ЦЕНТРАЛЬНЫЙ УНИВЕРСИТЕТ

УТВЕРЖДЕНА

Приказом Ректора АНО ВО
«Центральный университет»
Ивашкевич Е.В.
от «19» января 2024 г. № 0119.37

Рабочая программа дисциплины (модуля) «Промышленная разработка» дополнительной профессиональной программы – программы профессиональной переподготовки «Академия data science»

Траектория: Backend-разработка

Москва
2024

Электронный документ

Содержание

1. Краткая характеристика дисциплины (модуля)

Изучение дисциплины (модуля) «Промышленная разработка» имеет важное значение для будущих специалистов в области информатики и программной инженерии, поскольку оно позволяет им получить практические навыки и знания, необходимые для разработки сложных программных систем, и подготовить их к успешной карьере в этой области.

Изучение промышленной разработки также имеет значение для повышения качества и эффективности разработки программных систем, поскольку оно позволяет слушателям понять основные принципы и методы промышленной разработки, что может привести к созданию более надежных, масштабируемых и поддерживаемых систем.

Цель изучения дисциплины (модуля): овладение современными методами и технологиями разработки программных систем, позволяющими создавать сложные и масштабируемые приложения, соответствующие требованиям промышленной разработки.

Задачи изучения дисциплины (модуля):

— формирование знаний о видах облачных ресурсов и различий между способами доставки и развертывания ПО;

— формирование знаний основных гарантий обеспечения надежной инфраструктуры;

— формирование знаний категоризации ресурсов для разворачивания конвейеризируемой инфраструктуры;

— формирование умения работать с инструментами контейнеризации, собирать образы и запускать контейнеры с различными параметрами и использовать сложные сценарии, такие как мультистадийные сборки;

— формирование умения применять инструменты для развертывания мутабельной и иммутабельной инфраструктуры;

— формирование умения анализировать отчеты лучших devops-практик и применять их для улучшения процессов разработки;

— формирование навыка выгружать артефакты сборки проектов, включая образы Docker и результаты тестирования, и хранить секреты в системах непрерывной интеграции;

— формирование навыка настраивать окружение разработчика простого и среднего уровня.

2. Тематический план

№ п/п	Наименование раздела дисциплины (модуля)	Трудоемкость, академические часы				ТКУ (текущий контроль успеваемости)
		Очная форма				
		Аудиторная работа		Контроль	Самостоятельная работа	
		Лекции	Семинары (практические занятия)			
1	Devops: основные практики	4	4		10	Домашнее задание Подготовка к семинару
2	Тестирование ПО	4	4		12	Домашнее задание Подготовка к семинару
3	Основы контейнеризации	4	4		12	Домашнее задание Подготовка к семинару
4	Продвинутые темы контейнеризации	4	4		12	Домашнее задание Подготовка к семинару
5	Непрерывная интеграция и доставка	4	4		12	Домашнее задание Подготовка к семинару
6	Непрерывное развёртывание	4	4		12	Домашнее задание Подготовка к семинару
7	Kubernetes	4	5		12	Домашнее задание Контрольная работа
8	Observability	5	5		12	Домашнее задание
	Зачет с оценкой			4		
	Итого:	***33***	***34***	***4***	***94***	
Объем дисциплины (модуля) (в ак. ч.)		***165***				

3. Содержание дисциплины (модуля)

№п/п	Наименование раздела дисциплины (модуля)	Содержание дисциплины (модуля) по темам
1	Devops: основные практики	Истории развития devops. Понятия devops и DevOps — основные отличия. Культура и методы devops. Три принципа devops. Паттерны и антипаттерны devops. Отчёты State of Devops и деление по основным практикам devops
2	Тестирование ПО	Принципы тестирования. Психология тестирования. Пирамида тестирования: базис, цели, объекты тестирования. Уровни тестирования: модульное тестирование, интеграционное тестирование, системное тестирование, приёмочное тестирование. Связь V-модели и пирамиды тестирования. Методы динамического тестирования. Инструменты тестирования: JUnit. Инструменты Mock-тестирования: Mockito, Hamcrest. Эмуляция браузера в тестировании: Selenium. Перехват запросов: Postman, Charles. Разработка через тестирование. Артефакты автоматического тестирования: JUnit XML, Cobertura Code Coverage
3	Основы контейнеризации	Linux Namespaces. Отличия виртуализации от контейнеризации. Инструменты контейнеризации: Docker, Podman. Понятия образа и Volume, их отличия. Запуск контейнеров: параметры запуска, проброс портов, запуск в интерактивном режиме. Docker и томы.

		Виды томов: bind, mount, tmpfs. Преимущества использования tmpfs-томов. Образы: OverlayFS и слои. Виды слоёв. Основные команды работы со слоями. Сборки multi-stage. Загрузка образов в Registry, устройство Registry. Способы авторизации в Registry. Примеры Docker Registry: GitLab Container Registry, DockerHub. *Linux Namespaces*
4	Продвинутые темы контейнеризации	Docker Compose: сервисы, интеграция сервисов, Healthcheck, порядок запуска сервисов. Безопасность в Docker: Docker in Docker, Linux Capabilities. Kaniko как безопасный инструмент сборки образов. Исследование образов и контейнеров на безопасность
5	Непрерывная интеграция и доставка	Концепции непрерывной интеграции. Понятия стадий, задач. Передача артефактов между стадиями и задачами. Матрица сборки проекта. Примеры систем непрерывной интеграции — GitLab CI, GitHub Actions. Выгрузка артефактов сборки проектов: образов Docker, результатов тестирования, уязвимости сборки проектов. Хранение секретов в системах непрерывной интеграции. Агенты для интеграции сервисов
6	Непрерывное развёртывание	Отличие систем управления конфигурации от систем обеспечения облачной инфраструктуры. Виды облачных ресурсов. Настройка облачной инфраструктуры посредством инструмента Terraform. Понятие иммутабельной инфраструктуры. Виды развёртываний ПО. Интеграция систем доставки и систем развёртывания
7	Kubernetes	Отличия оркестраторов от систем контейнеризации. Основные понятия в Kubernetes. Как разворачивать кластер в Kubernetes при помощи MiniKube. Фундамент Kubernetes: containerd, etcd. Основные абстракции в Kubernetes-кластере по уровню развёртывания сервисов: Pod, Service — типы развёртывания сервисов, Ingress. Основные абстракции в Kubernetes-кластере по отказоустойчивости: ReplicaSet, Deployment. Основные абстракции в Kubernetes-кластере по сохранению состояния: PersistentVolumeClaim, StatefulSet, Volumes — эфемерные хранилища. Хранение секретов в Kubernetes-кластере. Подключение внешних хранилищ секретов. Управляющие ресурсы в Kubernetes-кластере: Job, CronJob, DaemonSet. Управление безопасностью в Kubernetes-кластере: RBAC, ServiceAccount, Network Policy. Настройка TLS
8	Observability	Понятие отслеживаемости. Основные гарантии обеспечения: SLA, SLO, SLI. Метрики для отслеживания стабильности инфраструктуры. Виды отслеживаемости: мониторинг, логирование, трассировка. Виды систем сбора метрик и логирования. Отличие стеков ELK (ElasticSearch, LogStash, Kibana) от стека LGTM (Loki, Grafana, Tempo, Mimir). Интеграция инфраструктуры отслеживания в Kubernetes-кластере: Prometheus Service Discovery

4. Учебно-методическое обеспечение

Университет располагает полным набором лицензионного и свободно распространяемого программного обеспечения, включая продукты отечественного производства.

Каждый слушатель в течение всего периода обучения получает индивидуальный неограниченный доступ к электронно-библиотечной системе и электронной информационно-образовательной среде университета. Эти системы предоставляют возможность доступа к ресурсам из любой точки, где есть подключение к сети Интернет, как на территории университета, так и за его пределами.

Слушателям обеспечен удаленный доступ к современным профессиональным базам данных и информационным справочным системам.

Основная литература:

1. Чернышев, С. А. Принципы, паттерны и методологии разработки программного обеспечения : учебник для вузов / С. А. Чернышев. — Москва : Издательство Юрайт, 2025. — 176 с. — (Высшее образование). — ISBN 978-5-534-14383-6. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/567946.

Дополнительная литература:

1. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. — СПб.: Питер, 2022. — 1120 с. — ISBN 978-5-4461-1155-8.

5. Материально-техническое обеспечение

Университет располагает материально-технической базой, соответствующей действующим противопожарным правилам и нормам и обеспечивающей проведение всех видов дисциплинарной и междисциплинарной подготовки, практической и научно-исследовательской работ обучающихся, предусмотренных учебным планом.

Помещения, которые представляют собой учебные аудитории для проведения занятий лекционного типа, занятий семинарского (практического) типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, а также помещения для самостоятельной работы и помещения для хранения и профилактического обслуживания учебного оборудования. Помещения укомплектованы специализированной мебелью и техническими средствами обучения, служащими для представления учебной информации большой аудитории.

Изучение дисциплины (модуля) обеспечивается в учебных аудиториях, оснащенных:

— столами и стульями;

— компьютерной техникой;

— механическими калькуляторами;

— специализированным оборудованием, включая демонстрационное оборудование.

Помещения для самостоятельной работы обучающихся, в том числе приспособленные для использования инвалидами и лицами с ограниченными возможностями здоровья, оснащены компьютерной техникой с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду Университета.

Обучающимся предоставляется доступ (в том числе удаленный) к ресурсам информационно-телекоммуникационной сети «Интернет», электронным ресурсам (в том числе электронным библиотечным системам, современным профессиональным базам данных и информационным справочным системам):

№	Наименование портала (издания, курса, документа)	Ссылка
1.	Научная электронная библиотека elibrary.ru библиотека	https://elibrary.ru/defaultx.asp
2.	База данных для IT-специалистов	https://habr.com
3.	База данных ScienceDirect	https://www.sciencedirect.com
4.	Официальный сайт Министерства науки и высшего образования Российской Федерации	https://minobrnauki.gov.ru/
5.	Федеральный портал «Российское образование"	https://www.edu.ru/
6.	Информационная система "Единое окно доступа к образовательным ресурсам"	http://window.edu.ru/
7.	Единая коллекция цифровых образовательных ресурсов	http://school-collection.edu.ru/
8.	Федеральный центр информационно - образовательных ресурсов	http://fcior.edu.ru/

Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), в том числе комплект лицензионного программного обеспечения, современные профессиональные базы данных и информационные справочные системы:

Наименование ПО	Производство	Лицензионное / свободно распространяемое
Операционные системы:		
Microsoft Imagine (Windows Client, Server)	зарубежное	лицензионное
Браузеры:		
Яндекс.Браузер	отечественное	свободно распространяемое
Google Chrome	зарубежное	свободно распространяемое
Офисные приложения:		
Microsoft Imagine (Visio, OneNote)	зарубежное	лицензионное
TeXstudio	зарубежное	свободно распространяемое
Adobe Acrobat Reader	зарубежное	свободно распространяемое
Программное обеспечение для планирования и учета времени:		
Toggle app	зарубежное	свободно распространяемое
Системы управления проектами:		
Microsoft Imagine (Project)	зарубежное	лицензионное
Системы управления базами данных:		
Microsoft Imagine (SQL Server)	зарубежное	лицензионное
Системы резервного копирования (backup):		
Acronis Backup Advanced for HyperV	зарубежное	лицензионное
Справочно-правовые системы:		
КонсультантПлюс: справочно-правовая система	отечественное	лицензионное
Средства антивирусной защиты:		
Kaspersky Endpoint Security для бизнеса Стандартный Russian Edition	отечественное	лицензионное
Среды разработки:		
Visual Studio Code	зарубежное	свободно распространяемое
Bash (Unix shell)	зарубежное	свободно распространяемое
Anaconda	зарубежное	свободно распространяемое
Robotic Operating System	зарубежное	свободно распространяемое
CopelliaSim	зарубежное	свободно распространяемое

Google Colaboratory	зарубежное	свободно распространяемое
Пакеты программных средств и библиотек:		
AutoPsy	зарубежное	свободно распространяемое
Interactive Disassembler (IDA)	зарубежное	свободно распространяемое
Системы управления библиографической информацией:		
Zotero	зарубежное	свободно распространяемое
Сервисы и службы:		
Bind	зарубежное	свободно распространяемое
Docker	зарубежное	свободно распространяемое

6. Методические и оценочные материалы

Методические указания для обучающихся по освоению дисциплины (модуля)

В процессе изучения дисциплины (модуля) «Промышленная разработка» в рамках текущего контроля успеваемости используются такие виды учебной работы, как лекции, семинары, контрольная работа, домашние задания, а также различные виды самостоятельной работы обучающихся по заданию преподавателя, направленные на развитие навыков профессиональной лексики, закрепление практических профессиональных компетенций, поощрение инициатив.

Лекция – систематическое, последовательное, монологическое изложение преподавателем учебного материала, как правило, теоретического характера.

В процессе лекций рекомендуется вести конспект лекций: кратко и схематично фиксировать основные идеи, выводы и обобщения лекции; выделять важные мысли, ключевые слова и термины. Необходимо отметить вопросы или материалы, которые вызывают затруднения, и попытаться найти ответы в рекомендованной литературе. Если разобраться в материале не удается, следует сформулировать вопрос и задать его преподавателю на консультации или во время семинарского (практического) занятия.

Участие в семинаре (аудиторная работа) – активная работа слушателя на семинаре, его ответы на вопросы преподавателя и участие в дискуссии.

Для успешного участия в семинаре слушателям рекомендуется заранее ознакомиться с темой обсуждения, прочитать необходимые материалы и подготовить вопросы. Важно активно слушать и вовлекаться в дискуссию, высказывая свои мнения и аргументируя их. При ответах на вопросы преподавателя стоит быть уверенным, четким и логичным, опираясь на изученный материал. Также полезно поддерживать диалог с однокурсниками, чтобы обогатить обсуждение и расширить свои знания.

Домашнее задание – набор задач по темам недели.

При работе над домашними заданиями важно внимательно ознакомиться с требованиями и сроками выполнения. Рекомендуется разбивать задания на этапы, чтобы избежать перегрузки и лучше усвоить материал. Использовать различные источники информации, включая учебники и онлайн-ресурсы, для более глубокого понимания темы.

Контрольная работа – письменная работа с набором задач, которые нужно решить за ограниченное время.

Цель контрольной работы – получить специальные знания по одной или нескольким темам дисциплины (модуля) и продемонстрировать навыки их практического применения.

Самостоятельная работа – работа слушателй, направленная на углубленное изучение отдельных тем и вопросов учебной дисциплины (модуля).

В процессе самостоятельной работы слушатели взаимодействуют с рекомендованными материалами при минимальном участии преподавателя. Задачи

слушателя включают работу с конспектами лекций (обработка текста), повторное изучение учебных материалов планов и тезисов ответов, изучение дополнительных тем, выполнение учебно-исследовательских заданий и другое.

Система оценивания результатов обучения по дисциплине (модулю)

Оценивание уровня учебных достижений обучающихся по дисциплине (модулю) осуществляется в виде текущего контроля успеваемости и промежуточной аттестации.

Промежуточная аттестация по дисциплине (модулю) осуществляется в форме ***зачета с оценкой***.

Для оценивания текущего контроля успеваемости и промежуточной аттестации используется десятибалльная шкала оценивания, которая соотносится с традиционной пятибалльной шкалой следующим образом:

Десятибалльная оценка	**Пятибалльная оценка**	**Оценка за зачет**	**Общая характеристика результата обучения по дисциплине (модулю)**
10	Отлично	Зачтено	Слушатель полностью владеет знаниями, изложенными в рабочей программе, и глубоко осмысляет дисциплину (модуль). Он самостоятельно и логически последовательно отвечает на все вопросы, акцентируя внимание на наиболее важном. Умеет анализировать, сравнивать, классифицировать, обобщать, конкретизировать и систематизировать изученный материал, выделяя ключевые моменты и устанавливая причинно-следственные связи. Четко формулирует ответы, уверенно интерпретирует результаты анализов и других исследований, а также решает сложные задачи. Слушатель хорошо знаком с методами исследования, необходимыми для практической деятельности, и умеет связывать теоретические аспекты дисциплины (модуля) с практическими задачами.
9	Отлично	Зачтено	
8	Отлично	Зачтено	
7	Хорошо	Зачтено	Слушатель обладает знаниями предмета почти в полном объеме рабочей программы и самостоятельно, логически последовательно и всесторонне отвечает на все вопросы, акцентируя внимание на наиболее значимых моментах. Он умеет анализировать, сравнивать, классифицировать, обобщать, конкретизировать и систематизировать изученный материал, выделяя его ключевые аспекты и устанавливая причинно-следственные связи.
6	Хорошо	Зачтено	

Десятибалльная оценка	Пятибалльная оценка	Оценка за зачет	Общая характеристика результата обучения по дисциплине (модулю)
			Формулирует свои ответы, уверенно интерпретирует результаты анализов и других исследований, а также решает сложные ситуационные задачи. Слушатель хорошо знаком с методами исследования, необходимыми для практической деятельности, и умеет связывать теоретические аспекты предмета с практическими задачами.
5	Удовлетворительно	Зачтено	Слушатель обладает базовыми знаниями по дисциплине (модулю), но испытывает трудности при самостоятельных ответах и использует неточные формулировки. В ходе ответов он допускает ошибки, касающиеся сути вопросов. Слушатель способен решать только самые простые задачи и владеет лишь минимальным набором методов исследования.
4	Удовлетворительно	Зачтено	
3	Не сдан	Не зачтено	Слушатель не овладел обязательным минимумом знаний по предмету и не может ответить на вопросы, даже если преподаватель задает дополнительные наводящие вопросы.
2	Не сдан	Не зачтено	
1	Не сдан	Не зачтено	

Дисциплина (модуль) «Промышленная разработка» оценивается следующим образом:

Активность	Вес	Описание
Домашние задания	70%	За каждое из заданий можно набрать 10 баллов
Аудиторная работа	15%	На каждом семинаре слушатель может заработать баллы за интересные вопросы, работу на семинаре и выполнение заданий
Контрольная работа	15%	Контрольная работа, на которой оценивается процент правильных ответов и конвертируется в количество набранных баллов (так, за 100% правильных ответов слушатель получает 10 баллов)

Формула расчёта итоговой оценки по дисциплине (модулю) «Промышленная разработка»: «0,7 × среднее за домашние задания + 0,15 × аудиторная работа + 0,15 × контрольная работа».

Текущий контроль успеваемости обучающихся по дисциплине (модулю)

Примерные вопросы для подготовки к семинарам

Тестирование ПО

1. Какие основные принципы тестирования ПО?
2. Какова роль психологии тестирования в процессе разработки?
3. Что такое пирамида тестирования и какие уровни она включает?
4. В чем разница между модульным и интеграционным тестированием?
5. Каковы цели системного тестирования?

6. Как V-модель соотносится с пирамидой тестирования?
7. Какие методы динамического тестирования вы знаете?
8. Как работает инструмент JUnit для тестирования ПО?
9. Что такое Mockito и как он используется в тестировании?
10. Чем Hamcrest отличается от Mockito?
11. Какова роль Selenium в тестировании веб-приложений?
12. Как использовать Postman для перехвата запросов?
13. В чем преимущества использования Charles для тестирования?
14. Что такое разработка через тестирование (TDD)?
15. Какие артефакты создаются при автоматическом тестировании с помощью JUnit?
16. Как Cobertura Code Coverage помогает в тестировании?
17. Каковы основные подходы к тестированию API?
18. Какие типы тестов являются наиболее важными для веб-приложений?
19. Каковы основные проблемы, с которыми сталкиваются тестировщики?
20. Какие метрики используются для оценки качества тестирования?

Основы контейнеризации

1. Что такое Linux Namespaces и как они работают?
2. В чем разница между виртуализацией и контейнеризацией?
3. Какие инструменты контейнеризации вы знаете?
4. Каковы основные функции Docker?
5. Что такое Podman и чем он отличается от Docker?
6. Что такое образ контейнера и как он создается?
7. Какова роль Volume в контейнеризации?
8. В чем разница между bind и mount томами?
9. Каковы преимущества использования tmpfs-томов?
10. Что такое OverlayFS и как он используется в Docker?
11. Какие виды слоев существуют в образах контейнеров?
12. Каковы основные команды для работы со слоями в Docker?
13. Что такое сборки multi-stage и в чем их преимущества?
14. Как загружать образы в Registry?
15. Какие типы Registry существуют для Docker?
16. Как происходит авторизация в Docker Registry?
17. Приведите примеры Docker Registry.
18. Как можно управлять ресурсами контейнеров?
19. Каковы основные проблемы, с которыми сталкиваются при работе с контейнерами?
20. Как обеспечивается безопасность контейнеров?

Продвинутые темы контейнеризации

1. Что такое Docker Compose и как он используется?
2. Как организуются сервисы в Docker Compose?
3. Что такое Healthcheck и как его настроить в Docker?
4. Каков порядок запуска сервисов в Docker Compose?
5. Какие меры безопасности применяются в Docker?
6. Что такое Docker in Docker и как он используется?
7. Как Linux Capabilities влияют на безопасность контейнеров?
8. Что такое Kaniko и как он помогает в сборке образов?
9. Как исследовать образы и контейнеры на безопасность?
10. Какие инструменты можно использовать для анализа безопасности контейнеров?
11. В чем преимущества использования Docker Compose для микросервисов?
12. Как управлять конфигурацией сервисов в Docker Compose?
13. Какие лучшие практики безопасности следует учитывать при работе с Docker?
14. Каковы основные ошибки, которые могут возникнуть при использовании Docker?
15. Как тестировать безопасность контейнеров перед развертыванием?

16. Каковы преимущества использования многослойной архитектуры в контейнерах?
17. Как обеспечивается мониторинг контейнеров?
18. Какие существуют методы управления сетевыми настройками в Docker?
19. Как реализовать автоматическое масштабирование контейнеров?
20. Каковы основные тренды в области контейнеризации на сегодняшний день?

Примерные домашние задания

Домашнее задание “Docker Compose Advanced”.

Для выполнения домашнего задания по теме “**Docker Compose Advanced**” ты можешь выбрать один из двух языков программирования (Python/Java):

1. **Docker Compose Advanced**. Java
2. **Docker Compose Advanced**. Python

Что нужно сделать:

- Перейти по выбранной ссылке и выполнить задания

- Добавь SSH-ссылку на репозиторий с выполненным заданием в систему для отслеживания прогресса.

- Приложи эту же SSH-ссылку в меню akhcheck и отправь на проверку

Важно: Задание проходит автоматическую проверку, поэтому убедись, что код оформлен корректно и все тесты успешно выполняются.

Docker Compose Advanced. Java

Описание задания

В этом задании Вам предстоит собрать конфигурацию Docker Compose под приложение на Java.

Для этого Вам необходимо будет сделать скопировать в свой репозитории содержимое ветки task-docker-compose в репозитории https://gitlab.akhcheck.ru/task-templates/DockerComposeJavaTemplate

Требования к сборке

- База Данных - MySQL, версия 5.7
- Версия Java - 8
- Файл docker-compose.yml должен быть в папке docker-compose-template.

Вам необходимо будет настроить подключение папки CREATE.sql из папки со скриптами во время создания контейнера.

Создайте init container, который будет заполнять данные в базу.

Подсказка: проект Веб собран при помощи Spring Boot.

Еще одна подсказка. Веб-приложение не стартует без базы данных. Вы должны найти способ создать базу данных перед запуском приложения.

Требования к запуску

- Приложение должно быть запущено на порту 8080
- Для заказов необходимо отображение:
- История заказов -> status MOVING
- Далее будет запущено приложение

Процедура сдачи

- Создайте Merge Request из ветки task-docker-compose в ветку main или в ветку master.

• Добавьте в ревьюеры проверяющего задание

Максимальный балл: 10

Docker Compose Advanced. Python

Описание задания

Преамбула

Коллега на работе разрабатывал проект на работе. Однако, он уволился, и компания перешла в контейнеризированную среду.

Ваша цель - запустить проект по ссылке. Вам даны права Reporter, поэтому вам необходимо ветку task-docker-compose в Ваш репозиторий.

Что от вас ждет начальство?

1. (2 балла) Docker-образ приложения backend и запуск контейнера с backend через docker compose (2 балла)
2. (1/3 балла) Запуск в docker compose Docker Container-а с базой данных. Если приложение будет backend будет запускаться без внешней базы (можно подключить SQLite), то будет добавлен 1 балл. Если будет запускаться приложение с базой данных Postgres, то будет ставиться 3 балла за эту часть задания. **Важно**: данные в базе данных должны храниться постоянно (в отдельном Volume-е)
3. (3 балла) Запуск сервиса frontend с использованием конфигурации NGINX. Из внешней сети контейнера сервис должен быть доступен на порту 8189.

• (2 балла) ставится за успешный запрос к API через сервис frontend

• (1 балл) ставится за успешный проброс стилевых файлов (этим должен заниматься Frontend). При успешном подключении вы увидите большие буквы по одному из URL.

1. (3 балла) Непоседливый разработчик забыл добавить тестовые данные в приложение. Для этого необходимо создать init container, который перед запуском сервисов backend и frontend будет заполнять базу данных 2 тестовыми пользователями:

```
users = [
    User(login='pavel', email='a@gmail.com'),
    User(login='yura', email='b@gmail.com')
]
```

• Инициализация базы данных должна стартовать перед запуском backend

• Frontend должен стартовать после успешного запуска сервиса backend (в ином случае, nginx не запустится)

Процедура сдачи задания

1. Скопируйте содержимое ветки task-docker-compose репозитория в репозиторий, в котором выполняете задания курса.
2. В ветке task-docker-compose в папке docker-compose-template создайте файл docker-compose.yml, в котором составьте корректную конфигурацию проектов.
3. Добавьте в репозиторий необходимые файлы для запуска проекта (например, нужны будут Dockerfile-ы для образов, которые собираетесь использовать для сборки проекта)
4. Перед тем, как отправлять задание на проверку, убедитесь, что проект запускается. Шаги по запуску проекта:

```
docker compose down -v
docker compose up
```

Если нет команды docker compose, то можно заменить команду на docker-compose.

Если у преподавателей не запустится проект, то, увы, задание не будет зачтено. Важно сделать запуск "под ключ".

Полезные ссылки

• https://fastapi.tiangolo.com/ru/deployment/manually/ - запуск приложения FastAPI.

Максимальный балл: 10

Домашнее задание: Практика по Ansible

Что нужно сделать:

• Перейти по ссылке: https://akhcheck.ru/course/74#656 и выполнить задание.

• Добавь SSH-ссылку на репозиторий с выполненным заданием в систему для отслеживания прогресса.

• Приложи эту же SSH-ссылку в меню akhcheck и отправь на проверку.

• **Важно:** Задание проходит автоматическую проверку, поэтому убедись, что код оформлен корректно и все тесты успешно выполняются.

Описание задания

В этом задании вам необходимо продемонстрировать умение настраивать тестовый стенд.

Шаги выполнения

1. Используйте репозиторий для одного из прошлых заданий https://gitlab.akhcheck.ru
2. Добавьте в задании по CI дополнительный шаг deliver, который
3. (Ручная проверка, 3 балла) Настройте Pipeline для сборки образа и Push-а в Registry. Образ в Registry должен иметь tag staging. Название Stage - BuildImage, название Job - BuildDebugImage
4. (Ручная проверка) (3 балла) Создайте ветку develop. В ветке develop настройте Pipeline в Ansible, который будет выкачивать созданный репозиторий на сервер node03.hadoop.akhcheck.ru (доступ был выдан в ходе запуска курса) в ветку develop. Репозиторий должен называться DevopsExam и находиться по пути: /home/<your user>/Pipeline, в репозитории будет выкачана ветка develop. Проверяющий будет проверять, что репозиторий создан и в репозитории выбрана конкретная ветка.
5. (Ручная проверка) (4 балла) Настройте установку необходимого окружения:

• для заданий по Python - настройте виртуальное окружение в Ansible и создание артефактов тестирования для тестовых стендов

• для заданий по Java - настройте выкачивание Docker-образа для Maven и запуск сервиса.

Максимальный балл: 10

Домашнее задание. Kubernetes - Basics.

Что нужно сделать:

• Перейти по ссылке: https://akhcheck.ru/course/74#737 и выполнить задание.

• Добавь SSH-ссылку на репозиторий с выполненным заданием в систему для отслеживания прогресса.

• Приложи эту же SSH-ссылку в меню akhcheck и отправь на проверку.

• **Важно:** Задание проходит автоматическую проверку, поэтому убедись, что код оформлен корректно и все тесты успешно выполняются.

Описание задания

Создайте настройку для Kubernetes-кластера. В качестве примера можно взять настроенное приложение с docker compose с предыдущих заданий

Что должно включаться в сборку:

1. Pod-ы для запуска приложения (3 балла).
2. Настройка сервиса с использованием кластера ClusterIP (3 балла)
3. Конфигурация сервера при помощи Ingress. При этом доступ должен быть как к базе данных, как к Frontend, так и в Backend (4 балла)

Если у вас есть собственный проект, который вы хотели поднять с помощью Kubernetes, то можно использовать его!

Формат сдачи задания

Необходимо прислать ссылку на репозиторий. В репозитории должен детально описан порядок сборки манифестов. Без порядка сборки манифестов задание не будет засчитано!

Максимальный балл: 10

Домашнее задание: Kubernetes - Продвинутое

Что нужно сделать:

- Перейти по ссылке: https://akhcheck.ru/course/74#738 и выполнить задание.
- Добавь SSH-ссылку на репозиторий с выполненным заданием в систему для отслеживания прогресса.
- Приложи эту же SSH-ссылку в меню akhcheck и отправь на проверку.

Важно: Задание проходит автоматическую проверку, поэтому убедись, что код оформлен корректно и все тесты успешно выполняются.

Описание задания

Создайте настройку для своего проекта в Kubernetes-кластере. Продолжаем выполнять задание:

Что должно включаться в сборку:

1. Настройка секретов для Docker Registry и для Basic Auth (1+2 балла).
2. Настройка Job для миграции или CronJob для ежедневных задач (3 балла)
3. Переформатирование базы данных в StatefulSet (2 балла)
4. Запуск базы данных в режиме High Availability (4 балла)

Если у вас есть собственный проект, который вы хотели поднять с помощью Kubernetes, то можно использовать его!

Формат сдачи задания

Необходимо прислать ссылку на репозиторий. В репозитории должен детально описан порядок сборки манифестов. Без порядка сборки манифестов задание не будет засчитано!

Примерные задания для контрольной работы

1. **Сравнительный анализ**. Опишите основные отличия между оркестраторами контейнеров и системами контейнеризации. Приведите примеры каждого типа и объясните, как они взаимодействуют друг с другом.
2. **Основные понятия**. Объясните ключевые термины, связанные с Kubernetes, такие как Pod, Node, Cluster, и Namespace. Какова их роль в управлении контейнерами?

3. **Разворачивание кластера**. Приведите пошаговую инструкцию по развертыванию кластера Kubernetes с использованием MiniKube. Укажите необходимые команды и настройки.

4. **Фундамент Kubernetes**. Объясните роль containerd и etcd в архитектуре Kubernetes. Как эти компоненты способствуют управлению контейнерами и состоянием кластера?

5. **Абстракции для развертывания сервисов**. Опишите разницу между Pod и Service в Kubernetes. Как используются Ingress и какие преимущества он предоставляет при развертывании приложений?

6. **Отказоустойчивость**. Объясните, как ReplicaSet и Deployment обеспечивают отказоустойчивость в Kubernetes. Приведите примеры их использования в реальных сценариях.

7. **Сохранение состояния**. Что такое PersistentVolumeClaim и как он используется для управления состоянием в Kubernetes? Опишите также, как StatefulSet и Volumes обеспечивают хранение данных.

8. **Хранение секретов**. Опишите процесс хранения секретов в Kubernetes-кластере. Как можно подключить внешние хранилища секретов, такие как HashiCorp Vault?

9. **Управляющие ресурсы**. Объясните, что такое Job, CronJob и DaemonSet в Kubernetes. В каких случаях их следует использовать и какие задачи они решают?

10. **Управление безопасностью**. Как реализуется управление безопасностью в Kubernetes с помощью RBAC, ServiceAccount и Network Policy? Приведите примеры их настройки и использования.