УТВЕРЖДАЮ
Генеральный директор
ООО «Мерион Нетворкс»
Тундайкина Т.Н.
_____
«01» апреля 2026 г.

Дополнительная профессиональная программа повышения квалификации

«DevOps-инженер»

г. Москва
2026

1. ОБЩИЕ ПОЛОЖЕНИЯ

1.1. Нормативные правовые основания разработки программы

Нормативную правовую основу разработки программы составляют:

- Федеральный закон РФ от 29 декабря 2012 г. № 273-ФЗ «Об образовании в Российской Федерации»;
- Приказ Министерства науки и высшего образования Российской Федерации от 24.03.2025 № 266 «Об утверждении Порядка организации и осуществления образовательной деятельности по дополнительным профессиональным программам»;
- Трудовой кодекс Российской Федерации от 30.12.2001 № 197-ФЗ;
- Приказ Министерства труда и социальной защиты РФ от 13.10.2014 № 716н «Об утверждении профессионального стандарта «Специалист по автоматизации и оркестрации IT-инфраструктуры»»;
- Профессиональный стандарт 06.001 «Программист», утвержденный приказом Министерства труда и социальной защиты РФ от 18.11.2013 № 679н;
- Приказ Министерства образования и науки Российской Федерации от 17 августа 2015 г. № 851 «Об утверждении федерального государственного образовательного стандарта высшего образования по направлению подготовки 09.03.04 Программная инженерия».

1.2. Цель и планируемые результаты обучения

Целью реализации программы является повышение профессионального уровня специалистов в области информационных технологий, направленное на совершенствование и получение ими новых компетенций в сфере DevOps-практик, необходимых для профессиональной деятельности по разработке, развертыванию и сопровождению программного обеспечения в современных IT-командах.

Данная программа соответствует требованиям профессионального стандарта 06.001 «Программист» и направлена на формирование компетенций специалиста уровня DevOpsинженер.

Профессиональные компетенции, качественное улучшение которых планируется по результатам освоения программы:

1. Управление инфраструктурой как кодом (Infrastructure as Code):

1. Проектирование и автоматизация инфраструктуры с использованием декларативных и императивных подходов.
2. Создание и управление инфраструктурой с помощью Terraform и Ansible.
3. Написание читаемого, масштабируемого кода IaC с использованием переменных, модулей и шаблонов.

2. Контейнеризация и управление образами:

4. Разработка и оптимизация Docker-образов, использование многоэтапной сборки.
5. Настройка многоконтейнерных приложений с Docker Compose.
6. Применение продвинутых техник работы с томами, сетями и безопасностью.

3. Оркестрация контейнерных нагрузок:

7. Развертывание и управление кластерами Kubernetes.
8. Применение Helm для установки и шаблонизации приложений.
9. Настройка сервисной сетки на базе Istio.

4. Построение CI/CD-конвейеров:

10. Использование Git как системы контроля версий.
11. Настройка автоматизированных пайплайнов в GitLab CI и Jenkins.
12. Организация процессов непрерывной интеграции, тестирования и доставки ПО.

5. Обеспечение безопасности инфраструктуры (DevSecOps): 13. Применение подходов Zero Trust, SASE и Defense in Depth.

14. Сканирование Docker-образов и кластеров на уязвимости.
15. Проведение автоматизированного анализа безопасности в рамках DevSecOps.

6. Мониторинг и наблюдаемость:

16. Развертывание и настройка систем мониторинга Zabbix, Prometheus, Grafana.
17. Настройка мониторинга приложений и инфраструктуры в Kubernetes-кластере.

В результате освоения программы повышения квалификации «DevOps-инженер с нуля» слушатель должен:

знать:

- принципы и культуру DevOps, историю возникновения и место в современных ITкомандах;
- архитектуру и принципы работы систем контейнеризации и оркестрации;
- инструменты автоматизации инфраструктуры (Terraform, Ansible);
- концепции CI/CD и архитектуру инструментов GitLab CI, Jenkins;
- современные подходы к обеспечению безопасности инфраструктуры и приложений;
- принципы построения систем мониторинга и наблюдаемости.

уметь:

- разворачивать и управлять инфраструктурой с помощью Terraform и Ansible;

- создавать, оптимизировать и запускать Docker-контейнеры и многоконтейнерные приложения;
- управлять кластерами Kubernetes, работать с Helm и Istio;
- настраивать CI/CD-пайплайны в GitLab CI и Jenkins;
- выявлять уязвимости в контейнерах и кластерах с помощью инструментов Trivy, Clair, Snyk, kube-hunter;
- развертывать и настраивать системы мониторинга Zabbix, Prometheus + Grafana.

владеть:

- практическими навыками работы в Linux-окружении (VirtualBox, WSL);
- навыками написания кода IaC и построения автоматизированных пайплайнов;
- навыками профессионального применения инструментов DevOps в рабочих сценариях.

1.3. Категория слушателей

К освоению дополнительных профессиональных программ допускаются лица старше 18 лет: 1) имеющие среднее профессиональное и (или) высшее образование; 2) получающие среднее профессиональное и (или) высшее образование (согласно части 4 статьи 76 Федерального закона от 29 декабря 2012 г. № 273-ФЗ «Об образовании в Российской Федерации»).

Приветствуется базовое знакомство с Linux-командной строкой и общее представление об IT-инфраструктуре. Предварительные знания в области DevOps не требуются.

1.4. Срок обучения

Трудоемкость обучения по данной программе – 200 академических часов учебной работы слушателя (в том числе 100 часов – лекционные занятия, 83 часа – практические занятия, 17 часов – самостоятельная работа и тестирование). Продолжительность учебного часа составляет 1 академический час (45 минут). Режим занятий: не более 8 академических часов в день.

1.5. Форма обучения

Форма обучения – дистанционная (исключительно с применением дистанционных образовательных технологий). Обучение проводится на платформе Merion Academy (https://merionet.ru) с предоставлением доступа к видеолекциям, практическим заданиям и тестовым материалам.

2. СТРУКТУРА И СОДЕРЖАНИЕ ПРОГРАММЫ

Структура и содержание программы представлены учебным планом, календарным учебным графиком и рабочими программами по учебным модулям.

2.1. Учебный план

Вид образования – дополнительное образование.

Подвид – дополнительное профессиональное образование.

Программа – повышение квалификации.

Наименование – «DevOps-инженер с нуля».

Категория обучающихся – лица, имеющие среднее профессиональное и (или) высшее образование; лица, получающие среднее профессиональное и (или) высшее образование старше 18 лет.

Срок обучения – 5 недель.

Форма обучения – дистанционная.

Режим занятий – до 8 академических часов в день.

Наименование модулей	Всего часов	Теорет. занятия	Практич. занятия	Самост. работа	Тест.	Итого	Форма контроля
Модуль 1. Введение в DevOps	8	7	0	0	1	8	Тестирование
Модуль 2. Infrastructure as Code (IaC)	32	21	10	0	1	32	Тестирование
Модуль 3. Контейнеризация	38	19	18	0	1	38	Тестирование
Модуль 4. Системы оркестрации	38	20	17	0	1	38	Тестирование
Модуль 5. CI/CD	30	15	13	0	2	30	Тестирование
Модуль 6. Безопасность (DevSecOps)	29	14	13	0	2	29	Тестирование
Модуль 7. Мониторинг	20	9	10	0	1	20	Тестирование
Финальное тестирование	5	0	0	0	5	5	Тестирование
ИТОГО	**200**	105	81	0	14	**200**	

2.2. Календарный учебный график

Календарный учебный график определяет количество учебных недель в соответствии с трудоемкостью и сроком освоения программы. Дата начала и окончания обучения устанавливаются по мере комплектации групп в течение всего календарного года.

Неделя/День	Нед. 1 Д.1	Нед.1 Д.2	Нед.1 Д.3	Нед.1 Д.4	Нед. 1 Д.5	Нед.2 Д.1	Нед.2 Д.2	Нед. 2 Д.3	...	Нед. 5 Д.5	Итого
Кол-во часов	8	8	8	8	8	8	8	8	...	8	200

Вид занятий	ТО	ТО/ПЗ	ТО/ПЗ	ТО/ПЗ	К	ТО/ПЗ	ТО/ПЗ	ПЗ	...	ИЭ	

ТО – теоретическое обучение; ПЗ – практическое занятие; К – консультация; ИЭ – итоговый экзамен.

2.3. Рабочие программы учебных модулей

Модуль 1. Введение в DevOps

Тема 1. Что такое DevOps и история его развития

Цели и принципы DevOps как подхода к разработке и эксплуатации ПО. Исторические предпосылки появления DevOps: Agile-манифест, системное мышление, движение Lean. Противоречие между командами разработки и эксплуатации. DevOps в современных IT-командах. Отличие DevOps от традиционных моделей разработки. Ключевые практики и ценности DevOps. Преимущества для бизнеса и процессов разработки. Обзор инструментального стека DevOps-инженера.

Тема 2. Промежуточное тестирование по модулю

Итоговое тестирование по блоку «Введение в DevOps».

Наименование тем	Всего часов	Теор. занятия	Практ. занятия	Самост. работа	Тестиров.	Форма контроля
Тема 1. Что такое DevOps и история его развития	7	7	0	0	0	
Тема 2. Тестирование по модулю	1	0	0	0	1	Тестирование
ИТОГО	**8**	7	0	0	1	

Модуль 2. Infrastructure as Code (IaC)

Тема 1. Введение в IaC. Средства автоматизации Terraform и Ansible. Декларативный и императивный подходы

Ключевые принципы Infrastructure as Code. Декларативный vs. императивный подходы. Обзор инструментов: Terraform и Ansible. Архитектура Terraform: провайдеры, состояние, план. Архитектура Ansible: инвентаризация, плейбуки, роли, модули. Настройка рабочей среды: установка ОС Linux в VirtualBox и WSL.

Тема 2. Работа с IaC через Terraform и Ansible

Практическая работа с Terraform: описание инфраструктуры, применение планов, управление состоянием. Практическая работа с Ansible: написание плейбуков, использование переменных, ролей и шаблонов Jinja2. Совместное использование Terraform и Ansible. Масштабируемый и читаемый код IaC. Практические задания: установка Ansible и Terraform, настройка Linux-среды, работа с модулями и ролями.

Тема 3. Тестирование по модулю «Infrastructure as Code»

Наименование тем	Всего часов	Теор. занятия	Практ. занятия	Самост. работа	Тестиров.	Форма контроля
Тема 1. Введение в IaC. Terraform и Ansible	12	12	0	0	0	

Наименование тем	Всего часов	Теор. занятия	Практ. занятия	Самост. работа	Тестиров.	Форма контроля
Тема 2. Практическая работа с IaC через Terraform и Ansible	19	9	10	0	0	
Тема 3. Тестирование по модулю	1	0	0	0	1	Тестирование
ИТОГО	**32**	21	10	0	1	

Модуль 3. Контейнеризация

Тема 1. Виртуализация и контейнеризация. Введение в Docker

Различия между виртуализацией и контейнеризацией. Архитектура Docker: daemon, образ, контейнер, реестр. Основные команды Docker. Установка Docker и Docker Compose. Запуск базы данных MariaDB и интерфейса Adminer в контейнерах.

Тема 2. Работа с данными и сетями в Docker. Docker Compose

Тома (volumes) и монтирование: bind mount, named volumes, tmpfs. Сетевые модели Docker: bridge, host, overlay, macvlan. Docker Compose: синтаксис, зависимости сервисов, настройка сетей. Развертывание многоконтейнерного проекта с настройкой сетей.

Тема 3. Сборка и оптимизация Docker Images. Multistaging

Dockerfile: инструкции, лучшие практики, кэширование слоев. Многоэтапная сборка (multi-stage build) для минимизации размера образов. Анализ и оптимизация образов. Практика по оптимизации Docker Images и многоэтапной сборке.

Тема 4. Тонкости и нюансы Docker. Углубленное изучение

Непривилегированные пользователи в контейнерах. Интеграция Docker с Ansible. Работа с bridge-сетями. Продвинутые техники управления volumes. Безопасность Dockerконтейнеров. Тестирование по блоку «Контейнеризация».

Наименование тем	Всего часов	Теор. занятия	Практ. занятия	Самост. работа	Тестиров.	Форма контроля
Тема 1. Виртуализация и контейнеризация. Введение в Docker	10	5	4	0	0	
Тема 2. Работа с данными и сетями в Docker. Docker Compose	10	5	5	0	0	
Тема 3. Сборка и оптимизация Docker Images. Multistaging	9	4	5	0	0	

Тема 4. Тонкости и нюансы Docker. Углубленное изучение	8	5	4	0	0	
Наименование тем	**Всего часов**	**Теор. занятия**	**Практ. занятия**	**Самост. работа**	**Тестиров.**	**Форма контроля**
Тема 5. Тестирование по модулю	1	0	0	0	1	Тестирование
ИТОГО	**38**	19	18	0	1	

Модуль 4. Системы оркестрации

Тема 1. Сравнительный анализ оркестраторов. Знакомство с Docker Swarm

Обзор оркестраторов контейнерных и неконтейнерных нагрузок. Сравнение Kubernetes, Docker Swarm, Nomad, Mesos. Работа с Docker Swarm: создание кластера, сервисы, стеки. Работа с etcd-хранилищем.

Тема 2. Введение в Kubernetes. Компоненты Control Plane и Data Plane

Архитектура Kubernetes: control plane (kube-apiserver, etcd, scheduler, controllermanager) и data plane (kubelet, kube-proxy, container runtime). Объекты Kubernetes: Pod, Node, Namespace. Установка и настройка кластера.

Тема 3. Deployment и ReplicaSet. Работа с данными (PV/PVC)

Объекты Kubernetes: Deployment, ReplicaSet, StatefulSet, DaemonSet. Стратегии обновления. Persistent Volumes (PV), Persistent Volume Claims (PVC), StorageClass. Развертывание приложений в Kubernetes с использованием Deployment и ReplicaSet.

Тема 4. Работа с шаблонизатором Helm

Архитектура Helm: chart, release, репозиторий. Создание собственных Helm Chart. Использование шаблонизатора для управления конфигурациями. Практика: установка приложения через Helm.

Тема 5. Сервисная сетка. Service Mesh на базе Istio

Концепция service mesh. Архитектура Istio: control plane (Pilot, Citadel, Galley) и data plane (Envoy sidecar). Управление трафиком, наблюдаемость, mTLS. Практика: настройка сервисной сетки на базе Istio. Тестирование по модулю.

Наименование тем	**Всего часов**	**Теор. занятия**	**Практ. занятия**	**Самост. работа**	**Тестиров.**	**Форма контроля**
Тема 1. Сравнительный анализ оркестраторов. Docker Swarm	8	5	3	0	0	

Тема 2. Введение в Kubernetes. Компоненты плоскостей управления и данных	7	4	3	0	0	
Наименование тем	**Всего часов**	**Теор. занятия**	**Практ. занятия**	**Самост. работа**	**Тестиров.**	**Форма контроля**
Тема 3. Deployment и ReplicaSet. Работа с данными (PV/PVC)	10	4	6	0	0	
Тема 4. Работа с шаблонизатором Helm	7	4	3	0	0	
Тема 5. Сервисная сетка на базе Istio	5	3	2	0	0	
Тема 6. Тестирование по модулю	1	0	0	0	1	Тестирование
ИТОГО	**38**	20	17	0	1	

Модуль 5. CI/CD

Тема 1. Система контроля версий. Знакомство с Git

Концепции системы контроля версий. Основные команды Git: init, clone, add, commit, push, pull, merge, rebase, branch. Стратегии ветвления: GitFlow, Trunk-Based Development. Практика: работа с основными командами Git.

Тема 2. CI/CD-конвейер. Знакомство с GitLab CI и Jenkins

Концепции непрерывной интеграции и доставки (CI/CD). Архитектура и компоненты GitLab CI: runner, pipeline, job, stage. Архитектура Jenkins: master, agent, плагины. Развертывание GitLab и настройка GitLab CI. Установка и настройка Jenkins.

Тема 3. CI/CD-конвейер. Построение пайплайнов

Построение пайплайнов для автоматической сборки, тестирования и доставки продукта. .gitlab-ci.yml: структура, переменные, артефакты, кэш, триггеры. Jenkinsfile: декларативный и скриптовый синтаксис. Создание пайплайнов в GitLab CI и Jenkins. Тестирование по модулю.

Наименование тем	**Всего часов**	**Теор. занятия**	**Практ. занятия**	**Самост. работа**	**Тестиров.**	**Форма контроля**
Тема 1. Система контроля версий. Знакомство с Git	10	5	5	0	0	
Тема 2. CI/CD-конвейер. GitLab CI и Jenkins	10	5	4	0	0	
Тема 3. Построение пайплайнов	8	5	4	0	0	

Тема 4. Тестирование по модулю	2	0	0	0	2	Тестирование
ИТОГО	**30**	15	13	0	2	

Модуль 6. Безопасность (DevSecOps)

Тема 1. Безопасность инфраструктуры. ZTNA, SASE, Defense in Depth

Современные модели сетевой безопасности: Zero Trust Network Access (ZTNA), Secure Access Service Edge (SASE), Defense in Depth (DiD). Принципы минимальных привилегий, сегментации и непрерывной верификации.

Тема 2. Основные подходы по обеспечению безопасности Docker

Векторы атак на Docker-контейнеры. Сканирование образов с помощью Trivy, Clair и Snyk. Сравнение результатов инструментов анализа. Hardening Docker: пользователи, capabilities, read-only FS, seccomp.

Тема 3. Основные подходы по обеспечению безопасности Kubernetes

Модель безопасности Kubernetes: RBAC, Network Policies, Pod Security Standards. Сканирование кластера с помощью kube-hunter. Практика по анализу и устранению уязвимостей.

Тема 4. Введение в DevSecOps. Виды анализа и типы проверок

Принципы DevSecOps: встраивание безопасности в CI/CD. Статический (SAST) и динамический (DAST) анализ. Анализ состава ПО (SCA). Тестирование по модулю.

Наименование тем	Всего часов	Теор. занятия	Практ. занятия	Самост. работа	Тестиров.	Форма контроля
Тема 1. Безопасность инфраструктуры. ZTNA, SASE, Defense in Depth	8	4	4	0	0	
Тема 2. Безопасность Docker	7	3	4	0	0	
Тема 3. Безопасность Kubernetes	7	4	3	0	0	
Тема 4. Введение в DevSecOps. Виды анализа	5	3	2	0	0	
Тема 5. Тестирование по модулю	2	0	0	0	2	Тестирование
ИТОГО	**29**	14	13	0	2	

Модуль 7. Мониторинг

Тема 1. Введение в мониторинг. Модели и принципы работы систем мониторинга

Основные модели мониторинга: push vs. pull. Метрики, логи, трейсы (три кита наблюдаемости). Архитектура Zabbix: сервер, агент, прокси. Развертывание Zabbix. Архитектура Prometheus: экспортеры, scrape-конфигурация, alertmanager. Grafana: дашборды, datasource, alerting. Стек Prometheus + Grafana: установка и настройка. Мониторинг в Kubernetes: kube-state-metrics, metrics-server, node-exporter. Настройка мониторинга приложений в Kubernetes. Тестирование по модулю.

Наименование тем	Всего часов	Теор. занятия	Практ. занятия	Самост. работа	Тестиров.	Форма контроля
Тема 1. Введение в мониторинг. Модели и принципы работы систем	18	9	9	0	0	
Тема 2. Практика: Zabbix, Prometheus + Grafana, мониторинг в Kubernetes	0	0	1	0	0	
Тема 3. Тестирование по модулю	1	0	0	0	1	Тестирование
ИТОГО	**20**	9	10	0	1	

Финальное тестирование

Обобщающее тестирование по всем модулям курса. Проверка остаточных знаний, закрепление ключевых тем. К итоговой аттестации допускаются слушатели, освоившие учебный план в полном объеме и прошедшие промежуточную аттестацию по всем модулям.

Наименование	Всего часов	Теор. занятия	Практ. занятия	Самост. работа	Тестиров.	Форма контроля
Финальное тестирование	5	0	0	0	5	Тестирование
ИТОГО	**5**	0	0	0	5	

3. ОРГАНИЗАЦИОННО-ПЕДАГОГИЧЕСКИЕ УСЛОВИЯ РЕАЛИЗАЦИИ ПРОГРАММЫ

3.1. Требования к квалификации педагогических кадров

Организационно-педагогические условия реализации Программы должны обеспечивать ее реализацию в полном объеме, соответствие качества подготовки обучающихся установленным требованиям. Продолжительность учебного часа занятий составляет 1 академический час (45 минут).

Преподаватели должны иметь высшее образование или среднее профессиональное образование по направлению подготовки в области информационных технологий, программной инженерии, автоматизации, а также практический опыт работы в качестве DevOps-инженера, системного администратора, разработчика или архитектора ITинфраструктуры не менее 3 лет. Допускается наличие сертификатов профессионального уровня (CKA, CKS, HashiCorp Certified, AWS/GCP/Azure DevOps).

3.2. Требования к материально-техническим условиям

Требования к оснащению: оборудованное рабочее место преподавателя с компьютером и выходом в сеть Интернет. Для проведения практических занятий слушателям необходим персональный компьютер или ноутбук с доступом в Интернет. Материально-техническое оснащение:

- стол – 1 шт.;
- стул – 1 шт.;
- ноутбук со встроенными динамиками и доступом в сеть Интернет – 1 шт.; – операционная система – Windows 10/11 или Linux (Ubuntu 22.04+) или MacOS; – программное обеспечение для виртуализации – VirtualBox 7.0+.

3.3. Требования к информационным и учебно-методическим условиям

Методическое обеспечение образовательной программы включает:

- видеолекции по каждой теме в формате screencast с демонстрацией рабочего окружения;
- практические задания в формате повтора за автором с пошаговыми инструкциями;
- тестовые задания для промежуточной и итоговой аттестации;
- учебно-методические материалы: конспекты, схемы, справочники команд; – доступ к платформе Merion Academy на два года с момента приобретения курса.

3.4. Общие требования к организации образовательного процесса

К освоению программы допускаются лица, имеющие среднее профессиональное и (или) высшее образование; лица, получающие среднее профессиональное и (или) высшее образование старше 18 лет.

В процессе обучения особое внимание уделяется практическому освоению инструментов DevOps в реальных рабочих сценариях: участники самостоятельно настраивают инфраструктуру, работают с принципами инфраструктура как код, строят

пайплайны и анализируют уязвимости. В результате обучения слушатели приобретают знания, навыки и практические умения, необходимые для работы в роли DevOps-инженера.

4. ОЦЕНКА РЕЗУЛЬТАТОВ ОСВОЕНИЯ ПРОГРАММЫ

Во время обучения проводится промежуточная аттестация обучающихся в форме тестирования по каждому модулю.

Промежуточная аттестация.

Для самоконтроля знаний слушателям по результатам освоения материалов каждого модуля предлагается пройти тест. Тест считается успешно пройденным при предоставлении более 80% правильных ответов. Количество попыток не ограничено. Результаты тестов учитываются при допуске к итоговому тестированию.

Итоговое тестирование.

К итоговому тестированию допускаются слушатели, освоившие учебный план в полном объеме и прошедшие промежуточную аттестацию по всем модулям. На прохождение итогового тестирования отводится 90 минут (2 академических часа). Тест считается успешно пройденным при предоставлении более 80% правильных ответов. Количество попыток неограничено.

Наименование модулей	Форма промежуточной аттестации	Методы контроля
Модуль 1. Введение в DevOps	Тестирование	Тестирование
Модуль 2. Infrastructure as Code (IaC)	Тестирование	Тестирование
Модуль 3. Контейнеризация	Тестирование	Тестирование
Модуль 4. Системы оркестрации	Тестирование	Тестирование
Модуль 5. CI/CD	Тестирование	Тестирование
Модуль 6. Безопасность (DevSecOps)	Тестирование	Тестирование
Модуль 7. Мониторинг	Тестирование	Тестирование
Итоговое тестирование	Тестирование	Тестирование

Критерии оценки промежуточной и итоговой аттестации в форме тестирования

Процент результативности (правильных ответов)	Качественная оценка образовательных достижений
80 – 100%	зачет

менее 80%	незачет

Лицам, успешно освоившим программу повышения квалификации и прошедшим итоговую аттестацию курса **на тарифе с наставником**, выдается удостоверение о повышении квалификации установленного образца.

5. ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ПРОГРАММЫ

Основная литература и документация:

18. Хасикова Л., Моуат Д. Kubernetes в действии. – М.: ДМК Пресс, 2019.
19. Хашимото М. Terraform: инфраструктура на уровне кода. – М.: ДМК Пресс, 2018.
20. Turnbull J. The Docker Book. – Lulu.com, 2021.
21. Hüttermann M. DevOps for Developers. – Apress, 2012.
22. Kim G., Behr K., Spafford G. Проект «Феникс». – М.: Манн, Иванов и Фербер, 2018.

Электронные источники и официальная документация:

- Официальная документация Kubernetes – https://kubernetes.io/docs/
- Официальная документация Docker – https://docs.docker.com/
- Официальная документация Terraform – https://developer.hashicorp.com/terraform/docs
- Официальная документация Ansible – https://docs.ansible.com/
- Официальная документация GitLab CI – https://docs.gitlab.com/ee/ci/
- Официальная документация Jenkins – https://www.jenkins.io/doc/
- Официальная документация Prometheus – https://prometheus.io/docs/
- Официальная документация Grafana – https://grafana.com/docs/
- Официальная документация Helm – https://helm.sh/docs/
- Официальная документация Istio – https://istio.io/latest/docs/
- Платформа Merion Academy – https://lms.merionet.ru/login/index.php