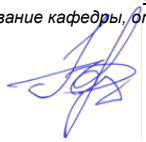
МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ВГУ»)

УТВЕРЖДАЮ

Заведующий кафедрой
информационных систем
наименование кафедры, отвечающей за реализацию дисциплины



(*Борисов Д.Н.*)
подпись, расшифровка подписи
3.05.2023 г.

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ
Б1.В.09 DevOps
Код и наименование дисциплины в соответствии с учебным планом

1. Код и наименование направления подготовки/специальности:
09.03.02 Информационные системы и технологии

2. Профиль подготовки/специализация: Информационные системы и сетевые технологии

3. Квалификация выпускника: *Бакалавриат*

4. Форма обучения: *очная*

5. Кафедра, отвечающая за реализацию дисциплины: *Кафедра информационных систем*

6. Составители программы: *Тырнов Ю.О. (y.tyrnov@dcrt.it)*

7. Рекомендована: рекомендована НМС ФКН 03.05.2023, протокол № 7

8. Учебный год: *2025/2026* **Семестр(ы):** ***6***

9.Цели и задачи учебной дисциплины
Формирование у студентов знаний, умений и навыков, необходимых для автоматизации процессов разработки, тестирования, развертывания и эксплуатации программного обеспечения с использованием технологий DevOps.
Задачи учебной дисциплины:
освоение принципов DevOps и их применения в разработке ПО;
изучение инструментов контейнеризации (Docker) и CI/CD (GitLab CI);
развитие навыков управления инфраструктурой через код (IaC) с использованием Ansible;
изучение методов мониторинга и логирования для обеспечения стабильности приложений.

10. Место учебной дисциплины в структуре ООП: *Дисциплина относится к части, формируемой участниками образовательных отношений (Б1.В.09). Для освоения дисциплины студент должен владеть компетенциями дисциплин: Б1.О.15 Введение в программирование, Б1.О.20 Операционные системы. Уметь работать в консоли Linux на уровне пользователя, пользоваться системой контроля версий (предпочтительно git). Иметь представление о культуре и стиле разработки ПО.*

11. Планируемые результаты обучения по дисциплине/модулю (знания, умения, навыки), соотнесенные с планируемыми результатами освоения образовательной программы (компетенциями) и индикаторами их достижения:

Код и название компетенции	Код и название индикатора компетенции	Знания, умения, навыки
ПК-2 Способен выполнять интеграцию программных модулей и компонент, выполнять верификацию программных продуктов	ПК-2.2 Собирает программные компоненты в программный продукт	Знать: алгоритм сборки программных компонентов в программный продукт Уметь: собирать программные компоненты в программный продукт Владеть: сборкой программных компонентов в программный продукт
ПК-3 Способен выполнять работы по созданию (модификации) и сопровождению информационных систем	ПК-3.1 Знает языки и методы программирования, инструменты и методики тестирования разрабатываемых ИС	Знать: языки и методы программирования, инструменты и методики тестирования для разработки ИС Уметь: использовать языки и методы программирования, инструменты и методики тестирования для разработки ИС Владеть: языками и методами программирования, инструментами и методиками тестирования при разработке ИС
ПК-4 Способен проводить анализ требований к программному обеспечению, выполнять работы по проектированию программного обеспечения	ПК-4.1 Знает принципы построения архитектуры программного обеспечения, методы и средства проектирования программного обеспечения	Знать: принципы построения архитектуры программного обеспечения, методы и средства проектирования программного обеспечения Уметь: использовать принципы построения архитектуры программного обеспечения, методы и средства проектирования программного обеспечения Владеть: принципами построения архитектуры программного обеспечения, методами и средствами проектирования программного обеспечения
ПК-4 Способен проводить анализ требований к программному обеспечению, выполнять работы по проектированию программного обеспечения	ПК-4.4 Описывает технологии обработки данных для возможности их использования в программной среде, включая вопросы параллельной обработки	Знать: технологии обработки данных для возможности их использования в программной среде, включая вопросы параллельной обработки Уметь: использовать технологии обработки данных для возможности их использования в программной среде, включая вопросы параллельной обработки Владеть: технологиями обработки данных для возможностей их использования в программной среде, включая вопросы параллельной обработки

12. Объем дисциплины в зачетных единицах/час

2/72

Форма промежуточной аттестации

Зачет, Курсовая работа

13. Трудоемкость по видам учебной работы

Вид учебной работы	Семестр 6	Всего
Аудиторные занятия		
Лекционные занятия	16	16
Практические занятия		
Лабораторные занятия	16	16
Самостоятельная работа	40	40
Курсовая работа		
Промежуточная аттестация		
Часы на контроль		
Всего	72	72

13.1. Содержание дисциплины

п/п	Наименование раздела дисциплины	Содержание раздела дисциплины	Реализация раздела дисциплины с помощью онлайн-курса, ЭУМК *
1. Лекции			
1.1	Введение в DevOps	Эволюция подходов разработки ПО: Waterfall → Agile → DevOps. Основные этапы DevOps: CI/CD, мониторинг, управление инфраструктурой.	-
1.2	Контейнеризация	Основы работы с Docker: создание контейнеров, работа с образами.	
1.3	Автоматизация процессов	Настройка CI/CD пайплайнов с использованием GitLab CI. Автоматизация инфраструктуры с помощью Ansible.	
1.4	Мониторинг и логирование	Инструменты мониторинга (Prometheus, Grafana). Логирование с использованием ELK Stack.	-
1.5	Облачные технологии	Работа с облаками для автоматизации DevOps-процессов.	-
2. Практические занятия			
3. Лабораторные занятия			
3.1	Введение в DevOps	Настройка сервера, раскатка приложения руками	
3.2	Контейнеризация	Контейнеризация приложение и выкладка разных версий в хаб	

3.3	Автоматизация процессов	Улучшение процесса по автоматизации сборки, добавление автоматической проверку кода, настройка уведомления для разработчиков по ходу проверки и сборки.	
3.4	Мониторинг и логирование	Интеграция мониторинга и алертинга в инфраструктуру, автоматизация обработки логов в приложении.	
3.5	Облачные технологии	Настройка Ansible для автоматического создания и конфигурирования инфраструктуры.	

13.2. Темы (разделы) дисциплины и виды занятий

№ п/п	Наименование темы (раздела) дисциплины	Виды занятий (количество часов)				
		Лекции	Практи-ческие	Лаборатор-ные	Самостоя-тельная работа	Всего
1	Введение в DevOps	2		2	4	8
2	Контейнеризация	2		2	4	8
3	Автоматизация процессов	6		6	16	28
4	Мониторинг и логирование	4		4	12	20
5	Облачные технологии	2		2	4	8
	Итого:	**16**		**16**	**40**	**72**

14. Методические указания для обучающихся по освоению дисциплины

Внеаудиторная самостоятельная работа студентов включает проработку материалов лекций, изучение рекомендованной литературы, подготовку к контрольным работам, подготовку к лабораторным работам и их защитам, подготовку к защите типовых проектов.

15. Перечень основной и дополнительной литературы, ресурсов интернет, необходимых для освоения дисциплины

а) основная литература:

№ п/п	Источник
1	Баланов, А. Н. DevOps: интеграция и автоматизация : учебное пособие для вузов / А. Н. Баланов. — 2-е изд., стер. — Санкт-Петербург : Лань, 2025. — 240 с. — ISBN 978-5-507-50491-6. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/440162— Режим доступа: для авториз. пользователей : https://reader.lanbook.com/book/440162#1
2	Баланов, А. Н. Построение микросервисной архитектуры и разработка высоконагруженных приложений: Учебное пособие для вузов / А. Н. Баланов. — 2-е изд., стер. — Санкт-Петербург : Лань, 2025. — 244 с. — ISBN 978-5-507-52652-9. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://reader.lanbook.com/book/456920. — Режим доступа: для авториз. пользователей : https://reader.lanbook.com/book/456920#1

б) дополнительная литература:

№ п/п	Источник
1	Херинг, М. DevOps для современного предприятия : учебное пособие / М. Херинг ; перевод с английского М. А. Райтмана.. — Москва : ДМК Пресс, 2020. — 232 с. — ISBN 978-5-97060-836-4. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/140580

2	Арзуманян, М. Ю. Архитектура предприятия : учебное пособие / М. Ю. Арзуманян. — Санкт-Петербург : СПбГУТ им. М.А. Бонч-Бруевича, 2016. — 86 с. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/180250
3	Грувер, Г. Запуск и масштабирование DevOps на предприятии / Г. Грувер. — Москва : ДМК Пресс, 2018. — 80 с. — ISBN 978-5-97060-704-6. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/116130
4	Скрынник, О. В. DevOps для ИТ-менеджеров: концентрированное структурированное изложение передовых идей / О. В. Скрынник. — 2-е изд. — Москва : ДМК Пресс, 2019. — 126 с. — ISBN 978-5-97060-692-6. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/112933

в) информационные электронно-образовательные ресурсы (официальные ресурсы интернет)***:**

№ п/п	Ресурс
1	*Официальная документация Docker (docker.com)*
2	*Курсы на платформе KodeCloud или Coursera*

16. Перечень учебно-методического обеспечения для самостоятельной работы

№ п/п	Источник
1	*Программное обеспечение: Docker Desktop, Ansible CLI*
2	*Платформы для практики: GitHub/GitLab*

17. Образовательные технологии, используемые при реализации учебной дисциплины, включая дистанционные образовательные технологии (ДОТ), электронное обучение (ЭО), смешанное обучение):

1. Электронно-библиотечная система «электронно-библиотечная система Лань», https://e.lanbook.com
2. Образовательный портал Moodle (сервер Moodle ВГУ)
3. OS Mbed. – Режим доступа : http://osmbed.com

18. Материально-техническое обеспечение дисциплины: Лекционная аудитория, оборудованная мультимедийным проектором. Компьютерные классы факультета для проведения лабораторных занятий. Программный эмулятор учебной ЭВМ для проведения лабораторных занятий. Образовательный портал «Электронный университет ВГУ» https://edu.vsu.ru.

19. Оценочные средства для проведения текущей и промежуточной аттестаций.

Текущий контроль освоения программы осуществляется на основе результатов выполнения лабораторных работ, в том числе с применением дистанционных образовательных технологий (электронный курс на образовательном портале «Электронный университет ВГУ» edu.vsu.ru). Перечень лабораторных работ приведен выше.

Порядок оценки освоения обучающимися учебного материала определяется содержанием следующих разделов дисциплины:

№ п/п	Наименование раздела дисциплины (модуля)	Компетенция(и)	Индикатор(ы) достижения компетенции	Оценочные средства
1	Введение в DevOps Контейнеризация Автоматизация процессов	ПК-2	ПК-2.2	*Тестовое задание* *Типовые проекты*

№ п/п	Наименование раздела дисциплины (модуля)	Компетенция(и)	Индикатор(ы) достижения компетенции	Оценочные средства
2	Автоматизация процессов Мониторинг и логирование	ПК-3	ПК-3.1	*Тестовое задание* *Типовые проекты*
3	Введение в DevOps Автоматизация процессов Облачные технологии	ПК-4	ПК-4.1	*Тестовое задание* *Типовые проекты*
4	Введение в DevOps Мониторинг и логирование Облачные технологии	ПК-4	ПК-4.4	*Тестовое задание* *Типовые проекты*

Промежуточная аттестация

Форма контроля - Зачет, Курсовая работа

Лабораторные работы после выполнения оцениваются преподавателем, и выставляется оценка «зачтено» при условии ответа на 80% вопросов преподавателя по предметной области лабораторной работы. По итогам лабораторных работ и устного ответа студента выставляется оценка «зачтено» или «не зачтено» по лабораторным работам всей дисциплины. К сдаче зачета с оценкой допускаются студенты, сдавшие 100% лабораторных работ.

Соотношение показателей, критериев и шкалы оценивания результатов обучения.

Критерии оценивания компетенций	Уровень сформирован-ности компетенций	Шкала оценок
Обучающийся владеет базовыми знаниями о DevOps и возможностью его практического применения. Обучающийся, может давать неполные ответы на дополнительные вопросы.	*Пороговый уровень*	*Зачтено*
Отсутствует владение основными понятиями DevOps или демонстрация существенных пробелов в знаниях. Обучающийся демонстрирует отрывочные, фрагментарные знания, допускает грубые ошибки в DevOps. Отсутствует способность выполнить задания даже с помощью преподавателя.	-	*Не зачтено*

20 Типовые оценочные средства и методические материалы, определяющие процедуры оценивания

20.1 Текущий контроль успеваемости

Контроль успеваемости по дисциплине осуществляется с помощью контрольных работ и тестовых заданий.

20.1.1 Тестовое задание

Вариант 1

1. Объясните основные принципы DevOps. Как они способствуют улучшению взаимодействия между командами разработки и эксплуатации?
2. Опишите процесс создания Docker-образа с использованием Dockerfile. Какие ключевые команды используются?
3. Какие метрики необходимо отслеживать для мониторинга серверов? Назовите инструменты, которые можно использовать для этой задачи.

Вариант 2

1. Чем контейнеризация отличается от традиционного подхода к развертыванию приложений? Какие преимущества она предоставляет?
2. Что такое автоматизация процессов в DevOps? Приведите примеры задач, которые можно автоматизировать.

3. Объясните, как централизованное логирование помогает анализировать работу системы. Укажите инструменты, которые можно использовать.

Вариант 3

1. Что такое CI/CD? Объясните разницу между непрерывной интеграцией (CI) и непрерывной доставкой (CD).
2. Как работает Docker Compose? Напишите пример файла `docker-compose.yml` для запуска веб-приложения и базы данных.
3. Какие преимущества дают облачные технологии для DevOps-процессов? Приведите примеры популярных платформ.

Вариант 4

1. Расскажите о роли контейнеризации в DevOps. Почему использование Docker стало стандартом в индустрии?
2. Опишите процесс настройки взаимодействия нескольких контейнеров с помощью Docker Compose.
3. Как облачные технологии обеспечивают безопасность данных? Приведите примеры практик и инструментов.

Вариант 5

1. Что такое мониторинг в DevOps? Какие метрики следует отслеживать для обеспечения стабильной работы приложения?
2. Опишите процесс создания и запуска контейнера с использованием Docker CLI (командной строки). Какие команды используются?
3. Как масштабируемость облачных технологий помогает DevOps-командам быстро адаптироваться к изменяющимся требованиям? Приведите примеры.

Приведённые ниже задания рекомендуется использовать при проведении диагностических работ для оценки остаточных знаний

ПК-2

Задания закрытого типа (в каждом задании необходимо выбрать один или несколько ответов)

1. Что означает принцип "Shift Left" в DevOps?
a) Перенос тестирования на более ранние этапы разработки
b) Использование левых компонентов в архитектуре
c) Удаление старых версий ПО
d) Запуск приложений только в левом дата-центре

2. Какой инструмент чаще всего используется для визуализации метрик и дашбордов в DevOps?
a) Nagios
b) Grafana
c) Jenkins
d) Kibana

3. Что означает аббревиатура CI/CD?
a) Centralized Infrastructure / Custom Development
b) Code Integration / Continuous Development
c) Continuous Integration / Continuous Delivery (или Deployment)
d) Cloud Integration / Container Deployment

4. Какой инструмент чаще всего используется для конфигурационного управления?
a) Jenkins
b) Ansible
c) Docker
d) Prometheus

5. Что такое Canary Deployment?
a) Обновление всех компонентов системы сразу
b) Постепенное развертывание обновлений на небольшой части пользователей
c) Резервное копирование всех сервисов
d) Метод тестирования безопасности

Задания открытого типа

1. Файлы в Ansible называются...?
2. Инструмент для управления инфраструктурой как кодом?
3. Название процесса автоматического развертывания после тестов?
4. Где хранятся конфигурации Kubernetes?
5. Графическая система визуализации метрик?

Задания с развернутым ответом

1. Опишите, как использовать Docker для разработки и развёртывания микросервиса. Что должно быть в Dockerfile и как организовать docker-compose.yml?
2. Опишите процесс настройки Jenkins для CI/CD пайплайна, который автоматически собирает, тестирует и разворачивает веб-приложение.
3. Опишите процесс настройки мониторинга микросервисов с использованием Prometheus и Grafana. Какие метрики нужно собирать?
4. Опишите, как с помощью Terraform можно развернуть веб-приложение в облаке AWS. Какие ресурсы нужно описать и как управлять состоянием?
5. Опишите процесс развертывания микросервиса в Kubernetes. Какие манифесты нужно создать и как обеспечить его доступность?

ПК-3

Задания закрытого типа (в каждом задании необходимо выбрать один или несколько ответов)

6. Какую задачу решает Helm в экосистеме Kubernetes?
a) Мониторинг подов
b) Хранение образов
c) Управление пакетами и шаблонами развертывания
d) Настройка облачной инфраструктуры

7. Какой компонент Kubernetes отвечает за планирование размещения подов?
a) Etcd
b) Scheduler
c) Kubelet
d) Controller Manager

8. Что происходит на этапе "Build" в CI/CD пайплайне?
a) Запускается приложение на проде
b) Производится сборка кода в исполняемый артефакт
c) Проводится нагрузочное тестирование
d) Настраивается инфраструктура

9. Какая команда в Git используется для объединения веток?
a) git push
b) git merge
c) git fork
d) git checkout

10. Что делает инструмент Docker?
a) Следит за использованием памяти
b) Позволяет запускать приложения в контейнерах
c) Мониторит сетевые соединения
d) Управляет хранилищем данных

Задания открытого типа

1. Как называется практика непрерывного объединения кода?
2. Какой инструмент используется для управления контейнерами?

3. Инструмент для мониторинга и сбора метрик?
4. Формат описания инфраструктуры в виде кода?
5. Инструмент для контейнеризации приложений?

Задания с развернутым ответом

1. Представьте, что вам нужно развернуть приложение в облаке (AWS или GCP). Опишите процесс настройки виртуальной сети (VPC) и обеспечения безопасности приложения в этой сети.
2. Объясните, как микросервисная архитектура влияет на разработку и эксплуатацию программных систем. Приведите примеры преимуществ и недостатков.
3. Опишите процесс настройки пайплайна CI/CD для веб-приложения. Какие этапы вы включите и какие инструменты будете использовать?
4. Объясните, что такое контейнеризация, и опишите основные преимущества использования Docker в DevOps-процессах.
5. Раскройте понятие Infrastructure as Code (IaC). Какой инструмент вы бы выбрали для управления инфраструктурой и почему?

ПК-4

Задания закрытого типа (в каждом задании необходимо выбрать один или несколько ответов)

1. Какая архитектура подразумевает разделение приложения на независимые сервисы?

a) Монолитная
b) Микросервисная
c) Слоистая
d) Модульная

2. Что обеспечивает горизонтальное масштабирование в Kubernetes?

a) Увеличение ресурсов отдельных узлов
b) Автоматическое добавление подов при высокой нагрузке
c) Ручное создание новых контейнеров
d) Запуск приложений на одном сервере

3. Какой инструмент используется для реализации принципа Infrastructure as Code (IaC)?

a) Jenkins
b) Terraform
c) Selenium
d) Grafana

4. Что такое Service Mesh?

a) Средство для балансировки нагрузки между серверами
b) Сетевой слой для управления коммуникацией между микросервисами
c) Инструмент для мониторинга производительности
d) База данных для хранения логов

5. Для чего используется Prometheus?

a) Автоматизация сборки приложений
b) Мониторинг и сбор метрик в реальном времени
c) Управление контейнерами
d) Настройка CI/CD пайплайнов

Задания открытого типа

1. Формат хранения контейнеров в Docker?
2. Как называется отдельная единица выполнения в Kubernetes?
3. Инструмент CI/CD от GitLab?
4. Как называется сеть в облачной инфраструктуре?
5. Популярная система управления версиями?

Задания с развернутым ответом

1. Опишите процесс настройки и внедрения микросервисной архитектуры для веб-приложения, включая использование инструментов DevOps.

2. Представьте, что вы администрируете кластер Kubernetes. Опишите процесс настройки автоматического масштабирования подов на основе нагрузки CPU.
3. Приведите пример сценария, в котором использование балансировщика нагрузки улучшает производительность приложения. Какие инструменты можно использовать для реализации балансировки?
4. Опишите основные этапы мониторинга приложения в DevOps-процессе. Какие метрики вы считаете наиболее важными для анализа производительности?
5. В чем заключается разница между Continuous Integration (CI), Continuous Delivery (CD) и Continuous Deployment (CD)? Приведите примеры их применения.
6. Объясните, как обеспечить безопасность данных в процессе работы с контейнерами Docker. Какие меры вы бы приняли для защиты контейнеров?

20.1.2 Курсовые работы (типовые проекты)
Выполнение курсовой работы (типового проекта) является необходимым условием допуска студентов к итоговому зачёту. Курсовая работа выполняется в три этапа:
Первый - выбор проекта и необходимых инструментов для его осуществления.
Второй этап - установка и настройка инструментов, написание необходимых скриптов.
Третий этап - запуск проекта в продакшн среде и его отладка, написание необходимой документации.

Примерный перечень курсовых работ (проектов):
1. Мониторинг серверов с Prometheus и Grafana.
2. Автоматизация развертывания веб-приложения с Docker Compose.
3. Централизованное логирование с использованием ELK-стека.
4. Настройка CI/CD пайплайна для статического сайта.
5. Автоматизация резервного копирования баз данных.
6. Облачное развертывание веб-приложения.
7. Система оповещений о сбоях через Telegram.
8. Контейнеризация монолитного приложения.
9. Автоматизация управления конфигурацией серверов.
10. Визуализация метрик производительности приложений.
11. Автоматизация тестирования API с Postman.
12. Настройка отказоустойчивой инфраструктуры в облаке.
13. Разработка системы управления логами.
14. Мониторинг доступности веб-сайтов.
15. Управление конфигурациями серверов.
16. Разработка системы управления версиями приложений.
17. Создание системы мониторинга микросервисов.
18. Автоматизация обновления серверного ПО.
19. Настройка масштабируемости приложений.
20. Централизованное управление логами.
21. Разработка дашборда для анализа метрик приложений.
22. Контейнеризация тестовой среды для CI/CD пайплайна.
23. Настройка автоматического деплоя приложений на серверы.
24. Разработка системы мониторинга сети.
25. Внедрение DevSecOps: автоматический анализ безопасности кода.
26. Управление облачными ресурсами.
27. Настройка системы алертинга для микросервисной архитектуры.

28. Разработка отказоустойчивого веб-приложения в облаке.
29. Автоматизация управления базами данных через Ansible.
30. Мониторинг производительности серверов.
31. Логирование событий безопасности приложения.
32. Разработка системы управления конфигурациями.
33. Внедрение Blue-Green деплоя для приложений.
34. Создание CI/CD пайплайна для микросервисной архитектуры.
35. Автоматизация развертывания приложений.
36. Управление секретами приложения.
37. Разработка системы тестирования производительности приложений.
38. Мониторинг сетевой активности.
39. Визуализация данных мониторинга.
40. Централизованное управление логами микросервисов.

20.2 Промежуточная аттестация

Оценка знаний, умений и навыков, характеризующая этапы формирования компетенций в рамках изучения дисциплины осуществляется в ходе текущей и промежуточной аттестаций.

Текущая аттестация проводится в соответствии с Положением о текущей аттестации обучающихся по программам высшего образования Воронежского государственного университета. Текущая аттестация проводится в форме индивидуального опроса в рамках рубежных аттестаций. Критерии оценивания приведены выше.

Промежуточная аттестация проводится в соответствии с Положением о промежуточной аттестации обучающихся по программам высшего образования.

Контрольно-измерительные материалы промежуточной аттестации включают в себя теоретические вопросы, позволяющие оценить уровень полученных знаний.

При оценивании используются качественные шкалы оценок. Критерии оценивания приведены выше.

Перечень вопросов к зачету:

Что такое DevOps и какие задачи он решает. Основные принципы и практики DevOps. Роль DevOps-инженера в проекте разработки и внедрения ПО. Жизненный цикл программного обеспечения в контексте DevOps. Что такое контейнеры Docker и как они работают. Преимущества использования Docker для разработки и развертывания приложений. Основы работы с Kubernetes: деплой, масштабирование, отказоустойчивость. Взаимодействие Kubernetes с Docker. Микросервисная архитектура: преимущества и вызовы. Что такое CI/CD, основные этапы и инструменты. Настройка пайплайнов CI/CD с использованием Jenkins или GitLab CI. Основы управления конфигурацией с использованием Ansible. Взаимодействие Ansible с Docker.