МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Львівський національний університет імені Івана Франка
Факультет електроніки та комп'ютерних технологій
Кафедра радіоелектронних і комп'ютерних систем

Затверджено
На засіданні
кафедри радіоелектронних і комп'ютерних систем
факультету електроніки та комп'ютерних технологій
Львівського національного університету імені Івана Франка
(протокол №11/24 від 28.08. 2023 р.)

Завідувач кафедри:

_____ Ігор ОЛЕНИЧ

Силабус з навчальної дисципліни
"Архітектура ПЗ ч.2",
що викладається в межах ОПП
" Високопродуктивний комп'ютинг "
першого (бакалаврського) рівня вищої освіти для здобувачів з спеціальності 121 – Інженерія програмного забезпечення

Львів 2023 р.

Назва дисципліни	Архітектура ПЗ (ч. 2)
Адреса викладання дисципліни	Корпус факультету електроніки та комп'ютерних технологій, Львівський національний університет імені Івана Франка, вул. Драгоманова 50, м. Львів, 79005, вул. Ген. Тарнавського 107, м. Львів, 79011
Факультет та кафедра, за якою закріплена дисципліна	Факультет електроніки та комп'ютерних технологій Кафедра радіоелектронних і комп'ютерних систем
Галузь знань, шифр та назва спеціальності	12 – інформаційні технології 121 – Інженерія програмного забезпечення
Викладачі дисципліни	Сінькевич О. О., доктор філософії з комп. н., доцент Оленич І.Б., д-р, фіз.-мат. наук, професор
Контактна інформація викладачів	oleh.sinkevych@lnu.edu.ua https://electronics.lnu.edu.ua/employee/o_sinkevych/ igor.olenych@lnu.edu.ua https://electronics.lnu.edu.ua/employee/olenych-i-b/
Консультації з питань навчання по дисципліні відбуваються	Консультації в день проведення лекцій/лабораторних занять (за попередньою домовленістю): ауд. 102, корпус факультету електроніки та комп'ютерних технологій, вул. Драгоманова 50, м. Львів
Сторінка курсу	https://e-learning.lnu.edu.ua/course/view.php?id=6022
Коротка анотація дисципліни	Дисципліна "Архітектура ПЗ (ч.2)" є нормативною дисципліною з спеціальності 121 – Інженерія програмного забезпечення для освітньої програми "Високопродуктивний комп'ютинг", яка викладається в 6-му семестрі в обсязі 5,5 кредитів (за Європейською Кредитно-Трансферною Системою ECTS).
Інформація про дисципліну	Навчальну дисципліну розроблено таким чином, щоб надати учасникам необхідні знання, обов'язкові для того, щоб брати участь у проектуванні програмного забезпечення, створенні типових програмних архітектур та управлінні структурою програмного забезпечення. Тому у дисципліні представлено як огляд сучасних підходів створення складних програмних систем, так і засобів та інструментів, які потрібні для проектування програмного забезпечення, реалізації основних методів і засобів автоматизації проектування, випробувань та оцінки якості програмного забезпечення.
Мета та цілі дисципліни	*Мета:* надати студентам основні поняття про структуру, інформаційні моделі та системи побудови програмного забезпечення, ознайомити студентів з типовими архітектурними стилями і шаблонами та показати, які архітектурні рішення слід використовувати при проектуванні програмного забезпечення. *Цілі:* ознайомлення студентів з основними підходами, архітектурними стилями та шаблонами при проектуванні програмного забезпечення, надати теоретичну базу для вибору технологій та набору програмних

	інструментів при проектуванні програмного забезпечення у межах специфічної архітектури.
Література для вивчення дисципліни	Основна література: 1. Martin R.C. Clean Architecture: A Craftsman's Guide to Software Structure and Design / Robert C. Martin. - Pearson, 2017. - 432 p. 2. Clements P., Bachmann F., Bass L. Documenting Software Architectures: Views and Beyond (2nd Edition). - Boston: Addison-Wesley Professional, 2010. 3. Gomaa H. Software Modeling and Design: UML, Use Cases, Patterns, and Software Architectures. – NY: Cambridge University Press, 2011. – 578 p. 4. Richards M., Ford N. Fundamentals of Software Architecture: An Engineering Approach. O'Reilly Media, 2020. 432 c. 5. Woods E., Pureur P., Erder M. Continuous Architecture in Practice: Software Architecture in the Age of Agility and DevOps. Pearson Education, Limited, 2021. 6. Software Architecture: The Hard Parts / M. Richards та ін. O'Reilly Media, Incorporated, 2021. 7. Hohpe G. Software Architect Elevator: Redefining the Architect's Role in the Digital Enterprise. O'Reilly Media, Incorporated, 2020. 366 c. 8. Software Architecture / ред.: S. Biffl та ін. Cham : Springer International Publishing, 2021.
Обсяг курсу	165 годин занять. З них 32 години лекцій, 32 години лабораторних робіт та 101 година самостійної роботи
Очікувані результати навчання	У результаті вивчення даного курсу студент буде: - *Знати* методологію та технологію збору основних вимог до програмного забезпечення та методи їх аналізу; технології проєктування програмного забезпечення; способи використання CASE засобів для аналізу предметного середовища та побудови архітектури програмного забезпечення. - *Вміти* виконувати аналіз вимог до програмного забезпечення, що розробляється; оцінювати трудомісткість і вибирати адекватні підходи до розробки програмного забезпечення; проектувати архітектуру програмного забезпечення з використанням засобів візуального моделювання; усвідомлено застосовувати методики випробувань і налагодження розроблюваного програмного забезпечення; проектувати компоненти архітектурного рішення. Після вивчення даного курсу «Архітектура ПЗ (ч. 2)» здобувачі набудуть таких Загальних(ЗК)/Фахових(ФК) компетентностей та Програмних результатів навчання (ПРН): ЗК01. Здатність до абстрактного мислення, аналізу та синтезу. ЗК02. Здатність застосовувати знання у практичних ситуаціях. ФК13. Здатність ідентифікувати, класифікувати та формулювати вимоги до програмного забезпечення. ФК14. Здатність брати участь у проектуванні програмного забезпечення, включаючи проведення моделювання (формальний опис) його структури, поведінки та процесів функціонування. ФК15. Здатність розробляти архітектури, модулі та компоненти програмних систем. ФК16. Здатність формулювати та забезпечувати вимоги щодо якості

	програмного забезпечення у відповідності з вимогами замовника, технічним завданням та стандартами. ФК17. Здатність дотримуватися специфікацій, стандартів, правил і рекомендацій в професійній галузі при реалізації процесів життєвого циклу. ФК18. Здатність аналізувати, вибирати і застосовувати методи і засоби для забезпечення інформаційної безпеки (в тому числі кібербезпеки). ФК19. Володіння знаннями про інформаційні моделі даних, здатність створювати програмне забезпечення для зберігання, видобування та опрацювання даних. ФК20. Здатність застосовувати фундаментальні і міждисциплінарні знання для успішного розв'язання завдань інженерії програмного забезпечення. ФК22. Здатність накопичувати, обробляти та систематизувати професійні знання щодо створення і супроводження програмного забезпечення та визнання важливості навчання протягом всього життя. ФК23. Здатність реалізовувати фази та ітерації життєвого циклу програмних систем та інформаційних технологій на основі відповідних моделей і підходів розробки програмного забезпечення. ФК24. Здатність здійснювати процес інтеграції системи, застосовувати стандарти і процедури управління змінами для підтримки цілісності, загальної функціональності і надійності програмного забезпечення. ФК25. Здатність обґрунтовано обирати та освоювати інструментарій з розробки та супроводження програмного забезпечення. ФК27. Здатність розробляти високопродуктивні програмні комплекси для вирішення задач наук про дані, систем штучного інтелекту, вбудованих та інших інноваційних систем. ПРН01. Аналізувати, цілеспрямовано шукати і вибирати необхідні для вирішення професійних завдань інформаційно-довідникові ресурси і знання з урахуванням сучасних досягнень науки і техніки. ПРН02. Знати кодекс професійної етики, розуміти соціальну значимість та культурні аспекти інженерії програмного забезпечення і дотримуватись їх в професійній діяльності. ПРН03. Знати основні процеси, фази та ітерації життєвого циклу програмного забезпечення. ПРН04. Знати і застосовувати професійні стандарти і інші нормативно-правові документи в галузі інженерії програмного забезпечення. ПРН05. Знати і застосовувати відповідні математичні поняття, методи доменного, системного і об'єктно-орієнтованого аналізу та математичного моделювання для розробки програмного забезпечення. ПРН06. Вміння вибирати та використовувати відповідну задачі методологію створення програмного забезпечення. ПРН07. Знати і застосовувати на практиці фундаментальні концепції, парадигми і основні принципи функціонування мовних, інструментальних і обчислювальних засобів інженерії програмного забезпечення. ПРН08. Вміти розробляти людино-машинний інтерфейс. ПРН09. Знати та вміти використовувати методи та засоби збору, формулювання та аналізу вимог до програмного забезпечення. ПРН11. Вибирати вихідні дані для проектування, куруючись формальними методами опису вимог та моделювання. ПРН12. Застосовувати на практиці ефективні підходи щодо проектування програмного забезпечення.

	ПРН14. Застосовувати на практиці інструментальні програмні засоби доменного аналізу, проектування, тестування, візуалізації, вимірювань та документування програмного забезпечення. ПРН15. Мотивовано обирати мови програмування та технології розробки для розв'язання завдань створення і супроводження програмного забезпечення. ПРН16. Мати навички командної розробки, погодження, оформлення і випуску всіх видів програмної документації. ПРН17. Вміти застосовувати методи компонентної розробки програмного забезпечення. ПРН19. Знати та вміти застосовувати методи верифікації та валідації програмного забезпечення. ПРН20. Знати підходи щодо оцінки та забезпечення якості програмного забезпечення. ПРН21. Знати, аналізувати, вибирати, кваліфіковано застосовувати засоби забезпечення інформаційної безпеки (в тому числі кібербезпеки) і цілісності даних відповідно до розв'язуваних прикладних завдань та створюваних програмних систем. ПРН24. Вміти проводити розрахунок економічної ефективності програмних систем. ПРН25. Вміти застосовувати інноваційні технологічні рішення при розробці високопродуктивних систем. ПРН26. Знати засоби інтеграції, розгортання та підтримки спеціалізованих програмних компонентів, розроблених на основі інноваційних технологій для вирішення завдань високопродуктивних технологій.
Ключові слова	Бізнес-логіка, структура програмного забезпечення, дизайн рівнів програмного забезпечення, патерни, CASE засоби, архітектурні шаблони, архітектурні стилі.
Формат курсу	Очний
Теми	Див. **Схема курсу**
Підсумковий контроль, форма	Залік в кінці семестру.
Пререквізити	Для вивчення даного курсу студентам потрібні базові знання з курсів: - архітектура ПЗ (ч.1); - основи програмування; - алгоритми і структури даних; - веб-технології та програмування (ч. 1) - об'єктно-орієнтоване програмування.
Навчальні методи та техніки, які будуть використовуватися під час викладання курсу	Інформаційні методи (лекції, презентації, лабораторні роботи, написання рефератів, виконання індивідуальних завдань, робота у групі, командна робота, обговорення, консультації для поглибленого розуміння тем, бесіда, ілюстрація, демонстрація), дедуктивні методи на основі узагальнень, евристичні методи (проблемна лекція), інтерактивні методи (дискусія).
Необхідне обладнання	Для проведення лекційних занять: комп'ютер (мінімальні характеристики: процесор Intel Core i3(4 ядра/8 потоків), 8ГБ оперативної пам'яті, 50ГБ вільного місця на диску, доступ до мережі Internet, засоби мультимедіа (в т.ч. проектор).

	Для проведення лабораторних занять: Комп'ютер (мінімальні характеристики: процесор Intel Core i3(4 ядра/8 потоків), 8ГБ оперативної пам'яті, 50ГБ вільного місця на диску. Необхідне програмне забезпечення включає в себе ОС Windows 10, або Ubuntu 22.04 LTS, середовище розробки MS Visual Studio/PyCharm/Vim/VS Code, компілятор мови програмування C++/Python.
Критерії оцінювання (окремо для кожного виду навчальної діяльності)	Оцінювання проводиться за 100-бальною шкалою. Бали нараховуються за наступним співвідношенням: • лабораторні роботи: 60% семестрової оцінки; максимальна кількість балів 60. • змістові модулі (2 модулі): 40% семестрової оцінки; максимальна кількість балів 40. Загалом упродовж семестру 100 балів. **Змістовий модуль** – самостійна робота студента оформлена у вигляді робочої програми та звіту – робота друкованим текстом, рекомендованим обсягом до 5 сторінок (шрифт Times New Roman, 14). Звіт включає в себе детальний розгляд завдання до модуля, приведення прикладів та лістингів коду програм, огляду технологій. Код програм повинен бути обов'язково прокоментований та пояснений, необхідно також продемонструвати його роботу у разі, якщо в якості прикладу наводяться не окремі елементи технології, а суцільна програма. Фінальна версія звіту у .pdf форматі разом з кодом завантажується на віддалений Git-репозиторій. **Академічна доброчесність:** Очікується, що роботи студентів будуть оригінальними дослідженнями чи міркуваннями. Списування та втручання в роботу інших студентів становлять, але не обмежують, приклади можливої академічної недоброчесності. Виявлення ознак академічної недоброчесності в написанні завдань є підставою для її незарахуванння викладачем, незалежно від масштабів плагіату чи обману. Жодні форми порушення академічної доброчесності не толеруються. **Відвідання занять** є важливою складовою навчання. Очікується, що всі студенти відвідають усі лекції та лабораторні зайняття курсу. Студенти повинні інформувати викладача про неможливість відвідати заняття. У будь-якому випадку студенти зобов'язані дотримуватися термінів визначених для виконання всіх видів робіт, передбачених курсом. **Література.** Уся література, яку студенти не зможуть знайти самостійно, буде надана викладачем виключно в освітніх цілях без права її передачі третім особам. Студенти заохочуються до використання також й іншої літератури та джерел, яких немає серед рекомендованих. **Політика виставлення балів.** Враховуються бали, набрані при поточному контролі та бали за виконання лабораторних робіт. При цьому обов'язково враховуються присутність на заняттях та активність студента під час практичного заняття; недопустимість пропусків та запізнень на заняття; користування мобільним телефоном, планшетом чи іншими мобільними пристроями під час заняття в цілях не пов'язаних з навчанням; списування та плагіат; несвоєчасне виконання поставленого завдання і т. ін. **Оцінювання лабораторних робіт** (8 лабораторних робіт, максимальна кількість балів: 60) відбувається шляхом оцінки роботи студента під час

	проведення лабораторної роботи в аудиторії та захисту звіту по виконаній лабораторній роботі (0-7.5 балів за одну роботу). Бали оцінювання лабораторних робіт нараховуються за наступним співвідношенням: 7.5 – студент в повному обсязі володіє навчальним матеріалом, має повне розуміння розглянутої теми, надає правильні відповіді на запитання по темі, код програми функціонує відповідно до завдання; 7 - студент в повному обсязі володіє навчальним матеріалом, має повне розуміння розглянутої теми, надає правильні відповіді на запитання по темі, код програми функціонує відповідно до завдання, проте код не є правильно оформлений згідно стандартів; 6 – студент достатньо розуміє розглянутий матеріал та принципи написаного ним коду програми, присутні неточності та незначні помилки у відповідях на запитання по темі, код програми функціонує відповідно до завдання (або з несуттєвими недоліками); 5 - студент достатньо розуміє розглянутий матеріал та принципи написаного ним коду програми, присутні неточності та незначні помилки у відповідях на запитання по темі, код програми функціонує відповідно до завдання, проте з помірними недоліками; 4 - студент не досить добре розуміє розглянутий матеріал та написаний ним код програми, вагається та надає неточні/не конкретні відповіді на запитання по темі, код програми функціонує неточно, або з помірними недоліками; 3 – студент не досить добре розуміє розглянутий матеріал та написаний ним код програми, вагається та надає неточні/не конкретні відповіді на запитання по темі, код програми функціонує неточно, або з вагомими недоліками; 2 – студент погано розуміє розглянутий матеріал та написаний ним код програми, студент в більшості надає помилкові відповіді на питання по темі, код програми функціонує з суттєвими недоліками; 1 - студент погано розуміє розглянутий матеріал та написаний ним код програми, код програми не функціонує належним чином; 0 - студент зовсім не засвоїв розглянутий матеріал, написаний ним код програми не відповідає темі/не функціонує взагалі. **Оцінювання змістових модулів** (2 змістових модулі, 20 балів за кожний) — за результатами написаних студентом звіту та коду програми. Бали оцінювання змістових модулів нараховуються за наступним співвідношенням: 20-16 - розглянута тема відтворюється в повному обсязі, правильно, обґрунтовано, логічно, містить аналіз і систематизацію, аргументовані висновки. Засвідчено глибоке володіння матеріалом. Наведений код повністю робочий та відповідає темі. Можуть бути присутні несуттєві помилки та невідповідності; 16-12 - відтворюється значна частина розглянутої теми. Виявлено знання і розуміння основних положень навчальної дисципліни, проте присутні неточності та/або невідповідності основній темі. Наведений код частково робочий, проте в загальному відповідають темі; 12-8 - відстежується загальне розуміння розглянутої теми. Виявлені множинні неточності та невідповідності, пояснення наведеного коду

	відсутні, код функціонує із значними неточностями (або відсутні приклади запуску коду на виконання взагалі); 8-4 – студент погано розуміє розглянуту тему. Виявлені суттєві неточності та невідповідності. Наведені приклади коду з суттєвими недоліками, або не відповідають темі; 4 – 0 – студент взагалі не розуміє розглянуту тему. Тему не розкрито, кількість викладеного матеріалу не відповідає загальним нормам обраного виду роботи. Наведений код не робочий, або відсутній як такий. **Критерії оцінювання результатів неформальної освіти:** Нарахування балів відбувається за написання студентом тез доповідей на конференціях, наукових статей, участь у діяльності наукових гуртків, участь у наукових семінарах та круглих столах, конкурсах, участь у заходах неформальної освіти за отримання сертифікатів про проходження навчання на різних освітніх платформах (Coursera, Prometheus тощо), курсах на провідних ІТ компаніях за тематикою навчальної дисципліни. Кількість балів визначається відсотком покриття результатів відповідної активності до вимог результатів навчання з навчальної дисципліни.
Питання до заліку чи екзамену.	Особливості розробки великих програмних систем. Архітектура і дизайн. Мета архітектури. Функціональність та структура ПЗ. Роль архітектури та архітектора. UML діаграми та різновиди. Загальні вимоги до технологій створення ПЗ. Правила та особливості розробки ПЗ. Проблеми, які виникають при розробці ПЗ. Моделі життєвого циклу ПЗ. Види та їх класифікація. Аналіз архітектури ПЗ з огляду на область застосування. Монолітна, сервісно-орієнтована, мікросервісна та безсерверна архітектура ПЗ. Модульність та архітектурні характеристики. Вимірювання модульності. Згуртованість, зчеплення, зближення. Компроміси. Виділення архітектурних характеристик. Монолітна архітектура. Проектування з використанням монолітної архітектури. Типи та її різновиди. Компоненти. Шари моноліту. Переваги та недоліки. Конвеєрний архітектурний стиль. Топологія конвеєра. Фільтри. Характеристики стилю та приклади. Переваги і недоліки. Архітектурний стиль мікроядро. Топологія мікроядра. Ядро системи. Компоненти Plug-in. Раєстрація та контракти. Характеристики стилю та приклади. Переваги і недоліки. Сервісно-орієнтований архітектурний стиль. Топологія та варіанти. Гранулярність. Розбиття бази даних Характеристики стилю та приклади. Переваги і недоліки. Архітектурний стиль, орієнтований на події. Топологія. Брокер та система оповіщення. Топологія медіатора. Асинхронні особливості. Обробка помилок та втрата даних. Оркестрована сервіс-орієнтована архітектура. Топологія. Бізнес сервіс, enterprise. Інфраструктурний сервіс. Механізм оркестрування. Потоки повідомлень. Мікросервісна архітектура. Топологія. Суть розподіленості. Гранулярність та ізоляція даних. Контекст та характеристики. Приклади.

	Вибір архітектурного стилю. Архітектурні рішення та анти-патерни. Вимір важливості та компромісів. Оцінка структури. ADR та стандарти.
Опитування	Анкету-оцінку з метою оцінювання якості курсу буде надано по завершенню курсу.

Схема курсу “Архітектура ПЗ (ч.2)” для студентів спеціальності 121 – Інженерія програмного забезпечення

Тиж.	Тема, план, короткі тези	Форма діяльності (заняття)* *лекція, самостійна, дискусія, групова робота)	Література.** * Ресурси в інтернеті	Завдання, год	Термін виконання
1	**Основні відомості про архітектуру ПЗ.** Особливості розробки великих програмних систем. Архітектура і дизайн. Мета архітектури. Функціональність та структура ПЗ. Роль архітектури та архітектора.	Лекція	[1], [2], [3], [8] Сайт курсу	2	кінець поточного тижня
1	Лаб. 1. Збір вимог до ПЗ. Визначення мети розробки, інструментів. Складання технічного завдання.	Лабораторна робота	Сайт курсу	2	протягом двох тижнів
2	**Застосування візуального моделювання в процесі розробки ПЗ.** Цілі та засоби мови UML. Діаграми взаємодії, класів, станів, діяльності, компонентів, розміщення.	Лекція	[1], [2], [3] Сайт курсу	2	кінець поточного тижня
2	**Опрацювання книжки** Fundamentals of Software Architecture: An Engineering Approach (перший розділ)	Самостійна робота	[4]	2	кінець поточного тижня
3	**Технології створення ПЗ.** Загальні вимоги до технологій створення ПЗ. Правила та особливості розробки ПЗ. Проблеми, які виникають при розробці ПЗ.	Лекція	[1], [2], [3], Сайт курсу	2	кінець поточного тижня
3	Лаб. 2. Побудова UML діаграм програмних компонент. Use Case діаграма.	Лабораторна робота	Сайт курсу	2	протягом двох тижнів
4	**Моделі життєвого циклу ПЗ.** Каскадна модель. Макетування. Інкрементна модель. Швидка розробка додатків. Спіральна та компонентно-орієнтована моделі. XP-процес.	Лекція	[1], [2], [3], [6] Сайт курсу	2	кінець поточного тижня
4	Опрацювання теорії щодо моделей життєвого циклу ПЗ, перегляд рекомендованих відео-лекцій.	Самостійна робота	[6], [8]	2	кінець поточного тижня
5	**Класифікація архітектури ПЗ.** Аналіз архітектури ПЗ з огляду на область застосування. Монолітна, сервісно-орієнтована, мікросервісна та безсерверна архітектура ПЗ.	Лекція	[1], [2], [3], [6] Сайт курсу	2	кінець поточного тижня

5	Лаб. 3. Вибір типу архітектури ПЗ. Реалізація схеми та прототипу програмного проекту.	Лабораторна робота	Сайт курсу	2	протягом двох тижнів
6	**Модульність та архітектурні характеристики**. Вимірювання модульності. Згуртованість, зчеплення, зближення. Компроміси. Виділення архітектурних характеристик.	Лекція	[1], [2], [3], [6] Сайт курсу	2	кінець поточного тижня
6	**Опрацьовування книжки** Fundamentals of Software Architecture: An Engineering Approach (розділи 4, 5)	Самостійна робота	[6], [8]	2	кінець поточного тижня
7	**Монолітна архітектура.** Проектування з використанням монолітної архітектури. Типи та її різновиди. Компоненти. Шари моноліту. Переваги та недоліки.	Лекція	[1], [2], [3], [6] Сайт курсу	2	кінець поточного тижня
7	Лаб. 4. Реалізація монолітної архітектури з використанням MVC та REST.	Лабораторна робота	Сайт курсу	2	протягом двох тижнів
8	**Пошаровий архітектурний стиль.** Проектування з використанням монолітної програмних шарів. Топологія. Рівні ізоляції. Переваги та недоліки.	Лекція	[1], [2], [3], [6] Сайт курсу	2	кінець поточного тижня
8	Опрацювання теорії та додаткових джерел по монолітній архітектурі, зокрема по топології та шарах ізоляції. Перегляд рекомендованих відео-лекцій.	Самостійна робота	[1], [2], [3], [6] Сайт курсу	2	кінець поточного тижня
9	**Конвеєрний архітектурний стиль.** Топологія конвеєра. Фільтри. Характеристики стилю та приклади. Переваги і недоліки.	Лекція	[1], [2], [3], [6] Сайт курсу	2	кінець поточного тижня
9	Модульне завдання (UML-діаграми, вимоги до ПЗ та монолітна архітектура).	Модульне завдання	Сайт курсу	2	кінець поточного тижня
10	**Архітектурний стиль мікроядро.** Топологія мікроядра. Ядро системи. Компоненти Plug-in. Раєстрація та контракти. Характеристики стилю та приклади. Переваги і недоліки.	Лекція	[1], [2], [3], [6] Сайт курсу	2	кінець поточного тижня
10	Лаб. 5. Реалізація простої мікроядерної архітектури. Інтеграції компонент.	Лабораторна робота	Сайт курсу	2	протягом двох тижнів
11	**Сервісно-орієнтований архітектурний стиль.** Топологія та варіанти. Гранулярність. Розбиття бази даних Характеристики стилю та приклади. Переваги і недоліки.	Лекція	[1], [2], [3], [6] Сайт курсу	2	кінець поточного тижня
11	Перегляд та засвоєння реалізації сервісно-орієнтованих систем. Ознайомлення з додатковими матеріалами, перегляд онлайн-лекцій.	Самостійна робота	[1], [2], [3], [6] Сайт курсу	2	кінець поточного тижня
12	**Архітектурний стиль, орієнтований на події.** Топологія. Брокер та система оповіщення. Топологія медіатора. Асинхронні особливості. Обробка помилок та втрата даних.	Лекція	[1], [2], [3], [6] Сайт курсу	2	кінець поточного тижня
12	Лаб. 6. Реалізація простого прототипу на базі event-driven архітектури.	Лабораторна робота	Сайт курсу	2	протягом двох тижнів

13	**Оркестрована сервіс-орієнтована архітектура.** Топологія. Бізнес сервіс, enterprise. Інфраструктурний сервіс. Механізм оркестрування. Потоки повідомлень.	Лекція	[1], [2], [3], [6] Сайт курсу	2	кінець поточного тижня
13	Перегляд та засвоєння реалізації оркестрованої архітектури. Ознайомлення з додатковими матеріалами.	Самостійна робота	[1], [2], [3], [6] Сайт курсу	2	кінець поточного тижня
14	**Мікросервісна архітектура.** Топологія. Суть розподіленості. Гранулярність та ізоляція даних. Контекст та характеристики. Приклади.	Лекція	[1], [2], [3], [6] Сайт курсу	2	кінець поточного тижня
14	Лаб. 7. Реалізація мікросервісної архітектури. Розподіленість простого Rest API.	Лабораторна робота	Сайт курсу	2	протягом двох тижнів
15	**Мікросервісна архітектура.** API шар. Операційне перевикористання компонент. Фронт-енд шар. Комунікація та оркестрування. Транзакції.	Лекція	[1], [2], [3], [6] Сайт курсу	2	кінець поточного тижня
15	Модульне завдання (сервіс-орієнтована та мікросервісна архітектури).	Модульне завдання	Сайт курсу	2	кінець поточного тижня
16	**Вибір архітектурного стилю.** Архітектурні рішення та анти-патерни. Вимір важливості та компромісів. Оцінка структури. ADR та стандарти.	Лекція	[1], [2], [3], [8] Сайт курсу	2	кінець поточного тижня
16	Лаб. 8. Реалізація мікросервісної архітектури. Розгортання та тестування.	Лабораторна робота	Сайт курсу	2	кінець поточного тижня