Національний технічний університет України
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Кафедра інженерії
програмного забезпечення
в енергетиці (ІПЗЕ)

Створення та оркестрація контейнерів
Робоча програма навчальної дисципліни (Силабус)

Реквізити навчальної дисципліни

Рівень вищої освіти	**Перший (бакалаврський)**
Галузь знань	12 Інформаційні технології
Спеціальність	121 Інженерія програмного забезпечення
Освітня програма	Інженерія програмного забезпечення інтелектуальних кібер-фізичних систем в енергетиці
Статус дисципліни	Вибіркова
Форма навчання	Очна (денна)
Рік підготовки, семестр	4 курс, весняний семестр
Обсяг дисципліни	4 кредитів ЄКТС/120 год (36 год. лекції, 18 год. практичні заняття, 66 год. самостійна робота)
Семестровий контроль/ контрольні заходи	Залік
Розклад занять	Згідно розкладу на весняний семестр поточного навчального року (http://roz.kpi.ua/)
Мова викладання	Українська
Інформація про керівника курсу / викладачів	Лектор: старший викладач, Колумбет В.П., kvplinux@gmail.com, тел. 093-405-35-39 Практичні: старший викладач, Колумбет В.П., kvplinux@gmail.com, тел. 093-405-35-39
Розміщення курсу	Кампус

Програма навчальної дисципліни

1. Опис навчальної дисципліни, її мета, предмет вивчання та результати навчання

В цій дисципліні детально вивчається концепції, принципи та методи інформаційних технологій. Етапи розробки програмного забезпечення. Міжнародні стандарти програмної інженерії. Моделі та процеси, які впроваджуються в розробленні програмного забезпечення. Вимоги до інформаційної системи. Документування розроблених вимог та моделей для створення інформаційної системи. Управління змінами до вимог протягом циклу розробки програмного забезпечення.

Метою дисципліни є опанування студентами теоретичних та практичних знань і навичок систем тестування, інструментальних засобів розробки програмних систем, розгортання та функціонування програмного забезпечення, розробляти, аналізувати та застосовувати ефективні алгоритми для розв'язання професійних завдань в області інформаційних технологій, та узгодження розробки й постачання програмного забезпечення із його використанням, сформувати цілісне уявлення про сучасні технології розробки додатків з мікросервісною архітектурою, їх розгортання, масштабування та тестування.

Предмет дисципліни - вивчення принципів побудови, архітектури, основних функцій, режимів роботи і елементи та комплекси практик оркестрації контейнерів, розроблення адаптивних алгоритмів розв'язування задач і системного адміністрування.

Завдання. В результаті вивчення дисципліни у студентів повинні сформуватися наступні компетентності:

Вивчення дисципліни «Створення та оркестрація контейнерів» сприяє формуванню у студентів фахової компетентності (ФК) за освітньою програмою:

ФК01 - Здатність ідентифікувати, класифікувати та формулювати вимоги до програмного забезпечення.

ФК08 - Здатність застосовувати фундаментальні і міждисциплінарні знання для успішного розв'язання завдань інженерії програмного забезпечення.

Вивчення дисципліни «Створення та оркестрація контейнерів» сприяє формуванню у студентів наступних програмних результатів навчання (ПРН) за освітньою програмою:

ПРН01 - Аналізувати, цілеспрямовано шукати і вибирати необхідні для вирішення професійних завдань інформаційно-довідникові ресурси і знання з урахуванням сучасних досягнень науки і техніки.

ПРН05 - Знати і застосовувати відповідні математичні поняття, методи доменного, системного і об'єктно-орієнтованого аналізу та математичного моделювання для розробки програмного забезпечення.

ПРН07 - Знати і застосовувати на практиці фундаментальні концепції, парадигми і основні принципи функціонування мовних, інструментальних і обчислювальних засобів інженерії програмного забезпечення.

ПРН09 - Знати та вміти використовувати методи та засоби збору, формулювання та аналізу вимог до програмного забезпечення.

ПРН14 - Застосовувати на практиці інструментальні програмні засоби доменного аналізу, проектування, тестування, візуалізації, вимірювань та документування програмного забезпечення.

ПРН18 - Знати та вміти застосовувати інформаційні технології обробки, зберігання та передачі даних.

2. Пререквізити та постреквізити дисципліни (місце в структурно-логічній схемі навчання за відповідною освітньою програмою)

Пререквізити дисципліни. Знання, отримані при вивченні дисциплін: "Компоненти програмної інженерії", "Алгоритми та структури даних" та "Операційні системи".

Постреквізити дисципліни. Отримані знання при вивченні дисципліни «Архітектура системного програмного забезпечення» формує базові знання для вивчення наступних дисциплін: "Теорія ймовірностей та математична статистика", "Алгоритми та структури даних", "Бази даних", "Моделювання та аналіз програмного забезпечення", "Основи розробки трансляторів", "Системи штучного інтелекту", "Організація баз даних та знань", "Криптографія та шифрування", "Системи керування версіями". Також викладений матеріал може бути використаний при вивченні дисциплін "Математичне моделювання систем і процесів", "Дослідження операцій", які викладаються в наступних семестрах. Компетенції, отримані студентами в процесі вивчення цієї дисципліни, використовуються ними при виконанні дипломної роботи.

3. Зміст навчальної дисципліни

Тема 1. Основи контейнеризації: концепції та інструменти.
Тема 2. Створення та управління контейнерами за допомогою Docker.
Тема 3. Оркестрація контейнерів: основи Kubernetes.
Тема 4. Масштабування та автоматизація контейнерних додатків.
Тема 5. Безпека контейнерних додатків.
Тема 6. Інтеграція контейнерів у процеси CI/CD.

4. Навчальні матеріали та ресурси

Основна література

1. Kubernetes and Docker - An Enterprise Guide: Effectively containerize applications, integrate enterprise systems, and scale applications in your enterprise 1st Edition, Kindle Edition by Scott Surovich (2020) pp. 110-128

2. Інженерія програмного забезпечення: Посібник для студентів вищих навчальних закладів / І. Л. Бородкіна, Г. О. Бородкін ; М-во освіти і науки України, Національний університет біоресурсів та природокористування України. – Київ: , 2018. – с.
3. Sommerville I. Software Engineering— 9th ed. / Ian Sommerville. – Addison-Wesley, 2011.
4. DevSecOps: [Електронний ресурс]. Synopsys – 2021. – Режим доступу до ресурсу: https://www.synopsys.com/glossary/what-is-devsecops.html
5. What is DevOps?: [Електронний ресурс]. AWS – 2020. – Режим доступу до ресурсу: https://aws.amazon.com/devops/what-is-devops.
6. The Docker Book: Containerization is the new virtualization Kindle Edition by James Turnbull (2014) pp.32-35.
7. Continuous Delivery: reliable software releases through build, test, and deployment automation. / Humble, Jez; Farley, David (2011). / Pearson Education Inc. ISBN 978-0-321-60191-9.
8. Управління версіями програмних засобів проекту [Електронний ресурс] : навчальний посібник для здобувачів ступеня бакалавра за освітньою програмою «Інженерія програмного забезпечення інтелектуальних кібер-фізичних систем в енергетиці» спеціальності 121 Інженерія програмного забезпечення / КПІ ім. Ігоря Сікорського ; уклад.: В. О. Кузьміних, О. В. Коваль, Р. А. Тараненко. – Електронні текстові дані (1 файл: 4,63 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2023. – 114 с. – Назва з екрана. (доступ за посиланням https://ela.kpi.ua/handle/123456789/56605)

Додаткова література

9. Програмна інженерія. Візуальне моделювання програмних систем: підручник для СПО / Е. А. Черткова. — 2-е вид., випр. та доп. — М.: Видавництво Юрайт, 2017. — 164 с. ISBN 978-5-534-04928-2
10. Лавріщева Є. М. Програмна інженерія. Парадигми, технології та case-засоби 2-ге вид. Видавництво: Юрайт : ISBN: 5534010568 ISBN-13(EAN):785534010565 Сторінки: 280
11. Основи інженерії якості програмних систем.-Андон Ф. І., Коваль Г. І., Коротун Т. М., Лавріщева О. М., Суслов В. Ю.-Київ: Академперіодика, 2007.-680 с.
12. What is DevOps?: [Електронний ресурс]. AWS – 2020. – Режим доступу до ресурсу: https://aws.amazon.com/devops/what-is-devops
13. Визначення предмету-програмна інженерія.-/Лавріщева К. М.-Проблеми програмування.-Спецвипуск.-2008.-№ 2-3.-с.191-204.
14. J.Mulder. Enterprise DevOps for Architects. Packt Publishing. BIRMINGHAM—MUMBAI, 2021. 178 c.
15. Методи програмування. Теорія, практика, інженерія. -/Лавріщева Е. М.-Наукова думка. -2006.-471с.
16. Peter, Naur; Brian Randell (7–11 October 1968). Software Engineering: Report of a conference sponsored by the NATO Science Committee (PDF). Garmisch, Germany: Scientific Affairs Division, NATO.
17. SWEBOK executive editors, Alain Abran, James W. Moore; editors, Pierre Bourque, Robert Dupuis. (2004). У Pierre Bourque and Robert Dupuis. Guide to the Software Engineering Body of Knowledge - 2004 Version. IEEE Computer Society. с. 1–1. ISBN 0-7695-2330-7.
18. What Is Git Explore. A Distributed Version Control Tool Edureka. Edureka. URL: https://www.edureka.com/biog-what-is-git
19. The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations / Gene Kim, Patrick Debois, John Willis, Jez Humble, John Allspaw / IT Revolution Press / October 6, 2016, 480p, ISBN-10 : 1942788002 ISBN-13 : 978-1942788003

Навчальний контент

5. Методика опанування навчальної дисципліни (освітнього компонента)

Лекційні заняття

№ з/п	Назва теми та лекції з переліком основних питань
1	**Тема 1. Основи контейнеризації: концепції та інструменти** *Лекція 1. Вступ до контейнеризації* Огляд базових принципів контейнеризації, її відмінностей від традиційної віртуалізації, а також огляд основних переваг використання контейнерів у сучасних IT-інфраструктурах.
2	**Тема 1. Основи контейнеризації: концепції та інструменти** *Лекція 2 Основи Docker.* Детальний розбір інструмента Docker: що таке контейнери, як їх створювати та керувати ними, огляд базових команд Docker для роботи з контейнерами та образами.
3	**Тема 2. Створення та управління контейнерами за допомогою Docker** *Лекція 3. Dockerfile: створення та оптимізація образів* Як створювати Dockerfile для автоматизації створення образів, основні інструкції Dockerfile, найкращі практики для зменшення розміру образів та підвищення їхньої продуктивності.
4	**Тема 2. Створення та управління контейнерами за допомогою Docker** *Лекція 4. Мережеві можливості Docker* Моделі мереж у Docker, налаштування приватних мереж, створення мостових, оверлейних мереж та використання їх у багатоконтейнерних додатках.
5	**Тема 2. Створення та управління контейнерами за допомогою Docker** *Лекція 5. Docker Compose: багатоконтейнерні додатки* Інструмент Docker Compose для керування багатоконтейнерними додатками, створення YAML-файлів для опису взаємодії між контейнерами, запуск та управління додатками.
6	**Тема 3. Оркестрація контейнерів: основи Kubernetes** *Лекція 6. Оркестрація контейнерів: вступ до Kubernetes* Основи Kubernetes як інструмента для оркестрації контейнерів. Огляд архітектури Kubernetes: компоненти кластера, ролі та функціонал.
7	**Тема 3. Оркестрація контейнерів: основи Kubernetes** *Лекція 7. Основні об'єкти Kubernetes: Pods, Services, Deployments* Огляд основних об'єктів у Kubernetes, їх призначення та способи використання для керування контейнерними додатками.
8	**Тема 4. Масштабування та автоматизація контейнерних додатків** *Лекція 8. Створення та налаштування Kubernetes маніфестів* Написання YAML-файлів для визначення Pods, Services, ConfigMap, Secret та інших об'єктів Kubernetes. Приклади та найкращі практики.
9	**Тема 4. Масштабування та автоматизація контейнерних додатків** *Лекція 9. Моніторинг та логування в Kubernetes* Інструменти для моніторингу продуктивності та стану контейнерів у Kubernetes, збір логів та використання Prometheus та Grafana для візуалізації даних.

10	**Тема 4. Масштабування та автоматизація контейнерних додатків** *Лекція 10. Автоматичне масштабування та балансування навантаження в Kubernetes* Налаштування горизонтального автоматичного масштабування додатків, балансування навантаження між контейнерами, стратегії деплою.
11	**Тема 5. Безпека контейнерних додатків** *Лекція 11. Безпека контейнерів та Kubernetes* Безпекові аспекти контейнеризації: контроль доступу (RBAC), використання секретів, безпека мережі в контейнерах, основи захисту даних.
12	**Тема 6. Інтеграція контейнерів у процеси CI/CD** *Лекція 12. CI/CD з Docker та Kubernetes* Автоматизація процесів безперервної інтеграції та доставки додатків з використанням контейнерів. Огляд інтеграції Docker та Kubernetes з інструментами CI/CD, такими як Jenkins та GitLab CI.

Практичні заняття

№ з/п	Назва теми заняття та перелік основних питань
1	**Установка Docker та запуск базового контейнера** Завдання: *Встановити Docker на локальну машину та запустити офіційний образ hello-world. Перевірити, що контейнер працює, та переглянути його логи.* **Мета**: Ознайомитись з базовими командами Docker для запуску контейнерів і перегляду інформації про них.
2	**Створення Docker-образа за допомогою Dockerfile** Завдання: Написати Dockerfile для простого веб-додатку на Python (або іншій мові за вибором), побудувати образ та запустити його в контейнері. **Мета**: Навчитися створювати Dockerfile для автоматизації процесу побудови образів.
3	**Налаштування Docker Compose для багатоконтейнерного додатка** Завдання: Створити Docker Compose файл для запуску веб-сервера та бази даних (наприклад, Nginx та PostgreSQL). Налаштувати їх взаємодію через мережу та запустити додаток. **Мета**: Опанувати інструмент Docker Compose для запуску складних додатків із кількома контейнерами.
4	**Розгортання додатку в Kubernetes** Завдання: Створити Kubernetes Pod для простого веб-додатку, налаштувати Service для доступу до Pod, та запустити цей додаток у кластері Kubernetes (локально або у хмарі). **Мета**: Опанувати основи розгортання додатків у Kubernetes, використовуючи Pods та Services.
5	**Використання ConfigMap та Secret в Kubernetes** Завдання: Налаштувати додаток у Kubernetes з використанням ConfigMap для зберігання конфігурації та Secret для зберігання конфіденційних даних (наприклад, паролів до бази даних). **Мета**: Зрозуміти, як зберігати та використовувати конфігураційні дані й секрети в Kubernetes.

6	**Моніторинг та автоматичне масштабування додатка в Kubernetes** Завдання: Налаштувати моніторинг стану Pods за допомогою Kubernetes метрик, а також автоматичне горизонтальне масштабування додатка на основі завантаження процесора або пам'яті. **Мета**: Навчитися використовувати інструменти моніторингу та масштабування додатків у Kubernetes для покращення продуктивності та стійкості додатків.

Самостійна робота студента

№ з/п	Вид самостійної роботи	Кількість годин СРС
1	**Дослідження концепцій контейнеризації та її переваг** Завдання: Провести аналіз різниці між традиційною віртуалізацією та контейнеризацією. Описати ключові переваги контейнерів у розробці додатків. Підготувати короткий звіт або презентацію з прикладами застосування контейнеризації у великих компаніях. **Мета**: Зрозуміти основні концепції та переваги контейнеризації в сучасних IT-середовищах.	10
2	**Створення та оптимізація Docker-образів** Завдання: Написати Dockerfile для простого додатка (наприклад, на PHP або Go), зібрати образ та оптимізувати його розмір, використовуючи багатошаровий підхід. Дослідити способи зменшення розміру образів (використання базових образів, очищення кешу). **Мета**: Розвинути навички створення ефективних Docker-образів і їхньої оптимізації для використання у продакшн середовищах.	10
3	**Налаштування багатоконтейнерного середовища з Docker Compose** Завдання: Самостійно створити конфігурацію *docker-compose.yml* для мікросервісної архітектури, що складається з декількох сервісів (наприклад, фронтенд на React та бекенд на Node.js з базою даних PostgreSQL). Налаштувати мережеві взаємодії між сервісами. **Мета**: Опанувати навички налаштування та керування багатоконтейнерними додатками.	10
4	**Налаштування кластеру Kubernetes у локальному середовищі** Завдання: Використовуючи Minikube або K3s, налаштувати локальний кластер Kubernetes. Додатково дослідити та підготувати звіт про можливості Kubernetes щодо масштабування, самовідновлення та балансування навантаження. **Мета**: Ознайомитися з основами Kubernetes та налаштувати кластер для подальших експериментів з оркестрацією контейнерів.	10
5	**Безпека контейнерів: дослідження та налаштування** Завдання: Дослідити методи безпеки контейнерних додатків у Docker та Kubernetes (ізоляція контейнерів, рольове управління доступом (RBAC), політики мереж). Налаштувати базові безпекові політики для вашого додатка у Docker або Kubernetes і підготувати рекомендації з безпеки. **Мета**: Навчитися налаштовувати безпеку контейнерів та розуміти ключові аспекти захисту контейнерних середовищ.	10
6	**Автоматизація розгортання додатків з використанням CI/CD та Kubernetes** Завдання: Налаштувати простий CI/CD конвеєр за допомогою GitLab CI або Jenkins для автоматичного розгортання додатка в Kubernetes після кожного комміту в репозиторій. Підготуйте документацію або відео з описом вашого процесу. **Мета**: Освоїти інтеграцію контейнерів у процеси CI/CD для автоматизації та прискорення розгортання додатків.	10
7	**Підготовка до заліку**	6

Контрольна робота

Метою контрольної роботи є закріплення та перевірка теоретичних знань із освітнього компонента, набуття студентами практичних навичок самостійного вирішення задач та складанні та компіляції програм.

Модульна контрольна робота (МКР) виконується після вивчення тем 1-6 та виконання практичних занять 1-5. Контрольна робота проводяться у середовищі Linux з використанням інструментарію Docker та Kubernetes. Кожен студент отримує індивідуальне завдання, відповідно до якого необхідно виконати розгортання програмного комплексу, реалізація поставленої задачі та моніторинг якості виконання програмної складової задачі.

Політика та контроль

6. Політика навчальної дисципліни (освітнього компонента)

Відвідування лекційних та лабораторних занять є обов'язковим за винятком поважних причин (хвороби, форс-мажорних обставин).

В разі пропущення занять з поважних причин викладач надає можливість студенту виконати усі або деякі лабораторні завдання (винятком є виконання деяких завдань у зв'язку із закінченням навчального процесу).

В разі пропущення занять без поважних причин, а також через порушення граничного терміну виконання завдання (deadline) студент може отримати 80% від максимальної оцінки відповідне завдання.

Протягом семестру студенти:

- виконують та захищають лабораторні роботи у відповідні терміни (на кожну лабораторну роботу відводиться два тижні для здачі),
- пишуть модульну контрольну роботу,
- повинні позитивно закрити дві атестації (в кінці березня та в середині травня),
- по закінченні навчального процесу складають екзамен.

Система вимог, які викладач ставить перед студентом:

- правила відвідування занять: заборонено оцінювати присутність або відсутність здобувача на аудиторному занятті, в тому числі нараховувати заохочувальні або штрафні бали. Відповідно до РСО даної дисципліни бали нараховують за відповідні види навчальної активності на лекційних та практичних заняттях.
- правила поведінки на заняттях: студент має можливість отримувати бали за відповідні види навчальної активності на лекційних та практичних заняттях, передбачені РСО дисципліни. Використання засобів зв'язку для пошуку інформації на гугл-диску викладача, в інтернеті, в дистанційному курсі на платформі Сікорський здійснюється за умови вказівки викладача;
- політика дедлайнів та перескладань: якщо студент не проходив або не з'явився на МКР (без поважної причини), його результат оцінюється у 0 балів. Перескладання результатів МКР не передбачено;
- політика щодо академічної доброчесності: Кодекс честі Національного технічного університету України «Київський політехнічний інститут» https://kpi.ua/files/honorcode.pdf встановлює загальні моральні принципи, правила етичної поведінки осіб та передбачає політику академічної доброчесності для осіб, що працюють і навчаються в університеті, якими вони мають керуватись у своїй діяльності, в тому числі при вивченні та складанні контрольних заходів з дисципліни «Системи автоматизації»;
- при використанні цифрових засобів зв'язку з викладачем (мобільний зв'язок, електронна пошта, переписка на форумах та у соцмережах тощо) необхідно дотримуватись загальноприйнятих етичних норм, зокрема бути ввічливим та обмежувати спілкування робочим часом викладача.

7. Види контролю та рейтингова система оцінювання результатів навчання (РСО)

Поточний контроль: вправи на лекційних заняттях, тестування, МКР, виконання завдань до практичних занять, виконання та захист лабораторних робіт.

Календарний контроль: провадиться двічі на семестр як моніторинг поточного стану виконання вимог силабусу.

Семестровий контроль: залік.

Система рейтингових (вагових) балів та критерії оцінювання

1) Робота на лекціях

На лекціях може бути проведено бліцопитування студентів. Такі опитування проводяться на довільних лекціях 5 разів протягом семестру, наприкінці лекції. Ваговий бал за вірну відповідь - 1. Максимальна кількість балів, що може отримати кожен студент за семестр - 5.

2) Лабораторні практикуми

Максимальна кількість балів за усі виконані комп'ютерні практикуми дорівнює 60 балів. Розподіл балів серед лабораторних практикумів наступний:

№ з/п	Назва лабораторного практикуму	Кількість балів
1	Установка Docker та створення першого контейнера	12
2	Створення Docker-образа для веб-додатка	12
3	Робота з Docker Compose для багатоконтейнерного додатка	12
4	Розгортання додатка в Kubernetes	12
5	Моніторинг та масштабування додатка в Kubernetes	12
	Всього:	60

Критерії оцінювання:

Виконання лабораторного практикуму:

- виконаний своєчасно (протягом двох тижнів з моменту видачі), у повному обсязі – відповідний бал згідно номеру комп'ютерного практикуму;
- виконаний із запізненням – знімається 10 – 30% від максимальної кількості балів в залежності від терміну запізнення;
- виконаний не самостійно, із запізненням – знімається 50% від максимальної кількості балів;
- невиконани1 протягом відведеного часу – 0 балів.

3) Модульна контрольна робота

Максимальна кількість балів за модульну контрольну роботу дорівнює 10 балів.

Якість виконання роботи:

- усі відповіді вірні та повні –10 балів,
- у відповідях допущені несуттєві неточності – 8 балів,
- половина відповідей вірна – 5 балів,
- відповіді з суттєвими неточностями, але без критичних помилок – 2 бали,
- менше половини відповідей вірна – 0 балів.

Штрафні та заохочувальні бали за:

- активність на комп'ютерних практикумах + 2 бали
- виконання комп'ютерного + 1 бали
практикуму з використанням власного оптимального алгоритму
- відсутність на занятті без поважної причини – 2 бали
- несвоєчасна здача комп'ютерного практикуму (пізніше ніж за тиждень) – 0,5 балів;

4) Складання заліку

Максимальний ваговий бал $r_{icп}$=40

На іспиті студент виконує письмову контрольну роботу, яка містить два теоретичних питання і одне практичне питання. Теоретичні питання оцінюються максимально по 10 балів, практичне – 20 балів.

Умови позитивної проміжної атестації

Для отримання „зараховано" з першої проміжної атестації студент матиме не менше ніж 30 балів (за умови, що за 8 тижнів згідно з календарним планом контрольних заходів „ідеальний" студент має отримати 12 +12 +12 = 36 бал).

Для отримання „зараховано" з другої проміжної атестації студент матиме не менше ніж 50 балів (за умови, що за 14 тижнів згідно з календарним планом контрольних заходів „ідеальний" студент має отримати 36 +12 +12 =60 балів).

Умови допуску до *заліку*

Необхідною умовою допуску до заліку є зарахування усіх комп'ютерних практикумів та виконання модульної контрольної роботи, а також стартовий рейтинг (R_c) не менше 40 балів.

Розрахунок шкали (R) рейтингу:

Сума вагових балів контрольних заходів протягом семестру (шкала рейтингу) складає:

$$R = r_{лек} + r_{практ} + r_{мод} + r_{icп} = 5+45 + 10 + 40= 100 \text{ балів.}$$

Максимальний стартовий рейтинг становить $R_c = r_{лек} + r_{практ} + r_{мод} = 60$ балів.

Рейтинг заліку дорівнює 40 балів. Мінімальний рейтинг допуску до заліку становить 40 балів.

Таким чином, рейтингова шкала з кредитного модуля складає

$$R = 60+40=100 \text{ балів.}$$

Для отримання студентом відповідних оцінок рейтингова оцінка студента переводиться згідно таблиці відповідності рейтингових балів оцінкам за університетською шкалою:

Бали	**Оцінка**
95 - 100	Відмінно
85 - 94	Дуже добре
75 - 84	Добре
65 - 74	Задовільно
60 - 64	Достатньо
Менше 60	Незадовільно
R < 40 є незараховані роботи комп'ютерного практикуму або не виконані інші умови допуску до екзамену	Не допущено

8. Додаткова інформація з дисципліни (освітнього компонента)

Перелік питань, які виносяться на семестровий контроль:

1. Контейнеризація і які її основні переваги у порівнянні з віртуалізацією.
2. Роль Docker у процесі контейнеризації? Охарактеризуйте основні команди Docker
3. Dockerfile. Описати його основні інструкції та структуру.
4. Яким чином Docker використовує системні ресурси для ізоляції контейнерів?
5. Охарактеризувати концепцію багатоконтейнерних додатків. Як Docker Compose допомагає в управлінні такими додатками?
6. Що таке образ Docker? Яким чином створюються та зберігаються образи?
7. Пояснити концепції мережевих моделей Docker. Які існують види мереж у Docker?
8. Kubernetes. Які завдання вирішує система оркестрації контейнерів?
9. Назвати основні компоненти архітектури Kubernetes та їхнє призначення.
10. Роль Pod у Kubernetes у системі оркестрації контейнерів?
11. Як працюють Service та LoadBalancer у Kubernetes? Яка їхня роль у маршрутизації трафіку?
12. Описати процес розгортання додатків у кластері Kubernetes. Що таке Deployment і як він використовується?

13. Роль ReplicaSet в автоматичному масштабуванні додатків?
14. Як забезпечується автоматичне горизонтальне масштабування додатків у Kubernetes?
15. Які типи зберігання даних підтримуються в Docker та Kubernetes? Що таке Volume та PersistentVolume?
16. Що таке ConfigMap та Secret у Kubernetes? Як ці об'єкти використовуються для зберігання конфігурацій?
17. Описати процес забезпечення безпеки контейнерних додатків у Docker та Kubernetes. Які є інструменти для цього?
18. Як Docker та Kubernetes інтегруються у процеси CI/CD? Опишіть основні етапи автоматизації розгортання додатків.
19. Що таке Helm та як він допомагає у керуванні додатками в Kubernetes?
20. Які інструменти використовуються для моніторингу контейнерів та кластерів Kubernetes? Опишіть принципи роботи Prometheus та Grafana.

Складено старшим викладачем ІПЗЕ, Колумбетом Вадимом Петровичем

Ухвалено кафедрою інженерії програмного забезпечення в енергетиці НН ІАТЕ (протокол № 34 від 10.05.2024 р.)
Погоджено Методичною комісією НН ІАТЕ (протокол № 9 від 31.05.2024 р.)