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
05.05.2025 г..

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ
Б1.В.03 Технологии DevOps

Код и наименование дисциплины в соответствии с учебным планом

1. Код и наименование направления подготовки:

09.04.02 Информационные системы и технологии

2. Профиль подготовки: Программные технологии в инфокоммуникационных системах

3. Квалификация выпускника: *Магистр*

4. Форма обучения: *заочная*

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

6. Составители программы: *асс. Тырнов Ю.О., доц. Борисов Д.Н.*

.__.

7. Рекомендована: *НМС факультета компьютерных наук, протокол № 7 от 05.05.2025 г*

(наименование рекомендующей структуры, дата, номер протокола,

отметки о продлении вносятся вручную)

8. Учебный год: *2027/2028* **Семестр/триместр(ы):** *триместр 7, курс 3*

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), оркестрации (Kubernetes) и CI/CD (Jenkins, GitLab CI).
- Развитие навыков управления инфраструктурой через код (IaC) с использованием Terraform и Ansible.
- Изучение методов мониторинга и логирования для обеспечения стабильности приложений.

10. Место учебной дисциплины в структуре ООП: Дисциплина относится части, формируемая участниками образовательных отношений. Для освоения дисциплины студент должен владеть компетенциями дисциплин Б1.О.10 Системная инженерия, Б1.В.04 Информационная безопасность интранет и IoT сетей.

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-4 Способен обеспечивать проверку программного обеспечения на различных этапах разработки и внедрения	ПК-4.1 Выполняет сборку программных компонентов и интеграцию в общий проект	**знать:** принципы сборки программных компонентов и интеграции в общий проект, принципы DevOps, инструменты CI/CD, контейнеризации и IaC, подходы к мониторингу и логированию **уметь:** выполнять сборку программных компонентов и интеграцию в общий проект **владеть:** методикой сборки программных компонентов и интеграции в общий проект
ПК-5 Способен описывать и сопровождать архитектуру вычислительных систем и комплексов, обеспечивает их работоспособность и безопасность	ПК-5.1 Администрирует встроенные подсистемы и средства защиты информации	**знать:** алгоритмы администрирования встроенных подсистем и средств защиты информации **уметь:** администрировать встроенные подсистемы и средства защиты информации, разрабатывать CI/CD пайплайны, автоматизировать управление инфраструктурой, внедрять системы мониторинга и логирования **владеть:** методикой администрирования встроенных подсистем и средств защиты информации

ПК-5 Способен описывать и сопровождать архитектуру вычислительных систем и комплексов, обеспечивает их работоспособность и безопасность	ПК-5.3 Выполняет оптимизацию распределения вычислительных ресурсов и компонентов вычислительной сети	**знать:** алгоритмы оптимизации распределения вычислительных ресурсов и компонентов вычислительной сети **уметь:** выполнять оптимизацию распределения вычислительных ресурсов и компонентов вычислительной сети **владеть:** методикой оптимизации распределения вычислительных ресурсов и компонентов вычислительной сети инструментами Docker, Kubernetes, Ansible, Terraform; навыками настройки Jenkins и GitLab CI.

12. Объем дисциплины в зачетных единицах/час. — 4/*144*.

Форма промежуточной аттестации *Экзамен.*

13. Трудоемкость по видам учебной работы

Вид учебной работы	3 курс, триместр 7	Всего
Аудиторные занятия	10	10
Лекционные занятия	4	4
Практические занятия		
Лабораторные занятия	6	6
Самостоятельная работа	125	125
Курсовая работа		
Промежуточная аттестация		
Часы на контроль	9	9
Всего	**144**	**144**

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.	https://edu.vsu.ru/course/view.php?id=31763
1.2	Контейнеризация	Основы работы с Docker: создание контейнеров, работа с образами. Оркестрация с Kubernetes: деплой приложений, масштабирование.	
1.3	Автоматизация процессов	Настройка CI/CD пайплайнов с использованием Jenkins или GitLab CI. Автоматизация инфраструктуры с помощью Terraform и Ansible.	
1.4	Мониторинг и логирование	Инструменты мониторинга (Prometheus, Grafana). Логирование с использованием ELK Stack.	https://edu.vsu.ru/course/view.php?id=31763
1.5	Облачные технологии	Работа с AWS/GCP/Azure для автоматизации DevOps-процессов.	https://edu.vsu.ru/course/view.php?id=31763
2. Практические занятия			
3. Лабораторные занятия			https://edu.vsu.ru/course/view.php?id=31763
3.1	Введение в DevOps	Настройка сервера, раскатка приложения	
3.2	Контейнеризация	Контейнеризация приложения и размещение разных версий в хаб	
3.3	Автоматизация процессов	Улучшение процесса по автоматизации сборки, добавление автоматической проверки кода, настройка уведомления для разработчиков на предмет 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 и Terraform 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	Практические	Лабораторные	Самостоятельная работа	Всего
1	Введение в DevOps	1		1	25	27
2	Контейнеризация	1		1	25	27
3	Автоматизация процессов	1		1	25	27
4	Мониторинг и логирование			2	25	27
5	Облачные технологии	1		1	25	27
	Итого:	4		6	125	135

14. Методические указания для обучающихся по освоению дисциплины

Внеаудиторная самостоятельная работа студентов включает проработку материалов лекций, изучение рекомендованной литературы, подготовку к лабораторным работам и их защитам, подготовку к устному опросу и зачету.

Самостоятельная работа в аудитории выполняется под непосредственным руководством преподавателя. Во время самостоятельной работы студенты используют электронно-библиотечные системы, доступные на портале Зональной Библиотеки ВГУ по адресу www.lib.vsu.ru. Для повышения эффективности руководства при проведении лабораторных занятий, призванных обеспечить выборочное использование лекционного материала для более глубокого изучения отдельных разделов дисциплины при решении соответствующих практических задач.

При использовании дистанционных образовательных технологий и электронного обучения выполнять все указания преподавателей по работе на LMS-платформе, своевременно подключаться к online-занятиям, соблюдать рекомендации по организации самостоятельной работы.

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	Электронно-библиотечная система «электронно-библиотечная система Лань», https://e.lanbook.com
2	Официальная документация Docker (docker.com)
3	Документация Kubernetes (kubernetes.io)

16. Перечень учебно-методического обеспечения для самостоятельной работы

№ п/п	Источник
1	Курсы на платформе KodeCloud или Coursera
2	Платформы для практики: GitLab, Jenkins

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

1. Образовательный портал Moodle (сервер Moodle ВГУ), www.lib.vsu.ru
2. Платформа GitLab https://gitlab.com
3. Платформа Jenkins, https://www.jenkins.io

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.

Для реализации учебного процесса используется:

VirtualBox (Oracle VM VirtualBox) — программный продукт для виртуализации операционных систем.

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	ПК-4	ПК-4.1	*Тестовое задание1 Практического задание 1*
2	Автоматизация процессов Мониторинг и логирование	ПК-5	ПК-5.1	*Тестовое задание 2 Практического задание 2*
3	Облачные технологии	ПК-5	ПК-5.3	*Контрольное задание 1 Контрольное задание 2*

Для оценивания результатов обучения на экзамене используется 4-балльная шала: «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

Соотношение показателей, критериев и шкалы оценивания результатов обучения

Критерии оценивания компетенций	**Уровень сформированности компетенций**	**Шкала оценок**
Глубокое понимание принципов DevOps, включая CI/CD, контейнеризацию, оркестрацию, управление конфигурацией и мониторинг. Уверенное владение теоретическими аспектами и способность объяснить их применение на практике. Выполнение задания без ошибок, включая настройку пайплайнов CI/CD, работу с Docker и Kubernetes, автоматизацию инфраструктуры с помощью Ansible или Terraform. Решение задач самостоятельно, с предложением оптимальных решений. Уверенное использование инструментов DevOps, свободная работа с системами контроля версий	*Повышенный уровень*	*Отлично*

(Git), контейнерами и облачными платформами. Демонстрация способности применять знания в сложных и нестандартных ситуациях. Все задания выполнены в полном объеме, решения эффективны и соответствуют требованиям.		
Демонстрация уверенных знаний ключевых концепций DevOps, с допущением незначительных неточностей в теории или терминологии. Выполнение заданий с небольшими недочетами, которые не влияют на общий результат. Способность настроить пайплайн CI/CD, работать с Docker и Kubernetes, с минимальной помощи преподавателя. Использование основных инструментов DevOps на хорошем уровне, возможны затруднения при решении сложных задач или работе с новыми инструментами. Задания выполнены в целом корректно, но возможны небольшие отклонения от требований.	*Базовый уровень*	*Хорошо*
Обладание базовыми знаниями о DevOps без понимания их взаимосвязи и практического применения. Теоретические ответы поверхностны. Задания выполнены частично или с существенными ошибками. Требуется значительная помощь преподавателя для выполнения задач по настройке CI/CD или работе с контейнерами. Умение использования только базовых функций и инструментов DevOps. Трудности при работе с автоматизацией инфраструктуры или мониторингом. Задания выполнены частично или содержат значительные ошибки.	*Пороговый*	*Удовлетворительно*
Отсутствует владение основными понятиями DevOps или демонстрация существенных пробелов в знаниях. Отсутствует способность выполнить задания даже с помощью преподавателя. Ошибки критически влияют на результат. Отсутствует владение необходимыми инструментами и технологиями для выполнения задач. Задания не выполнены или выполнены менее чем на половину.		*Неудовлетворительно*

20 Типовые оценочные средства и методические материалы, определяющие процедуры оценивания

20.1 Текущий контроль успеваемости

Текущая аттестация проводится в течение всего обучения в рамках освоения дисциплины и заключается в выполнении студентом пяти лабораторных работ, успешного прохождения тестовых заданий (не менее 50 % правильных ответов), а также успешного выполнения практических заданий (получение не менее 50 баллов из 100 баллов для каждого задания). Тестовые задания выполняются студентами после прослушивания лекций, практические задания выполняются после успешного написания теста. Лабораторные работы выполняются в течении отчетной сессии, но не позже итоговой аттестации. Контрольное задание 1 и контрольное задание 2 выполняются после предоставления отчетов по всем лабораторным работам.

Учебный процесс предусматривает две текущие аттестации. Результаты тестовых заданий и практических заданий учитываются преподавателей в качестве первой текущей аттестации. Отчеты лабораторных работ и результаты контрольных заданий учитываются преподавателем в качестве второй текущей аттестации.

Перечень заданий

Примеры тестовых заданий 1:

1. Что означает термин CI в DevOps?

a) Central Integration
b) Continuous Improvement
c) Continuous Integration
d) Code Injection

2. Какой инструмент используется для автоматической сборки и запуска пайплайнов?

a) Grafana
b) Jenkins
c) Docker
d) Nginx

3. Что такое контейнер в DevOps?

a) Виртуальная машина
b) Архив кода
c) Изолированная среда выполнения приложения
d) Сетевая топология

4. Какой файл описывает слои контейнера Docker?

a) docker.run
b) Dockerfile
c) docker-compose.yml
d) docker.conf

5. Что обеспечивает инструмент Ansible?

a) Мониторинг логов
b) Оркестрацию кластеров
c) Контейнеризацию
d) Автоматизацию конфигураций

Примеры тестовых заданий 2:

1. Какой тип деплоя позволяет откатиться без простоя при ошибке?

a) Rolling
b) Blue-Green
c) Manual
d) Shadow

2. Что такое Helm в контексте Kubernetes?

a) Контейнерный движок
b) Система логирования
c) Менеджер пакетов для Kubernetes
d) Средство мониторинга

3. Как называется процесс хранения инфраструктурных конфигураций в виде кода?

a) Code as Config
b) Infrastructure as Code
c) Virtual Infrastructure
d) Cloud Scripting

4. Что делает команда git merge?

a) Удаляет ветку
b) Создает коммит
c) Объединяет ветки
d) Скачивает изменения

5. Какой инструмент предназначен для сбора метрик и мониторинга?

a) Ansible
b) Docker
c) Prometheus
d) GitLab

Пример практических заданий 1

Вариант 1

Вам необходимо подготовить изменения в репозитории для их интеграции в основную ветку. Выполните следующие шаги:

- Создайте новую ветку feature/devops-task.
- Внесите изменения в файл README.md.
- Зафиксируйте изменения с комментарием "Updated README for DevOps task".
- Слейте ветку feature/devops-task с основной веткой master, разрешив возможные конфликты.

Вариант 2

Вам необходимо подготовить исправление багов в репозитории. Выполните следующие шаги:

- Создайте новую ветку bugfix/logging-fix.
- Измените файл logging.conf, добавив новый уровень логирования.
- Зафиксируйте изменения с комментарием: "Add debug level to logging config".
- Отправьте ветку в удалённый репозиторий и создайте merge request в ветку main.

Вариант 3

Настройте CI-пайплайн для автоматической проверки кода:

- Создайте .gitlab-ci.yml (или .github/workflows/ci.yml для GitHub).
- Добавьте этап lint, проверяющий код с помощью flake8.
- Настройте выполнение пайплайна при каждом push в ветку develop.

Пример практических заданий 2

Вариант 1

Настройте простой CI/CD пайплайн для веб-приложения на платформе GitLab CI:

- Создайте файл .gitlab-ci.yml с этапами сборки, тестирования и деплоя.
- Укажите шаги для установки зависимостей и запуска тестов (например, npm install и npm test).
- Настройте этап деплоя, чтобы приложение развертывалось на сервере через SSH.

Вариант 2

Подготовьте проект к запуску в контейнере:

- Создайте Dockerfile для Python-приложения.
- Настройте установку зависимостей из requirements.txt.
- Соберите образ и запустите его локально.

Вариант 3

Разверните приложение в Kubernetes с помощью Helm:

- Создайте Helm-чарт с названием my-app.
- Опишите конфигурацию deployment и service.
- Установите Helm-чарт в namespace dev.

Контрольные (практико-ориентированные) задания

Пример контрольного задания 1

1. Настройте автоматическое масштабирование приложения в Kubernetes для эффективного распределения вычислительных ресурсов.
2. Настройте пайплайн CI/CD, который автоматически запускает тесты и развёртывает приложение в тестовой среде после коммита в ветку develop.
3. Реализуйте развёртывание контейнеризированного приложения с помощью Docker Compose, включая базу данных и веб-сервер.
4. Настройте автоматическое масштабирование приложения в Kubernetes для эффективного распределения вычислительных ресурсов.
5. Создайте Helm-чарт для Kubernetes-приложения с параметризированной конфигурацией (реплика, имя образа, порты).

Пример контрольного задания 2

1. Реализуйте инфраструктуру как код с использованием Terraform для создания виртуальной машины с установленным Nginx.
2. Настройте мониторинг состояния приложения с использованием Prometheus и визуализацию метрик в Grafana.
3. Создайте playbook Ansible, который устанавливает и настраивает веб-сервер Apache на удалённой машине.
4. Настройте Blue-Green deployment для обновления приложения без простоев и потери доступности.
5. Реализуйте хранение чувствительных данных в переменных окружения с использованием .env файлов и исключением их из репозитория.

Контрольные, практические и тестовые задания оцениваются по 100-бальной шкале каждая.

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

ПК-4

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

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?
6. Файлы в Ansible называются...?
7. Инструмент для управления инфраструктурой как кодом?
8. Название процесса автоматического развертывания после тестов?
9. Где хранятся конфигурации Kubernetes?
10. Графическая система визуализации метрик?

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-5

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

6. Что обеспечивает балансировщик нагрузки в микросервисной архитектуре?

a) Равномерное распределение запросов между серверами
b) Хранение данных приложения
c) Автоматическое масштабирование баз данных
d) Шифрование сетевого трафика

7. Какая практика DevOps обеспечивает автоматическую проверку кода перед слиянием в основную ветку?

a) Continuous Integration (CI)
b) Continuous Deployment (CD)
c) Blue-Green Deployment
d) Infrastructure as Code

8. Что обеспечивает принцип «неизменяемой инфраструктуры»?

a) Частое изменение конфигурации серверов
b) Замена серверов вместо их обновления
c) Ручное управление ресурсами
d) Использование виртуальных машин

9. Какой инструмент используется для оркестрации контейнеров?

a) Docker Compose
b) Kubernetes
c) Ansible
d) GitLab CI

10. Что такое VPC (Virtual Private Cloud)?

a) Виртуальная сеть в облачном провайдере (AWS, GCP)
b) Инструмент для управления базами данных
c) Система мониторинга трафика
d) Механизм шифрования данных

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?
6. Как называется стратегия развертывания с двумя средами?
7. Как называется шаблонизатор для Kubernetes?
8. Процесс запуска множества контейнеров вместе?
9. Какой файл описывает Docker-среду?

10. Как называется автоматическое масштабирование подов в Kubernetes?

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?

20.2 Промежуточная аттестация

Промежуточная аттестация проводится в соответствии с Положением о промежуточной аттестации обучающихся по программам высшего образования и Положением о балльно-рейтинговой системе факультета компьютерных наук.

Форма контроля - Экзамен

Промежуточная аттестация проводится на основании итогов выполнения студентом лабораторных работ по всем темам (100% выполненных работ), успешного прохождения двух тестов (не менее 50% правильных ответов) и успешного выполнения практических и контрольных заданий (50 баллов из 100 баллов по всем видам заданий).

По итогам выполнения лабораторных работ, учета прохождения тестов, выполнений практических и контрольных заданий, а также устного ответа (собеседование со студентом в конце триместра по вопросам из перечня вопросов к экзамену и использования контрольно-измерительных материалов) студенту выставляется оценка «отлично», «хорошо», «удовлетворительно» или «не удовлетворительно» согласно оценочным средствам для проведения текущей и промежуточной аттестаций рабочей программы.

Перечень вопросов к экзамену:

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. Что такое Infrastructure as Code и зачем она нужна. Основы работы с Terraform: создание и управление инфраструктурой. Проблемы "configuration drift" и способы их решения. Основные подходы к мониторингу приложений в реальном времени. Инструменты для мониторинга и логирования: Prometheus, Grafana, ELK Stack. Как настраивать алерты для критических ошибок. Основы работы сетей в Linux: настройка IP, маршрутизация, DNS. Принципы работы NAT и VLAN. Основы информационной безопасности в DevOps: защита данных и контроль доступа.

Различия между IaaS, PaaS и SaaS. Использование облачных платформ (AWS, Google Cloud, Azure) для автоматизации DevOps-процессов. Что такое VPC (Virtual Private Cloud) и его компоненты.

Пример контрольно-измерительного материала

Направление подготовки 09.04.02 Информационные системы и технологии

Дисциплина _Технологии DevOps_

Курс___3___

Форма обучения ____заочная_

Вид аттестации __промежуточная_

Вид контроля ____экзамен___

Контрольно-измерительный материал № 1

1. Поясните преимущества использования Docker для разработки и развертывания приложений.
2. Как настраивать алерты для критических ошибок?
3. Приведите алгоритм настройки пайплайн CI/CD, который автоматически запускает тесты и развёртывает приложение в тестовой среде.