ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ

Основы методологии Development Operation

Код модуля	Модуль
1163893(1)	Основы методологии Development Operation

Екатеринбург

Оценочные материалы составлены автором(ами):

№ п/п	Фамилия, имя, отчество	Ученая степень, ученое звание	Должность	Подразделение
1	Лавров Владислав Васильевич	доктор технических наук, доцент	Профессор	теплофизики и информатики в металлургии

Согласовано:

Управление образовательных программ Е.А. Смирнова

Авторы:

- **Лавров Владислав Васильевич, Профессор, теплофизики и информатики в металлургии**

1. СТРУКТУРА И ОБЪЕМ ДИСЦИПЛИНЫ Основы методологии Development Operation

1.	**Объем дисциплины в зачетных единицах**	4	
2.	**Виды аудиторных занятий**	Лекции Практические/семинарские занятия	
3.	**Промежуточная аттестация**	Зачет	
4.	**Текущая аттестация**	Домашняя работа	1

2. ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ (ИНДИКАТОРЫ) ПО ДИСЦИПЛИНЕ МОДУЛЯ Основы методологии Development Operation

Индикатор – это признак / сигнал/ маркер, который показывает, на каком уровне обучающийся должен освоить результаты обучения и их предъявление должно подтвердить факт освоения предметного содержания данной дисциплины, указанного в табл. 1.3 РПМ-РПД.

Таблица 1

Код и наименование компетенции	**Планируемые результаты обучения (индикаторы)**	**Контрольно-оценочные средства для оценивания достижения результата обучения по дисциплине**
1	**2**	**3**
ПК-17 -Способность оценивать качество программного обеспечения, проводить тестирование и исследование результатов.	З-1 - Перечислить методы и средства верификации работоспособности выпусков программных продуктов, разработки процедур для развертывания программного обеспечения, миграции и преобразования данных, интерфейсов взаимодействия с внешней средой, интерфейсов взаимодействия внутренних модулей системы с учетом возможностей языков, утилит и сред программирования, средств пакетного выполнения процедур З-2 - Перечислить основные показатели качества, методы и средства обеспечения и	Домашняя работа Зачет Лекции Практические/семинарские занятия

	контроля качества программного обеспечения. П-1 - Выполнить работы по верификации работоспособности выпусков программных продуктов, созданию процедур для развертывания программного обеспечения, миграции и преобразования данных, интерфейсов взаимодействия с внешней средой, интерфейсов взаимодействия внутренних модулей системы с учетом возможностей языков, утилит и сред программирования, средств пакетного выполнения процедур П-2 - Выполнить программную реализацию тестов для проверки качества программного обеспечения П-3 - Выполнить работы по определению и описанию тестовых случаев, анализу результатов тестирования программного продукта У-1 - Определять последовательность действий по проверке работоспособности версий программного продукта У-2 - Выбирать стратегию тестирования и перечень документов по управлению качеством программного обеспечения. У-3 - Проводить анализ результатов тестирования программного продукта	
ПК-19 -Способность создавать техническую документацию на продукцию в сфере информационных технологий, управления технической информацией.	З-1 - Определить список нормативных документов, регламентирующих оформление технических, методических, рекламных (маркетинговых) материалов З-2 - Перечислить способы разработки эксплуатационных пользовательских документов, а также стандартных технических документов на продукцию в	Домашняя работа Зачет Практические/семинарские занятия

	сфере информационных технологий. П-1 - Разрабатывать с применением технических стандартов эксплуатационные пользовательские документы и комплекты технической документации на продукцию в сфере информационных технологий на основе предоставленного материала. П-2 - Разработать электронную справочную систему на программный продукт в заданном стандартном формате. У-1 - Определять последовательность разработки эксплуатационных пользовательских документов, а также комплектов стандартной технической документации на продукцию в сфере информационных технологий на основе предоставленного материала У-2 - Определить последовательность разработки документов информационно-маркетингового назначения на продукцию в сфере информационных технологий	

3. ПРОЦЕДУРЫ КОНТРОЛЯ И ОЦЕНИВАНИЯ РЕЗУЛЬТАТОВ ОБУЧЕНИЯ В РАМКАХ ТЕКУЩЕЙ И ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ПО ДИСЦИПЛИНЕ МОДУЛЯ В БАЛЬНО-РЕЙТИНГОВОЙ СИСТЕМЕ (ТЕХНОЛОГИЧЕСКАЯ КАРТА БРС)

3.1. Процедуры текущей и промежуточной аттестации по дисциплине

1. **Лекции**: **коэффициент значимости совокупных результатов лекционных занятий – 0.5**		
Текущая аттестация на лекциях	**Сроки – семестр, учебная неделя**	**Максимальная оценка в баллах**
Домашняя работа	7,16	*60*
Активность работы на лекциях	7,16	*40*
Весовой коэффициент значимости результатов текущей аттестации по лекциям – 0.5		
Промежуточная аттестация по лекциям – зачет		

Весовой коэффициент значимости результатов промежуточной аттестации по лекциям – 0.5		
2. Практические/семинарские занятия: коэффициент значимости совокупных результатов практических/семинарских занятий – 0.5		
Текущая аттестация на практических/семинарских занятиях	**Сроки – семестр, учебная неделя**	**Максимальная оценка в баллах**
Отчет по практическим работам	7,16	100
Весовой коэффициент значимости результатов текущей аттестации по практическим/семинарским занятиям– 1		
Промежуточная аттестация по практическим/семинарским занятиям–зачет **Весовой коэффициент значимости результатов промежуточной аттестации по практическим/семинарским занятиям– не предусмотрено**		
3. Лабораторные занятия: коэффициент значимости совокупных результатов лабораторных занятий –не предусмотрено		
Текущая аттестация на лабораторных занятиях	**Сроки – семестр, учебная неделя**	**Максимальная оценка в баллах**
Весовой коэффициент значимости результатов текущей аттестации по лабораторным занятиям -не предусмотрено		
Промежуточная аттестация по лабораторным занятиям –нет **Весовой коэффициент значимости результатов промежуточной аттестации по лабораторным занятиям – не предусмотрено**		
4. Онлайн-занятия: коэффициент значимости совокупных результатов онлайн-занятий –не предусмотрено		
Текущая аттестация на онлайн-занятиях	**Сроки – семестр, учебная неделя**	**Максимальная оценка в баллах**
Весовой коэффициент значимости результатов текущей аттестации по онлайн-занятиям -не предусмотрено		
Промежуточная аттестация по онлайн-занятиям –нет **Весовой коэффициент значимости результатов промежуточной аттестации по онлайн-занятиям – не предусмотрено**		

3.2. Процедуры текущей и промежуточной аттестации курсовой работы/проекта

Текущая аттестация выполнения курсовой работы/проекта	**Сроки − семестр, учебная неделя**	**Максимальная оценка в баллах**
Весовой коэффициент текущей аттестации выполнения курсовой работы/проекта– не предусмотрено		
Весовой коэффициент промежуточной аттестации выполнения курсовой работы/проекта– защиты – не предусмотрено		

4. КРИТЕРИИ И УРОВНИ ОЦЕНИВАНИЯ РЕЗУЛЬТАТОВ ОБУЧЕНИЯ ПО ДИСЦИПЛИНЕ МОДУЛЯ

4.1. В рамках БРС применяются утвержденные на кафедре/институте критерии (признаки) оценивания достижений студентов по дисциплине модуля (табл. 4) в рамках контрольно-оценочных мероприятий на соответствие указанным в табл.1 результатам обучения (индикаторам).

Таблица 4

Критерии оценивания учебных достижений обучающихся

Результаты обучения	**Критерии оценивания учебных достижений, обучающихся на соответствие результатам обучения/индикаторам**
Знания	Студент демонстрирует знания и понимание в области изучения на уровне указанных индикаторов и необходимые для продолжения обучения и/или выполнения трудовых функций и действий, связанных с профессиональной деятельностью.
Умения	Студент может применять свои знания и понимание в контекстах, представленных в оценочных заданиях, демонстрирует освоение умений на уровне указанных индикаторов и необходимых для продолжения обучения и/или выполнения трудовых функций и действий, связанных с профессиональной деятельностью.
Опыт /владение	Студент демонстрирует опыт в области изучения на уровне указанных индикаторов.
Другие результаты	Студент демонстрирует ответственность в освоении результатов обучения на уровне запланированных индикаторов. Студент способен выносить суждения, делать оценки и формулировать выводы в области изучения. Студент может сообщать преподавателю и коллегам своего уровня собственное понимание и умения в области изучения.

4.2 Для оценивания уровня выполнения критериев (уровня достижений обучающихся при проведении контрольно-оценочных мероприятий по дисциплине модуля) используется универсальная шкала (табл. 5).

Таблица 5

Шкала оценивания достижения результатов обучения (индикаторов) по уровням

Характеристика уровней достижения результатов обучения (индикаторов)				
№ п/п	**Содержание уровня выполнения критерия оценивания результатов обучения (выполненное оценочное задание)**	**Шкала оценивания**		
		Традиционная характеристика уровня		**Качественная характеристика уровня**
1.	Результаты обучения (индикаторы) достигнуты в полном объеме, замечаний нет	Отлично (80-100 баллов)	Зачтено	Высокий (В)
2.	Результаты обучения (индикаторы) в целом достигнуты, имеются замечания, которые не требуют обязательного устранения	Хорошо (60-79 баллов)		Средний (С)

3.	Результаты обучения (индикаторы) достигнуты не в полной мере, есть замечания	Удовлетворительно (40-59 баллов)		Пороговый (П)
4.	Освоение результатов обучения не соответствует индикаторам, имеются существенные ошибки и замечания, требуется доработка	Неудовлетворительно (менее 40 баллов)	Не зачтено	Недостаточный (Н)
5.	Результат обучения не достигнут, задание не выполнено	Недостаточно свидетельств для оценивания		Нет результата

5. СОДЕРЖАНИЕ КОНТРОЛЬНО-ОЦЕНОЧНЫХ МЕРОПРИЯТИЙ ПО ДИСЦИПЛИНЕ МОДУЛЯ

5.1. Описание аудиторных контрольно-оценочных мероприятий по дисциплине модуля

5.1.1. Лекции

Самостоятельное изучение теоретического материала по темам/разделам лекций в соответствии с содержанием дисциплины (п. 1.2. РПД)

5.1.2. Практические/семинарские занятия

Примерный перечень тем

1. Основы администрирования ОС.
2. Системы и сети передачи данных.
3. Системы хранения данных, типы и особенности.
4. Администрирование баз данных.
5. Масштабируемость и отказоустойчивость.
6. Мониторинг, логирование и оповещение событий.
7. Виртуализация в DevOps.
8. Облачные решения.
9. Работа с Terraform. Управление облачной инфраструктурой.
10. Конфигурационное управление. IaC.
11. Системы контроля версий. Распределённая система управления версиями Git.
12. Жизненный цикл ПО.
13. Практические навыки работы с Docker.
14. Микросервисы и микросервисная архитектура.
15. Оркестровка контейнеров, кластеры Kubernetes.
16. Kubernetes конфигурация развертывания.
17. Планирование безопасности для кластера Kubernetes.

Примерные задания

Студенты должны самостоятельно выполнить все нижеперечисленные задания.

7.1. Сборка и запуск контейнера в Docker

Этап 1.1. *Установить Docker на свою операционную систему и выполнить проверку установки*

Чтобы установить Docker на свою операционную систему, необходимо перейти на официальный сайт Docker (https://www.docker.com/) и ознакомиться с инструкцией по установке для различных операционных систем. Выберите инструкции, соответствующие вашей ОС, и следуйте им. После завершения установки, проверьте, что Docker установлен правильно, выполнив команду docker --version в командной строке или терминале.

Этап 1.2. *Разработать тестовое приложение Visual Studio .NET Core*

Для начала необходимо убедиться, что на рабочем компьютере установлена Visual Studio и .NET Core SDK. Затем создайте новый проект в Visual Studio и выберите шаблон, соответствующий типу приложения, который необходимо разработать – веб-приложение ASP.NET Core MVC. Напишите программный код приложения «Калькулятор» – простой калькулятор с веб-интерфейсом, который будет выполнять основные арифметические операции (сложение, разность, умножение и деление) и отображать результат. В Приложении А представлены примеры листинга программного кода контроллера CalculatorController.cs, представления Index.cshtml, файла appsettings.json. Обратите внимание, что каждый студент должен указать индивидуальный номер порта, на котором будет работать приложение. Номер порта должен соответствовать порядковому номеру из студенческого журнала.

На главной странице веб-приложения, в футере должны отображаться ФИО и номер группы студента.

Выполните сборку и протестируйте работу приложения, чтобы убедиться, что оно выполняет требуемые функции.

Этап 1.2. *Разработать тестовое приложение Visual Studio .NET Core*

Для начала необходимо убедиться, что на рабочем компьютере установлена Visual Studio и .NET Core SDK. Затем создайте новый проект в Visual Studio и выберите шаблон, соответствующий типу приложения, который необходимо разработать – веб-приложение ASP.NET Core MVC. Напишите программный код приложения «Калькулятор» – простой калькулятор с веб-интерфейсом, который будет выполнять основные арифметические операции (сложение, разность, умножение и деление) и отображать результат. В Приложении А представлены примеры листинга программного кода контроллера CalculatorController.cs, представления Index.cshtml, файла appsettings.json. Обратите внимание, что каждый студент должен указать индивидуальный номер порта, на котором будет работать приложение. Номер порта должен соответствовать порядковому номеру из студенческого журнала.

На главной странице веб-приложения, в футере должны отображаться ФИО и номер группы студента.

Выполните сборку и протестируйте работу приложения, чтобы убедиться, что оно выполняет требуемые функции.

Этап 1.3. *Создать контейнер с использованием Docker для тестового приложения ASP.NET Core Web Application, определить и настроить окружение контейнера с помощью Dockerfile*

В качестве тестового приложения используйте созданный проект «Калькулятор» – простой калькулятор с веб-интерфейсом, который выполняет основные арифметические операции (сложение, разность, умножение и деление) и отображает результат.

Сначала необходимо создать файл с именем "Dockerfile" в корневой папке тестового проекта. Для этого откройте текстовый редактор и добавьте следующий код в файл:

```
# Используем базовый образ с ASP.NET 6.0
FROM mcr.microsoft.com/dotnet/aspnet:6.0 AS base

# Устанавливаем рабочую директорию внутри контейнера
WORKDIR /app

# Используем SDK
FROM mcr.microsoft.com/dotnet/sdk:6.0 AS build

# Копируем файлы в контейнер
COPY . /src
```

Этап 1.4. *Разместить приложение в системе контроля версий GitHub*

После того, как тестовое приложение будет работать без ошибок, создайте личный репозиторий на GitHub. Для этого перейдите на сайт GitHub (https://github.com/), войдите в свой аккаунт (или зарегистрируйтесь, если аккаунта нет) и создайте новый репозиторий. Затем, добавьте тестовый проект в Git и привяжите его к личному репозиторию на GitHub.

Этап 1.5. *Выполнить сборку образа с использованием команды docker build*

Сначала в терминальном режиме на удаленном сервере необходимо клонировать проект из личного репозитория GitHub. Для этого выполнить команду:

```
git clone <HTTPS>
```

Здесь <HTTPS> – это HTTPS-адрес проекта в GitHub.

Далее необходимо осуществить сборку Docker-образа. Откройте командную строку или терминал в папке с вашим Dockerfile, затем выполните команду для сборки Docker-образа:

```
docker build -t <test-web-app-image>:latest .
```

Здесь <test-web-app-image> – это имя образа. Вы можете заменить его на любое удобное вам имя.

Проверьте существование созданного образа в списке активных контейнеров и их идентификаторов с помощью команды:

```
docker ps -a
```

Этап 1.6. *Создать контейнер с использованием Docker-образа и выполнить его*

Для создания контейнера с использованием Docker-образа воспользуйтесь командой docker run:

```
docker run -d -p 5001:5001 <CONTAINER_ID или CONTAINER_NAME>
```

Здесь:

-d означает, что контейнер будет запущен в фоновом режиме (daemon mode);

-p 5001:5001 пробрасывает порт 5001 на вашем хосте к порту 5001 в контейнере (на котором запущено ваше ASP.NET Core приложение). Номер порта должен соответствовать

LMS-платформа − не предусмотрена

5.2. Описание внеаудиторных контрольно-оценочных мероприятий и средств текущего контроля по дисциплине модуля

Разноуровневое (дифференцированное) обучение.

Базовый

5.2.1. Домашняя работа

Примерный перечень тем

1. Практические навыки работы с Docker.

Примерные задания

Разработка контейнерного приложения на языке C# (.NET) и настройка конвейера CI/CD с помощью GitHub Actions.

Подключение образа контейнера СУБД MariaDB для взаимодействия с контейнерным приложением на языке C# (.NET).

Разработка контейнерного приложения Python и настройка конвейера CI/CD с помощью GitHub Actions.

Разработка контейнерного приложения Node.js и настройка конвейера CI/CD с помощью GitHub Actions.

Подключение образа контейнера СУБД MariaDB для взаимодействия с контейнерным приложением на языке Node.js.

Подключение образа контейнера СУБД PostgreSQL для взаимодействия с контейнерным приложением на языке Rust.

Подключение образа контейнера СУБД MariaDB для взаимодействия с контейнерным приложением на языке Python.

Подключение образа контейнера СУБД PostgreSQL для взаимодействия с контейнерным приложением на языке Java.

Разработка контейнерного приложения Java и настройка конвейера CI/CD с помощью GitHub Actions.

Разработка контейнерного приложения Rust и настройка конвейера CI/CD с помощью GitHub Actions.

Разработка контейнерного приложения Go и настройка конвейера CI/CD с помощью GitHub Actions.

Взаимодействие контейнерного приложения C# (.NET) и контейнера СУБД MariaDB.

Подключение образа контейнера СУБД PostgreSQL для взаимодействия с контейнерным приложением на языке Go.

LMS-платформа − не предусмотрена

5.3. Описание контрольно-оценочных мероприятий промежуточного контроля по дисциплине модуля

5.3.1. Зачет

Список примерных вопросов

1. Цели методологии DevOps. Общее представление об инфраструктуре современной разработки.
2. Основные принципы DevOps.
3. Различие между традиционным и DevOps-ориентированным подходами к разработке и эксплуатации ПО.
4. Ключевые роли в DevOps-команде.
5. Цели внедрения DevOps в организации. Преимущества и риски.
6. Российские и иностранные разработки в области DevOps.
7. Принципы работы современных компьютеров: процессоры, память, накопители.
8. Модель доставки программного продукта «Непрерывная интеграция и непрерывная доставка».
9. Модель доставки программного продукта «Облачная доставка».
10. Модель доставки программного продукта «Доставка с поддержкой контейнеров».
11. Что такое инфраструктура как код?
12. Преимущества IaC.
13. Основные принципы IaC.
14. Инструменты для IaC.
15. Основные принципы мониторинга и логирования.

16. Преимущества мониторинга и логирования.
17. Системы для мониторинга.
LMS-платформа − не предусмотрена

5.4 Содержание контрольно-оценочных мероприятий по направлениям воспитательной деятельности

Направление воспитательной деятельности	Вид воспитательной деятельности	Технология воспитательной деятельности	Компетенция	Результаты обучения	**Контрольно-оценочные мероприятия**
Профессиональное воспитание	проектная деятельность учебно-исследовательская, научно-исследовательская целенаправленная работа с информацией для использования в практических целях	Технология образования в сотрудничестве Технология повышения коммуникативной компетентности Технология формирования уверенности и готовности к самостоятельной успешной профессиональной деятельности Технология проектного образования Технология самостоятельной работы	ПК-17 ПК-19	П-1 П-2	Домашняя работа Зачет Лекции Практические/семинарские занятия