Н. Ельцина. – Екатеринбург : Издательство Уральского университета, 2020. – 135 с. : схем., табл. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=699140.

Дополнительная литература:

4. Никулова, Г. А. Web-технологии : введение в программирование на JavaScript : защита контента средствами JS и CSS : учебно-методическое пособие : [16+] / Г. А. Никулова, А. С. Терлецкий ; Липецкий государственный педагогический университет им. П. П. Семенова-Тян-Шанского. – Липецк : Липецкий государственный педагогический университет им. П.П. Семенова-Тян-Шанского, 2023. – 77 с. : ил., табл. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=714543.

5. Диков, А. В. Курс программирования на JavaScript : учебное пособие / А. В. Диков. – Москва : Директ-Медиа, 2024. – 268 с. : ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=713572.

6. Брылёва, А. А. Программные средства создания интернет-приложений : учебное пособие / А. А. Брылёва. – Минск : РИПО, 2022. – 485 с. : ил., табл., схем. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=711495.

6. Профессиональные базы данных и информационные справочные системы

1) Официальный сайт программной платформы node.js — URL: https://nodejs.org/
2) Официальный документация программной платформы node.js – URL: https://nodejs.org/docs/latest/api/
3) Официальный сайт программной платформы Deno – URL: https://deno.com
4) Официальный сайт программной платформы Go (Golang) – URL: https://go.dev

7. Перечень лицензионного и свободно распространяемого программного обеспечения, в том числе отечественного производства

1) Программная платформа: node.js / Deno / Go
2) Интегрированная среда разработки: OpenIDE
3) Редактор кода: Geany / VS Codium / VS Code
4) Дистрибутив операционной системы Linux: Debian / ALT Linux / Astra Linux / RedOS
5) Инструменты управления виртуальными машинами: Virt-manager / VirtualBox
6) Офисный пакет: LibreOffice / Р7-Офис / МойОфис
7) Браузер: Chromium / Firefox / Яндекс Браузер

8. Обеспечение освоения дисциплины лиц с инвалидностью и ОВЗ

Обучающиеся из числа инвалидов и лиц с ОВЗ обеспечиваются печатными и (или) электронными образовательными ресурсами в формах, адаптированных к ограничениям их здоровья.

9. Материально-техническое обеспечение дисциплины (модуля) представлено в приложении к ОПОП «Материально-технические условия реализации образовательной программы» и включает:

- учебные аудитории для проведения учебных занятий, предусмотренных программой бакалавриата, оснащенные оборудованием и техническими средствами обучения;
- помещения для самостоятельной работы обучающихся, оснащенные компьютерной техникой с возможностью подключения к сети «Интернет» и обеспечением доступа к электронной информационно-образовательной среде МАУ;

Допускается замена оборудования его виртуальными аналогами.

10. Распределение трудоемкости по видам учебной деятельности

Таблица 1 - Распределение трудоемкости

Вид учебной деятельности	Распределение трудоемкости дисциплины (модуля) по формам обучения			
	Очная		Заочная	
	Семестр	Всего часов	Семестр	Всего часов
	6		2	
Лекции	18	18	4	4
Практические занятия				
Лабораторные работы	36	36	6	6
Самостоятельная работа	54	54	125	125
Подготовка к промежуточной аттестации				
Всего часов по дисциплине / из них в форме практической подготовки	**144**	**144**	**144**	**144**
Формы промежуточной аттестации и текущего контроля				
Экзамен	1	1	1	1
Зачёт с оценкой	-	-	-	-
Курсовая работа (проект)	-	-	-	-
Количество расчетно-графических работ	1	1	1	1
Количество контрольных работ	-	-	-	-
Количество рефератов	-	-	-	-
Количество эссе	-	-	-	-

Перечень лабораторных работ по формам обучения

№ п\п	Темы лабораторных работ
1	**2**
	Очная форма
	Модуль 1. Основы серверной разработки на платформе Node.js
1.	Установка Node.js и npm. Создание первого HTTP-сервера на встроенном модуле http.
2.	Освоение асинхронного программирования: колбэки, промисы, async/await. Наблюдение за циклом событий.
3.	Разработка сервера на Express.js. Маршрутизация, обработка различных HTTP-методов.
4.	Создание и подключение middleware (логгирование, парсинг тела запроса, раздача статики). Обработка ошибок.
5.	Работа с файловой системой: чтение, запись, использование потоков (streams). Запуск с nodemon.
	Модуль 2. Проектирование интерфейсов и архитектуры приложений
6.	Проектирование REST API. Разработка спецификации ресурсов, методов, кодов ответов.
7.	Документирование API с помощью OpenAPI (Swagger). Генерация интерактивной документации.
8.	Разработка технических спецификаций на программные компоненты (контроллеры, сервисы, репозитории).

9.	Реализация архитектуры MVC на Node.js. Разделение логики, внедрение зависимостей.
10.	Создание микросервисного взаимодействия: обмен данными между двумя сервисами через HTTP/JSON.
11.	Оформление технической документации на компоненты и интерфейсы в соответствии со стандартами (Markdown, UML-диаграммы).
	Модуль 3. Работа с базами данных и бизнес-логика
12.	Проектирование логической и физической схем реляционной БД (PostgreSQL). Написание миграций.
13.	Подключение ORM (Sequelize или TypeORM). Создание моделей, связей, выполнение запросов.
14.	Работа с NoSQL БД на примере MongoDB и Mongoose (схемы, документы, агрегации).
15.	Реализация бизнес-логики: сервисы, репозитории, DTO. Валидация входных данных (Joi, class-validator).
16.	Внедрение аутентификации и авторизации: JWT, хеширование паролей (bcrypt), middleware проверки ролей.
17.	Юнит-тестирование сервисов (Jest) и интеграционное тестирование API (Supertest).
	Модуль 3. Работа с базами данных и бизнес-логика
18.	Контейнеризация приложения: создание Dockerfile, настройка docker-compose для Node.js + база данных.
19.	Организация CI/CD (например, GitHub Actions): автоматическая сборка, тестирование, деплой.
20.	Нагрузочное тестирование (k6 или Artillery). Профилирование производительности с clinic.js.
21.	Анализ рисков: производительность event loop, масштабирование (PM2, кластеризация), утечки памяти.
22.	Обеспечение безопасности: защита от инъекций, XSS, CSRF, rate limiting, Helmet.
23.	Финальная сборка проекта. Развертывание на облачной платформе. Создание проектной документации для кодирования и развертывания. Демонстрация работающего приложения.
	Заочная форма
	Модуль 1. Основы серверной разработки на платформе Node.js
1.	Разработка HTTP-сервера на Express.js: маршрутизация, middleware, обработка запросов и ошибок.
2.	Асинхронное программирование (promises, async/await) и работа с файловой системой (чтение/запись, стримы).
	Модуль 2. Проектирование интерфейсов и архитектуры приложений
3.	Проектирование REST API. Документирование спецификации в OpenAPI (Swagger). Создание технической документации на компоненты.
4.	Реализация архитектуры MVC на Node.js. Внедрение микросервисного взаимодействия (HTTP/JSON).
	Модуль 3. Работа с базами данных и бизнес-логика
5.	Проектирование схемы реляционной БД (PostgreSQL). Подключение ORM (TypeORM/Sequelize). Создание моделей и миграций.
6.	Реализация бизнес-логики (сервисы, DTO, валидация). Аутентификация и авторизация (JWT, bcrypt). Юнит-тестирование (Jest).
	Модуль 4. Развертывание, оптимизация и анализ рисков
7.	Контейнеризация приложения: Dockerfile, docker-compose (Node.js + БД). Организация CI/CD (GitHub Actions).
8.	Нагрузочное тестирование (k6) и профилирование производительности (clinic.js). Анализ рисков. Развертывание на облачной платформе. Демонстрация готового приложения.