Аннотация рабочей программы дисциплины

Б1.О.34 «Микросервисная Архитектура»

Курс 4 Семестр 7 Количество з.е. 2
Объем трудоемкости: 2 зачетных единиц (72 ч., из них – 54,2 час. аудиторной нагрузки: лекционных 16 ч., лабораторных работ - 34 ч., 17,8 часов самостоятельной работы, 4 часа КСР, 0,2 часа ИКР.), форма контроля – зачет.

Цель дисциплины: Цели дисциплины: формирование у студентов общих компетенций, формирующих способность решать задачи профессиональной деятельности в области проектирования, развертывания, эксплуатации и администрирования надежных, масштабируемых и эволюционирующих распределенных систем, основанных на микросервисной архитектуре.

Задачи дисциплины:

- **Привить студентам понимание эволюции архитектуры:** От монолитов к сервисно-ориентированной архитектуре (SOA) и, наконец, к микросервисам.
- **Умение проводить сравнительный анализ:** Умение объективно оценивать преимущества и недостатки микросервисов по сравнению с монолитной архитектурой. Понимание, когда микросервисы оправданы, а когда нет (избегание "архитектурного хайпа").
- **Изучение принципов:** Освоение ключевых принципов, таких как слабая связанность, сильная связность сервисов, независимость развертывания и автономность команд.

Приобретение практических навыков проектирования и разработки:

- Декомпозиция: Научить правильно разбивать бизнес-домен на отдельные, слабосвязанные сервисы на основе методологий типа Domain-Driven Design (DDD), определения ограниченных контекстов (Bounded Context) и выделения сервисов вокруг бизнес-возможностей.
- Проектирование API и взаимодействия: Научить проектировать эффективные API (REST, gRPC, GraphQL) и организовывать взаимодействие между сервисами (синхронное vs асинхронное, с использованием messaging брокеров как RabbitMQ или Kafka).
- Управление данными: Понимание принципов децентрализованного управления данными, когда каждый сервис владеет своей собственной базой данных и общается с другими через четко определенные API.
- Независимое развертывание: **Научить настраивать CI/CD (Continuous Integration / Continuous Deployment) пайплайны для независимого развертывания отдельных сервисов.**
- Контейнеризация и оркестрация: **Дать понимание технологий контейнеризации (Docker) и оркестрации (Kubernetes), которые являются фундаментом для управления жизненным циклом микросервисов.**

Место дисциплины в структуре ООП ВО:

Дисциплина «Микросервисная архитектура» относится к базовой части блока Б1 Дисциплины (модули).

Для изучения дисциплины студент должен владеть знаниями, умениями и навыками по дисциплинам: Дискретная математика, Конструирование алгоритмов и структур данных, Организация вычислительных систем, Алгоритмы и структуры данных, Теория вероятностей и математическая статистика, с которыми дисциплина связана логически и содержательно-методически.

Дисциплина в значительной степени взаимодействует для формирования компетенций с дисциплинами: Объектно-ориентированное программирование и шаблоны проектирования

Операционные системы; Web-разработка; Компьютерные сети; DevOps; Анализ и проектирование информационных систем; Разработка мобильных приложений.

Результаты обучения (знания, умения, опыт, компетенции):

ОПК-3 ***Способен к разработке алгоритмических и программных решений в области системного и прикладного программирования, математических, информационных и имитационных моделей, созданию информационных ресурсов глобальных сетей, образовательного контента, прикладных баз данных, тестов и средств тестирования систем и средств на соответствие стандартам и исходным требованиям***

ОПК-3.1 Аргументировано применяет методы проектирования, разработки и реализации программных продуктов и программных комплексов в различных областях человеческой деятельности

Знает методы проектирования, разработки и реализации программных продуктов и программных комплексов в различных областях человеческой деятельности.

Умеет: спроектировать архитектуру системы, в зависимости от поставленной задачи. **Владеет** средствами разработки и реализации микросервисных архитектур.

ОПК-3.2 Использует инструментальные, программные и аппаратные средства измерений для оценки качества программного обеспечения

Знает инструментальные, программные и аппаратные средства измерений для оценки качества программного обеспечения

Умеет: использовать их на практике

Владеет навыками оценки качества программного обеспечения.

ОПК-4 ***Способен участвовать в разработке технической документации программных продуктов и комплексов с использованием стандартов, норм и правил, а также в управлении проектами создания информационных систем на стадиях жизненного цикла***

ОПК-4.4 Имеет практический опыт анализа и интерпретации информационных систем

Знает *правила* анализа и интерпретации информационных систем

Умеет аргументировано применять методы проектирования, разработки и реализации программных продуктов и программных комплексов в различных областях человеческой деятельности. **Владеет** навыками использования инструментальных, программных и аппаратных средств измерений для оценки качества программного обеспечения

ОПК-5 ***Способен инсталлировать и сопровождать программное обеспечение информационных систем и баз данных, в том числе отечественного происхождения, с учетом информационной безопасности***

ОПК-5.3 Имеет практические навыки установки и инсталляции программных комплексов, применения основ сетевых технологий.

Знает принципы разработки и функционирования ИС и БД.
Умеет документировать и сопровождать ИС и БД.
Владеет практическими навыками установки и инсталляции программных комплексов, применения основ сетевых технологий.

ОПК-6 ***Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности***

ОПК-6.1 Аргументировано применяет современные информационные технологии, в том числе отечественные, при создании программных продуктов и программных комплексов различного назначения.

Знает: принципы микросервисной архитектуры (MSA): слабая связанность, сильная когерентность, независимое развертывание. • Ключевые технологические паттерны MSA: API Gateway, Service Discovery, Circuit Breaker, внешний конфигурационный сервер.• Современные протоколы взаимодействия сервисов (REST/gRPC, асинхронная коммуникация через брокеры сообщений - RabbitMQ, Kafka).

Умеет: Выбирать технологический стек (язык, фреймворк, БД) для каждого сервиса исходя из его задач, а не из единого стандарта.• Реализовывать сервисы на современных фреймворках (например, Spring Boot, FastAPI,) и контейнеризировать их с помощью Docker.• Настраивать межсервисное взаимодействие, выбирая тип коммуникации (синхронный/асинхронный) в зависимости от требований к согласованности и производительности.

Владеет: Навыками проектирования и развертывания отказоустойчивых микросервисных систем с использованием оркестратора (Kubernetes). • **Методами** обоснования выбора технологий (включая отечественные) для конкретного проекта, оценки компромиссов (производительность/сложность).

ОПК-6.2 Ориентируется в современных положениях и концепциях прикладного и системного программного обеспечения, архитектуры компьютеров и сетей (в том числе и глобальных), технологии создания и сопровождения программных продуктов и программных комплексов

Знает: Принципы работы сетей: модель OSI/TCP-IP, DNS, HTTP/• Принципы мониторинга и observability в распределенных системах (логи, метрики, трейсинг - ELK Stack, Prometheus, Grafana, Zipkin).

Умеет: Проектировать сетевое взаимодействие между сервисами, понимая последствия выбора портов, протоколов и политик безопасности. • Настраивать пайплайн CI/CD для отдельного микросервиса, обеспечивая его автоматизированное тестирование и развертывание. • Настраивать централизованный сбор логов и мониторинг ключевых метрик

(health, latency, traffic) для группы сервисов.• Анализировать проблемы в распределенной системе, используя логи и трейсы для идентификации "больного" сервиса.

Владеет: Методами диагностики и решения проблем, характерных для распределенных систем (сетевые задержки, частичные отказы, проблемы согласованности данных).**Системным мышлением** для понимания полного жизненного цикла микросервисного приложения — от разработки и до эксплуатации и масштабирования в глобальной сети.

LC-3 ***Способен проектировать и поддерживать архитектуру систем искусственного интеллекта***

LC-3.1 Создает и развивает архитектуры системы ИИ на всех этапах жизненного цикла

Знает архитектуры системы ИИ.
Умеет определять архитектуру ИИ, подходы к проектированию, выбор подходящего стека технологий ИИ для RnD и промышленной разработки.
Владеет способностью применять различные принципы и паттерны при проектировании архитектуры систем ИИ.

PL-1 ***Способен применять язык программирования Python для решения задач в области ИИ***

PL-1.1 Разрабатывает и отлаживает прикладные решения разной сложности и для разного круга конечных пользователей с использованием языка программирования Python, тестирует, испытывает и оценивает качество таких решений.
Знает основы синтаксиса языка, пишет небольшие скрипты для автоматизации ручной работы по обработке небольших объемов данных с помощью встроенных модулей и внешних библиотек (csv, json, requests).
Владеет основными библиотеками для выполнения большинства рутинных задач в крупных проектах: ввод-вывод, серверное программирование (FastAPI, Flask, Django REST Framework), применение многопоточности (модуль threading).
Умеет разрабатывать серверные приложения и поддерживать их. Использует особенности виртуальной машины Python (например, GIL), разрабатывает библиотечный код общего пользования, а также документацию к нему. Профилирует и оптимизирует приложения на Python, используя встроенные инструменты (например, cPython).

PL-1.2 Осуществляет выбор инструментов разработки на Python, приемлемых для создания прикладной системы обработки научных данных, машинного обучения и визуализации с заданными требованиями
Знает и применяет основные библиотеки для научных вычислений, такие как NumPy, SciPy и Pandas. **Знает** и применяет основные библиотеки для визуализации данных, например, Matplotlib и Seaborn
Умеет разрабатывать собственные компоненты для библиотек машинного обучения с учётом интеграции с ними.
Владеет приемами оптимизации кода с использованием библиотек для научных вычислений. **Владеет** навыками применения библиотек машинного обучения, в том числе глубокого обучения, такие как scikit-learn, PyTorch и TensorFlow

PL-1.3 Разрабатывает и поддерживает системы обработки больших данных различной степени сложности

Знает принципы работы систем обработки больших данных различной степени сложности.

Умеет разрабатывать и поддерживать простейшие ETL-скрипты в пайплайнах обработки данных, применять основные функции фреймворка PySpark, может самостоятельно построить процесс обработки больших данных с использованием Airflow. **Владеет** инструментами профилирования и оптимизации ETL процессы для обработки больших данных в рамках Spark/Mapreduce фреймворка. Самостоятельно поддерживает инфрастуктуру обработки больших данных.

PL-2 ***Способен применять JVM-совместимые языки программирования для решения задач в области ИИ***

PL-2.1 Разрабатывает и отлаживает прикладные решения разного уровня сложности и для широкого круга конечных пользователей с использованием JVM-совместимых языков программирования, тестирует, испытывает и оценивает качество таких решений.

Знает и понимает модель памяти Java и способен поддерживать приложения с высоким параллелизмом и конкуренцией. Понимает алгоритмы сборки мусора и способен оптимизировать сборку мусора. **Умеет** применять основные библиотеки для решения рутинных задач в серверном программировании: ввод-вывод. **Владеет** навыками применения простейших примитивов многопоточного программирования, интеграция с базами данных.

PL-2.2 Разрабатывает и поддерживает системы обработки больших данных различной степени сложности.

Знает принципы работы систем обработки больших данных различной степени сложности.

Умеет разрабатывать и поддерживать простые ETL алгоритмы в пайплайнах обработки данных. Эффективно применяет фреймворки для пакетной обработки данных (Spark, Mapreduce) и адаптирует алгоритмы для вычислений на малых объемах данных под большие данные. **Владеет** инструментами организации потоковых вычислений (Kafka, Flink) и интеграции с аналитическими БД (ClickHouse, Greenplum)

Содержание и структура дисциплины:

Распределение видов учебной работы и их трудоемкости по разделам дисциплины.
Разделы дисциплины, изучаемые в _7_ семестре *(очная форма)*

№	Наименование разделов (тем)	Количество часов				
		Всего	Аудиторная работа			Внеаудиторная работа
			Л	ПЗ	ЛР	СРС
1	2	3	4	5	6	7
1.	Введение в Микросервисную архитектуру. От Монолита к Микросервисам	8	2		4	2

№	Наименование разделов (тем)	Количество часов				
		Всего	Аудиторная работа			Внеаудиторная работа
			Л	ПЗ	ЛР	СРС
1	2	3	4	5	6	7
2.	**Декомпозиция и проектирование сервисов. Антипаттерны**	8	2		4	2
3.	**Взаимодействие между микросервисами. Синхронные и асинхронные паттерны**	8	2		4	2
4.	Управление данными в распределенной системе	8	2		4	2
5.	Обнаружение сервисов, конфигурирование и безопасность	8	2		4	2
6.	Наблюдаемость (Observability) и мониторинг	8	2		4	2
7.	Развертывание и оркестрация микросервисов	8	2		4	2
8.	Тестирование и надежность микросервисных систем	11,8	2		6	3,8
ИТОГО по разделам дисциплины		**69,8**	**16**		**34**	**17,8**
Контроль самостоятельной работы (КСР)		4				
Промежуточная аттестация (ИКР)		0,2				
Подготовка к текущему контролю		0				
Общая трудоемкость по дисциплине		**72**				

Примечание: Л – лекции, КСР – контрольные и самостоятельные работы, ЛР – лабораторные занятия, СРС – самостоятельная работа студента

Курсовые проекты или работы.
Не предусмотрены учебным планом

Вид аттестации: ЛР, проект по кейсам индустриальных партнеров, зачет.

Автор Приходько Т.А. – кандидат технических наук, доцент кафедры вычислительных технологий;