Лекція 2. Архітектура корпоративних інформаційних систем

- Клієнт-серверна та багаторівнева архітектура КІС
- Монолітні, модульні та сервісорієнтовані системи.
- Принципи побудови масштабованих і надійних корпоративних платформ.
- Роль баз даних, прикладного рівня та інтерфейсу користувача. Порівняння on-premise та cloud-архітектур

Клієнт-серверна (Client-Server) та багаторівнева (Multi-tier/N-tier) архітектури є фундаментальними підходами до побудови корпоративних інформаційних систем (КІС), які забезпечують ефективне управління ресурсами, обробку даних та взаємодію між користувачами.

Слайд 1. Клієнт-серверна архітектура (2-рівнева)

Це класична модель, де функціональність розділена між двома сторонами:

- клієнт (Клієнтська частина): користувацький інтерфейс (frontend), який формує запити;
- сервер (Серверна частина): сховище даних та прикладних програм, що обробляє запити та забезпечує зв'язок.

https://kitrum.com/blog/client-server-architecture-advantages-and-disadvantages/

Слайд 2. Основні характеристики 2-рівневої моделі:

- прямий зв'язок: клієнт звертається безпосередньо до бази даних (БД) на сервері.
- бізнес-логіка: може бути розміщена як на клієнті ("товстий клієнт"), так і на сервері.
- переваги: простота, легкість впровадження, висока швидкість обробки невеликих обсягів даних.
- недоліки: обмежена масштабованість, складнощі з безпекою, навантаження на мережу при збільшенні користувачів.

https://dou.ua/forums/topic/44636/

Слайд 3. Багаторівнева (N-рівнева) архітектура КІС

Багаторівнева архітектура (часто 3-рівнева) – це продовження клієнт-серверної моделі, де компоненти програмного забезпечення фізично та логічно розділені на більше ніж два рівні.

Типова 3-рівнева структура:

1. Клієнтський рівень (Presentation Tier): Інтерфейс користувача (наприклад, веб-браузер або мобільний застосунок), який відображає дані.
2. Рівень програми/бізнес-логіки (Application/Logic Tier): Проміжний сервер (middleware), який обробляє дані, виконує бізнес-правила та приймає рішення.
3. Рівень даних (Data/Database Tier): Сервер баз даних, де зберігається інформація. Він ізольований від клієнта рівнем програми.

Слайд 4. Переваги багаторівневої архітектури:

- Масштабованість: Можна легко додавати нові сервери для обробки, не змінюючи клієнтську частину.
- Краща безпека: Клієнт не має прямого доступу до бази даних, що зменшує ризик несанкціонованого доступу.
- Гнучкість та підтримка: Різні рівні можуть розроблятися та оновлюватися незалежно.
- Багаторазове використання: Бізнес-логіка може використовуватися різними типами клієнтів (веб, мобільний).

Слайд 5. Порівняльна характеристика

Характеристика	Клієнт-сервер (2-tier)	Багаторівнева (3-ti
Масштабованість	Низька (обмежена ресурсами БД)	Висока (можна дода логіки)
Безпека	Клієнт має прямий доступ до БД	Клієнт не бачить баз сервер логіки)
Обслуговування	Потребує оновлення ПЗ на кожному ПК	Основні зміни вносят застосунків

Слайд 6. Чому КІС обирають багаторівневу архітектуру?

Корпоративні системи вимагають високої стійкості, безпеки та масштабованості.
Багаторівнева модель забезпечує розділення обов'язків, де кожен компонент - представлення, логіка та дані - ефективно виконує свою функцію. Це дозволяє уникнути "вузьких місць", коли один сервер не справляється з навантаженням.

https://www.habitat3.com.au/single-post/6-benefits-of-3-tier-cloud-architecture-over-a-single-server-architecture

Слайд 7. Яку архітектуру обрати?

Дволанкова підходить для малих офісів, де кількість користувачів не перевищує 10–20 осіб, а бізнес-логіка проста.

Багаторівнева є стандартом для сучасних КІС великих підприємств, оскільки дозволяє працювати тисячам користувачів через інтернет та легко інтегрувати нові функції.

Слайд 8. Монолітні, модульні та сервісорієнтовані КІС.
Ці три типи архітектури визначають ступінь внутрішньої зв'язності та гнучкості КІС.

Монолітна архітектура (Monolithic)

Система будується як єдиний, нерозривний блок коду, де всі функції (бухгалтерія, склад, кадри) об'єднані в одну програму з єдиною базою даних.
Принцип: Всі компоненти тісно пов'язані (tightly coupled). Зміна в одному модулі може вимагати перескладання всієї системи.
Переваги: Висока швидкість внутрішніх операцій, простота розгортання для невеликих компаній.
Недоліки: Важко масштабувати; помилка в одному місці може зупинити всю систему.

Слайд 9. Модульні КІС.

Модульна архітектура (Modular)

Система складається з відносно незалежних блоків (модулів), які мають чітко визначені інтерфейси для взаємодії.

- Принцип: Логічне розділення функцій. Ви можете купити та впровадити лише модуль "Склад", а пізніше додати "Фінанси".
- Переваги: Гнучкість у виборі функціоналу, легше тестування та оновлення окремих частин.
- Недоліки: Потребує складнішої інтеграції між модулями порівняно з монолітом.

https://www.flowsense.solutions/compare/modular-vs-monolithic-erp

Слайд 10. Сервісорієнтована архітектура (SOA)

Найсучасніший підхід, де система – це набір незалежних сервісів, що спілкуються між собою через стандартні мережеві протоколи (наприклад, веб-сервіси).

Принцип: Кожен сервіс (наприклад, "Генерація рахунку") є автономним і може використовуватись іншими системами компанії.
Переваги: Максимальна масштабованість, можливість використовувати різні мови програмування для різних сервісів.
Недоліки: Висока складність управління та залежність від стабільності мережі.

https://medium.com/@lpramithamj/monolithic-vs-soa-vs-microservices-architecture-a-java-perspective-6d3d9fb26ac7

Слайд 11. Порівняльна характеристика

Критерій	Монолітна	Модульна	Сервісоріє
Зв'язність	Дуже висока	Середня	Низька (авт
Складність оновлення	Потребує зупинки всієї системи	Можна оновлювати частинами	Окремі серв оновлюють
Масштабованість	Вертикальна (потужніше залізо)	Горизонтальна (за модулями)	Глобальна (

Слайд 12. Приклади світових лідерів ERP, що використовують ту чи іншу архітектуру

Світові лідери ринку ERP постійно трансформують свої архітектури, щоб відповідати вимогам часу. Цікаво, що більшість сучасних гігантів зараз перебувають у стані "гібридної" трансформації, переходячи від монолітів до сервісорієнтованих структур. Ось приклади лідерів ринку, розподілені за домінуючим типом архітектури:

Слайд 13. Монолітна архітектура (Традиційні On-premise рішення)

Хоча термін "моноліт" часто сприймається як застарілий, він забезпечує неймовірну цілісність даних.

- Microsoft Dynamics GP (Great Plains): Класичний приклад монолітної системи для малого та середнього бізнесу. Всі функції (фінанси, інвентар, продажі) тісно інтегровані в єдине ядро.
- 1C:Enterprise (старіші версії): Хоча вона має модульний вигляд, внутрішня структура бази даних та логіка часто настільки взаємозалежні, що систему важко розділити на незалежні сервіси.

Слайд 14. Модульна архітектура (Гнучкі бізнес-платформи)

Це "золотий стандарт" для великих підприємств, де клієнт обирає потрібні блоки.

- SAP S/4HANA (Cloud/On-premise): Це еталон модульності. Ви можете впровадити модуль FI (Finance) незалежно від MM (Material Management), але вони працюють на єдиній платформі SAP NetWeaver.
- Odoo: Один із найяскравіших представників сучасної модульності. Кожна функція (CRM, eCommerce, HR) — це окремий додаток (module), який можна встановити або видалити, не порушуючи роботу системи.
- Oracle NetSuite: Повністю хмарна ERP, побудована за модульним принципом, де клієнт активує функції за потреби.

Слайд 15. Сервісорієнтована та мікросервісна архітектура (Cloud-Native)

Ці системи побудовані як набір незалежних веб-сервісів.

- Salesforce (Sales/Service Cloud): Хоча це більше CRM, Salesforce є лідером у SOA. Кожна функція доступна через API, що дозволяє легко інтегрувати її з будь-якими іншими системами.
- Workday: Сучасна хмарна система для HR та фінансів. Вона побудована за принципами мікросервісів, що дозволяє оновлювати систему щотижня без зупинки роботи користувачів (zero-downtime updates).
- Infor CloudSuite: Використовує платформу Infor OS, яка працює як сервісна шина (ESB), з'єднуючи різні спеціалізовані сервіси в єдину екосистему.

Слайд 16. Підсумок

Вендор / Продукт	Тип архітектури	Ключова особливість
SAP S/4HANA	**Модульна**	Висока надійність через ін
Odoo	**Модульна**	Найбільша гнучкість (App S бізнесу).
Workday	**SOA / Microservices**	Створена виключно для хм оновлення.

Важливе зауваження: Зараз спостерігається тренд на "Композитну ERP" (Composable ERP). Це коли компанія не купує один моноліт, а збирає свою КІС як конструктор: фінанси від SAP, HR від Workday, а CRM від Salesforce, з’єднуючи їх через хмарні сервіси.

Слайд 17. Композитна ERP (Composable ERP)

— це сучасна стратегія побудови корпоративних систем, яку Gartner визначив як майбутнє корпоративного софту. Це перехід від ідеї «одного великого вендора» до ідеї «найкращої комбінації сервісів».
Якщо класична ERP –це масивний монолітний блок, то композитна ERP – це набір окремих «будівельних блоків» (модулів або мікросервісів), які можна легко з'єднувати, замінювати або масштабувати незалежно один від одного.

https://www.riministreet.com/blog/composable-erp-strategy-benefits/

Слайд 18. Композитна ERP (Composable ERP)

Ключові принципи

Best-of-breed (Найкращий у своєму роді): Ви не зобов’язані використовувати складський модуль від SAP лише тому, що у вас їхня бухгалтерія. Ви можете взяти бухгалтерію SAP, HR від Workday, а логістику від вузькоспеціалізованого стартапу.
API-first: Всі компоненти спілкуються між собою через відкриті інтерфейси (API), що дозволяє «вмикати» нові сервіси як у розетку.
Гнучкість (Agility): Система підлаштовується під бізнес, а не бізнес під жорсткі рамки програми. Потрібно додати модуль ШІ для аналізу продажів? Ви просто інтегруєте його як новий сервіс, не переписуючи всю ERP.

Слайд 19. Порівняння: Традиційна vs Композитна ERP

Характеристика	Традиційна (Монолітна)	Композитна (Со
Структура	Тісно пов'язані модулі (Hard-wired)	Незалежні сервіс
Вендор	Повна залежність від одного постачальника	Свобода вибору постачальників
Оновлення	Глобальне, дороге, ризиковане	Поетапне оновле частин
Швидкість змін	Повільна (місяці/роки)	Швидка (тижні/м

Слайд 20. Переваги для бізнесу

Відсутність Vendor Lock-in: Ви не стаєте «заручником» одного розробника. Якщо модуль логістики застарів — ви замінюєте лише його, а не всю систему.
Масштабованість: Можна розширювати лише ті частини системи, які цього потребують (наприклад, під час сезону розпродажів посилити лише модуль обробки замовлень).
Готовність до інновацій: Легко додавати сучасні технології, такі як IoT (інтернет речей), Blockchain або ШІ, просто підключаючи їх як нові компоненти екосистеми.

Основною проблемою композитної ERP є складність інтеграції.
Компанії потрібна потужна внутрішня IT-експертиза або використання спеціальних платформ (iPaaS, як-от MuleSoft або Boomi), які будуть «склеювати» всі ці незалежні сервіси в єдине ціле.

https://www.riministreet.com/blog/composable-erp-strategy-benefits/

Слайд 21. Принципи побудови масштабованих і надійних корпоративних платформ: Горизонтальне масштабування (Scale-Out)

Побудова сучасних КІС, особливо за моделлю Composable ERP, базується на здатності системи витримувати зростання навантаження (масштабованість) та працювати без збоїв (надійність).
Ось ключові принципи, які використовують архітектори при розробці таких платформ:
1. Горизонтальне масштабування (Scale-Out)
Замість збільшення потужності одного сервера (Vertical Scaling), система проектується так, щоб навантаження розподілялося між десятками або сотнями дрібних вузлів.
Розподіл навантаження (Load Balancing): Використання балансувальників навантаження, які спрямовують запити користувачів на найменш завантажені сервери.
Stateless-архітектура: Сервери застосунків не зберігають дані про стан сесії користувача. Це дозволяє миттєво додавати нові сервери в кластер.

Слайд 22. Принципи побудови масштабованих і надійних корпоративних платформ: принцип ізоляції та надмірності (Redundancy)

2. Принцип ізоляції та надмірності (Redundancy)
Надійність досягається через дублювання компонентів, щоб уникнути «єдиної точки відмови» (Single Point of Failure).

Реплікація баз даних: Дані одночасно записуються на основний (Master) та резервні (Slave) сервери.
Microservices Isolation: Якщо в системі виходить з ладу модуль «Звіти», модулі «Продажі» та «Склад» продовжують працювати.

Слайд 23. Принципи побудови масштабованих і надійних корпоративних платформ: асинхронна взаємодія (Asynchronous Processing)

3. Асинхронна взаємодія (Asynchronous Processing)
Щоб система не «зависала» під час важких операцій (наприклад, генерація річного звіту), використовується черга повідомлень.
Черги повідомлень (Message Queues): Використання інструментів на кшталт RabbitMQ або Apache Kafka. Користувач натискає «Сформувати», запит іде в чергу, а система звільняється для інших завдань, видаючи результат, коли він готовий.

Слайд 24. Принципи побудови масштабованих і надійних корпоративних платформ: робота з даними: Розподілення та Кешування

4. Робота з даними: Розподілення та Кешування
Кешування (Caching): Дані, що часто запитуються (наприклад, прайс-лист), зберігаються в оперативній пам'яті (через Redis або Memcached), що в рази прискорює відгук системи.
Шардинг (Sharding): Розподіл гігантської бази даних на частини за певною ознакою (наприклад, дані клієнтів Європи на одному сервері, Азії — на іншому).

Слайд 25. Принципи побудови масштабованих і надійних корпоративних платформ: принцип Cloud-Native та Контейнеризація

5. Принцип Cloud-Native та Контейнеризація

Docker та Kubernetes: Платформа розбивається на контейнери, якими керує Kubernetes. Він автоматично перезапускає компоненти, що вийшли з ладу, та масштабує їх залежно від трафіку.
Автоматичне відновлення (Self-healing): Система сама виявляє збій і піднімає нову копію сервісу без втручання адміністратора.

Слайд 26. Порівняння підходів

Порівняння підходів

Принцип	Для чого потрібен	Технологічний при
Масштабованість	Обробка 1000+ замовлень на секунду	Kubernetes, Horizor Autoscaling
Надійність	Робота 24/7 без зупинок	Multi-AZ (доступніс зонах)

Порада: Для перевірки надійності великих систем часто використовують "Chaos Engineering" (метод, популяризований Netflix), коли в системі навмисно відключають сервери в робочий час, щоб перевірити, чи зможе вона автоматично відновитися.

Слайд 27. Роль баз даних, прикладного рівня та інтерфейсу користувача КІС

У багаторівневій архітектурі КІС кожен із цих елементів виконує свою критичну роль, забезпечуючи життєвий цикл інформації: від її введення до безпечного зберігання.

Слайд 28. Інтерфейс користувача (Рівень представлення / Presentation Tier)

Це «обличчя» системи, з яким взаємодіє працівник підприємства.
Роль: Візуалізація даних, збір вхідної інформації та передача запитів на прикладний рівень.
Типи:
«Товстий клієнт»: Повноцінна програма, встановлена на ПК (наприклад, десктопний додаток SAP GUI). Виконує частину обчислень самостійно.
«Тонкий клієнт»: Веб-браузер або мобільний додаток. Вся логіка — на сервері, інтерфейс лише відображає результат.
Тренди: Розвиток UX/UI дизайну для мінімізації помилок персоналу (наприклад, адаптивні інтерфейси в Oracle Modern Best Practice).

Слайд 29. Прикладний рівень (Рівень логіки / Application Tier)

Це «мозок» КІС, де зосереджена вся бізнес-логіка компанії.
Роль: Обробка запитів від інтерфейсу, виконання розрахунків (нарахування зарплати, списання залишків, калькуляція собівартості) та контроль прав доступу.
Функції:

- Перевірка даних на відповідність правилам (наприклад, чи не перевищено кредитний ліміт клієнта).
- Координація транзакцій між різними модулями.
- Взаємодія з зовнішніми сервісами через API.

Приклад: Сервер застосунків SAP NetWeaver або сервіси Microsoft Dynamics 365.

Слайд 30. **Рівень баз даних (Рівень даних / Data Tier)**

Це «пам'ять» системи, де інформація зберігається в структурованому вигляді.
Роль: Надійне зберігання, швидка вибірка, забезпечення цілісності (Integrity) та безпеки даних.
Складові:

- СУБД (Система управління базами даних): Microsoft SQL Server, Oracle Database, PostgreSQL.
- In-memory DB: Технології типу SAP HANA, що тримають дані в оперативній пам'яті для миттєвої аналітики.

Ключові задачі: Резервне копіювання, обробка тисяч одночасних запитів та підтримка ACID-транзакцій (гарантія, що фінансова операція або виконана повністю, або не виконана зовсім).

Слайд 31. Взаємодія рівнів (Приклад: Оформлення замовлення)

Інтерфейс: Менеджер вводить назву товару та кількість.

Клієнтське ПЗ відправляє JSON-запит на сервер.

Прикладний рівень: Програма перевіряє, чи є товар на складі, розраховує персональну знижку клієнта та формує фінансову проводку.

База даних: Сервер БД отримує команду «Update», змінює кількість товару на складі та записує нове замовлення в таблицю.

Слайд 32.

Розподіл ролей у цифрах

Компонент	Частка коду	Основний ресурс	Критичний п
Інтерфейс	20-30%	Відеокарта / RAM клієнта	Швидкість від
Прикладний рівень	50-60%	Процесор (CPU) сервера	Пропускна зд (Throughput)

Слайд 33. Порівняння on-premise та cloud-архітектур

Вибір між on-premise (локальною) та cloud (хмарною) архітектурою — це стратегічне рішення, яке визначає витрати, рівень контролю та швидкість розвитку компанії на роки вперед.

Слайд 34. Основні відмінності

Характеристика	Локальна (On-premise)	Хмарна (Cloud/Saa
Розгортання	На власних серверах компанії	На серверах провай AWS)
Модель витрат	**CapEx:** Великі початкові інвестиції в залізо та ліцензії	**OpEx:** Щомісячна пі поріг входу
Оновлення	Ручне, складне, часто потребує зупинки системи	Автоматичне, безпер виконується вендор
Масштабованість	Повільна (потрібна закупівля та монтаж заліза)	Майже миттєва (дод або потужності)

Слайд 35. Детальний розгляд ключових аспектів

Безпека та комплаєнс:

- On-premise обирають банки та стратегічні підприємства, де закон вимагає фізичного знаходження даних всередині країни.
- Cloud пропонує вищий рівень захисту від кібератак (DDoS, шифрування), оскільки гіганти як Microsoft чи Oracle інвестують мільярди у безпеку, що недоступно окремій компанії.

Сумарна вартість володіння (TCO):

- Протягом перших 12–18 місяців хмара значно дешевша.
- На горизонті 5–7 років сукупні витрати на підписку можуть перевищити вартість локальних ліцензій, проте хмара все одно часто виграє за рахунок відсутності витрат на IT-штат, електроенергію та оновлення заліза.

Слайд 36. Гібридна архітектура (Hybrid):

Компромісний варіант, де критичні дані зберігаються локально, а аналітика, CRM або поштові сервіси виносяться в хмару. Це дозволяє зберегти контроль над таємницею, використовуючи гнучкість хмарних обчислень.

Слайд 37. Порівняння архітектурних підходів для КІС, структуроване за ключовими критеріями для прийняття рішення:

1. Економічна модель (CapEx vs OpEx)
   - On-premise: Потребує значних капітальних інвестицій (CapEx) на старті: купівля серверів, мережевого обладнання, джерел безперебійного живлення та безстрокових ліцензій на ПЗ.
   - Cloud: Базується на операційних витратах (OpEx). Ви сплачуєте лише за використані ресурси або кількість користувачів (модель SaaS). Це дозволяє почати впровадження з мінімальним бюджетом.
2. Масштабованість та гнучкість
   - On-premise: Масштабування обмежене фізичним залізом. Якщо кількість користувачів подвоїться, доведеться чекати тижні на поставку та налаштування нових серверів.

- Cloud: Має майже безмежну горизонтальну масштабованість. Нові потужності або модулі активуються за кілька кліків.

3. Безпека та контроль

- On-premise: Ви маєте повний фізичний контроль. Дані не покидають контур компанії, що критично для оборонних чи державних структур. Проте ви самі відповідаєте за захист від хакерів.
- Cloud: Провайдери (AWS, Azure, Google Cloud) забезпечують рівень захисту, який важко реалізувати локально. Однак виникає питання юридичного доступу до даних та їх фізичного розташування (наприклад, згідно з GDPR).

4. Оновлення та обслуговування

- On-premise: Кожне оновлення КІС — це окремий складний проект, який може тривати місяцями через ризик конфліктів із кастомними доробками.

- Cloud: Оновлення відбуваються автоматично на стороні вендора. Компанія завжди працює на найактуальнішій версії без участі власних системних адміністраторів.

Слайд 38.

Порада: Для великих компаній часто оптимальним є Гібридний підхід: основне ядро КІС (фінанси, виробництво) — локально, а мобільні сервіси, аналітика та CRM — у хмарі.

Слайд 39. Порівняльна таблиця

Критерій	On-premise (Локальна)	Cloud (Хмарна)
Швидкість розгортання	Місяці	Дні/Тижні
Відповідальність за Backup	IT-відділ компанії	Провайдер (автома
Доступність (Uptime)	Залежить від локального живлення	Гарантується SLA (99.9%+)

https://www.ecisolutions.com/blog/cloud-erp-vs-on-premise-solutions-making-the-right-choice-for-your-business/

https://go-erp.eu/erp-system-features-comparison-cloud-vs-on-premise-erp/