Черкаський національний університет імені Богдана Хмельницького
ННІ Інформаційних та освітніх технологій
Кафедра прикладної математики та інформатики

ЗАТВЕРДЖЕНО
Завідувач кафедри
ПМ та інформатики
_____/О.В. Піскун

РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ «АРХІТЕКТУРА ТА РОЗРОБКА ВЕБ-ДОДАТКІВ»

1. Загальна інформація про курс

Назва курсу, мова викладання	АРХІТЕКТУРА ТА РОЗРОБКА ВЕБ-ДОДАТКІВ Курс викладається українською та англійською мовами.
Статус дисципліни	Обов'язкова
Викладачі	Дідковський Р.М., д.т.н., доцент, доцент кафедри прикладної математики та інформатики
Код класу	
Корпоративна пошта:	didkovskyirm@vu.cdu.edu.ua
Затвердження та перегляд робочої навчальної програми	Розглянуто та затверджено на засіданні кафедри 28.08.2024, протокол № 1

2. Анотація до курсу

Навчальна дисципліна «Архітектура та розробка веб-додатків» є курсом циклу професійної та практичної підготовки фахівця з інформаційних систем та технологій. Навчальна дисципліна є теоретичною та практичною основою сукупності знань та вмінь, що формують у фахівця в області інформаційних систем навички розробки програмних додатків, орієнтованих на роботу у глобальній мережі Інтернет.

Вивчення навчальної дисципліни рекомендується планувати, починаючи з другого семестру.

3. Мета та цілі курсу

Метою курсу є формування у студентів теоретичних знань та практичних навичок з розробки веб-орієнтованих додатків мовою програмування Java з використанням спеціалізованих бібліотек, фреймворків та інструментів, таких як: Spring, Spring JDBC, Swagger, Gradle, Hibernate.

Завданнями вивчення навчальної дисципліни «Архітектура та розробка веб-додатків» передбачено:

- набуття та систематизація студентами знань і практичних навичок з розробки програм мовою програмування Java;
- набуття та систематизація студентами знань і практичних навичок з використання при розробці програм фреймворку Spring та його додатків: Spring JDBC, Spring Boot;
- набуття та систематизація студентами знань і практичних навичок з розробки REST-додатків;
- набуття та систематизація студентами знань і практичних навичок з використання спеціалізованих інструментів для організації неперервної інтеграції та неперервної доставки розроблених додатків на сервер;
- формування відповідних компетентностей, необхідних для виконання вказаних вище задач.

4. Компетентності та очікувані результати навчання

Навчальна дисципліна «Архітектура та розробка веб-додатків» забезпечує формування таких компетентностей, передбачених освітньою програмою підготовки магістрів спеціальності 126 Інформаційні системи та технології:

Інтегральна компетентність: здатність розв'язувати задачі дослідницького та інноваційного характеру у сфері інформаційних систем та технологій.

Загальні компетентності:

ЗК04. Здатність розробляти проекти та управляти ними.

ЗК05. Здатність оцінювати та забезпечувати якість виконуваних робіт.

ЗК06. Здатність застосовувати знання у практичних ситуаціях.

Фахові компетентності *(визначені стандартом та освітньою програмою компетентності, формування яких забезпечує ця навчальна дисципліна)*

СК01. Здатність розробляти та застосувати ІСТ, необхідні для розв’язання стратегічних і поточних задач.

СК02. Здатність формулювати вимоги до етапів життєвого циклу сервіс-орієнтованих інформаційних систем.

СК03. Здатність проектувати інформаційні системи з урахуванням особливостей їх призначення, неповної/недостатньої інформації та суперечливих вимог.

СК08. Здатність управляти та користуватися сучасними інформаційно-комунікаційними системами та технологіями, у першу чергу, орієнтованими на роботу у локальній та глобальній мережі.

СК09. Здатність розв'язувати практичні завдання, використовуючи знання систем хмарних обчислень, архітектури та стандартів комунікаційних засобів розподілених обчислень, концепцій паралельної обробки інформації.

Згідно з вимогами освітньо-професійної програми, **програмними результатами вивчення** дисципліни «Архітектура та розробка веб-додатків» є такі:

РН01. Відшуковувати необхідну інформацію в науковій і технічній літературі, базах даних, інших джерелах, аналізувати та оцінювати цю інформацію.

РН03. Приймати ефективні рішення з проблем розвитку інформаційної інфраструктури, створення і застосування ІСТ.

РН04. Управляти процесами розробки, впровадження та експлуатації у сфері ІСТ, які є складними, непередбачуваними і потребують нових стратегічних та командних підходів.

РН06. Обґрунтовувати вибір технічних та програмних рішень з урахуванням їх взаємодії та потенційного впливу на вирішення організаційних проблем, організовувати їх впровадження та використання.

РН07. Здійснювати обґрунтований вибір проектних рішень та проектувати сервіс-орієнтовану інформаційну архітектуру підприємства (установи, організації тощо).

РН12. Демонструвати знання сучасного рівня технологій інформаційних систем з метою їх запровадження у професійній діяльності; знати принципи функціонування та технології віртуалізації серверних систем, архітектури, та стандарти комунікаційних засобів розподілених обчислень; вміти розробляти програмне забезпечення різного рівня складності, що входить до складу інформаційних систем та технологій, при розв’язанні прикладних науково-виробничих задач і задач бізнесу.

5. Обсяг і характеристика курсу

Найменування показників	**Характеристика навчального курсу**
	денна форма навчання
Освітня програма, спеціальність	Веб-орієнтовані інформаційні системи, 126 Інформаційні системи та технології
Рік навчання	1, 2
Семестр вивчення	2, 3
обов’язкова /вибіркова	обов’язкова

Кількість кредитів ЄКТС	6
Загальний обсяг годин	180
Кількість годин навчальних занять	60
Лекційні заняття	20
Практичні заняття	0
Семінарські заняття	0
Лабораторні заняття	40
Самостійна та індивідуальна робота	120
Форма підсумкового контролю	екзамен

6. Пререквізити курсу

Для вивчення курсу студенти не потребують базових знань з навчальних дисциплін, що викладаються на освітньому ступені магістра, але повинні мати базові знання з алгоритмізації та програмування, мов програмування, об'єктно-орієнтованого програмування, ОС, операційної системи Linux та її адміністрування, комп'ютерних мереж, що викладаються на освітньому ступені бакалавра.

7. Технічне забезпечення

Для вивчення курсу потрібно встановити набір для розробки програм мовою Java – JDK, що вільно розповсюджуваний, інструментальне середовище на зразок вільно розповсюджуваного Visual Studio Code чи Sublime Text, операційну систему Linux на фізичній чи віртуальній машині, програмне забезпечення для створення та запуску веб-сервера. Для виконання домашніх завдань додатково потрібні загальновживані офісні програми та онлайн-сервіси.

8. Політика курсу

Академічна доброчесність. Очікується, що роботи студентів будуть їх оригінальними розробками. Фабрикування результатів, списування, втручання у роботу інших студентів можуть бути кваліфіковані як академічна недоброчесність. Виявлення ознак академічної недоброчесності у вихідному коді студента є підставою для незарахування домашнього завдання викладачем, незалежно від масштабів плагіату.

Відвідування занять. Відвідування занять є важливою складовою навчання. Очікується, що всі студенти відвідають усі лекції і практичні заняття курсу. Студенти мають інформувати викладача про неможливість відвідати заняття (за наявності поважної причини), у такому випадку допускається підключення студентів онлайн і їх дистанційна робота. Допускається по 2 пропуски лекцій та лабораторних робіт з поважних причин, які не впливатимуть на систему оцінювання. Студенти зобов'язані дотримуватися усіх строків, визначених для виконання усіх видів робіт, передбачених курсом.

9. Схема курсу

Тема, основні питання / завдання	Розподіл годин за темами та формами занять	Форми та методи проведення	Література. Ресурси в інтернеті	Завдання для самостійної роботи, год	Форма контролю, бали
ЗМІСТОВИЙ МОДУЛЬ 1. ОСНОВИ МОВИ ПРОГРАМУВАННЯ JAVA					
Лекція 1. Вступ до курсу. Мова програмування Java 1. Огляд мови програмування Java. 2. Поняття JVM, JRE, JDK. 3. Основи Java: змінні, типи даних, оператори, структури. 4. Інструменти, що будуть використовуватись у курсі: система контролю версій git.	1	Лекція-візуалізація (з використанням презентації)	Основна: 1, 2 Додаткова: 1	1. Опрацювання літератури з теми лекції (2 год.)	
Лабораторне заняття 1. Мова програмування Java 1. Огляд мови програмування Java. 2. Поняття JVM, JRE, JDK. 3. Основи Java: змінні, типи даних, оператори, структури. 4. Інструменти, що будуть використовуватись у курсі: система контролю версій git.	2	Практична робота. Матеріали: ознайомчі та демонстраційні версії програмного забезпечення, вихідний код програми	Основна: 1, 2 Додаткова: 1	1. Завершіть реалізацію системи кодування/декодування, представленої на занятті. Алфавіти та правила подані на сайті https://cryptii.com/. (4 год.)	Завдання 1: 0-2 бали
Лекція 2. Вбудовані класи 1. Вбудовані класи: String, StringBuilder та StringBuffer, класи Util. 2. Класи-обгортки: ієрархія, упаковка та розпаковка. 3. Інформація про систему.	1	Лекція-візуалізація (з використанням презентації)	Основна: 1, 2 Посилання: 1, 2, 3, 4, 5, 6	1. Опрацювання літератури з теми лекції (2 год.)	
Лабораторне заняття 2. Вбудовані класи 1. Вбудовані класи: String, StringBuilder та StringBuffer, класи Util. 2. Класи-обгортки: ієрархія, упаковка та розпаковка. 3. Інформація про систему.	2	Практична робота. Матеріали: ознайомчі та демонстраційні версії програмного забезпечення, вихідний код	Основна: 1, 2 Посилання: 1, 2, 3, 4, 5, 6	1. Використовуючи реалізацію попереднього домашнього завдання, розробіть повністю "дружній до користувача" консольний додаток, який приймає вхідні дані з клавіатури та виводить результат на консоль. (4 год.)	Завдання 1: 0-2 бали

		програми			
Лекція 3. Об'єктно-орієнтована парадигма 1. Структура класу. 2. Реалізація інкапсуляції. 3. Реалізація успадкування, включаючи модифікатори видимості та об'єднання. 4. Реалізація поліморфізму.	1	Лекція-візуалізація (з використанням презентації)	Основна: 1, 2 Додаткова:2 Посилання: 7, 8, 9	1. Опрацювання літератури з теми лекції (2 год.)	
Лабораторне заняття 3. Об'єктно-орієнтована парадигма 1. Структура класу. 2. Реалізація інкапсуляції. 3. Реалізація успадкування, включаючи модифікатори видимості та об'єднання. 4. Реалізація поліморфізму.	2	Практична робота. Матеріали: ознайомчі та демонстраційні версії програмного забезпечення, вихідний код програми	Основна: 1, 2 Додаткова:2 Посилання: 7, 8, 9, 10	1. Розширте реалізацію попереднього домашнього завдання наступними функціями: - реалізуйте кодер та декодер шифру Віженера - реалізуйте шифратор та дешифратор шифру Віженера2x (звичайний шифр Віженера, який застосовує кодування/дешифрування двічі) - реалізуйте шифр Віженера над шифром Цезаря - реалізуйте декодер шифру Віженера над шифром Цезаря. (4 год.)	Завдання 1: 0-2 бали
Лекція 4. Узагальнене програмування. Фреймворк колекцій. 1. Повторне використання коду за допомогою дженериків. 2. Використання вбудованих контейнерів: List, Set, Map.	1	Лекція-візуалізація (з використанням презентації)	Основна: 1, 2 Посилання: 11, 12, 13	1. Опрацювання літератури з теми лекції (2 год.)	
Лабораторне заняття 4. Узагальнене програмування. Фреймворк колекцій. 1. Повторне використання коду за допомогою дженериків. 2. Використання вбудованих контейнерів: List, Set, Map.	2	Практична робота. Матеріали: ознайомчі та демонстраційні версії програмного забезпечення, вихідний код програми	Основна: 1, 2 Посилання: 11, 12, 13	1. Оновіть кодек Морзе для зберігання фіксованих відображень алфавіту (Морзе) як Map. (0.5 год.) 2. Реалізуйте вказану аналітику у edu.chnu.javaforweb.analytics.CodingAudit. (0.5 год.) 3. Додайте можливість взаємодії з CodingAudit з існуючого меню. (0.5 год.) 4. Оновіть реалізацію історії дій користувача для використання списку замість простого масиву. (0.5 год.)	Завдання 1-3: 0-1 бал Завдання 4-7: 0-1 бал

				5. Додайте можливість очищення історії. (0.5 год.) 6. Додайте можливість видалення останнього елемента з історії. (0.5 год.) 7. Додайте можливість кодування/ декодування речень, що складаються з кількох слів. (1 год.)	
Модульна контрольна робота	2	Тестування	Основна: 1, 2 Додаткова: 1, 2 Посилання: 1-13	Повторити навчальний матеріал тем 1-4.	4 бали
Всього балів за змістовим модулем 1					**12**
Всього годин за змістовим модулем 1	**36**				
Лекцій	**4**				
Семінарських занять	**0**				
Лабораторних занять	**8**				
Самостійна робота	**24**				
ЗМІСТОВИЙ МОДУЛЬ 2. РОЗШИРЕНІ ТА СПЕЦІАЛЬНІ МОЖЛИВОСТІ JAVA					
Лекція 5. Тестування коду. 1. Поняття тестування коду 2. Написання тестів. 3. Покриття коду. 4. Методики тестування. 5. Unit-тестування. 6. Робота з IDE. 7. Test Driven Development.	2	Лекція-візуалізація (з використанням презентації)	Додаткова: 3, 4 Посилання: 14, 15, 16, 17	1. Опрацювання літератури з теми лекції (4 год.)	
Лабораторне заняття 5. Тестування коду. 1. Поняття тестування коду 2. Написання тестів. 3. Покриття коду. 4. Методики тестування. 5. Unit-тестування. 6. Робота з IDE. 7. Test Driven Development.	4	Практична робота. Матеріали: ознайомчі та демонстраційні версії програмного забезпечення, вихідний код програми	Додаткова: 3, 4 Посилання: 14, 15, 16, 17	1. Напишіть тести на існуючі кодеки (Caesar, Morse, Vigenere). (4 год.) 2. Напишіть тести на CodingAudit. (2 год.) 3. Напишіть тести на CodingHistory. (2 год.)	Завдання 1: 0-2 бали Завдання 2-3: 0-2 бали
Лекція 6. Функціональне	1	Лекція-візуалізація (з	Основна: 1,	1. Опрацювання літератури з теми лекції	

програмування. 1. Реалізація функціонального програмування у Java: функціональний інтерфейс, lambda-функції 2. Stream API: java.util.Stream, java.util.Optional		використанням презентації)	2 Посилання: 18, 19, 20, 21, 22	(2 год.)	
Лабораторне заняття 6. Функціональне програмування. 1. Реалізація функціонального програмування у Java: функціональний інтерфейс, lambda-функції 2. Stream API: java.util.Stream, java.util.Optional.	2	Практична робота. Матеріали: ознайомчі та демонстраційні версії програмного забезпечення, вихідний код програми	Основна: 1, 2 Посилання: 18, 19, 20, 21, 22	1. Реалізуйте кодери та декодери Цезаря, Морзе, Віженера з використанням: - лямбда-функцій - посилань на метод. (1.5 год.) 2. Змініть реалізацію CodingAudit за допомогою Stream API (всі 3 методи можна реалізувати за допомогою потоків). (1 год.) 3. Змініть реалізацію кодеків Caeser, Morse, Vigenere за допомогою Stream API (використовуйте потоки тільки якщо вони можуть спростити реалізацію алгоритму або читання коду). (1.5 год.)	Завдання 1: 0-1 бал Завдання 2-3: 0-1 бал
Лекція 7. Виключні ситуації та засоби введення/виведення. 1. Виключні ситуації: Exceptions, Assertions. 2. Засоби введення/виведення: - IO, NIO/NIO2 - керування ресурсами - серіалізація - кодування.	2	Лекція-візуалізація (з використанням презентації)	Основна: 1, 2 Додаткова: 5 Посилання: 23, 24, 25, 26	1. Опрацювання літератури з теми лекції (4 год.)	
Лабораторне заняття 7. Виключні ситуації та засоби введення/виведення. 1. Виключні ситуації: Exceptions, Assertions. 2. Засоби введення/виведення: - IO, NIO/NIO2 - керування ресурсами - серіалізація - кодування.	4	Практична робота. Матеріали: ознайомчі та демонстраційні версії програмного забезпечення, вихідний код програми	Основна: 1, 2 Додаткова: 5 Посилання: 23, 24, 25, 26	1. Реалізуйте ієрархію виключних ситуацій. Оновіть додаток "Coding" для повернення помилок та правильної обробки їх у споживачах: - IllegalCharacterException - коли на вхід кодеку(ам) передано непідтримувані дані (символ/слово); - CodecUnsupportedException - якщо вибрано непідтримуваний кодек; - EmptyHistoryException - коли не вдається знайти найпопулярніший кодек, оскільки ще не було виконано жодної операції	Завдання 1: 0-1 бал

				кодування; - OperationUnsupportedException - коли вибрано непідтримувану операцію. (2 год.) 2. Зробіть історію постійною. Зберігайте та відновлюйте при запуску історію операцій кодування. У якості компонента для зберігання слід використовувати файл. (2 год.) 3. Записуйте усі повідомлення до консолі та до файлу у домашньому каталозі користувача через фасад логера (клас логера повинен записувати журнали у кілька місць, має бути можливість легко додавати місця). (2 год.) 4. Покрийте винятки та домашні завдання з введення-виведення відсутніми модульними тестами. (2 год.)	Завдання 2: 0-1 бал Завдання 3: 0-1 бал Завдання 4: 0-1 бал
Лекція 8. Reflection API, анотації. 1. Віртуальна машина Java та керування пам'яттю 2. Клас java.lang.reflect 3. Клас java.lang.annotation.	2	Лекція-візуалізація (з використанням презентації)	Основна: 1, 2 Посилання: 27, 28, 29, 30	1. Опрацювання літератури з теми лекції (4 год.)	
Лабораторне заняття 8. Reflection API, анотації. 1. Віртуальна машина Java та керування пам'яттю 2. Клас java.lang.reflect 3. Клас java.lang.annotation.	4	Практична робота. Матеріали: ознайомчі та демонстраційні версії програмного забезпечення, вихідний код програми	Основна: 1, 2 Посилання: 27, 28, 29, 30	1. Реалізуйте власні анотації: - @Codec(algorithm = Algorithms.MORSE/ Algorithms.CAESAR/…) – додайте цю анотацію до всіх класів кодеків зі вказаним алгоритмом; - @Key – може бути реалізованим лише для алгоритму Віженера; - @Shift - може бути реалізованим лише для алгоритму Цезаря. (4 год.) 2. Змініть реалізацію класів DecodersFactory та EncodersFactory: - приберіть оператор switch; - знайдіть клас з анотацією @Codec та алгоритмом = {algorithm} й реалізуйте інтерфейс Decoder/Encoder з використанням Reflection API;	Завдання 1: 0-2 бали Завдання 2: 0-2 бали

				- створіть екземпляр знайденого класу з конструктором без аргументів, використовуючи Reflection API; - встановіть значення key/shift (де можливо), використовуючи Reflection API (key/shift повинні бути позначені відповідною анотацією); - згенеруйте виключну ситуацію у випадку неправильного використання: -- @Codec – використовується для класу, що не реалізує інтерфейс Decoder/Encoder; -- @Key – використовується для класу, що не викристовує key (наприклад, Caesar); -- @Shift – використовується для класу, що не використовує shift (наприклад, Morse). (4 год.)	
Модульна контрольна робота	2	Тестування	Основна: 1, 2 Додаткова: 3-5 Посилання: 14-30	Повторити навчальний матеріал тем 5-8.	7 балів
Всього балів за змістовим модулем 2					21
Всього годин за змістовим модулем 2	**63**				
Лекцій	**7**				
Семінарських занять	**0**				
Лабораторних занять	**14**				
Самостійна робота	**42**				
ЗМІСТОВИЙ МОДУЛЬ 3. ВИКОРИСТАННЯ JAVA ДЛЯ РОЗРОБКИ WEB-ДОДАТКІВ					
Лекція 9. Servlet API (Web). 1. Протокол HTTP. 2. Архітектура клієнт-сервер. 3. Статичний та динамічний контент. 4. Java HTTP-сервлет. 5. Веб-сервер Tomcat. 6. Запуск веб-додатку на Java.	2	Лекція-візуалізація (з використанням презентації)	Посилання: 31, 32, 33, 34	1. Опрацювання літератури з теми лекції (4 год.)	
Лабораторне заняття 9. Servlet API	4	Практична робота.	Посилання:	1. Створіть пакет edu.chnu.javaworweb.	Завдання 1: 0-1

(Web). 1. Протокол HTTP. 2. Архітектура клієнт-сервер. 3. Статичний та динамічний контент. 4. Java HTTP-сервлет. 5. Веб-сервер Tomcat. 6. Запуск веб-додатку на Java.		Матеріали: ознайомчі та демонстраційні версії програмного забезпечення, вихідний код програми	31, 32, 33, 34	crypto.web для зберігання класів, пов'язаних з реалізацією веб-інтерфейсу. (2 год.) 2. Реалізуйте веб-версію криптографічного додатку з використанням HTTP-сервлетів. (2 год.) 3. Реалізуйте форму входу для аутентифікації користувача (обов'язковим є лише поле логіну). Додаток повинен мати 2 типи користувачів. (2 год.) 4. Реалізуйте фільтр логування запитів. (2 год.)	бал Завдання 2: 0-1 бал Завдання 3: 0-1 бал Завдання 4: 0-1 бал
Лекція 10. Фреймворк Spring (IoC, MVC, Boot). 1. Основи Spring: - інверсія контролю - ін'єкція залежності - ядро Spring - порівняння конфігурації XML, конфігурації анотацій та конфігурації Java. 2. Spring Web: - Шаблон Model-View-Controller - Spring MVC: конфігурація, контролери, конвертери повідомлень, перехоплювачі.	2	Лекція-візуалізація (з використанням презентації)	Додаткова: 6 Посилання: 35, 36, 37, 38	1. Опрацювання літератури з теми лекції (4 год.)	
Лабораторне заняття 10. Фреймворк Spring (IoC, MVC, Boot). 1. Основи Spring: - інверсія контролю - ін'єкція залежності - ядро Spring - порівняння конфігурації XML, конфігурації анотацій та конфігурації Java. 2. Spring Web: - Шаблон Model-View-Controller - Spring MVC: конфігурація, контролери, конвертери повідомлень, перехоплювачі.	4	Практична робота. Матеріали: ознайомчі та демонстраційні версії програмного забезпечення, вихідний код програми	Додаткова: 6 Посилання: 35, 36, 37, 38	1. Зробіть усі кодеки (Caesar, Morse, Vigenere тощо) компонентами Spring: видаліть анотації @Codec, @Key, @Shift та використайте властивості, які можуть бути введені Spring для встановлення ключа та зсуву. (2 год.) 2. Змініть заводську реалізацію кодеків: - вставте усі кодеки у фабрику (використовуючи Spring) - повертайте той же екземпляр кодека з фабрики за запитом. (1 год.) 3. Використайте контролери замість сервлетів. (2 год.) 4. Використайте перехоплювачі для	Завдання 1: 0-2 бали Завдання 2: 0-1 бал Завдання 3-5: 0-1 бал

				реалізації логування запитів. (2 год.) 5. Використайте обробники виключень замість відображень помилок у web.xml. (1 год.)	
Лекція 11. JDBC, Spring JDBC, Flyway. 1. JDBC: - підключення до бази даних (БД) - маніпулювання структурою бази даних - робота зі вмістом бази даних. 2. Spring JdbcTemplate 3. Flyway.	2	Лекція-візуалізація (з використанням презентації)	Посилання: 39, 40, 41, 42, 43	1. Опрацювання літератури з теми лекції (4 год.)	
Лабораторне заняття 11. JDBC, Spring JDBC, Flyway. 1. JDBC: - підключення до бази даних (БД) - маніпулювання структурою бази даних - робота зі вмістом бази даних. 2. Spring JdbcTemplate 3. Flyway.	4	Практична робота. Матеріали: ознайомчі та демонстраційні версії програмного забезпечення, вихідний код програми	Посилання: 39, 40, 41, 42, 43	1. Інтегруйте базу даних до проекту: - таблиці повинні автоматично заповнюватися при запуску програми. Якщо таблиці вже існують - пропустіть цей крок; - використовуйте Flyway для створення структури бази даних (і заповнення даними) при запуску програми; - оголосіть хост, логін, пароль та ім'я бази даних у файлі application.properties, розташованому в ресурсах проекту; - використовуйте Spring Jdbc та джерело даних Hikari у вашому додатку замість прямого використання з'єднань та операторів. (4 год.) 2. Перенесіть сховище історії з файлу до реляційної бази даних (PostgreSQL). (4 год.)	Завдання 1: 0-2 бали Завдання 2: 0-2 бали
Лекція 12. REST, Swagger. 1. Поняття REST. 2. Принципи REST. 3. API Swagger.	1	Лекція-візуалізація (з використанням презентації)	Додаткова: 7 Посилання: 44, 45, 46	1. Опрацювання літератури з теми лекції (2 год.)	
Лабораторне заняття 12. REST, Swagger. 1. Поняття REST. 2. Принципи REST. 3. API Swagger.	2	Практична робота. Матеріали: ознайомчі та демонстраційні версії програмного забезпечення,	Додаткова: 7 Посилання: 44, 45, 46	1. Реалізуйте REST API зі збереженням існуючих веб-інтерфейсу та консольного інтерфейсу. Надайте можливість використовувати всі існуючі функції (кодування/декодування, історія,	Завдання 1: 0-1 бали

		вихідний код програми		аналітика), але працюйте з необробленим JSON через Swagger UI. Використовуйте відповідні HTTP-методи та HTTP-коди стану. (2 год.) 2. Використовуйте ControllerAdvice для зіставлення винятків з об'єктами описової відповіді з відповідними кодами стану HTTP. (2 год.)	Завдання 2: 0-1 бали
Лекція 13. Gradle, інтеграційне тестування. 1. Gradle. 2. Безперервна інтеграція (Continuous Integration). 3. Безперервна доставка (Continuous Delivery). 4. Корисні функції Spring Boot. 5. Моніторинг. 6. Інтеграційне тестування: - піраміда тестування; - макети; - концепція інтеграційного тестування; - інтеграційне тестування Spring Boot.	1	Лекція-візуалізація (з використанням презентації)	Посилання: 47, 48, 49, 50, 51	1. Опрацювання літератури з теми лекції (2 год.)	
Лабораторне заняття 13. Gradle, інтеграційне тестування. 1. Gradle. 2. Безперервна інтеграція (Continuous Integration). 3. Безперервна доставка (Continuous Delivery). 4. Корисні функції Spring Boot. 5. Моніторинг. 6. Інтеграційне тестування: - піраміда тестування; - макети; - концепція інтеграційного тестування; - інтеграційне тестування Spring Boot.	4	Практична робота. Матеріали: ознайомчі та демонстраційні версії програмного забезпечення, вихідний код програми	Посилання: 47, 48, 49, 50, 51	1. Налаштуйте Spring Actuator: - увімкніть кінцеву точку /shutdown - увімкніть кінцевий пункт /metrics - увімкніть інші кінцеві точки на ваш смак - налаштуйте актуатор на окремому порту. (2 год.) 2. Додайте інтеграцію з Sentry: - https://sentry.io/for/spring/ - Sentry.sentry-spring-boot-starter. (2 год.) 3. Покрийте існуючий функціонал інтеграційними тестами. (4 год.)	Завдання 1: 0-1 бал Завдання 2: 0-1 бал. Завдання 3: 0-2 бали.
Лекція 14. ORM: Hibernate, Spring Data. 1. Поняття ORM.	1	Лекція-візуалізація (з використанням	Посилання: 52, 53, 54,	1. Опрацювання літератури з теми лекції (2 год.)	

2. Огляд Hibernate. 3. Конфігурація Hibernate. 4. Порівняння SessionFactory та EntityManager. 5. Типи вибірки (eager, lazy). 6. Режими вибірки (вибірка, об'єднання, підвибірка). 7. Проблема N+1. 8. Огляд Spring Data.		презентації)	55, 56		
Модульна контрольна робота	2	Тестування	Основна: 1, 2 Додаткова: 6-7 Посилання: 31-56	Повторити навчальний матеріал тем 9-14.	9 балів
Всього балів за змістовим модулем 3					**27**
Всього годин за змістовим модулем 3	**81**				
Лекцій	**9**				
Семінарських занять	**0**				
Лабораторних занять	**18**				
Самостійна робота	**54**				
Підсумковий контроль: екзамен		Тестування			**40** .

10. Система оцінювання та вимоги

Навчальні досягнення студентів оцінюються за 100-бальною шкалою Університету, чотирибальною шкалою (5 «відмінно», 4 «добре», 3 «задовільно», 2 «незадовільно»), і шкалою оцінок ЄКТС. На поточний контроль відводиться 100 балів.

Оцінювання поточної успішності студентів на окремих навчальних заняттях та за виконання завдань самостійної роботи визначається диференційовано, відповідно до рівня складності завдань, та встановлюється в межах від 0 до 4 балів.

Кількість балів за роботу з теоретичним матеріалом та на лабораторних заняттях залежить від дотримання таких вимог:

- своєчасність виконання навчальних завдань;
- повний обсяг їх виконання;
- якість виконання навчальних завдань;
- самостійність виконання;
- творчий підхід у виконанні завдань.

Виконання модульних контрольних робіт та завдань самостійної роботи є обов'язковим. До їх виконання допускаються всі студенти. Студент, який не виконав поточних завдань, не підготувався до лабораторних занять, отримує 0 балів. Поточну заборгованість, пов'язану з непідготовленістю або недостатньою підготовленістю до навчальних занять, студент повинен ліквідувати шляхом виконання у визначений термін завдань, передбачених програмою. За виконані завдання нараховуються від 0 до 6 балів.

Студенти, які за результатами поточного контролю набрали менше 20 балів, вважаються такими, що мають академічну заборгованість, ліквідація якої є обов'язковою. Студенти, які не мають академічної заборгованості за результатами поточного контролю, допускаються до екзамену.

11. Критерії оцінювання успішності навчання

1. Завданням **поточного контролю** є систематична перевірка розуміння та засвоєння програмного матеріалу шляхом тестування, аналіз виконання завдань самостійної роботи, умінь у електронному форматі представляти певний матеріал.

Критеріями оцінювання у ході поточного контролю є:

а) під час поточної аудиторної роботи на лекційних та семінарських заняттях:

- активна участь у дискусіях та пропонованих формах роботи на лекційних та лабораторних заняттях;
- доповнення та запитання на лекційних та лабораторних заняттях;

б) при виконанні завдань для самостійної та індивідуальної роботи:

- повнота виконання завдання;
- творчість та самостійність виконання.

2. Критерії оцінювання **модульної контрольної роботи**. Модульна контрольна робота містить кілька (у залежності від модуля) тестових завдань, кожне з яких оцінюється 1 балом.

Завданням **підсумкового контролю** (екзамену) є комплексна діагностика результатів навчання, глибини засвоєння студентом програмного матеріалу з навчальної дисципліни, логіки та взаємозв'язків між окремими його змістовими модулями, здатності до творчого використання набутих знань.

Екзамен містить 30 тестових завдань, кожне з яких оцінюється 1-2 балами.

Список рекомендованої літератури / інтернет-ресурси / нормативні документи

Основна

1. Schildt H. Java. The Complete Reference. Comprehensive Civerage of the Java Language / McGraw-Hill Education. – 2022.
2. Horstmann C.S. Core Java. Volume I: Fundamentals / Addison-Wesley. – 2022.
3. Horstmann C.S. Core Java. Volume II: Advanced Features / Addison-Wesley. – 2022.

Додаткова

1. Oggl B., Kofler M. Git. Project Management for Developers and DevOps Teams / Rheinwerk Publishing Inc., Boston (MA). – 2023.
2. Weisfeld M. The Object-Oriented Thought Process / Pearson Education, Inc. – 2019.
3. Gulati S., Sharma R. Java Unit Testing with JUnit 5. Test Driven Development with JUnit 5 / Apress. – 2017.
4. Langr J., Hunt A., Thomas D. Pragmatic Unit Testing in Java 8 with JUnit / The Pragmatic Programmers, LLC. – 2015.
5. Lumi N. Exceptions in Java. Basics, advanced concepts, and real API examples. – 2022.
6. Sarcar V. Java Design Patterns. A Hands-On Experience with Real-World Examples / Apress. – 2022.
7. Fielding R.T. Architectural Styles and the Design of Network-based Software Architectures. Dissertation submitted in partial satisfaction of the requirements for the degree of DOCTOR OF PHILOSOPHY in Information and Computer Science / University of California, Irvine. – 2000.

Посилання на інтернет-ресурси

1. The Numbers Classes: https://docs.oracle.com/javase/tutorial/java/data/numberclasses.html
2. Autoboxing and Unboxing: https://docs.oracle.com/javase/tutorial/java/data/autoboxing.html
3. Strings: https://docs.oracle.com/javase/tutorial/java/data/strings.html
4. Java User Input (Scanner): https://www.w3schools.com/java/java_user_input.asp
5. Arrays: https://docs.oracle.com/javase/tutorial/java/nutsandbolts/arrays.html
6. Java – Internal Caching in Wrapper: https://howtodoinjava.com/java-examples/internal-cache-wrapper-classes/
7. Lesson: Classes and Objects: https://docs.oracle.com/javase/tutorial/java/javaOO/index.html
8. Lesson: Interfaces and Inheritance: https://docs.oracle.com/javase/tutorial/java/IandI/index.html
9. Lesson: Object-Oriented Programming Concepts: https://docs.oracle.com/javase/tutorial/java/concepts/index.html
10. Vigenyre cipher: https://en.wikipedia.org/wiki/Vigen%C3%A8re_cipher
11. Lesson: Generics (Updated): https://docs.oracle.com/javase/tutorial/java/generics/index.html
12. Trail: Collections: https://docs.oracle.com/javase/tutorial/collections/index.html
13. Java Collections Cheat Sheet: https://www.jrebel.com/blog/java-collections-cheat-sheet
14. 7 Popular Unit Test Naming Conventions: https://dzone.com/articles/7-popular-unit-test-naming
15. Best Practices for Unit Testing in Java: https://www.developer.com/java/best-practices-unit-testing-java/
16. A Quick JUnit vs TestNG Comparison: https://www.baeldung.com/junit-vs-testng
17. The different types of software testing: https://www.atlassian.com/continuous-delivery/software-testing/types-of-software-testing
18. Lambda Expressions: https://docs.oracle.com/javase/tutorial/java/javaOO/lambdaexpressions.html
19. Java SE 8: Lambda Quick Start: https://www.oracle.com/webfolder/technetwork/tutorials/obe/java/Lambda-QuickStart/index.html
20. Chapter 15. Expressions: https://docs.oracle.com/javase/specs/jls/se8/html/jls-15.html
21. Package java.util.stream: https://docs.oracle.com/javase/8/docs/api/java/util/stream/package-summary.html
22. Java Streams Cheat Sheet: https://www.jrebel.com/blog/java-streams-cheat-sheet

23. Lesson: Exceptions: https://docs.oracle.com/javase/tutorial/essential/exceptions/index.html
24. Lesson: Basic I/O: https://docs.oracle.com/javase/tutorial/essential/io/
25. Introducing JSON: https://www.json.org/json-en.html
26. JSON in Java: https://www.baeldung.com/java-json
27. Trail: The Reflection API: https://docs.oracle.com/javase/tutorial/reflect/
28. Using Java Reflection: https://www.oracle.com/technical-resources/articles/java/javareflection.html
29. Lesson: Annotations: https://docs.oracle.com/javase/tutorial/java/annotations/
30. Class Loaders in Java: https://www.baeldung.com/java-classloaders
31. Introduction to Servlets and Servlet Containers: https://www.baeldung.com/java-servlets-containers-intro
32. What is Servlet Container: https://www.programcreek.com/2013/04/what-is-servlet-container/
33. Introduction to Java Servlets — Servlets in a Nutshell: https://medium.com/edureka/java-servlets-62f583d69c7e
34. Apache Tomcat 8: https://tomcat.apache.org/tomcat-8.5-doc/setup.html
35. Spring Tutorial: https://www.tutorialspoint.com/spring/index.htm
36. Spring Framework Documentation: https://docs.spring.io/spring-framework/reference/
37. Intro to Inversion of Control and Dependency Injection with Spring: https://www.baeldung.com/inversion-control-and-dependency-injection-in-spring
38. Spring - MVC Framework: https://www.tutorialspoint.com/spring/spring_web_mvc_framework.htm
39. Trail: JDBC Database Access: https://docs.oracle.com/javase/tutorial/jdbc/index.html
40. Introduction to JDBC: https://www.baeldung.com/java-jdbc
41. Accessing Relational Data using JDBC with Spring: https://spring.io/guides/gs/relational-data-access/
42. Spring JDBC: https://www.baeldung.com/spring-jdbc-jdbctemplate
43. Flyway Documentation: https://documentation.red-gate.com/fd/
44. Building REST services with Spring: https://spring.io/guides/tutorials/rest/
45. Testing the Web Layer: https://spring.io/guides/gs/testing-web/
46. Swagger UI: https://swagger.io/tools/swagger-ui/
47. Building an Application with Spring Boot: https://spring.io/guides/gs/spring-boot/
48. What is Continuous Integration (CI): https://www.jetbrains.com/teamcity/ci-cd-guide/continuous-integration/
49. Spring Boot Actuator: https://www.baeldung.com/spring-boot-actuators
50. Spring Profiles: https://www.baeldung.com/spring-profiles
51. Introduction to Spring Testing: https://docs.spring.io/spring-framework/reference/testing/introduction.html
52. Hibernate ORM: https://hibernate.org/orm/
53. Hibernate Tutorial: https://www.tutorialspoint.com/hibernate/index.htm
54. Accessing Data with JPA: https://spring.io/guides/gs/accessing-data-jpa/
55. Spring Data JPA - Reference Documentation: https://docs.spring.io/spring-data/jpa/docs/current/reference/html/
56. Spring Data JDBC: https://spring.io/projects/spring-data-jdbc