

**БАКАЛАВРСЬКА РОБОТА**

**БР. ІІ - 78.00.00.000 ІІЗ**

**Група ІІ-22-3**

**Надиберний Андрій**

**2026**

**Івано-Франківський національний технічний університет нафти і газу**

**Факультет інформаційних технологій**

**Кафедра інженерії програмного забезпечення**

**Надиберний Андрій Олегович**

(прізвище, ім'я, по батькові)

УДК 004.4  
(індекс)

## **БАКАЛАВРСЬКА РОБОТА**

**Веб-базована ієрархічна система розподіленої взаємодії користувачів**

(назва роботи)

**Інженерія програмного забезпечення**

(назва освітньої програми)

**121 - Інженерія програмного забезпечення**

(шифр і назва спеціальності)

**Робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело**

Здобувач освітнього рівня Надиберний А.О.  
(підпис, ініціали та прізвище здобувача)

Науковий керівник Ваврик Тетяна Олександрівна, асистент  
(підпис, прізвище, ім'я, по батькові, науковий ступінь, вчене звання керівника)

**Допущено до захисту**  
**Завідувач кафедри**

доц. Бандура В.В.  
(посада) (підпис) (дата) (ініціали та прізвище)

**Івано-Франківськ – 2026**

**Івано-Франківський національний технічний університет нафти і газу**

Інститут, факультет інформаційних технологій

Кафедра інженерії програмного забезпечення

Ступінь вищої освіти бакалавр

Спеціальність 121 – Інженерія програмного забезпечення

**ЗАТВЕРДЖУЮ:**

Зав. кафедрою ІІЗ

доц.

В.В. Бандура

“        ”                                  2026 р.

## **ЗАВДАННЯ**

### **НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТОВІ**

**Надиберному Андрію Олеговичу**

(прізвище, ім'я, по-батькові)

**1. Тема проекту (роботи) “ Веб-базована ієрархічна система розподіленої взаємодії користувачів”**

керівник проекту (роботи) Ваврик Тетяна Олександрівна, асистент

затверджені наказом закладу вищої освіти від “ 21 ” квітня 2026р. № 179/7

**2. Строк подання студентом проекту (роботи) 09 червня 2026 р.**

**3. Вихідні дані до проекту (роботи) Результати і матеріали отримані під час проходження переддипломної практики**

**4. Зміст розрахунково - пояснювальної записки (перелік питань, які потрібно розробити)**

1. Аналіз проблематики розробки ієрархічної системи розподіленої взаємодії користувачів

2. Архітектура та принципи побудови ієрархічної системи онлайн-конференцій

3. Проектування структури ієрархічної системи онлайн взаємодії користувачів

4. Моделювання та програмна реалізація ієрархічної системи взаємодії користувачів

5. Реалізація основних модулів ієрархічної системи

**5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)**

1. Класифікація систем онлайн-конференцій за типом медіаконтенту (рис. 1.1, ст.15)

2. Класифікація онлайн-конференцій за механізмами управління (рис. 1.2, ст.16 )

3. Класифікація систем онлайн-конференцій за архітектурними підходами (рис. 1.3, ст.17)

4. Діаграма активності процесу проведення онлайн-конференції (рис. 1.4, ст.20)

5. Чотирирівнева архітектура системи онлайн-конференцій (рис. 2.1, ст.24 )

## 6. Консультанти розділів проекту (роботи)

| Розділ | Консультант | Підпис, дата   |                  |
|--------|-------------|----------------|------------------|
|        |             | Завдання видав | Завдання прийняв |
|        |             |                |                  |
|        |             |                |                  |
|        |             |                |                  |

7. Дата видачі завдання 21 квітня 2026 р.

Керівник

\_\_\_\_\_ (підпис)

Завдання прийняв до виконання

\_\_\_\_\_ (підпис)

## КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів дипломного проекту (роботи)                                             | Строк виконання етапів проекту | Примітка |
|-------|--------------------------------------------------------------------------------------|--------------------------------|----------|
| 1     | Аналіз проблематики розробки ієрархічної системи розподіленої взаємодії користувачів | 04.05.2026                     | виконано |
| 2     | Архітектура та принципи побудови ієрархічної системи онлайн-конференцій              | 15.05.2026                     | виконано |
| 3     | Проектування структури ієрархічної системи онлайн взаємодії користувачів             | 21.05.2026                     | виконано |
| 4     | Моделювання та програмна реалізація ієрархічної системи взаємодії користувачів       | 28.05.2026                     | виконано |
| 5     | Реалізація основних модулів ієрархічної системи                                      | 03.06.2026                     | виконано |
| 6     | Оформлення пояснювальної записки бакалаврської роботи завідувачем кафедри            | 09.06.2026                     | виконано |

Студент – дипломник

Надиберний А.О.

(підпис)

Керівник роботи

Ваврик Т.О.

(підпис)

## АНОТАЦІЯ

Бакалаврська робота містить 79 сторінок, 29 рисунків, список використаних джерел із 34 найменуваннями, 1 додаток.

**Мета роботи:** розробка та дослідження веб-базованої ієрархічної системи розподіленої взаємодії користувачів для організації та управління онлайн-конференціями.

**Об'єкт дослідження** - процеси розподіленої взаємодії користувачів у веб-орієнтованих інформаційних системах.

**Предмет дослідження** - методи, моделі та програмні засоби побудови ієрархічних систем онлайн-взаємодії користувачів.

**В першому розділі** проведено аналіз теоретичних засад та існуючих підходів до побудови систем онлайн-конференцій і визначено доцільність використання ієрархічної моделі.

**В другому розділі** розроблено архітектуру ієрархічної системи та визначено принципи її функціонування

**В третьому розділі** виконано проєктування структури системи та змодельовано основні сценарії взаємодії користувачів

**В четвертому розділі** реалізовано програмну систему та підтверджено її працездатність на основі змодельованих процесів.

**Висновок:** розроблено та реалізовано ефективну веб-базовану ієрархічну систему розподіленої взаємодії користувачів, що забезпечує гнучке управління онлайн-комунікаціями.

**КЛЮЧОВІ СЛОВА:** ВЕБ-БАЗОВАНА СИСТЕМА, ОНЛАЙН-КОНФЕРЕНЦІЇ, ІЄРАРХІЧНА СТРУКТУРА, РОЗПОДІЛЕНА ВЗАЄМОДІЯ, КОРИСТУВАЧІ, АРХІТЕКТУРА СИСТЕМИ, MVC

## ANNOTATION

The bachelor's thesis contains 79 pages, 29 figures, a list of used sources with 34 names, 1 appendix.

**Purpose of the work:** development and research of a web-based hierarchical system of distributed user interaction for organizing and managing online conferences.

**The object of the study** is the processes of distributed user interaction in web-oriented information systems.

**The subject of the study** is methods, models and software tools for building hierarchical systems of online user interaction.

**The first section** analyzes the theoretical foundations and existing approaches to building online conference systems and determines the feasibility of using a hierarchical model.

**The second section** develops the architecture of the hierarchical system and determines the principles of its functioning.

**The third section** designs the system structure and models the main user interaction scenarios.

**The fourth section** implements the software system and confirms its operability based on the simulated processes.

**Conclusion:** an effective web-based hierarchical system of distributed user interaction has been developed and implemented, which provides flexible management of online communications.

**KEYWORDS:** WEB-BASED SYSTEM, ONLINE CONFERENCES, HIERARCHICAL STRUCTURE, DISTRIBUTED INTERACTION, USERS, SYSTEM ARCHITECTURE, MVC

## ЗМІСТ

|                                        |          |
|----------------------------------------|----------|
| <b>ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ .....</b> | <b>8</b> |
| <b>ВСТУП.....</b>                      | <b>9</b> |

|                                                                                                            |           |
|------------------------------------------------------------------------------------------------------------|-----------|
| <b>РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМАТИКИ РОЗРОБКИ ІЄРАРХІЧНОЇ СИСТЕМИ РОЗПОДІЛЕНОЇ ВЗАЄМОДІЇ КОРИСТУВАЧІВ.....</b> | <b>12</b> |
|------------------------------------------------------------------------------------------------------------|-----------|

|                                                                                                             |    |
|-------------------------------------------------------------------------------------------------------------|----|
| 1.1. Архітектурні принципи та сфери застосування веб-орієнтованих систем онлайн-конференцій .....           | 12 |
| 1.2. Класифікація, функціональні моделі та архітектурні підходи до побудови систем онлайн-конференцій ..... | 14 |
| 1.3. Функціональна архітектура та механізми управління в ієрархічних системах онлайн-конференцій.....       | 18 |
| 1.4 Висновки по розділу .....                                                                               | 22 |

|                                                                                                |           |
|------------------------------------------------------------------------------------------------|-----------|
| <b>РОЗДІЛ 2. АРХІТЕКТУРА ТА ПРИНЦИПИ ПОБУДОВИ ІЄРАРХІЧНОЇ СИСТЕМИ ОНЛАЙН-КОНФЕРЕНЦІЙ .....</b> | <b>24</b> |
|------------------------------------------------------------------------------------------------|-----------|

|                                                                                                                        |    |
|------------------------------------------------------------------------------------------------------------------------|----|
| 2.1. Архітектурна модель та програмний базис ієрархічної системи онлайн-конференцій.....                               | 24 |
| 2.2. Опис робочих процесів та функціональних сценаріїв управління сесіями онлайн-конференцій .....                     | 27 |
| 2.3. Реалізація архітектурного шаблону Model-View-Controller у середовищі ієрархічної системи онлайн-конференцій ..... | 29 |
| 2.4 Висновки по розділу .....                                                                                          | 37 |

|                                                                                                 |           |
|-------------------------------------------------------------------------------------------------|-----------|
| <b>РОЗДІЛ 3. ПРОЕКТУВАННЯ СТРУКТУРИ ІЄРАРХІЧНОЇ СИСТЕМИ ОНЛАЙН ВЗАЄМОДІЇ КОРИСТУВАЧІВ .....</b> | <b>38</b> |
|-------------------------------------------------------------------------------------------------|-----------|

|                                                                      |    |
|----------------------------------------------------------------------|----|
| 3.1. Методологія проектування та розробки системи онлайн-конференцій | 38 |
|----------------------------------------------------------------------|----|

|           |      |               |        |      |                                                                                                       |  |  |                 |      |         |   |  |
|-----------|------|---------------|--------|------|-------------------------------------------------------------------------------------------------------|--|--|-----------------|------|---------|---|--|
|           |      |               |        |      | БР.ІП – 78.00.00.000 ПЗ                                                                               |  |  |                 |      |         |   |  |
|           |      |               |        |      |                                                                                                       |  |  |                 |      |         |   |  |
| Змн.      | Арк. | № докум.      | Підпис | Дата | Веб-базована ієрархічна система<br>розподіленої взаємодії<br>користувачів<br><br>Пояснювальна записка |  |  | Літ.            | Арк. | Акрушів |   |  |
| Розроб.   |      | Надиберний А. |        |      |                                                                                                       |  |  |                 |      |         | 6 |  |
| Перевір.  |      | Ваврик Т.О.   |        |      |                                                                                                       |  |  |                 |      |         |   |  |
| Реценз.   |      | Процюк Г.Я.   |        |      |                                                                                                       |  |  |                 |      |         |   |  |
| Н. Контр. |      | Піх М.М.      |        |      |                                                                                                       |  |  |                 |      |         |   |  |
| Затверд.  |      | Бандура В.В.  |        |      |                                                                                                       |  |  |                 |      |         |   |  |
|           |      |               |        |      |                                                                                                       |  |  | ІФНТУНГ ІП-22-3 |      |         |   |  |

|                                                                                                                             |    |
|-----------------------------------------------------------------------------------------------------------------------------|----|
| 3.2. Функціональне моделювання сценаріїв взаємодії та аналіз прецедентів системи.....                                       | 41 |
| 3.2.1. Роль адміністратора системи .....                                                                                    | 43 |
| 3.2.2. Сценарії взаємодії в межах закритих конференцій .....                                                                | 44 |
| 3.2.3. Роль учасника у відкритих конференціях.....                                                                          | 46 |
| 3.3. Об'єктно-орієнтована декомпозиція та архітектурна організація компонентів ієрархічної системи онлайн-конференцій ..... | 48 |
| 3.3.1. Рівень керування .....                                                                                               | 49 |
| 3.3.2. Рівень даних та бізнес-логіки.....                                                                                   | 49 |
| 3.3.3. Рівень представлення.....                                                                                            | 50 |
| 3.4 Висновки по розділу                                                                                                     |    |

## **РОЗДІЛ 4. МОДЕЛЮВАННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ ІЄРАРХІЧНОЇ СИСТЕМИ ВЗАЄМОДІЇ КОРИСТУВАЧІВ ..... 54**

|                                                                                                                |    |
|----------------------------------------------------------------------------------------------------------------|----|
| 4.1. Динамічне моделювання процесів взаємодії системи онлайн-конференцій на основі діаграм послідовності ..... | 54 |
| 4.1.1. Сценарії доступу учасників.....                                                                         | 54 |
| 4.1.2. Сценарії адміністратора.....                                                                            | 58 |
| 4.2. Алгоритмізація робочих процесів на основі діаграм активності .....                                        | 64 |
| 4.3. Реалізація модуля адміністрування ієрархічної системи онлайн-конференцій.....                             | 69 |
| 4.4. Інтерфейс модуля управління сесіями в межах ієрархічної системи....                                       | 71 |
| 4.5 Висновки по розділу .....                                                                                  | 74 |

## **ВИСНОВКИ ..... 75**

## **ПЕРЕЛІК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ ..... 77**

## **ДОДАТКИ**

## **БІБЛОГРАФІЧНА ДОВІДКА**

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                         | 7    |

## ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

**CCR** – Closed Conference Room

**GUI** – Graphical User Interface

**KPI** – Key Performance Indicators

**MVC** – Model-View-Controller

**OCR** – Open Conference Room

**RBAC** – Role-Based Access Control

**REST** – Representational State Transfer

**RMI** – Remote Method Invocation

**SBME** – Simulation-Based Medical Education

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
|      |      |          |        |      |                         | 8    |
| Змн. | Арк. | № докум. | Підпис | Дата |                         |      |

## ВСТУП

Сучасний етап розвитку інформаційних технологій характеризується стрімким поширенням веб-орієнтованих систем, які забезпечують ефективну взаємодію користувачів у розподіленому середовищі. Особливого значення набувають системи онлайн-конференцій, що дозволяють організовувати комунікацію, спільну роботу та обмін інформацією незалежно від географічного розташування учасників. Зростання вимог до масштабованості, надійності, безпеки та гнучкості таких систем обумовлює необхідність розробки нових підходів до їх проєктування.

Одним із перспективних напрямів є використання ієрархічних моделей організації взаємодії, які дозволяють структурувати користувачів, розмежовувати права доступу та оптимізувати процеси управління. Веб-базовані ієрархічні системи розподіленої взаємодії забезпечують ефективну координацію великої кількості учасників, що є особливо актуальним для освітніх, наукових і корпоративних середовищ.

У межах даної роботи досліджуються теоретичні засади побудови таких систем, розробляється архітектурна модель, здійснюється проєктування структури та реалізація програмного забезпечення для підтримки онлайн-взаємодії користувачів. Основна увага приділяється інтеграції сучасних архітектурних підходів, методів моделювання та програмної реалізації з метою створення ефективної, масштабованої та зручної у використанні системи.

### Актуальність роботи

Актуальність теми дослідження обумовлена зростаючою потребою у засобах організації дистанційної взаємодії користувачів, що особливо посилилася в умовах глобальної цифровізації суспільства. Системи онлайн-конференцій стали невід'ємною складовою освітнього процесу, бізнес-

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
|      |      |          |        |      |                         | 9    |
| Змн. | Арк. | № докум. | Підпис | Дата |                         |      |

комунікацій та наукової діяльності, однак існуючі рішення не завжди забезпечують достатній рівень гнучкості, масштабованості та керованості.

Особливої уваги потребує проблема організації багаторівневої взаємодії користувачів із різними ролями та правами доступу. Більшість сучасних систем орієнтовані на плоску модель взаємодії, що обмежує можливості ефективного управління великими групами користувачів. Впровадження ієрархічних структур дозволяє вирішити зазначені проблеми шляхом чіткого розподілу функцій, делегування повноважень та оптимізації інформаційних потоків.

Таким чином, розробка веб-базованої ієрархічної системи розподіленої взаємодії користувачів є актуальним науково-практичним завданням, спрямованим на підвищення ефективності організації онлайн-комунікації та управління користувацькими процесами.

**Метою роботи** є розробка та дослідження веб-базованої ієрархічної системи розподіленої взаємодії користувачів для організації та управління онлайн-конференціями.

**Об'єкт дослідження** - процеси розподіленої взаємодії користувачів у веб-орієнтованих інформаційних системах.

**Предмет дослідження** - методи, моделі та програмні засоби побудови ієрархічних систем онлайн-взаємодії користувачів.

**Завдання дослідження:**

- проаналізувати існуючі підходи до побудови систем онлайн-конференцій;
- дослідити архітектурні принципи та моделі веб-орієнтованих систем;
- розробити архітектуру ієрархічної системи взаємодії користувачів;
- сформулювати функціональні сценарії та моделі взаємодії;
- здійснити об'єктно-орієнтоване проєктування системи;
- реалізувати програмні модулі системи онлайн-конференцій;
- оцінити ефективність запропонованого рішення.

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
|      |      |          |        |      |                         | 10   |
| Змн. | Арк. | № докум. | Підпис | Дата |                         |      |

## **Методи дослідження**

У роботі використано методи системного аналізу для дослідження предметної області, методи об'єктно-орієнтованого проєктування для побудови структури системи, а також методи моделювання із застосуванням UML-діаграм для формалізації процесів взаємодії. Для реалізації програмного забезпечення використано сучасні підходи веб-розробки та архітектурні шаблони, зокрема MVC.

## **Наукова новизна**

Наукова новизна отриманих результатів полягає у розробці моделі веб-базованої ієрархічної системи розподіленої взаємодії користувачів, яка забезпечує ефективне управління ролями, доступом і сесіями в умовах багаторівневої організації користувачів, а також у вдосконаленні підходів до структуризації онлайн-комунікаційних процесів.

## **Практичне застосування**

Розроблена система може бути використана в освітніх платформах, корпоративних інформаційних системах та наукових середовищах для організації онлайн-конференцій, дистанційного навчання та колективної роботи користувачів. Практична цінність полягає у підвищенні ефективності управління взаємодією користувачів та можливості адаптації системи до різних умов експлуатації.

Бакалаврська робота містить 79 сторінок, 29 рисунків, 4 розділи, перелік використаних джерел налічує 34 найменування, 1 додаток.

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
|      |      |          |        |      |                         | 11   |
| Змн. | Арк. | № докум. | Підпис | Дата |                         |      |

# РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМАТИКИ РОЗРОБКИ ІЄРАРХІЧНОЇ СИСТЕМИ РОЗПОДІЛЕНОЇ ВЗАЄМОДІЇ КОРИСТУВАЧІВ

## 1.1. Архітектурні принципи та сфери застосування веб-орієнтованих систем онлайн-конференцій

Стрімкий розвиток глобальної мережі Інтернет та високопродуктивних обчислювальних технологій зумовив інтенсивне впровадження систем розподіленого співробітництва в режимі реального часу. Серед таких рішень особливе місце посідають системи онлайн-конференцій, які забезпечують синхронну багатокористувацьку комунікацію у веб-середовищі, об'єднуючи географічно розподілених користувачів.

Відповідно до досліджень [1], [3-5], онлайн-конференцію слід розглядати як формалізовану, конфіденційну, інтерактивну та заздалегідь сплановану електронну комунікаційну сесію. Ключовими атрибутами таких систем є:

- Структурованість: наявність чіткого порядку денного, регламенту та авторизованого кола учасників.
- Конфіденційність та безпека: захищена передача даних (зокрема, із застосуванням наскрізного шифрування), що гарантує доставку інформації виключно цільовим реципієнтам.
- Інтерактивність: підтримка двостороннього або багатостороннього зв'язку в реальному часі, що імітує ефект «живої» присутності.
- Електронна природа: використання комп'ютерних мереж, серверного обладнання та термінальних пристроїв як базового середовища обміну даними.
- Плановість: цілеспрямована організація події, що концептуально відрізняє онлайн-конференцію від спонтанних комунікаційних актів (наприклад, звичайних телефонних дзвінків).

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
|      |      |          |        |      |                         | 12   |
| Змн. | Арк. | № докум. | Підпис | Дата |                         |      |

Сучасна веб-орієнтована система онлайн-конференцій [2, 6] використовує Інтернет як базову комунікаційну інфраструктуру. У такій архітектурі веб-браузер виконує роль універсального клієнтського інтерфейсу, а спеціалізовані сервери (або розподілені хмарні кластери) забезпечують перехоплення запитів, маршрутизацію медіапотоків та синхронізацію даних між учасниками.

Застосування веб-орієнтованого підходу має низку суттєвих переваг. По-перше, такі системи відзначаються високим рівнем масштабованості, оскільки обчислювальне навантаження та потоки даних можуть динамічно розподілятися між багатьма серверними вузлами. По-друге, використання стандартизованих мережевих протоколів (таких як HTTP/HTTPS, TCP/IP), а також сучасних технологій передачі потокового мультимедіа в реальному часі (зокрема WebRTC та WebSocket), забезпечує кросплатформну сумісність. Це мінімізує вимоги до апаратного забезпечення клієнта та усуває необхідність інсталяції спеціалізованого програмного забезпечення — стандартний веб-браузер стає повноцінним і єдиним необхідним інструментом для участі в конференції.

Спектр практичного застосування систем онлайн-конференцій [1, 3-6] постійно розширюється і виходить далеко за межі традиційних корпоративних нарад. У сучасному інформаційному суспільстві ці технології є критично важливим інструментом для:

- Корпоративного сектору: проведення засідань рад директорів, антикризового управління, фінансового та бюджетного планування, маркетингових презентацій, а також узгодження контрактів.
- Інженерії та розробки: організації спільних сесій архітектурного проектування, огляду технічної документації та обговорення етапів розробки нових продуктів.
- Освітньої галузі: формування базису для систем дистанційного навчання (e-learning), проведення віртуальних лекцій, семінарів, академічних

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                         | 13   |

дискусій та забезпечення регулярних консультацій (віртуальні робочі години).

- Медицини (телемедицини): проведення віддалених консилиумів, надання експертних медичних консультацій та обміну діагностичними даними між клініками.

Таким чином, інтеграція веб-систем онлайн-конференцій у сучасні бізнес-процеси, освітні та наукові практики дозволяє суттєво оптимізувати часові й логістичні витрати, руйнуючи географічні бар'єри та підвищуючи загальну ефективність управління і співробітництва.

## **1.2. Класифікація, функціональні моделі та архітектурні підходи до побудови систем онлайн-конференцій**

Сучасні системи онлайн-конференцій [1, 4, 5] є складними розподіленими програмно-апаратними комплексами, які можна систематизувати за низкою базових критеріїв. Для глибокого аналізу предметної області доцільно виокремити три ключові категорії класифікації: за типом підтримуваного медіаконтенту, за механізмами управління сесією та за базовими архітектурними парадигмами.

### **1. Класифікація за типом медіаконтенту**

Залежно від протоколів передачі даних та підтримуваних мультимедійних форматів, системи онлайн-конференцій поділяються на такі види:

- відеоконференції - мультимодальні системи, що забезпечують синхронний обмін відеопотоками, аудіоданими, текстовою інформацією та графікою між віддаленими вузлами через веб-середовище. Базові принципи таких систем були закладені в класичних програмних рішеннях (наприклад, IBM Lotus SameTime, Google Meet та zoom і сьогодні еволюціонували у високопродуктивні хмарні платформи.

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                         | 14   |

- аудіо конференції - системи, оптимізовані для передачі виключно голосового трафіку (VoIP) у режимі реального часу з мінімальними затримками.

- текстові конференції - асинхронні або синхронні комунікаційні середовища, що використовують текстові повідомлення як основний носій інформації.



Рисунок 1.1 – Класифікація систем онлайн-конференцій за типом медіаконтенту

## 2. Класифікація за механізмами управління

Системи онлайн-конференцій суттєво різняться за набором адміністративних функцій, які можна згрупувати у такі підсистеми:

- Управління безпекою [1, 4-6]: відповідає за авторизацію та обмеження доступу до віртуального середовища й інформаційних потоків. Аутентифікація може здійснюватися через парольні політики або криптографічні механізми (наприклад, інфраструктуру відкритих ключів — PKI). За рівнем доступу онлайн-конференції поділяються на: відкриті (публічний доступ без обмежень), закриті (додавання виключно адміністратором), обмежені (доступ за специфічними критеріями) та

захищені/приховані (метадані сесії недоступні для неавторизованих користувачів).

- Конфігурація та політики доступу: забезпечує реалізацію рольової моделі доступу (RBAC), управління привілеями та облік ресурсів. До базових ролей належать: гість (read-only права), учасник (права на читання та базовий запис/виступ), доповідач (розширені права на презентацію контенту) та модератор (повний адміністративний контроль).

- Управління доступом до спільного простору [10]: механізми координації доступу до поділюваних ресурсів (право голосу, трансляція відео, керування віртуальною дошкою). Математично це виражається як надання активних прав  $k$  учасникам із загальної множини  $n$  (де  $0 < k \leq n$ ) у певний дискретний момент часу. Моделі управління варіюються від вільного доступу до суворого маркерного доступу із запитом, підтвердженнями та можливістю примусового переривання модератором.

- Мета-управління (meta-management): відповідає за сигналізацію, service discovery (виявлення доступних сесій) та маршрутизацію запрошень між потенційними учасниками.



Рисунок 1.2 - Класифікація онлайн-конференцій за механізмами управління

### 3. Класифікація за архітектурними підходами

Фундаментальною характеристикою системи онлайн-конференцій є її топологія.



Рисунок 1.3 – Класифікація систем онлайн-конференцій за архітектурними підходами

Виділяють два базові підходи та їхні гібридні варіації:

- Централізована архітектура - базується на топології «зірка», де єдиний сервер-мультиплексор обробляє медіапотоки та керує станом сесії. Перевагою є простота реалізації та глобальна консистентність даних (весь стан зберігається в одному вузлі). Головний недолік — наявність єдиної точки відмови та вразливість до мережеских перевантажень при зростанні кількості клієнтів.

- Децентралізована архітектура - функції управління та реплікації даних розподіляються між множиною серверних вузлів. Цей підхід забезпечує високу відмовостійкість і балансування навантаження, проте вимагає імплементації складних алгоритмів консенсусу та синхронізації станів між серверами.

- Гібридні стратегії - спрямовані на поєднання простоти управління з високою масштабованістю. До цієї категорії належать каскадні ієрархічні

топології (де сервери утворюють деревоподібну структуру), а також кільцеві архітектури (де кілька серверів об'єднані в логічне кільце і підпорядковані єдиному керуючому вузлу). Сучасним розвитком цього підходу є мікросервісна архітектура з використанням контейнеризації для динамічного розгортання релейних серверів залежно від навантаження.

### **1.3. Функціональна архітектура та механізми управління в ієрархічних системах онлайн-конференцій**

У межах даної роботи розглядається архітектура веб-орієнтованої системи онлайн-конференцій, побудована на принципах багаторівневого ієрархічного структурування серверів середнього рівня. Такий підхід дозволяє забезпечити високу масштабованість та відмовостійкість системи за рахунок раціонального розподілу навантаження між вузлами ієрархії. Проектована система орієнтована на забезпечення текстової комунікації в реальному часі та реалізує комплексний набір функцій управління сесіями.

Ключові функціональні модулі системи охоплюють чотири основні напрями управління:

#### **1. Управління безпекою**

Система підтримує диференційовані рівні доступу, що дозволяє класифікувати онлайн-конференції на два типи:

- Відкриті конференції: Орієнтовані на публічний доступ. Процедура приєднання не потребує попередньої автентифікації через ідентифікатор та пароль. Натомість реалізовано механізм спрощеної реєстрації, де користувач визначає псевдонім (аліас) для ідентифікації в межах поточної сесії.

- закриті конференції: Призначені для обмеженого кола верифікованих учасників. Доступ регламентується суворою політикою автентифікації (логін/пароль). Адміністратор системи здійснює попередню генерацію облікових даних та їх розсилку учасникам до початку заходу.

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
|      |      |          |        |      |                         | 18   |
| Змн. | Арк. | № докум. | Підпис | Дата |                         |      |

## 2. Конфігурація та рольова модель

У системі імплементовано трирівневу рольову модель доступу (RBAC), що визначає повноваження суб'єктів:

- Адміністратор: Має повні привілеї щодо створення, модерації та видалення віртуальних залів. У закритих конференціях адміністратор додатково керує реєстром облікових записів користувачів.

- Учасник: Володіє правами на інтерактивну взаємодію (читання та надсилання повідомлень).

- Гість: Наділений виключно правами на моніторинг контенту (читання) без можливості внесення змін у потік даних.

## 3. Управління доступом до слова

Для забезпечення керованості дискусії в системі реалізовано механізми динамічного модераторського контролю. Адміністратор, виконуючи функції модератора, має можливість у реальному часі змінювати статус учасників, надаючи або відкликаючи права на запис (виступ). Це особливо актуально для «режиму доповідача», де право на активну комунікацію спочатку має лише лектор, а іншим учасникам доступ відкривається лише під час сесії запитань та відповідей. Такий детермінований підхід мінімізує інформаційний шум та структурує перебіг онлайн-конференції.

## 4. Мета-менеджмент

Інформаційне супроводження конференцій (анонсування, розсилка запрошень) реалізовано через інтеграцію з протоколами електронної пошти. Система не містить вбудованого SMTP-сервера, натомість використовує інтерфейси взаємодії із зовнішніми поштовими шлюзами для виконання розсилок.

Процес взаємодії користувачів із системою під час проведення сесії формалізовано у вигляді діаграми активності (рис. 1.1). Ця діаграма відображає логічну послідовність дій від моменту ініціалізації конференції адміністратором до завершення комунікаційного акту учасниками.

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                         | 19   |

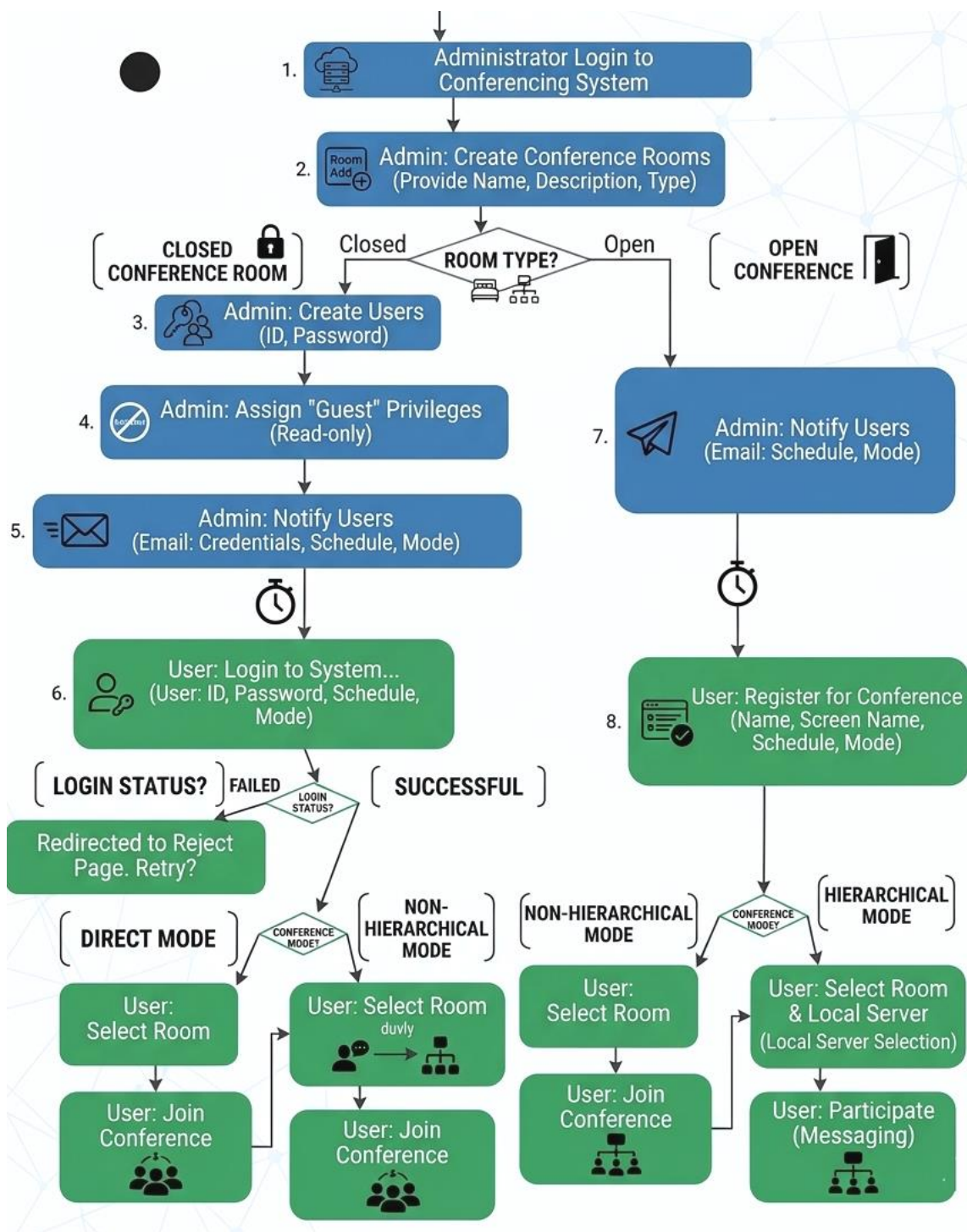


Рисунок 1.4 - Діаграма активності процесу проведення онлайн-конференції

Для кращого розуміння того, як розподіляються права та привілеї між різними типами користувачів у описаній системі, нижче наведено інтерактивну модель рольового доступу.

Рисунок 1.4 (діаграма активності) детермінує логіку взаємодії між адміністратором та кінцевими користувачами в процесі ініціалізації, конфігурування та проведення сесій онлайн-конференцій. Алгоритм структуровано за ієрархічним принципом із розгалуженням залежно від типу доступу до віртуального середовища.

#### 1. Етап ініціалізації та конфігурування системи

Процес розпочинається з етапу автентифікації адміністратора, що передбачає введення ідентифікатора (ID) та пароля для отримання доступу до панелі управління. Після успішного входу адміністратор здійснює ініціалізацію конференц-залів, визначаючи їхні атрибути: назву, опис та технічний тип. На цьому етапі відбувається ключове розгалуження логіки системи залежно від обраного типу конференції: закритого (Closed) або відкритого (Open) типу.

#### 2. Диференціація механізмів управління доступом

У закритих конференціях реалізується сувора політика безпеки. Адміністратор власноруч ініціює створення облікових записів для кожного учасника, генеруючи відповідні ідентифікатори та паролі. Наступним кроком є делегування повноважень, зокрема призначення привілеїв «Гість» (Guest) із правами виключно на читання (read-only) для відповідних категорій користувачів. Заключним кроком цього блоку є розсилка електронних сповіщень, що містять конфіденційні облікові дані, розклад та обраний режим функціонування системи.

У відкритих конференціях механізм доступу є ліберальнішим. Адміністратор лише анонсує подію через поштову розсилку, вказуючи часовий регламент та технічні параметри з'єднання.

#### 3. Автентифікація та реєстрація користувачів

Після настання запланованого часу розпочинається фаза входу користувачів:

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                         | 21   |

У закритому режимі користувач проходить процедуру верифікації через систему логіна/пароля. У разі неуспішної автентифікації спрацьовує механізм перенаправлення на сторінку відмови з можливістю повторної спроби.

У відкритому режимі користувач проходить процедуру самореєстрації, вказуючи ім'я та власні дані, після чого отримує доступ до вибору параметрів сесії.

#### 4. Вибір архітектурного режиму та завершення алгоритму

Фінальний етап алгоритму передбачає вибір режиму обробки даних (Conference Mode). Система пропонує три варіанти маршрутизації:

- Прямий режим (Direct Mode): Безпосередній вибір кімнати та приєднання до сесії.

- Неієрархічний режим (Non-Hierarchical Mode): Стандартна процедура вибору віртуального залу.

- Ієрархічний режим (Hierarchical Mode): Передбачає додатковий рівень вибору локального сервера (Local Server Selection) для оптимізації мережевого навантаження, що є критичним для територіально розподілених систем.

Алгоритм завершується фазою активної участі, що включає обмін мультимедійними та текстовими повідомленнями у межах обраної програмної інфраструктури. Така модель забезпечує високу керованість процесом та адаптивність до різних сценаріїв безпеки.

### 1.4 Висновки по розділу

У першому розділі проведено системний аналіз проблематики розробки веб-базованих систем розподіленої взаємодії користувачів, зокрема в контексті онлайн-конференцій. Розглянуто основні архітектурні принципи побудови таких систем та визначено ключові сфери їх застосування в

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                         | 22   |

сучасному цифровому середовищі. Здійснено класифікацію існуючих підходів і функціональних моделей, що дозволило виокремити характерні особливості різних типів систем. Проаналізовано архітектурні підходи до побудови ієрархічних систем, які забезпечують ефективне управління ролями, доступом та взаємодією користувачів. Отримані результати стали теоретичною основою для подальшого проєктування та реалізації досліджуваної системи.

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
|      |      |          |        |      |                         | 23   |
| Змн. | Арк. | № докум. | Підпис | Дата |                         |      |

РОЗДІЛ 2. АРХІТЕКТУРА ТА ПРИНЦИПИ ПОБУДОВИ ІЄРАРХІЧНОЇ СИСТЕМИ ОНЛАЙН-КОНФЕРЕНЦІЙ

2.1. Архітектурна модель та програмний базис ієрархічної системи онлайн-конференцій

Для забезпечення високої масштабованості, надійності та модульності проєктованої системи онлайн-конференцій обрано ієрархічну чотирирівневу архітектуру (рисунок 2.1). Такий підхід дозволяє чітко розмежувати зони відповідальності між різними функціональними компонентами системи, що є критично важливим для розподілених веб-додатків реального часу.

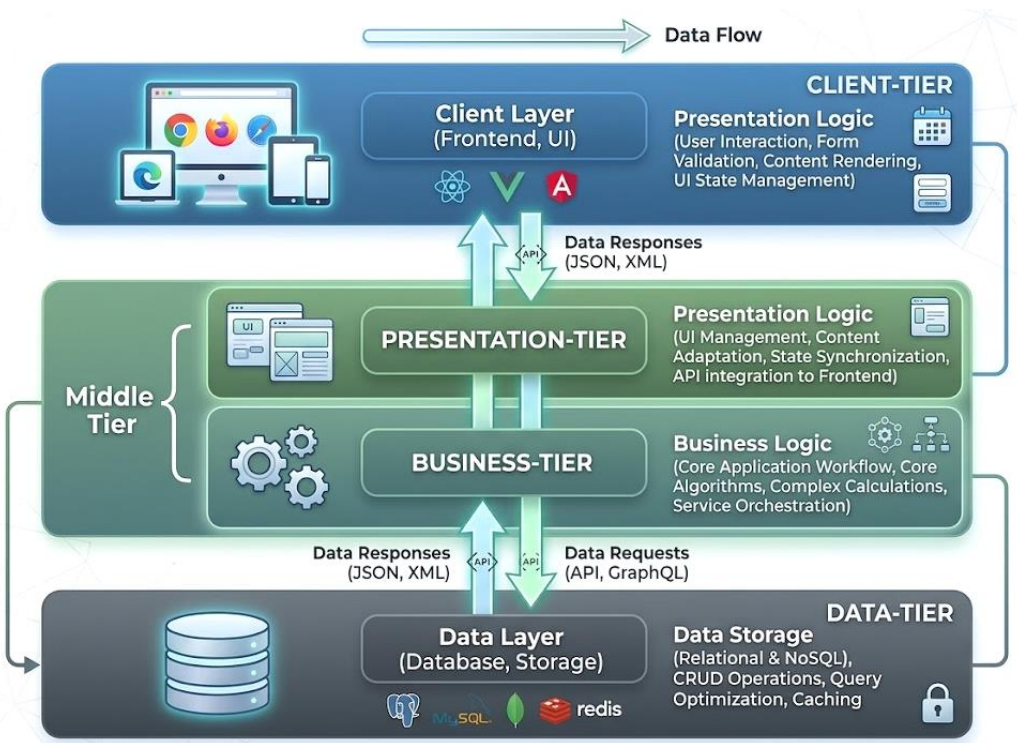


Рисунок 2.1 — Чотирирівнева архітектура системи онлайн-конференцій

Структура архітектурних рівнів системи:

- Клієнтський рівень (Client Tier) - забезпечує інтерфейсну взаємодію кінцевого користувача з додатком. Основним інструментом доступу є стандартний веб-браузер, що мінімізує вимоги до термінального обладнання.

- Рівень представлення (Presentation Tier): серверний рівень, відповідальний за генерацію динамічного вмісту інтерфейсу користувача та первинну обробку HTTP-запитів.

- Рівень бізнес-логіки (Business Tier): ядро системи, де імплементовано правила проведення онлайн-конференцій, механізми управління сесіями та логіку взаємодії компонентів.

- Рівень даних (Data Tier): забезпечує персистентне зберігання структурованої інформації (дані користувачів, логи сесій, налаштування конференц-залів) у реляційній базі даних.

Технологічним фундаментом для реалізації зазначених рівнів є платформа Java 2 Enterprise Edition (J2EE), яка надає стандартизоване розподілене середовище для виконання корпоративних додатків (рисунок 2.2).

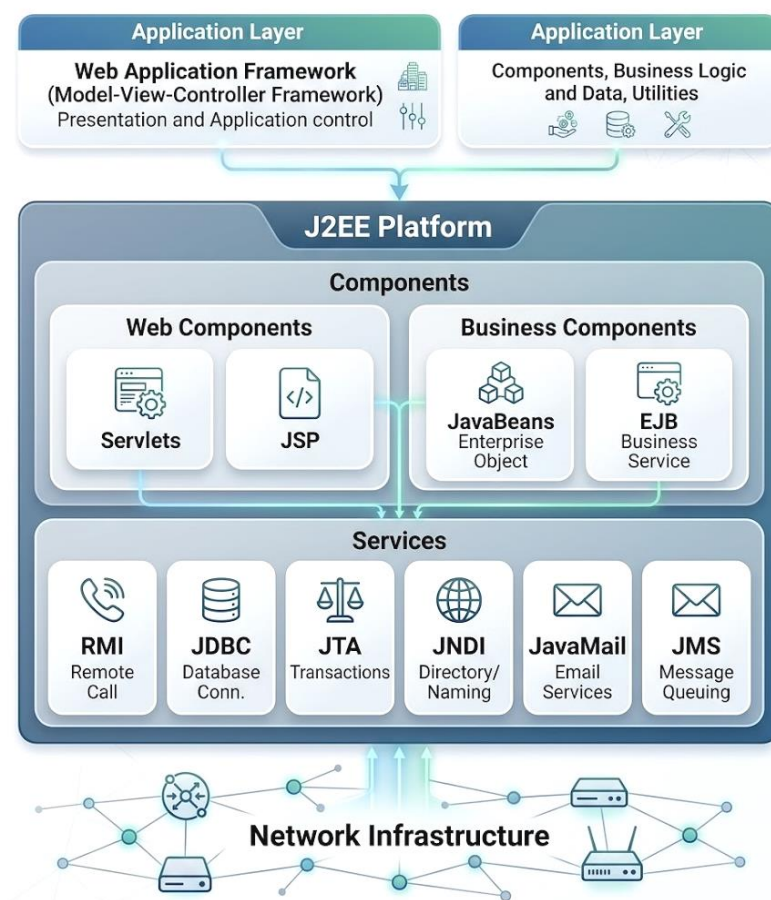


Рисунок 2.2 — Служби та компоненти програмно-технологічного базису системи

Основним інструментарієм розробки виступає набір прикладних програмних інтерфейсів (API) Java Enterprise, ключові з яких систематизовано в таблиці 2.1.

Таблиця 2.1 - Характеристика базових API платформи J2EE в контексті системи онлайн-конференцій

| Технологія (API)               | Опис та призначення в системі                                                                          |
|--------------------------------|--------------------------------------------------------------------------------------------------------|
| Java Servlets                  | Компонентно-орієнтований метод розширення функціональності веб-сервера для динамічної обробки запитів. |
| JSP (JavaServer Pages)         | Технологія створення динамічного контенту на основі статичних шаблонів HTML/XML.                       |
| JDBC (Database Connectivity)   | Уніфікований інтерфейс для взаємодії з широким спектром реляційних СКБД.                               |
| EJB (Enterprise JavaBeans)     | Модель компонентів для інкапсуляції складної бізнес-логіки та забезпечення транзакційності.            |
| JTA (Transaction API)          | Забезпечує цілісність даних через управління розподіленими транзакціями.                               |
| JMS (Messaging Services)       | Надає сервіси асинхронного обміну повідомленнями між компонентами системи.                             |
| JavaMail                       | Фреймворк для реалізації функцій сповіщення та запрошення учасників електронною поштою.                |
| JNDI (Naming Interface)        | Служба іменування та каталогів для ідентифікації ресурсів у мережевому середовищі.                     |
| RMI (Remote Method Invocation) | Протокол для виклику методів об'єктів, що розташовані на віддалених серверах.                          |

Інфраструктура виконання (Execution Infrastructure) системи (рисунок 2.3) базується на взаємодії наступних компонентів:

- веб-сервери: здійснюють обробку вхідних з'єднань та обслуговування клієнтських запитів. Архітектура системи передбачає сумісність із серверами стандартів J2EE (наприклад, Apache Tomcat), що забезпечує гнучкість розгортання.

- реляційна база даних: виконує роль централізованого сховища даних. Використання JDBC-сумісних драйверів дозволяє інтегрувати систему з сучасними СКБД (MySQL, PostgreSQL тощо).

- сервери розгалуження - спеціалізовані модулі на базі RMI-об'єктів,

призначені для інтенсивної обробки та маршрутизації повідомлень онлайн-конференцій між учасниками. Це дозволяє розвантажити основний веб-сервер від задач передачі поточкових даних.

клієнтське середовище - кросплатформний веб-браузер, що взаємодіє із сервером за протоколом HTTP/HTTPS.

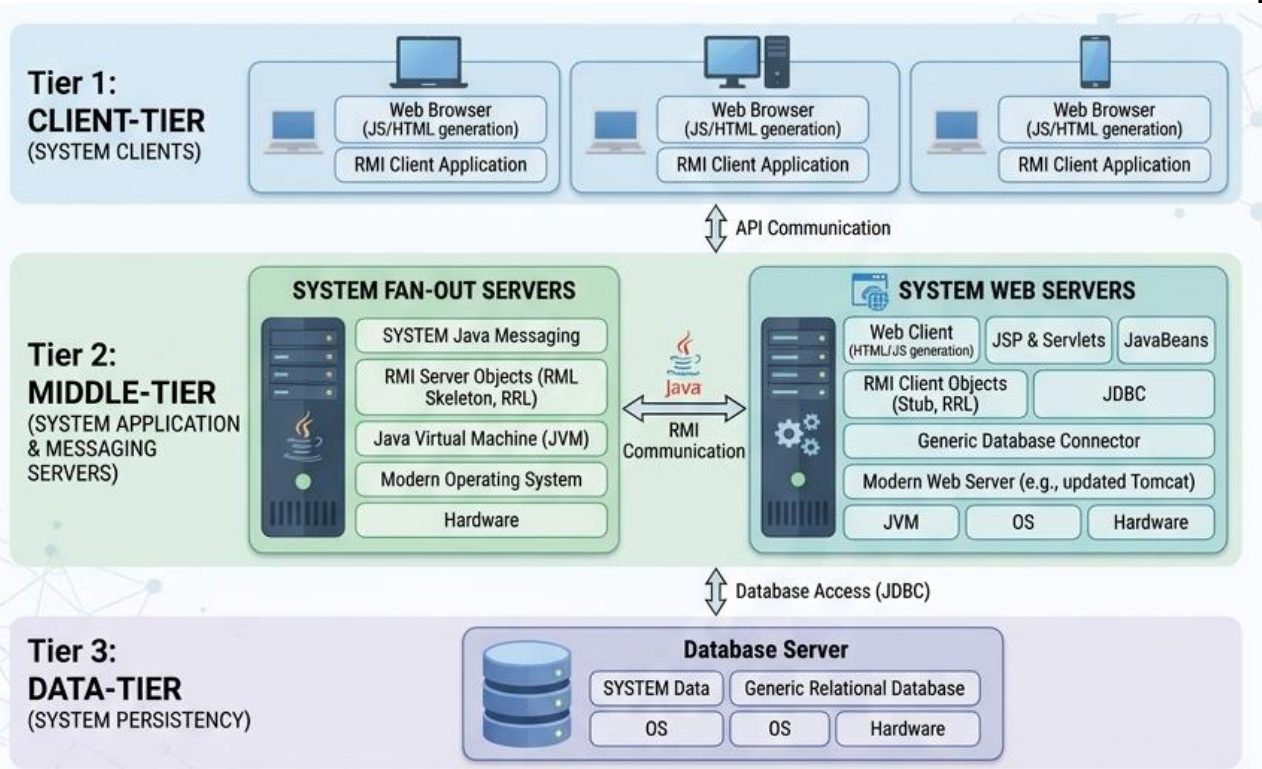


Рисунок 2.3 - Інфраструктура виконання та топологія системи

Дана архітектурна конфігурація забезпечує необхідний рівень абстракції, що дозволяє модернізувати окремі рівні системи без необхідності повної перебудови її інфраструктури, забезпечуючи високу живучість та адаптивність до навантажень.

2.2. Опис робочих процесів та функціональних сценаріїв управління сесіями онлайн-конференцій

Представлена діаграма активності (рисунок 2.4) детермінує алгоритміч-

ну послідовність взаємодії суб'єктів системи - адміністратора та кінцевих користувачів - у процесі організації та проведення сесій онлайн-конференцій. Логіка робочого процесу структурована за допомогою розподілу зон відповідальності та враховує варіативність архітектурних конфігурацій і політик безпеки.

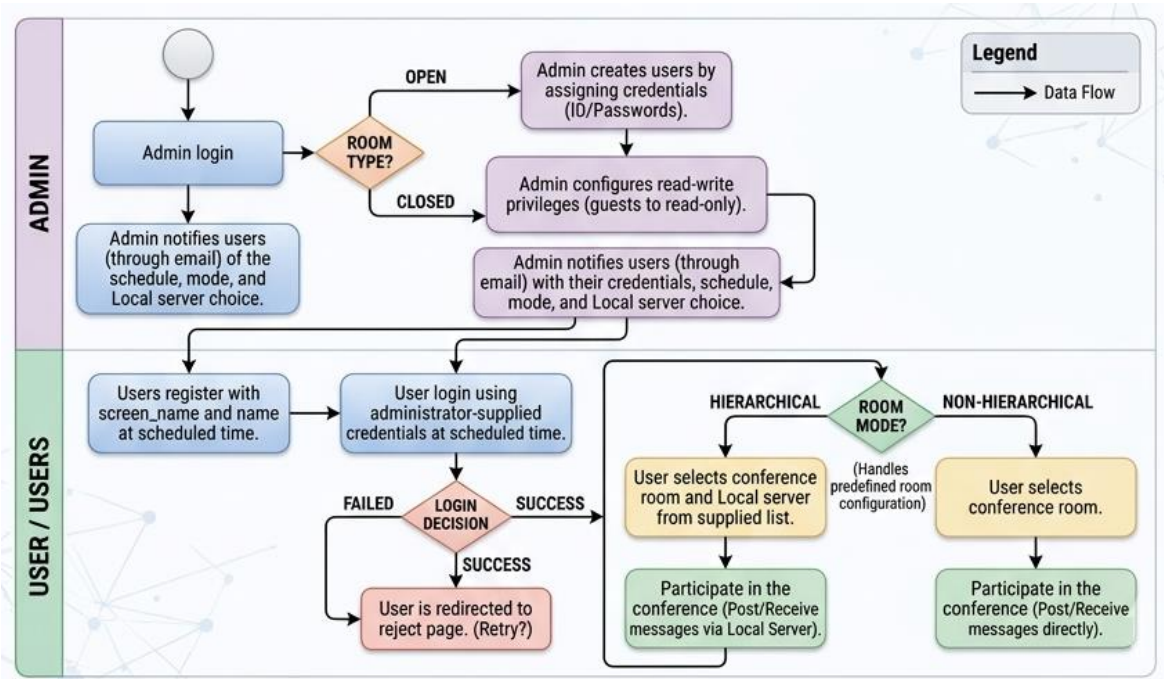


Рисунок 2.4 – Діаграма активності

Життєвий цикл сесії розпочинається з автентифікації адміністратора, після чого здійснюється вибір типу віртуальної кімнати (Room Type). Алгоритм передбачає два основні сценарії розгортання:

1. Відкритий режим (OPEN) характеризується мінімальними бар'єрами для входу. Адміністратор здійснює інформаційну розсилку щодо параметрів сесії, а користувачі проходять процедуру самореєстрації, вказуючи ідентифікаційне ім'я (screen\_name) безпосередньо перед приєднанням.
2. Закритий режим (CLOSED): базується на суворій моделі автентифікації. Адміністратор ініціює створення облікових записів (ID/Password) та здійснює детерміновану конфігурацію прав доступу, зокрема обмежуючи привілеї певних груп учасників до рівня «тільки

читання» (read-only). Після завершення етапу преконфігурації система генерує сповіщення з персональними обліковими даними для кожного учасника.

Для закритих конференцій критичним етапом є прийняття рішення щодо автентичності суб'єкта (Login Decision). У разі негативного результату верифікації спрацьовує механізм обробки винятків із перенаправленням користувача на сторінку відмови та можливістю повторної ітерації входу. Успішна авторизація дозволяє суб'єкту перейти до вибору функціонального режиму роботи.

Фінальний етап алгоритму визначається обраним режимом онлайн-конференції (Room Mode), що безпосередньо корелює із мережевою топологією системи:

- Ієрархічний режим (HIERARCHICAL): орієнтований на оптимізацію навантаження у територіально розподілених мережах. Користувач здійснює вибір не лише віртуального залу, а й конкретного локального сервера (Local Server) зі списку запропонованих. Подальша інтеракція (Post/Receive messages) реалізується через вибраний проміжний вузол.

- Прямий режим (NON-HIERARCHICAL): передбачає спрощену процедуру вибору кімнати та безпосередній обмін інформаційними потоками в межах єдиної програмної інфраструктури.

Системний потік завершується фазою активного співробітництва, що полягає у синхронному або асинхронному обміні повідомленнями, забезпечуючи реалізацію мети функціонування розробленої програмної моделі.

### **2.3. Реалізація архітектурного шаблону Model-View-Controller у середовищі ієрархічної системи онлайн-конференцій**

Проектування архітектури запропонованої системи онлайн-конференцій

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                         | 29   |

базується на використанні де-факто стандарту розробки інтерактивних веб-додатків — архітектурного шаблону Model-View-Controller (MVC). Вибір даного патерну обумовлений необхідністю чіткої сегрегації обов'язків між функціональними компонентами, що забезпечує гнучкість масштабування та спрощує процес модернізації системи. Реалізація MVC у межах розробленого рішення базується на сервісах та компонентах платформи J2EE, як це було ілюстровано раніше (рис. 2.2).

Відповідно до парадигми MVC, програмний комплекс розподіляється на три автономні, але взаємопов'язані шари: модель, представлення та контролер. Кожен із рівнів виконує специфічні функції та має чітко визначені інтерфейси взаємодії:

#### 1. Шар моделі (Model)

Компонент Model є центральною ланкою архітектури, що інкапсулює бізнес-дані системи та логіку їх обробки. В контексті онлайн-конференцій модель відповідає за управління станом віртуальних кімнат, зберігання повідомлень та автентифікацію користувачів.

Як реалізація, у пропонованій системі роль моделі виконують об'єкти JavaBeans, які інкапсулюють бізнес-сутності, та RMI-об'єкти (Remote Method Invocation), що забезпечують доступ до стану додатка з віддалених вузлів ієрархії.

Для довготривалого зберігання даних використовуються таблиці реляційної бази даних, доступ до яких здійснюється через рівень моделі. Вона надає контролеру методи для модифікації даних, а представленню — можливість запиту актуального стану.

#### 2. Шар представлення (View)

Компонент View відповідає за візуалізацію даних та формування динамічного інтерактивного інтерфейсу користувача. У сучасній архітектурі цей шар функціонує як автономний клієнтський додаток, що дозволяє відокремити логіку відображення від серверних обчислень.

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
|      |      |          |        |      |                         | 30   |
| Змн. | Арк. | № докум. | Підпис | Дата |                         |      |

Рівень представлення базується на парадигмі Single Page Application (SPA). Основним інструментарієм є сучасні компонентно-орієнтовані фреймворки та бібліотеки (наприклад, React, Angular або Vue.js).

На відміну від класичних підходів, View не генерується сервером у вигляді готових HTML-сторінок. Натомість клієнтська частина асинхронно запитує дані у контролера через RESTful API у форматі JSON. Це забезпечує високу реактивність інтерфейсу та ефект «безшовного» оновлення контенту, що є критично важливим для систем передачі повідомлень та мультимедійних потоків у реальному часі.

Шар представлення динамічно реагує на зміни в моделі, трансформуючи внутрішній стан системи у формат, придатний для відображення у веб-браузері клієнта (HTML/XML/JSON). Це дозволяє відокремити логіку відображення від механізмів обробки даних.

### 3. Шар контролера (Controller)

Компонент Controller функціонує як стратегічний координатор (оркестратор), що детермінує логіку обробки подій, керує маршрутизацією запитів та забезпечує взаємодію між клієнтським інтерфейсом і бізнес-логікою системи.

У сучасній архітектурі роль контролера виконує набір REST-контролерів (наприклад, реалізованих на базі фреймворку Spring Boot або Spring MVC). На відміну від класичних сервлетів, вони орієнтовані на архітектурний стиль REST (Representational State Transfer).

Контролер ідентифікує тип вхідного запиту (GET, POST, PUT, DELETE), здійснює валідацію вхідних даних та делегує виконання бізнес-операцій відповідним сервісам моделі. Результат операції серіалізується у формат JSON або XML і повертається клієнтській частині (View).

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                         | 31   |

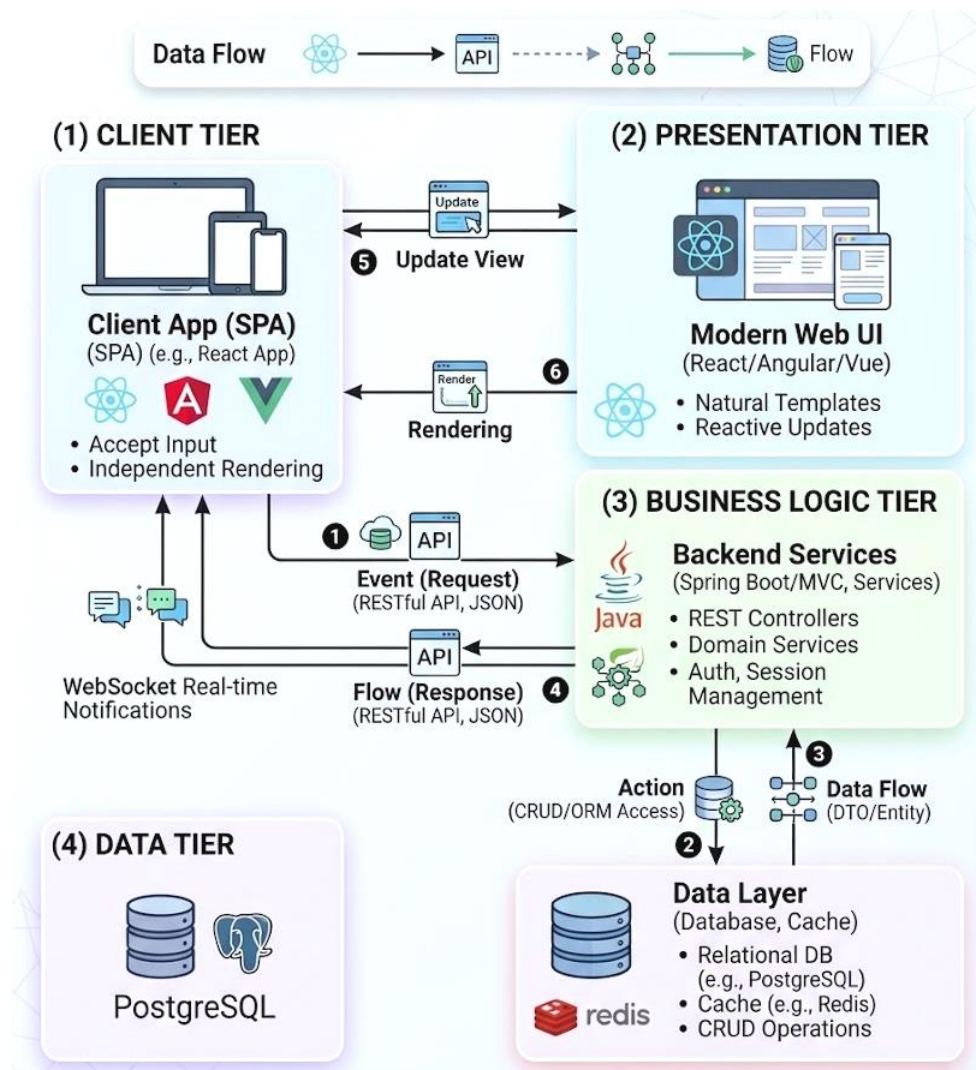


Рисунок 2.5 — Схема взаємодії компонентів MVC у ієрархічній системі онлайн-конференцій

Враховуючи специфіку онлайн-конференцій, шар контролера також включає обробники протоколу WebSockets (або контролери повідомлень STOMP/MQTT). Це дозволяє реалізувати подієво-орієнтовану модель, де контролер ініціює миттєву розсилку повідомлень усім активним учасникам конференції без прямого запиту з їхнього боку.

Механізм роботи: Контролер інтерпретує дії користувача, ініціює відповідні операції в об'єктах моделі (створення JavaBeans або активація RMI-об'єктів) та вибирає наступне представлення для відображення результату. Таким чином, контролер забезпечує цілісність логічного

переходу між станами системи онлайн-конференції.

Взаємозв'язок між структурними компонентами шаблону MVC та циклічність обміну даними схематично зображено на рисунку 2.5.

Використання даної архітектурної моделі дозволяє досягти високої ступені автономності компонентів. Зокрема, розробник може змінювати візуальний інтерфейс (View) без втручання в бізнес-логіку (Model), або мігрувати на іншу систему управління базами даних, не змінюючи алгоритми контролера. Це є критично важливим для систем онлайн-конференцій, що працюють у гетерогенних мережевих середовищах.

## **2.4. Ієрархічна модель середнього рівня та механізми масштабованості системи**

Середній рівень архітектури (Middle Tier), схематично представлений на рисунку 2.3, відіграє критичну роль у забезпеченні функціонування систем онлайн-конференцій, оскільки саме на ньому реалізується логіка обробки запитів та маршрутизації медіапотоків. Основними структурними одиницями цього рівня є веб-сервери та сервери розгалуження.

За умови низької інтенсивності навантаження (невелика кількість активних учасників) система може функціонувати на базі єдиного веб-сервера, який одноосібно виконує завдання з обробки клієнтських запитів та дистрибуції повідомлень. Проте зі зростанням кількості одночасних сесій та учасників виникає проблема обмеження апаратних ресурсів одного вузла. Для вирішення цієї проблеми застосовується стратегія горизонтального масштабування шляхом ієрархічної організації множини серверів розгалуження та веб-серверів. Така топологія дозволяє ефективно розподіляти обчислювальне навантаження та забезпечувати реплікацію повідомлень у реальному часі.

В основі ієрархічної архітектури лежить деревоподібна структура, де

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                         | 33   |

кожен об'єкт (вузол) має строго одного батьківського вузла та може мати необмежену кількість дочірніх вузлів. Вершиною такої структури є корінь (root). Приклад подібної топології наведено на рисунку 2.6, де вузол R1 виступає кореневим сервером, P1 та P2 є дочірніми вузлами першого рівня, а Q1–Q4 — термінальними або проміжними вузлами наступних рівнів.

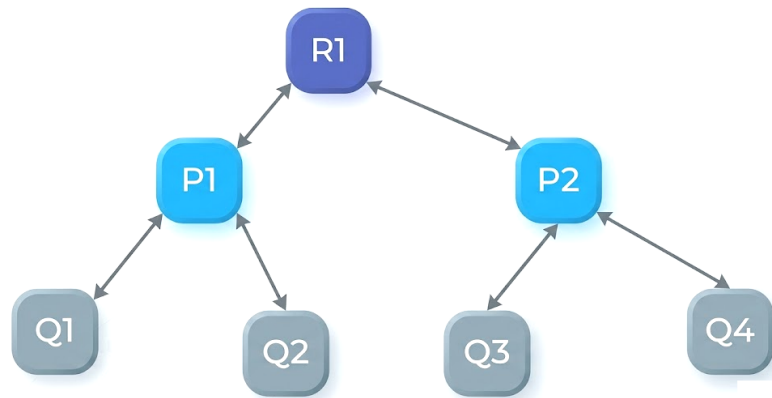


Рисунок 2.6 — Приклад топології ієрархічної архітектури онлайн-конференції

Для практичної реалізації розподілених об'єктів у межах ієрархії середнього рівня доцільно використовувати технологію Java Remote Method Invocation (RMI). RMI забезпечує прозору взаємодію між об'єктами, розташованими у різних віртуальних машинах Java (JVM), що є критично важливим для масштабованості. Архітектура RMI базується на трьох рівнях абстракції:

- Шар Stub/Skeleton: рівень клієнтських заглушок (stubs) та серверних скелетів (skeletons). Він діє як проксі-об'єкт, що перехоплює виклики клієнта, здійснює маршалінг (серіалізацію) даних та передає їх на наступний рівень. Скелет на стороні сервера демаршалізує дані та передає виклик безпосередньо віддаленому об'єкту.

- Шар віддалених посилань (Remote Reference Layer, RRL): відповідає

за семантику викликів та управління з'єднаннями (наприклад, підтримку одноадресної або багатоадресної передачі).

- Транспортний шар (Transport Layer): забезпечує фізичне встановлення та підтримку з'єднань. У стандартних реалізаціях JVM цей шар базується на протоколі TCP/IP [54], що гарантує надійність доставки пакетів даних.

Реалізація ієрархії за допомогою RMI представлена на рисунку 2.7. Механізм взаємодії передбачає, що скелет вузла P1 демаршалізує дані [26], отримані від стаба Q1, після чого вони можуть бути репліковані далі за ієрархією (наприклад, до стаба P1 для передачі R1).

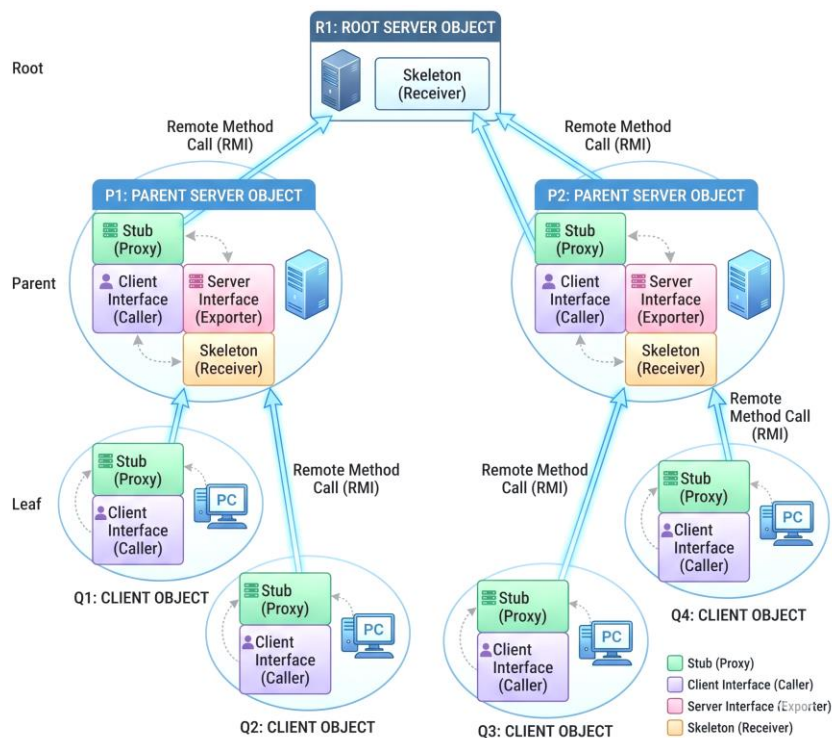


Рисунок 2.7 — Реалізація ієрархічної взаємодії за допомогою механізмів RMI

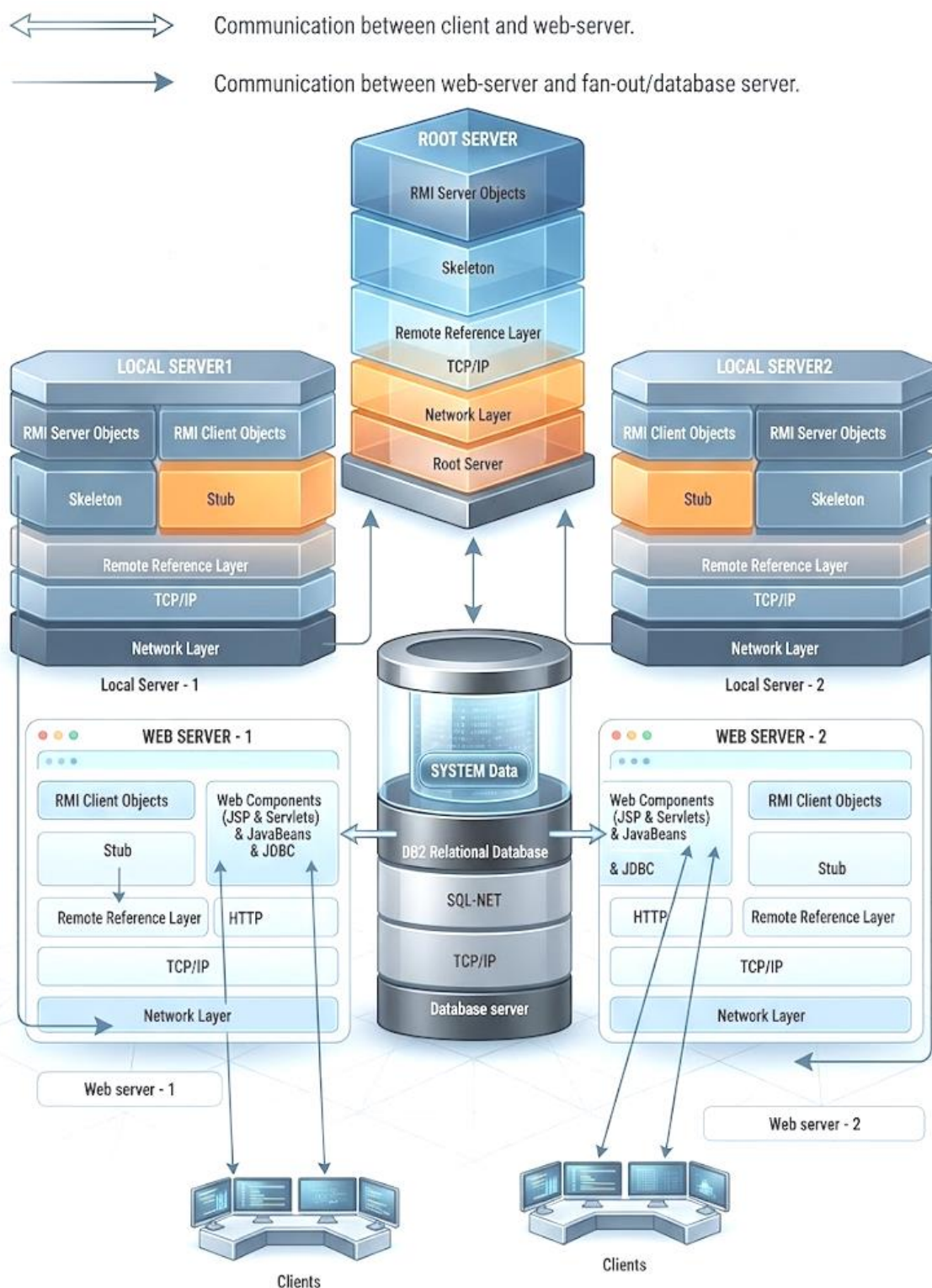


Рисунок 2.8 — Модель каскадної реплікації даних в ієрархічній системі

Практична модель реплікації повідомлень у системі онлайн-конференцій передбачає наступний алгоритм (див. рисунок 2.8):

- Кореневий сервер розгалуження (Root Fan-out Server, R1): здійснює глобальну координацію та синхронізацію медіапотоків між гілками дерева.

- Локальні сервери розгалуження (P1, P2): виступають проміжними ланками (батьківськими вузлами) для групи веб-серверів, забезпечуючи локальну реплікацію.

- Веб-сервери (Q1–Q4): безпосередньо взаємодіють із клієнтськими додатками учасників. Отримане від клієнта повідомлення веб-сервер Q1 передає своєму батьківському вузлу P1. Останній реплікує дані серед усіх своїх «дітей» (інших веб-серверів своєї групи) та паралельно передає пакет кореневому серверу R1. Корінь, у свою чергу, розсилає повідомлення іншим локальним серверам (P2), які доставляють його кінцевим веб-серверам (Q3,Q4).

Така багаторівнева реплікація дозволяє уникнути перевантаження центрального вузла та мінімізувати затримки при обслуговуванні великої кількості географічно розподілених користувачів.

## 2.5 Висновки по розділу

У другому розділі сформовано архітектурну модель ієрархічної системи онлайн-конференцій та обґрунтовано вибір її програмного базису. Визначено ключові принципи побудови системи, що забезпечують її масштабованість, модульність і ефективність функціонування. Описано робочі процеси та функціональні сценарії управління сесіями, що дозволяє формалізувати взаємодію між користувачами та системними компонентами. Реалізація архітектурного шаблону Model-View-Controller сприяла структуризації програмної системи та підвищенню її підтримуваності. Запропонована архітектура створює передумови для подальшого розвитку та інтеграції системи.

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
|      |      |          |        |      |                         | 37   |
| Змн. | Арк. | № докум. | Підпис | Дата |                         |      |

## РОЗДІЛ 3. ПРОЕКТУВАННЯ СТРУКТУРИ ІЄРАРХІЧНОЇ СИСТЕМИ ОНЛАЙН ВЗАЄМОДІЇ КОРИСТУВАЧІВ

### 3.1. Методологія проектування та розробки системи онлайн-конференцій

Процес створення складної розподіленої системи онлайн-конференцій потребує застосування структурованого підходу до проектування, який забезпечує узгодженість між функціональними вимогами та технічною реалізацією. Методологія розробки, обрана для даної роботи, базується на ітеративній моделі життєвого циклу програмного забезпечення, що візуалізована на рисунку 3.1.

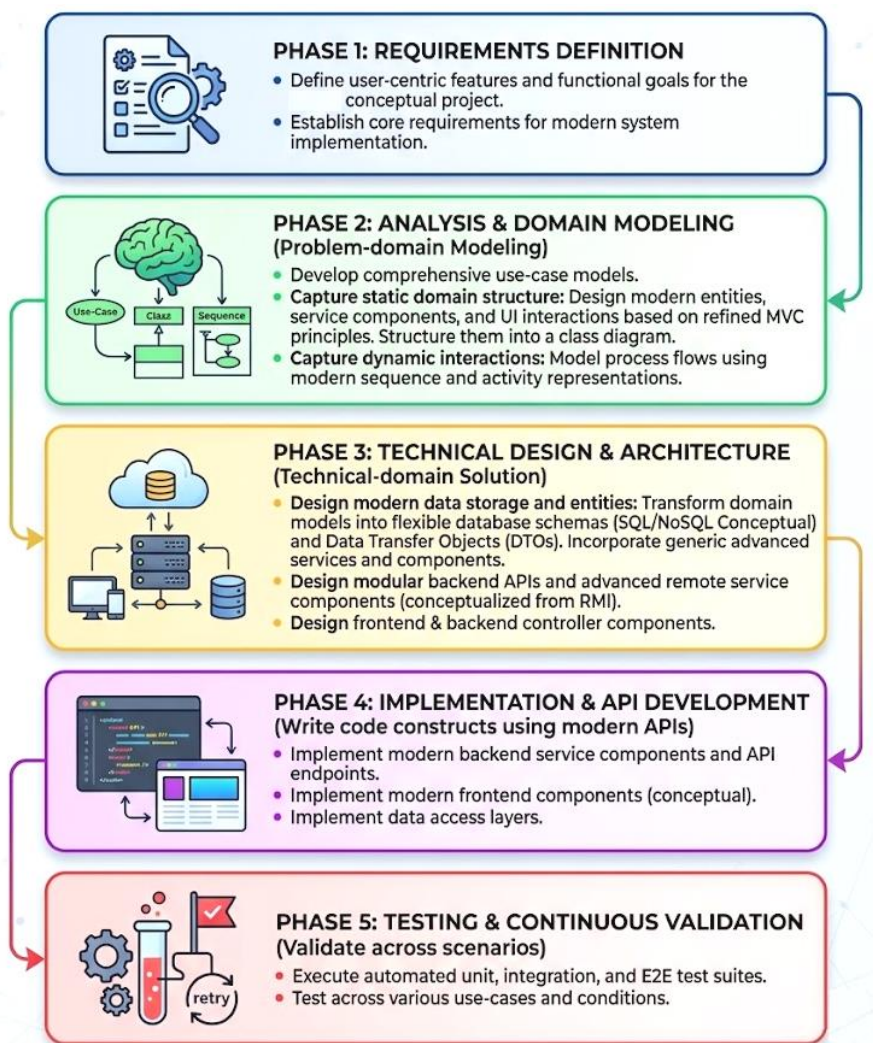


Рисунок 3.1 — Методологія проектування та етапи розробки системи

Ключовою особливістю даної методології є використання інструментарію уніфікованої мови моделювання UML для специфікації, візуалізації та документування артефактів системи. Використання UML дозволяє формалізувати архітектурні рішення на різних рівнях абстракції, що мінімізує ризики виникнення помилок на етапі імплементації.

Представлена на рисунку 3.1 методологія відображає систематизований, ітеративний підхід до проектування та програмної реалізації сучасної системи онлайн-конференцій. Алгоритм розробки структуровано за п'ятьма послідовними фазами, кожна з яких детермінує специфічний набір артефактів та інженерних рішень.

#### Фаза 1: Визначення вимог (Requirements Definition)

Початковий етап присвячений концептуалізації проекту та формалізації функціональних цілей. Основна увага зосереджена на орієнтованих на користувача функціях (user-centric features), що стають фундаментом для специфікації системи. На цій стадії закладаються базові вимоги до архітектурної стійкості та масштабованості майбутнього програмного продукту.

#### Фаза 2: Аналіз та доменне моделювання (Analysis & Domain Modeling)

Другий етап передбачає перехід від абстрактних вимог до побудови логічних моделей проблемної області. Процес включає:

- Розробку комплексних діаграм прецедентів (Use Case Models) для опису взаємодії акторів із системою.
- Формування статичної структури домену через діаграми класів, що базуються на рефлексивних принципах MVC.
- Моделювання динамічних аспектів системи за допомогою діаграм послідовності та активності, що детермінують потоки керування та передачу повідомлень у реальному часі.

#### Фаза 3: Технічне проектування та архітектура (Technical Design & Architecture)

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                         | 39   |

На даному етапі здійснюється трансформація доменних моделей у конкретні технічні рішення. Особлива увага приділяється:

- Проектуванню рівнів зберігання даних (Data persistent layers) із використанням гнучких схем SQL/NoSQL та об'єктів передачі даних (DTO).
- Розробці модульної архітектури бекенд-інтерфейсів (APIs) та компонентів віддалених сервісів, що забезпечують ієрархічну реплікацію повідомлень.
- Сегрегації обов'язків між фронтенд- та бекенд-контролерами згідно з сучасною декопльованою моделлю MVC.

#### Фаза 4: Імплементация та розробка API (Implementation & API Development)

Етап реалізації полягає у безпосередньому написанні програмного коду з використанням сучасних стеків технологій та інструментарію Java Enterprise. Ключові операції включають розгортання серверних компонентів, створення кінцевих точок API (endpoints) та імплементацию рівнів доступу до даних. Процес кодування базується на принципах чистої архітектури, що полегшує подальшу підтримку системи.

#### Фаза 5: Тестування та безперервна валідація (Testing & Continuous Validation)

Заключна фаза методології спрямована на забезпечення якості та верифікацію системи за різними сценаріями експлуатації. Цикл валідації охоплює автоматизоване модульне тестування (Unit testing), інтеграційне тестування компонентів та наскрізне (E2E) тестування всієї інфраструктури. Це дозволяє гарантувати відповідність реалізованого функціоналу вихідним специфікаціям та стійкість системи до високих навантажень.

Загалом, запропонована методологія забезпечує цілісність процесу розробки — від ідентифікації потреб користувача до розгортання верифікованої та масштабованої системи онлайн-конференцій. Застосована методологія забезпечує прозорість процесу розробки та дозволяє адаптувати

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                         | 40   |

систему онлайн-конференцій до специфічних умов експлуатації, зберігаючи при цьому цілісність її архітектурного каркаса.

### 3.2. Функціональне моделювання сценаріїв взаємодії та аналіз прецедентів системи

Важливим етапом проектування системи онлайн-конференцій є формалізація функціональних вимог шляхом побудови моделей прецедентів. Даний підхід дозволяє чітко детермінувати межі системи, ідентифікувати ключових акторів та визначити їхні ролі в процесі ініціалізації та проведення віртуальних засідань. Загальна модель використання системи, що відображає консолідовану структуру взаємодії суб'єктів із програмним середовищем, представлення на рисунку 3.2.



Рисунок 3.2 — Узагальнена модель прецедентів системи онлайн-конференцій

Представлена діаграма відображає концептуальну модель функціональних вимог до системи онлайн-конференцій, детермінуючи межі взаємодії між різними категоріями користувачів та ключовими сервісами системи. Нижче наведено науково-технічний опис структури та логіки взаємодії, відображеної на діаграмі:

### 1. Суб'єкти взаємодії (Актори)

Архітектура системи передбачає три рівні доступу, що представлені відповідними акторами:

- Administrator - суб'єкт із найвищим рівнем повноважень, відповідальний за стратегічне управління та модерацію.
- Conferee - основний зареєстрований користувач системи із розширеним функціоналом інтеракції.
- Guest - Суб'єкт із обмеженими правами доступу, що бере участь у специфічних сценаріях конференцій.

### 2. Аналіз функціональних сценаріїв

На рисунку 3.2 формалізовано наступні вектори функціональності:

Управлінський вектор (Administrator):

- Administer: Прецедент, що охоплює системне адміністрування, конфігурування параметрів онлайн-середовища та управління обліковими записами.
- Moderate conferences: Процес модерації активних сесій, що включає контроль за дотриманням регламенту та управління правами учасників у реальному часі.

Кооперативний вектор (Conferee):

- Participate in open conferences: Сценарій вільного приєднання до публічних заходів, що не потребують попередньої авторизації в межах закритих груп.
- Participate in closed conferences: Взаємодія в межах захищених сесій, що передбачає обов'язкову автентифікацію та перевірку прав доступу.

Обмежений вектор (Guest):

- Participate in closed conferences: Спеціалізований сценарій доступу для незареєстрованих або тимчасових користувачів до закритих заходів (зазвичай у режимі обмежених прав або пасивного спостерігача).

### 3. Системна інтерпретація

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
|      |      |          |        |      |                         | 42   |
| Змн. | Арк. | № докум. | Підпис | Дата |                         |      |

Графічна репрезентація підтверджує декопльований підхід до проектування інтерфейсів. Кожен прецедент виступає як точка входу до бізнес-логіки системи, що обробляється відповідним контролером. Використання чітко розмежованих ролей дозволяє реалізувати модель безпеки на основі ролей (RBAC), що є критично важливим для забезпечення конфіденційності та цілісності даних у системах онлайн-конференцій.

Ця модель слугує базою для подальшої розробки діаграм послідовності та класів, забезпечуючи відповідність програмної реалізації вихідним функціональним вимогам проекту.

Для детального аналізу функціональних можливостей кожної категорії користувачів розроблено серію підмоделей, що описують специфічні сценарії використання системи залежно від рівня повноважень та типу конференції.

### 3.2.1. Роль адміністратора системи

Адміністратор виступає ключовим суб'єктом управління інфраструктурою. Підмодель використання для цієї ролі (рисунок 3.3) охоплює процеси конфігурування віртуальних кімнат, генерацію облікових записів для закритих сесій, призначення привілеїв доступу та моніторинг стану серверів розгалуження.

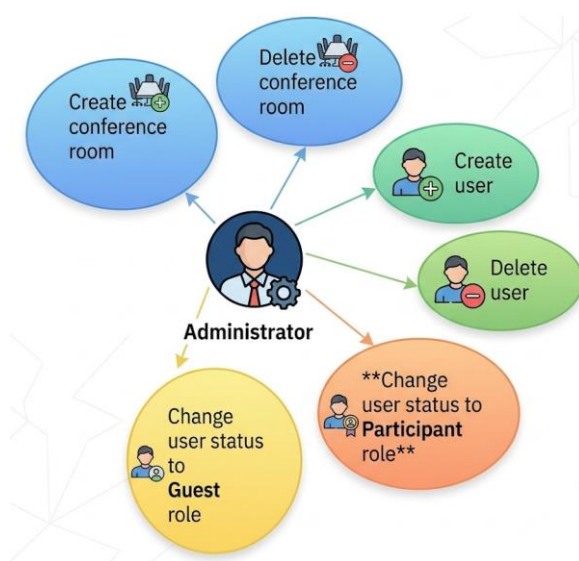


Рисунок 3.3 – Підмодель функціональних можливостей адміністратора

У центрі моделі розташований актор Administrator, який виступає основним ініціатором усіх представлених прецедентів. Його роль полягає в оркестрації ресурсів системи та забезпеченні дотримання політик безпеки через механізми управління доступом.

На діаграмі (рис. 3.3) виділено три ключові вектори адміністративної діяльності, що забезпечують цілісність функціонування середовища онлайн-конференцій:

Управління віртуальною інфраструктурою:

- Create conference room / Delete conference room: прецеденти, що відповідають за життєвий цикл інфраструктурних об'єктів. Це передбачає динамічне виділення серверних ресурсів для нових сесій та їх деалокацію після завершення заходів.

Адміністрування облікових записів користувачів:

- Create user / Delete user: базові операції управління ідентифікацією (Identity Management). Вони забезпечують формування реєстру легітимних користувачів та видалення суб'єктів, що втратили право доступу до системи.

Декомпозиція повноважень та управління ролями:

- Change user status to Participant / Guest role: ключові сценарії реалізації моделі управління доступом на основі ролей. Ці прецеденти дозволяють адміністратору гнучко змінювати рівень функціональних можливостей користувачів у межах конкретної конференції, трансформуючи їхній статус від пасивного спостерігача до активного учасника.

### 3.2.2. Сценарії взаємодії в межах закритих конференцій

У закритих системах онлайн-конференцій реалізується сувора ієрархія прав доступу, що відображено у двох підмоделях.

Роль гостя (Guest): Сценарій використання (рисунок 3.4) передбачає обмежений функціонал, що зазвичай зводиться до пасивного спостереження за ходом конференції (режим read-only) без права ініціації медіапотоків або

редагування спільних ресурсів.

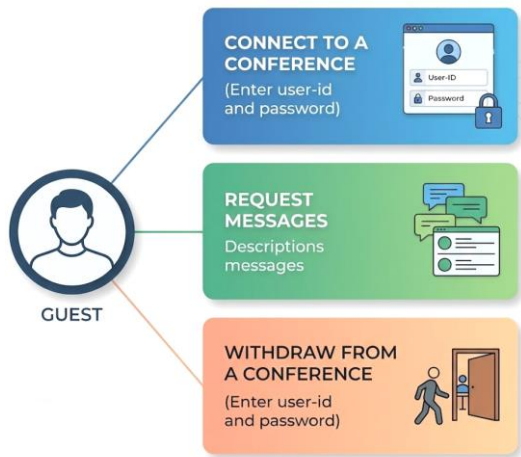


Рисунок 3.4 — Підмодель використання для ролі гостя у закритій конференції

Роль учасника (Participant): На відміну від гостя, авторизований учасник закритої конференції (рисунок 3.5) володіє повним набором інструментів інтеракції, включаючи можливість трансляції аудіо/відео даних, обміну текстовими повідомленнями та використання інструментів спільної роботи після проходження процедури автентифікації.



Рисунок 3.5 — Підмодель використання для учасника закритої конференції

Основним актором на даній діаграмі (рис. 3.5) є conferee (учасник) — зареєстрований суб'єкт системи, який володіє повноваженнями для активної взаємодії в межах конференційної сесії. На відміну від ролі адміністратора, функціонал учасника зосереджений на процесах споживання та генерації контенту в реальному часі.

Діаграма виділяє чотири фундаментальні сценарії використання, що забезпечують повний життєвий цикл участі в онлайн-заході:

- Автентифікація та доступ (Connect to a conference): Прецедент, що реалізує механізми безпеки. Процес передбачає введення ідентифікатора (user-id) та пароля (password), що є тригером для сервісів автентифікації на рівні моделі (Model) та встановлення сесійного з'єднання.

- Інтерактивна взаємодія (Post messages): Сценарій, що відповідає за ініціацію вихідного інформаційного потоку. Учасник генерує повідомлення, які через контролер (Controller) реплікуються ієрархічною мережею серверів розгалуження для інших суб'єктів.

- Отримання інформації (Request messages): Функція запиту або асинхронного отримання оновлень. У сучасній інтерпретації цей прецедент зазвичай реалізується через протокол WebSockets, забезпечуючи реактивне оновлення інтерфейсу (View) при надходженні нових даних.

- Термінація сесії (Withdraw from a conference): Сценарій коректного виходу з конференції. Виконання цього прецеденту ініціює процедури звільнення системних ресурсів вузла та оновлення статусу учасника в базі даних.

### 3.2.3. Роль учасника у відкритих конференціях

Сценарій участі у відкритих онлайн-конференціях (рисунок 3.6) характеризується ліберальнішою моделлю доступу. Основний акцент зміщується на процедуру самореєстрації учасника безпосередньо перед приєднанням до сесії. Функціонал такого користувача залишається широким,

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                         | 46   |

проте не потребує попереднього адміністрування облікових записів, що спрощує масштабування публічних заходів.



Рисунок 3.6 — Підмодель використання для учасника відкритої конференції

Основним логічним актором на діаграмі виступає Conferee — авторизований користувач, чия діяльність зосереджена на безпосередній участі в інформаційному обміні. У межах ієрархічної моделі системи цей суб'єкт представляє кінцевий вузол, що ініціює запити на обслуговування до серверів вищого рівня.

Функціональна структура взаємодії учасника класифікується за чотирма фундаментальними сценаріями:

- Ініціалізація доступу (Register for conference): Даний прецедент описує процедуру реєстрації учасника в конкретній конференційній сесії. Процес передбачає передачу ідентифікаційних параметрів користувача до системи для проведення валідації та надання відповідних повноважень у межах обраного віртуального простору.

- Генерація інформаційного потоку (Post messages): Прецедент, що забезпечує реалізацію активної комунікації. Він ініціює вихідний потік даних від клієнтського термінала, який через рівень контролерів (Controller) реплікується по всій ієрархічній структурі серверів для доставки іншим

учасникам.

- Синхронізація інформаційного стану (Request messages): Операція, відповідальна за отримання актуальних даних сесії. Цей сценарій реалізує механізм «підписки» на оновлення, що дозволяє підтримувати консистентність діалогового середовища для всіх активних вузлів системи.

- Термінація сесійної взаємодії (Withdraw from a conference): Заключний прецедент, що відповідає за логічне завершення участі в конференції. Виконання цієї дії активує процедури звільнення зарезервованих системних ресурсів, коректного розірвання з'єднання та оновлення статусу суб'єкта в глобальному реєстрі системи.

Деталізація цих підмоделей дозволяє забезпечити високий рівень безпеки та керованості в системі онлайн-конференцій, адаптуючи інтерфейс та доступні сервіси під конкретні потреби кожного суб'єкта взаємодії.

### **3.3. Об'єктно-орієнтована декомпозиція та архітектурна організація компонентів ієрархічної системи онлайн-конференцій**

Проектування архітектури сучасної системи онлайн-конференцій базується на принципах модульності та сепарації відповідальності. Проблемну область розроблюваної системи доцільно класифікувати на чотири основні функціональні модулі, які визначають логічну структуру класів та механізми їхньої взаємодії (таблиця 3.1).

Для забезпечення гнучкості та масштабованості програмного забезпечення реалізація об'єктної моделі здійснюється на основі архітектурного шаблону Model-View-Controller (MVC). Даний підхід дозволяє відокремити бізнес-логіку від інтерфейсу користувача та механізмів обробки запитів.

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
|      |      |          |        |      |                         | 48   |
| Змн. | Арк. | № докум. | Підпис | Дата |                         |      |

Таблиця 3.1 - Функціональна декомпозиція модулів системи

| Модуль                                              | Призначення та ключові функції                                                                                                                                                    |
|-----------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Модуль Front/Login</b>                           | Забезпечує механізми автентифікації та авторизації: перевірку облікових даних адміністраторів та користувачів закритих сесій, а також реєстрацію учасників відкритих конференцій. |
| <b>Модуль адміністрування</b>                       | Реалізує управлінські функції: менеджмент облікових записів (додавання/видалення), конфігурування віртуальних залів та оперативну модерацію активних сесій.                       |
| <b>Модуль локальної конференції (Single-Server)</b> | Відповідає за обробку подій у межах одного сервера: прийом публікацій від учасників та дистрибуцію медіапотоків усім підключеним клієнтам.                                        |
| <b>Модуль ієрархічної конференції</b>               | Керує розподіленою взаємодією: забезпечує багаторівневу реплікацію повідомлень між серверами ієрархії та глобальну синхронізацію стану сесії.                                     |

### 3.3.1. Рівень керування

Класи контролера виступають як посередники, що перехоплюють системні події, ініційовані користувачем, та трансформують їх у виклики методів моделей. Деталізацію компонентів цього рівня представлено в таблиці 3.2.

Таблиця 3.2 - Специфікація класів рівня контролера

| Назва класу       | Функціональна роль                                                                                         |
|-------------------|------------------------------------------------------------------------------------------------------------|
| frontController   | Централізована точка входу для запитів на доступ; здійснює валідацію облікових даних.                      |
| adminController   | Оркестратор адміністративних дій: обробляє запити на алокацію ресурсів залів та керування правами доступу. |
| SingleConfRoom    | Обробник логіки взаємодії в ізольованому серверному середовищі.                                            |
| HierarchyConfRoom | Контролер розподіленої взаємодії; реалізує алгоритми міжсерверної реплікації даних.                        |

### 3.3.2. Рівень даних та бізнес-логіки

Рівень моделі містить сутності проблемної області, що описують стан системи та правила обробки даних (таблиця 3.3). Важливою особливістю є використання успадкування: базовий клас ConferenceRoom детермінує

загальні властивості, які спеціалізуються у класах відкритих (OCR) та закритих (CCR) залів.

Таблиця 3.3 - Опис об'єктів рівня моделі

| Назва класу                         | Опис сутності                                                              |
|-------------------------------------|----------------------------------------------------------------------------|
| Admin, User                         | Профілі користувачів із відповідними атрибутами ідентифікації.             |
| ClosedRoomConferee, ClosedRoomGuest | Спеціалізовані ролі учасників закритих сесій із різними рівнями привілеїв. |
| OpenRoomConferee                    | Об'єкт ідентифікації учасника публічної сесії.                             |
| ConferenceRoom (OCR/CCR)            | Абстракція віртуального простору для комунікації.                          |
| ConferenceMessage                   | Агрегатор контенту (текст, метадані), що генерується в реальному часі.     |
| RootServer, LocalServer             | Технічні компоненти, що описують топологію розподіленої мережі.            |

### 3.3.3. Рівень представлення

Класи представлення відповідають за візуалізацію даних та забезпечення інтерактивного досвіду користувача. Структуру UI-компонентів, адаптованих під специфічні ролі та сценарії, наведено в таблиці 3.4.

Таблиця 3.4 - Компоненти інтерфейсу користувача

| Назва класу                                 | Призначення інтерфейсу                                                                |
|---------------------------------------------|---------------------------------------------------------------------------------------|
| adminLoginUI, closedUserLoginUI, registerUI | Форми входу та реєстрації для різних категорій суб'єктів.                             |
| AdminSelectUI, adminUserUI, adminRoomUI     | Панелі керування системними ресурсами та обліковими записами.                         |
| moderateRoomUI                              | Інструментарій модератора для керування привілеями учасників <i>ex post</i> .         |
| OCR_SelectUI, CCR_SelectUI                  | Навігаційні екрани вибору доступних конференц-залів.                                  |
| OCR_UI, CCR_UI                              | Робочі середовища конференції, що інтегрують засоби публікації та перегляду контенту. |

Цілісну структуру взаємозв'язків між описаними компонентами, включаючи відношення асоціації, агрегації та успадкування, відображено на

діаграмі класів (рисунок 3.7).

Представлена діаграма класів відображає статичну архітектуру системи онлайн-конференцій, побудовану за архітектурним шаблоном Model-View-Controller (MVC). Вона детермінує склад програмних сутностей, їхні атрибути та типи ієрархічних взаємозв'язків, забезпечуючи сегрегацію обов'язків у межах розподіленої системи.

#### 1. Рівень моделі

Базовий клас `ConferenceRoom` через механізм успадкування спеціалізується у відкриті (`OpenConferenceRoom`) та закриті (`ClosedConferenceRoom`) конференц-зали. Абстракція `User` виступає суперкласом для ролей `Admin`, `ClosedRoomConferee`, `ClosedRoomGuest` та `OpenRoomConferee`, що дозволяє реалізувати уніфіковану обробку суб'єктів із різними рівнями доступу. Інфраструктурні компоненти включає класи `RootServer` та `LocalServer` для опису топології мережі розгалуження, а також агрегацію `ConferenceMessage` для представлення контенту сесії.

#### 2. Рівень контролера

Рівень представлений набором спеціалізованих обробників. `frontController` виступає центральною точкою входу для автентифікації, `adminController` інкапсулює логіку управління ресурсами, а `SingleConfRoom` та `HierarchyConfRoom` відповідають за диспетчеризацію повідомлень у локальному та ієрархічному режимах відповідно.

#### 3. Рівень представлення

Охоплює сукупність екранних форм (UI), що забезпечують візуалізацію станів системи та збір вхідних даних від користувачів залежно від їхньої ролі та функціонального контексту (реєстрація, адміністрування, участь у конференції).

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                         | 51   |

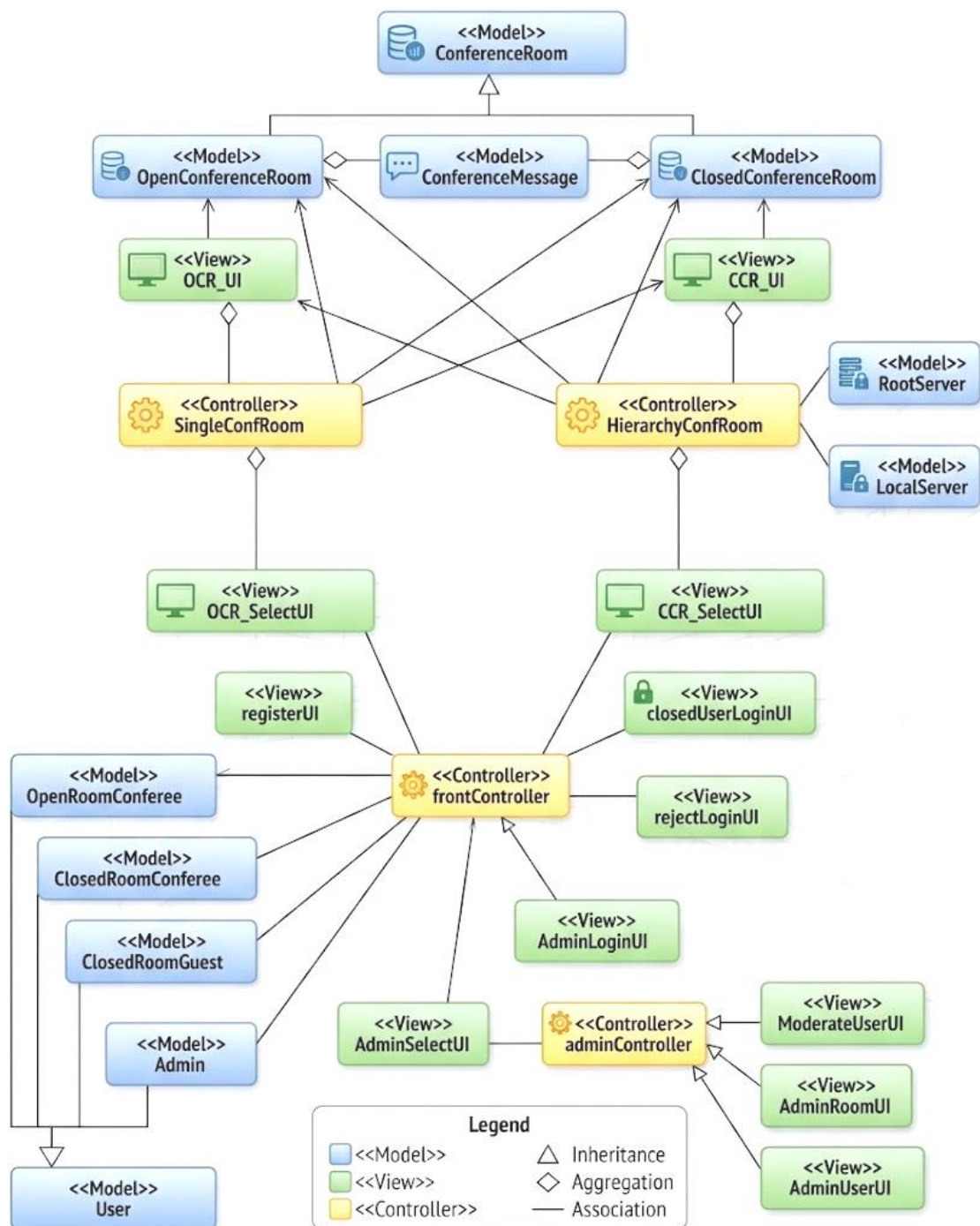


Рисунок 3.7 — Об'єднана діаграма класів архітектури системи онлайн-конференцій

Діаграма демонструє принцип низької зв'язності (low coupling): компоненти взаємодіють через чітко визначені асоціації та інтерфейси. Використання ієрархій успадкування та шаблону MVC гарантує масштабованість системи та можливість незалежної модернізації логіки

відображення без втручання в ядро бізнес-процесів. Застосування такої багаторівневої структури дозволяє мінімізувати зв'язність між модулями та підвищити стійкість системи до змін у технологічному стеку або функціональних вимогах.

### 3.4 Висновки по розділу

У третьому розділі розроблено методологічні засади проектування ієрархічної системи онлайн-взаємодії користувачів. Проведено функціональне моделювання сценаріїв взаємодії з урахуванням ролей основних акторів системи, що дозволило деталізувати її поведінку. Здійснено аналіз прецедентів, який забезпечив формалізацію типових сценаріїв використання системи. Виконано об'єктно-орієнтовану декомпозицію системи з виділенням рівнів керування, бізнес-логіки та представлення. Отримані результати дозволили сформувати узгоджену структуру системи, придатну для подальшої реалізації.

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                         | 53   |

## РОЗДІЛ 4. МОДЕЛЮВАННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ ІЄРАРХІЧНОЇ СИСТЕМИ ВЗАЄМОДІЇ КОРИСТУВАЧІВ

### 4.1. Динамічне моделювання процесів взаємодії системи онлайн-конференцій на основі діаграм послідовності

Для деталізації часової послідовності обміну повідомленнями між програмними об'єктами та визначення логіки управління потоками даних у межах розробленої архітектури застосовано інструментарій динамічного моделювання. Використання діаграм послідовності та активності дозволяє формалізувати поведінку системи на етапі проектування, що є критично важливим для забезпечення масштабованості та відмовостійкості розподілених ІТ-рішень.

Діаграми сценаріїв дозволяють візуалізувати часову впорядкованість викликів методів між об'єктами рівнів Model, View та Controller. У межах дослідження розроблено серію діаграм, що описують ключові прецеденти функціонування системи.

#### 4.1.1. Сценарії доступу учасників

Процеси приєднання користувачів до віртуального середовища різняться залежно від типу доступу до сесії. Формалізація приєднання до закритої конференц-зали, що потребує обов'язкової автентифікації, представлена на рисунку 4.1, тоді як алгоритм доступу до відкритої зали відображений на рисунку 4.2.

Представлена діаграма послідовності (рис. 4.1) формалізує динаміку прецеденту входу учасника до закритої конференції, ілюструючи транзакційну взаємодію між компонентами архітектури Model-View-Controller (MVC). Процес детермінує послідовний перехід від збору вхідних даних до валідації і подальшої зміни стану інтерфейсу.

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                         | 54   |

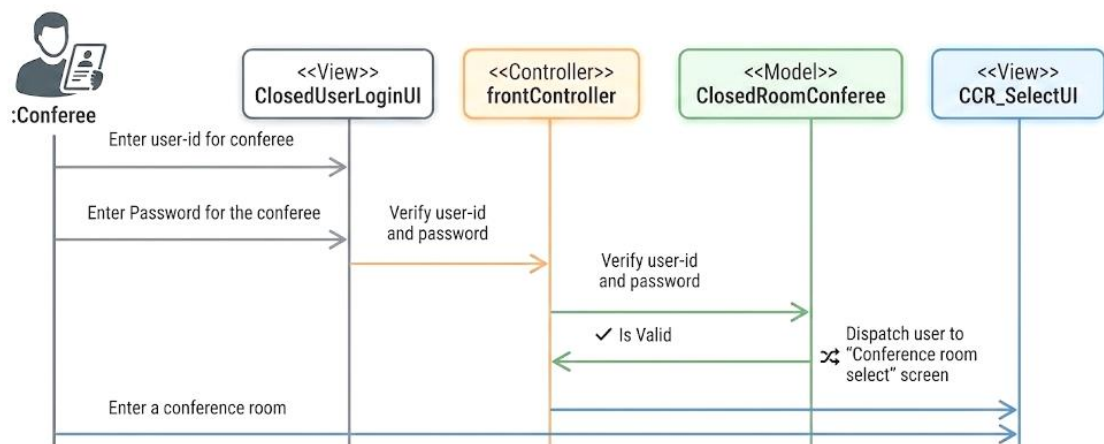


Рисунок 4.1 - Діаграма сценарію учасника, що приєднується до закритої конференц-зали

Алгоритм розпочинається з активності актора :Conferee, який передає ідентифікаційні дані (user-id та password) об'єкту класу ClosedUserLoginUI. На цьому етапі реалізується відокремлення інтерфейсу від бізнес-логіки, де компонент View відповідає виключно за інтеракцію з користувачем та агрегацію вхідних параметрів.

Зібрані облікові дані передаються від рівня представлення до компонента frontController за допомогою повідомлення Verify user-id and password. Контролер виконує роль медіатора, координуючи потік даних між інтерфейсом та ядром системи, не реалізуючи при цьому логіку перевірки самостійно, що забезпечує низьку зв'язність компонентів.

Остаточна перевірка легітимності ідентифікаторів делегується класу ClosedRoomConferee, який представляє рівень моделі. У відповідь на запит контролера модель повертає логічний результат Is Valid. Це підтверджує цілісність даних та дозволяє системі прийняти рішення про надання доступу на основі ролі учасника.

Після успішної валідації frontController ініціює операцію Dispatch user, що переводить систему до нового стану відображення — екрану CCR\_SelectUI. Сценарій завершується безпосереднім вибором конференц-

залу учасником, що демонструє успішний перехід від стадії автентифікації до стадії оперативної взаємодії.

Ця модель забезпечує надійну інкапсуляцію процесів безпеки та дозволяє масштабувати систему шляхом незалежної модернізації будь-якого з рівнів архітектури.

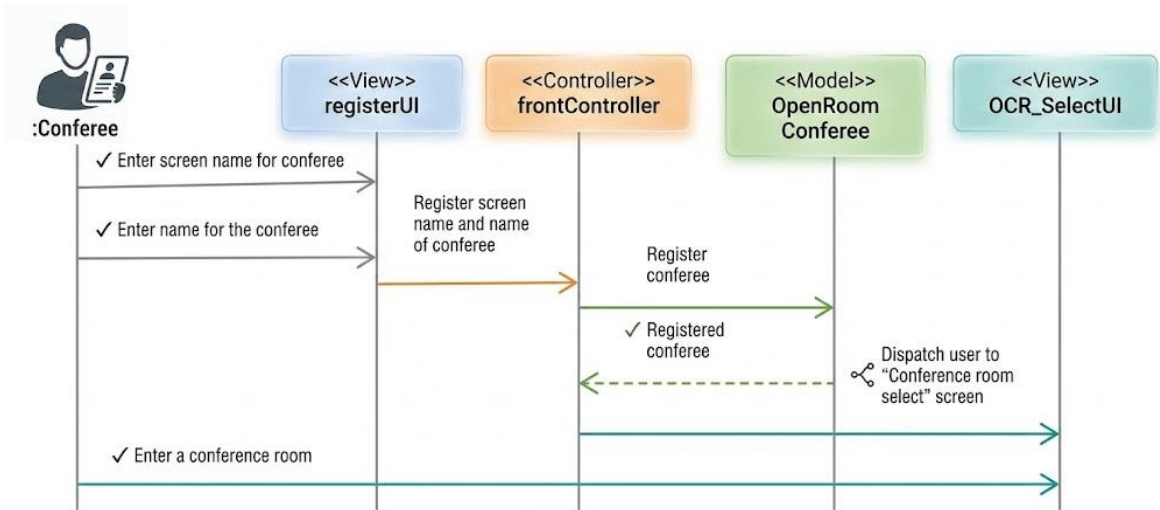


Рисунок 4.2 - Діаграма сценарію учасника, що приєднується до відкритої конференц-зали

Представлена діаграма послідовності (рис. 4.2) формалізує динамічні аспекти взаємодії суб'єктів та програмних компонентів під час реалізації прецеденту приєднання учасника до відкритої онлайн-конференції. Опис сценарію базується на архітектурному шаблоні MVC, що забезпечує чітку детермінацію функцій рівнів представлення, керування та моделі.

Процес активується актором :Conferee (Учасник), який взаємодіє з інтерфейсним об'єктом registerUI. Сценарій передбачає введення учасником первинних ідентифікаційних атрибутів: екранного імені (screen name) та власного імені (name). Дана взаємодія ініціює формування запиту на реєстрацію, що передається від шару представлення до системного логічного шару.

Об'єкт класу представлення делегує обробку зібраних даних компоненту frontController. Контролер виконує функцію медіатора, перехоплюючи запит на реєстрацію та координуючи подальші дії між шарами ієрархії. Це дозволяє ізолювати інтерфейс користувача від логіки обробки даних, забезпечуючи стабільність архітектури при зміні технічних параметрів реєстрації.

Для персистенції та валідації ідентифікаторів учасника frontController звертається до об'єкта моделі OpenRoomConferee через виклик методу Register conferee. На цьому рівні реалізується логіка створення облікового запису в межах поточної сесії онлайн-конференції. Після успішного завершення операції модель повертає підтвердження (Registered conferee) до контролера, що свідчить про зміну стану системи та легітимність статусу учасника.

Отримавши підтвердження від моделі, frontController ініціює операцію диспатчеризації (Dispatch user), спрямовану на об'єкт OCR\_SelectUI. Це призводить до перенаправлення учасника на екран вибору доступних відкритих конференц-залів.

Заключною дією сценарію є безпосередній вибір віртуальної аудиторії учасником через активований інтерфейс OCR\_SelectUI.

Аналіз діаграми підтверджує дотримання принципів низької зв'язності:

- Інтерфейс реєстрації (registerUI) не володіє інформацією про структуру даних у базі, а модель (OpenRoomConferee) не впливає на логіку відображення.

- Використання frontController дозволяє уніфікувати процеси реєстрації для різних типів учасників, забезпечуючи гнучкість системи при масштабуванні.

- Послідовний ланцюг повідомлень гарантує, що учасник отримає доступ до вибору залів лише після успішної фіксації його даних у системі.

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
|      |      |          |        |      |                         | 57   |
| Змн. | Арк. | № докум. | Підпис | Дата |                         |      |

#### 4.1.2. Сценарії адміністратора

Управління конфігурацією системи онлайн-конференцій потребує суворого розмежування повноважень. Процедура ініціалізації сесії адміністратора показана на рисунку 4.3. Сценарії оперативного управління ресурсами та суб'єктами, що включають додавання або видалення учасників та регулювання параметрів закритих конференц-залів, деталізовані на рисунках 4.4 та 4.5 відповідно. Процес модерації активної конференції, спрямований на контроль за дотриманням регламенту, візуалізовано на рисунку 4.6.

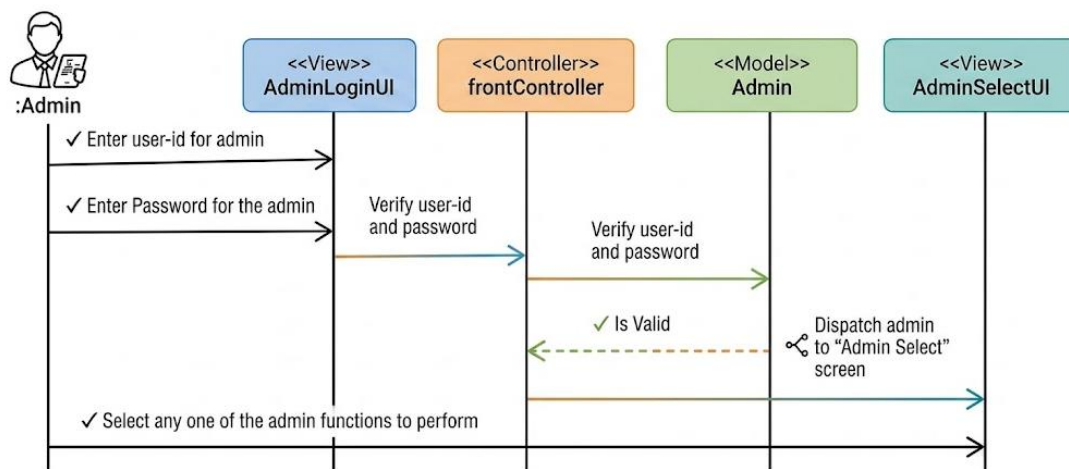


Рисунок 4.3 - Діаграма сценарію входу адміністратора для адміністрування конференції

Представлена діаграма послідовності (рис. 4.3) формалізує динаміку прецеденту автентифікації та входу системного адміністратора до керуючого модуля системи онлайн-конференцій. Побудова сценарію відповідає принципам архітектурного шаблону MVC, забезпечуючи суворе розмежування функцій між інтерфейсом користувача, логікою обробки запитів та рівнем даних.

Процес ініціюється актором :Admin, який взаємодіє з об'єктом класу AdminLoginUI. Сценарій передбачає введення ідентифікаційних атрибутів

(ідентифікатора та пароля адміністратора), що є необхідною умовою для запуску механізму перевірки прав доступу в межах моделі безпеки системи.

Компонент інтерфейсу передає агреговані облікові дані об'єкту frontController через повідомлення Verify user-id and password. На цьому етапі контролер виконує роль координатора, який перехоплює вхідний потік і спрямовує його до відповідного бізнес-об'єкта для валідації, реалізуючи принцип інкапсуляції логіки доступу.

Для безпосередньої верифікації легітимності ідентифікаторів frontController звертається до об'єкта моделі Admin. Модель виконує процедуру зіставлення вхідних параметрів із еталонними даними та повертає підтвердження Is Valid. Такий підхід гарантує цілісність системи, оскільки лише рівень моделі має прямий доступ до механізмів перевірки ідентичності.

Після успішної валідації контролер ініціює зміну стану системи, виконуючи операцію Dispatch admin до об'єкта AdminSelectUI. Це призводить до перенаправлення адміністратора на спеціалізований екран вибору функцій адміністрування. Фінальним кроком сценарію є вибір адміністратором конкретної операції (наприклад, управління користувачами або конференц-залами) через активований інтерфейс керування.

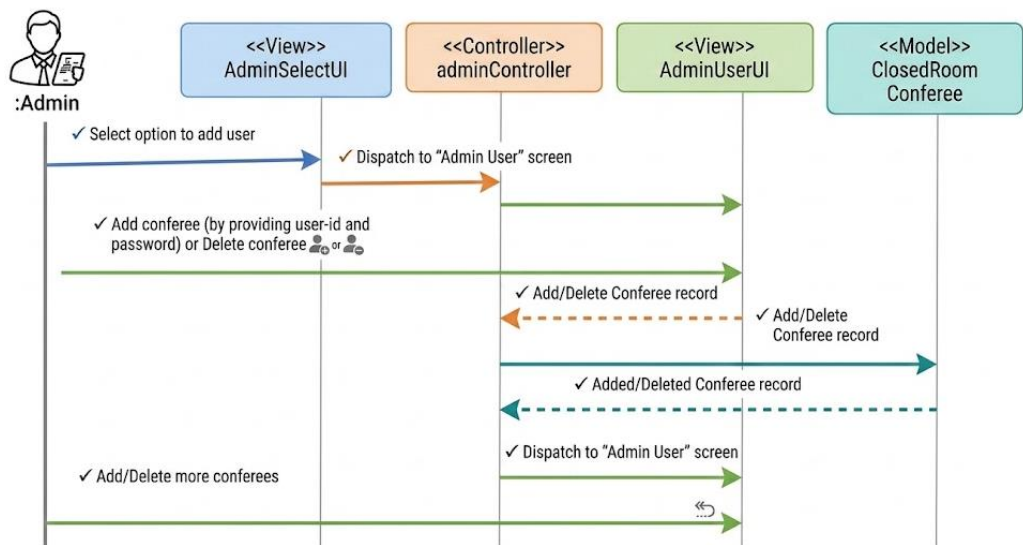


Рисунок 4.4 - Діаграма сценарію додавання/видалення адміністратором учасника конференції

Представлена діаграма послідовності (рис. 4.4) формалізує динамічний аспект прецеденту управління обліковими записами в системі онлайн-конференцій. Вона детермінує взаємодію між адміністратором та структурними рівнями архітектури MVC, забезпечуючи реалізацію функцій додавання та видалення учасників закритих конференцій.

Процес розпочинається з активності актора :Admin, який взаємодіє з об'єктом інтерфейсу AdminSelectUI, вибираючи опцію додавання нового користувача. Ця дія ініціює запит на диспатчеризацію до об'єкта adminController. Контролер, виконуючи роль координатора потоків, активує спеціалізований інтерфейс управління — AdminUserUI, забезпечуючи зміну контексту взаємодії на рівні представлення.

У межах активованого інтерфейсу AdminUserUI адміністратор вносить необхідні ідентифікаційні дані (ідентифікатор користувача та пароль) для створення нового запису або вказує існуючий запис для видалення. Рівень представлення агрегує ці дані та передає повідомлення Add/Delete Conferee record до adminController. Таке розмежування гарантує, що інтерфейс не містить логіки обробки даних, а лише слугує засобом введення.

Контролер делегує виконання операції об'єкту моделі ClosedRoomConferee. На цьому етапі реалізується безпосередня модифікація стану системи: додавання нового об'єкта учасника до реєстру або його термінація. Після успішного виконання транзакції модель повертає підтвердження (Added/Deleted Conferee record) до контролера, що гарантує цілісність інформаційного середовища.

Отримавши підтвердження від моделі, adminController оновлює стан представлення, знову активуючи екран AdminUserUI. Це дозволяє адміністратору отримати візуальне підтвердження успішності операції та забезпечує можливість ітераційного виконання подальших дій без необхідності повторної навігації по головному меню системи.

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
|      |      |          |        |      |                         | 60   |
| Змн. | Арк. | № докум. | Підпис | Дата |                         |      |

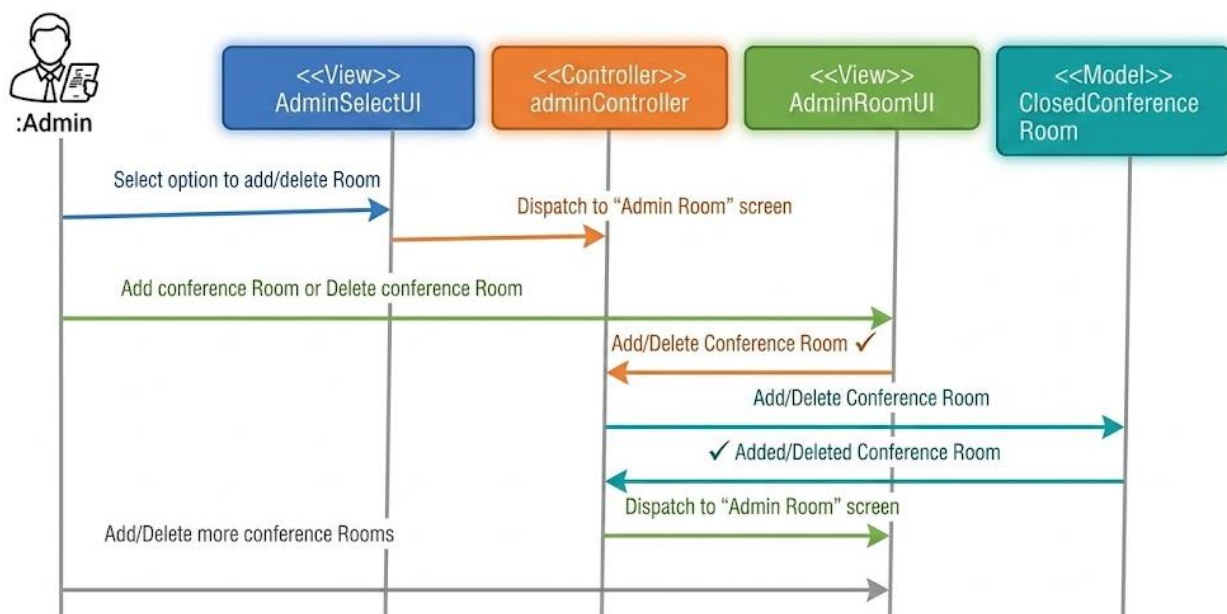


Рисунок 4.5 - Діаграма сценарію додавання/видалення адміністратором закритої конференц-зали

Діаграма послідовності (рис. 4.5) формалізує динамічні аспекти та логіку управління системними ресурсами під час реалізації прецеденту адміністрування конференц-залів. Процес детермінує взаємодію між адміністратором та компонентами архітектури MVC, забезпечуючи транзакційну цілісність операцій створення та видалення об'єктів закритих конференцій.

Алгоритм активується актором :Admin, який через інтерфейс головного меню AdminSelectUI ініціює запит на управління залами. Відповідно до логіки архітектурного шаблону, об'єкт представлення делегує перехід до adminController. Контролер виконує операцію диспатчеризації, активуючи спеціалізований компонент AdminRoomUI, що забезпечує зміну візуального контексту для виконання цільових дій.

У межах активованого інтерфейсу AdminRoomUI адміністратор виконує операцію додавання або видалення конкретної конференц-зали. Запит Add/Delete Conference Room надходить до adminController, який виступає посередником (Mediator) між інтерфейсом користувача та ядром

системи. Це гарантує, що рівень представлення залишається ізольованим від механізмів маніпуляції даними.

Контролер транслює операційний запит до об'єкта моделі ClosedConferenceRoom. На цьому етапі реалізується бізнес-логіка: створення нового екземпляра зали з відповідними атрибутами або термінація існуючого об'єкта в базі даних. Після успішного завершення операції модель повертає підтвердження Added/Deleted Conference Room до контролера, що підтверджує консистентність інформаційного середовища.

Отримавши статус виконання від моделі, adminController оновлює стан представлення через повторну диспатчеризацію до AdminRoomUI. Це дозволяє адміністратору отримати актуальну інформацію про стан залів та забезпечує можливість ітераційного виконання подальших адміністративних функцій без необхідності повторного проходження всього шляху навігації.

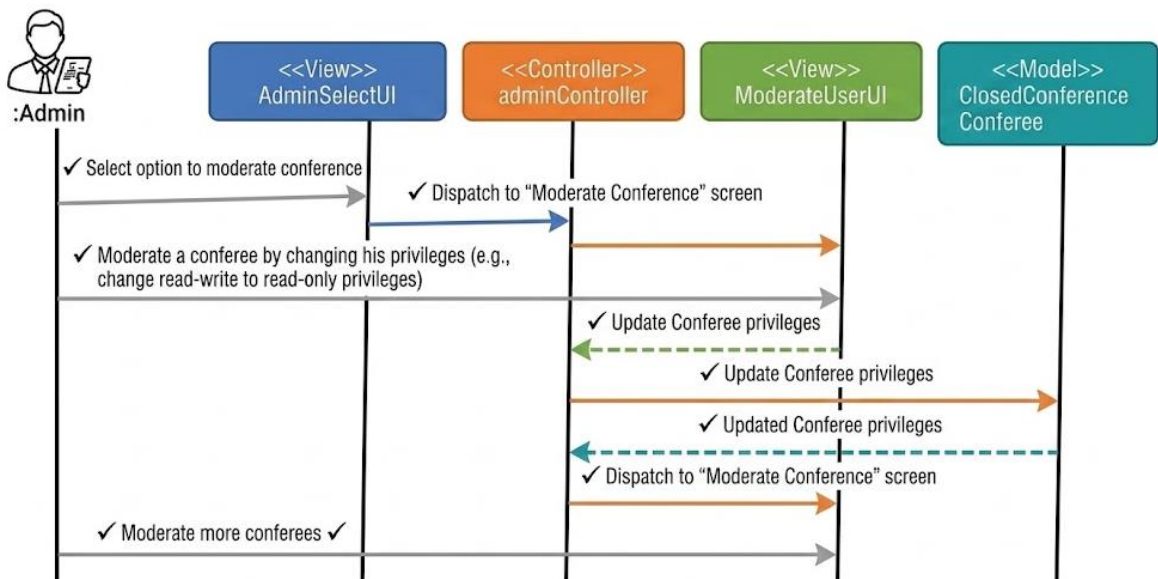


Рисунок 4.6 - Діаграма сценарію модерації адміністратором конференц-зали

Діаграма (рис. 4.6) формалізує динамічні аспекти та логіку управління правами доступу учасників під час реалізації прецеденту модерації конференції адміністратором. Процес детермінує взаємодію між системним

адміністратором та компонентами архітектури MVC, забезпечуючи транзакційну цілісність операцій зі зміни привілеїв користувачів у закритих конференц-залах.

Алгоритм активується актором :Admin, який через інтерфейс вибору функцій AdminSelectUI ініціює запит на модерацію конференції. Відповідно до логіки архітектурного шаблону, об'єкт представлення делегує перехід до adminController. Контролер виконує операцію диспатчеризації, активуючи спеціалізований компонент представлення — ModerateUserUI (екран модерації), що забезпечує перехід до цільового інтерфейсу управління.

У межах активованого інтерфейсу ModerateUserUI адміністратор виконує дії з модерації конкретного учасника, що полягають у модифікації його привілеїв (наприклад, переведення зі статусу з правом запису у статус «тільки для читання»). Запит Update Conferee privileges надходить до adminController, який виступає посередником (Mediator) між інтерфейсом користувача та ядром системи. Це гарантує ізоляцію рівня представлення від механізмів маніпуляції даними.

Контролер транслює операційний запит до об'єкта моделі ClosedRoomConferee. На цьому етапі реалізується безпосередня бізнес-логіка: актуалізація параметрів доступу учасника в реєстрі системи. Після успішної фіксації змін у базі даних модель повертає підтвердження Updated Conferee privileges до контролера, що свідчить про успішне завершення транзакції та оновлення стану системи.

Отримавши статус виконання від моделі, adminController оновлює стан представлення через повторну диспатчеризацію до ModerateUserUI. Це забезпечує адміністратору зворотний зв'язок про успішність операції та дозволяє продовжувати процес модерації інших учасників у поточному сеансі без необхідності повторної навігації по ієрархії меню.

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
|      |      |          |        |      |                         | 63   |
| Змн. | Арк. | № докум. | Підпис | Дата |                         |      |

Сценарій передбачає ітераційний характер роботи адміністратора, що оптимізує часові витрати на управління великою кількістю учасників у розподіленому середовищі.

#### **4.2. Алгоритмізація робочих процесів на основі діаграм активності**

Для опису логіки виконання завдань та розгалуження потоків керування в системі використано діаграми активності. Це дозволяє глибше зрозуміти специфіку обробки повідомлень у різних інфраструктурних режимах:

- режим одного сервера: Діаграма активності, що описує взаємодію учасника в умовах локальної архітектури, представлена на рисунку 4.7. У цьому режимі фокус зосереджено на лінійній обробці та миттєвому розповсюдженні даних у межах одного вузла.

Представлена діаграма активності (рис. 4.7) формалізує алгоритм взаємодії учасника з системою у режимі одного сервера, детермінуючи послідовність операцій від ініціалізації сесії до її термінації.

Нижче наведено стислий опис ключових етапів робочого процесу:

##### **1. Автентифікація та ініціалізація сеансу**

Процес розпочинається з вибору типу конференції на вітальній сторінці системи. Сценарій передбачає проходження процедури автентифікації через інтерфейс `ClosedUserLoginUI`, де виконується перевірка ідентифікатора та пароля. Система реалізує механізм обробки винятків: у разі невалідності даних актор перенаправляється на екран `rejectLoginUI` з наступним поверненням до форми входу.

##### **2. Операційна стадія**

Після успішної авторизації учасник здійснює вибір віртуальної аудиторії через компонент `CCR_SelectUI`, орієнтуючись на рекомендації адміністратора щодо вибору сервера.

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                         | 64   |

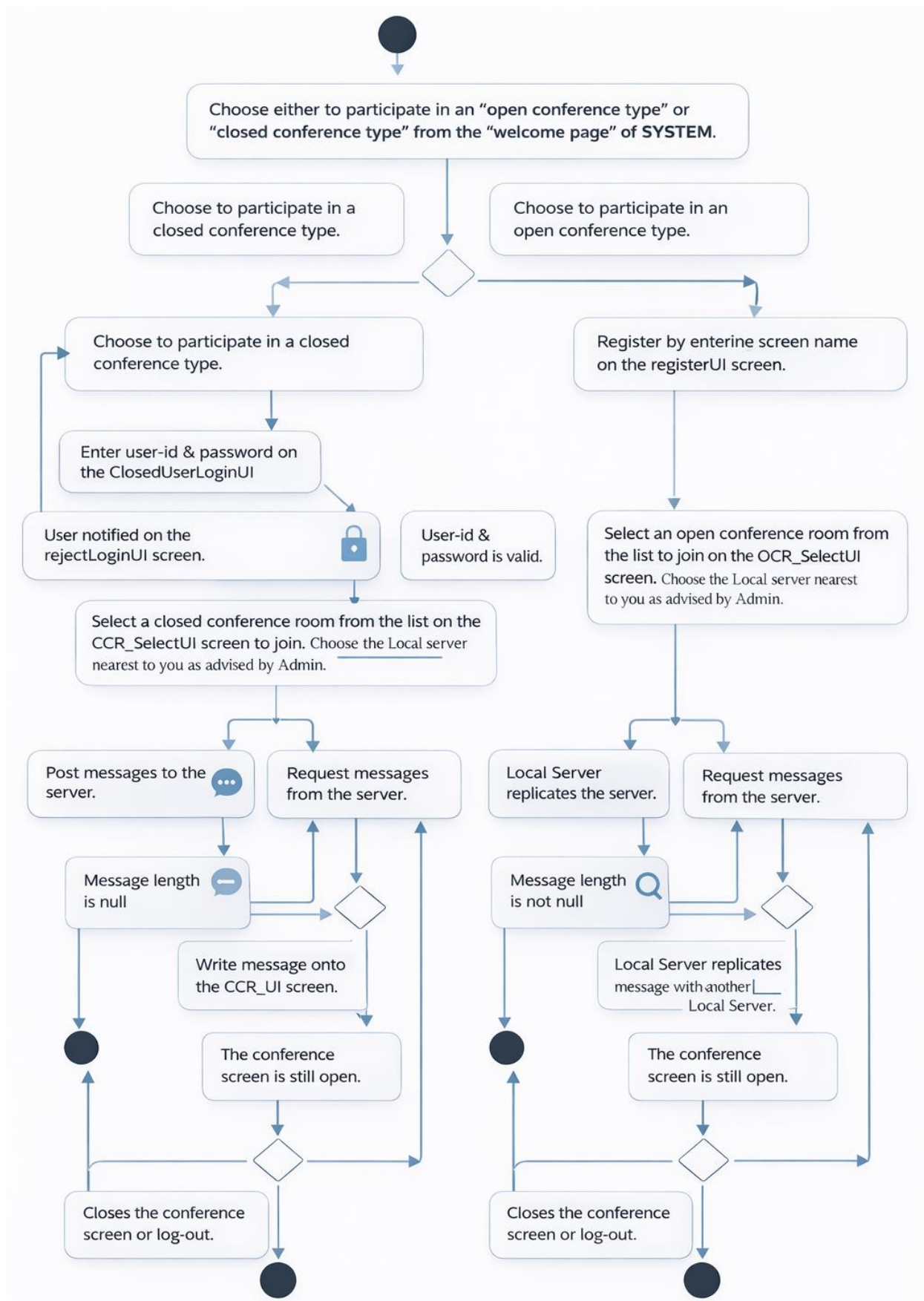


Рисунок 4.7 - Діаграма активності користувача, що бере участь у конференції в режимі одного сервера

Стадія активної участі базується на паралельному виконанні двох потоків керування:

- Генерація контенту: публікація повідомлень на локальний сервер із попередньою валідацією довжини тексту.

- Отримання даних: постійний запит оновлень із сервера для синхронізації стану конференції.

Результати взаємодії візуалізуються безпосередньо в інтерфейсі користувача CCR\_UI.

### 3. Термінація активності

Логіка завершення сесії базується на циклічній перевірці стану вікна конференції. У разі ініціації закриття екрана або виконання команди log-out, система виконує фіналізацію процесів і переходить до кінцевого стану алгоритму.

Дана модель демонструє високу ступінь детермінованості процесів, що забезпечує стабільне функціонування онлайн-конференції в межах локальної інфраструктури.

- Ієрархічний режим. Складніший алгоритм функціонування, притаманний розподіленим системам із багаторівневою структурою серверів розгалуження, показано на рисунку 4.8. Тут ключову роль відіграють процеси реплікації даних між ієрархічними рівнями та забезпечення консистентності інформаційного стану всіх вузлів системи онлайн-конференцій.

Представлена діаграма активності (рис. 4.8) формалізує логіку функціонування системи SYSTEM у розподіленому ієрархічному середовищі, детермінуючи процеси багатопотокової обробки даних та міжсерверної синхронізації.

Нижче наведено опис ключових етапів цього алгоритму:

#### 1. Ініціалізація та ідентифікація в розподіленій мережі

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                         | 66   |

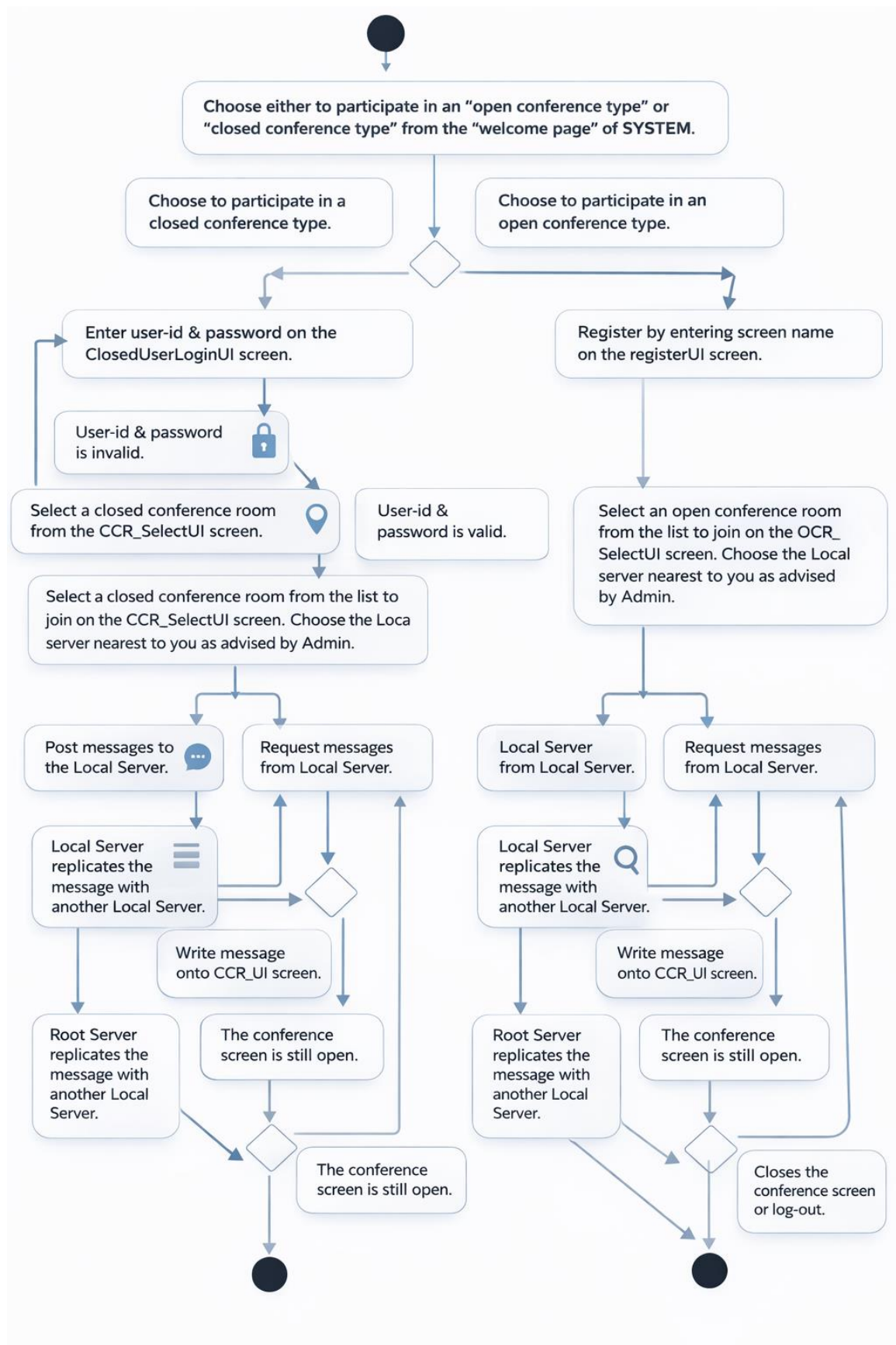


Рисунок 4.8 - Діаграма активності користувача, що бере участь у конференції в ієрархічному режимі

Процес розпочинається з вибору типу конференції на головній сторінці, після чого учасник проходить процедуру реєстрації або автентифікації залежно від обраного прецеденту. Особливістю ієрархічного режиму є вибір локального сервера (Local Server), територіально найближчого до вузла учасника, що оптимізує мережеві затримки при передачі медіаконтенту.

## 2. Механізм ієрархічної реплікації даних

На відміну від односерверного режиму, повідомлення, опубліковане учасником на локальний сервер, проходить багаторівневий цикл реплікації. Локальний сервер реплікує дані на кореневий сервер (Root Server), який, у свою чергу, розповсюджує інформаційний об'єкт серед інших локальних серверів системи. Це забезпечує консистентність стану конференції в межах всієї глобальної ієрархії.

## 3. Операційний цикл та валідація станів

Система реалізує паралельну обробку запитів на отримання оновлень. Візуалізація контенту на інтерфейсі (OCR\_UI або CCR\_UI) відбувається лише після успішної перевірки цілісності та наявності даних (умова Message length is not null).

## 4. Завершення сесії та деалокція ресурсів

Алгоритм передбачає циклічний моніторинг стану активного вікна конференції. Процес завершується автоматично при ініціації виходу (log-out) або закритті графічного інтерфейсу, що супроводжується коректною термінацією сесійних з'єднань у розподіленій структурі.

Дана модель демонструє архітектурну логіку, спрямовану на підтримання цілісності інформаційного простору в багатовузлових системах онлайн-конференцій.

Застосований підхід до моделювання дозволяє не лише верифікувати архітектурні рішення, а й слугує базою для програмної імплементації методів

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                         | 68   |

реплікації та балансування навантаження, що є актуальним для сучасних систем управління конференціями.

4.3. Реалізація модуля адміністрування ієрархічної системи онлайн-конференцій

Представлений графічний інтерфейс користувача (рис. 4.9) є програмною реалізацією модуля адміністрування системи онлайн-конференцій, побудованої на основі ієрархічної топології. Дизайн інтерфейсу відображає архітектурну логіку, закладену в об’єктно-орієнтованій моделі системи, де ключовими суб’єктами управління виступають адміністратори, локальні сервери та конференц-зали.

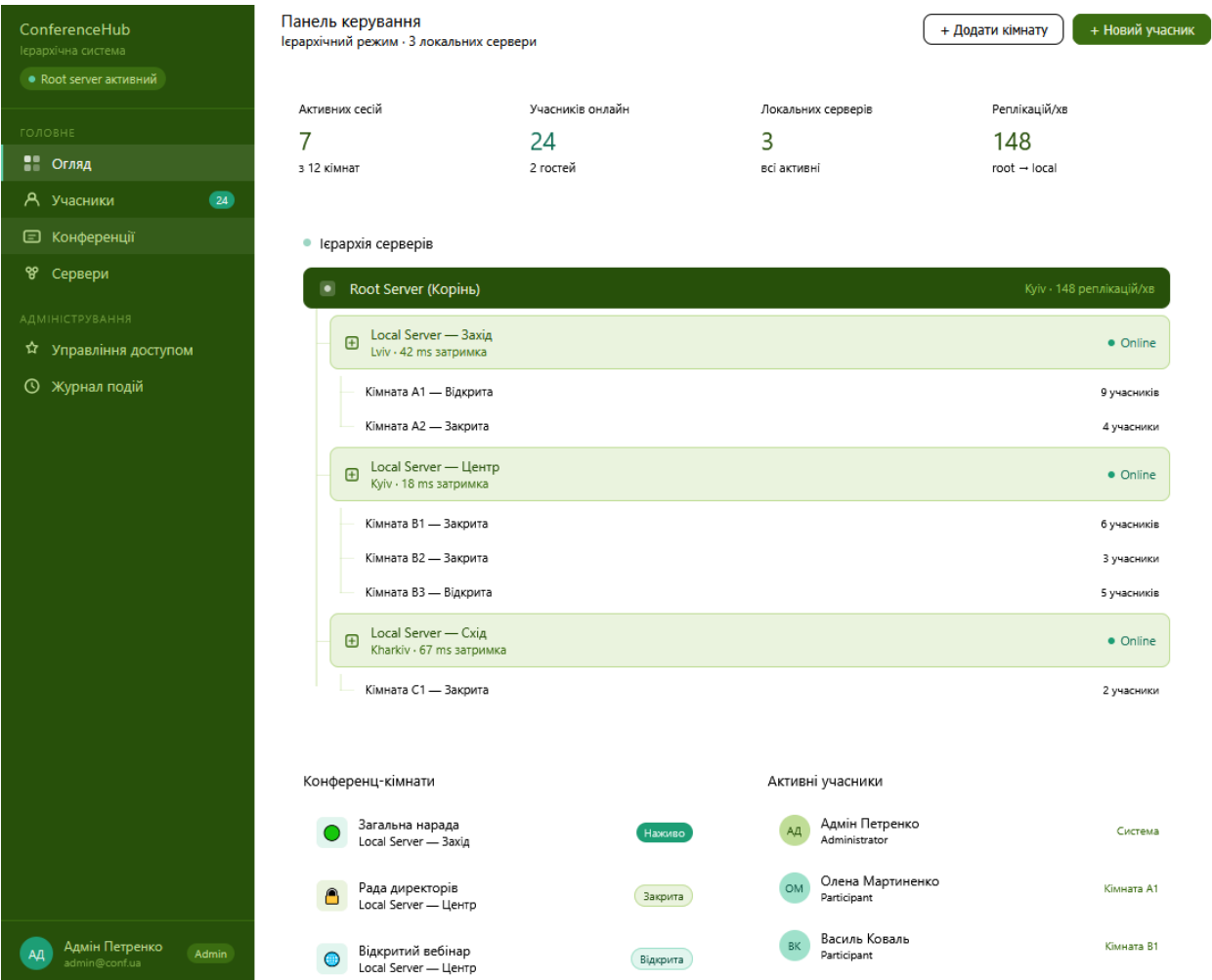


Рисунок 4.9 – Інтерфейс модуля адміністрування

Виконаємо опис функціональних блоків даного інтерфейсу:

### 1. Моніторинг системних метрик та KPI

Верхня частина робочої області містить панель індикаторів стану, що дозволяють здійснювати оперативний контроль продуктивності системи в реальному часі. До складу ключових метрик входять:

- Завантаженість ресурсів: відображення кількості активних сесій відносно загальної місткості залів (7 із 12) та поточна чисельність учасників онлайн.

- Інфраструктурний стан: статус локальних серверів (3 одиниці) та інтенсивність процесів реплікації даних (148 реплікацій/хв) від кореневого вузла (Root Server) до периферійних.

### 2. Візуалізація ієрархічної топології серверів

Центральним елементом інтерфейсу є динамічне дерево ієрархії серверів, що відповідає класам Root Server та Local Server в аналітичній моделі.

Вузлова структура: інтерфейс демонструє підпорядкованість трьох локальних серверів («Захід», «Центр», «Схід») кореновому серверу в Києві.

Параметри зв'язку: для кожного вузла відображається мережева затримка, що дозволяє адміністратору оцінювати якість з'єднання в різних географічних кластерах.

Декомпозиція залів: під кожним локальним сервером відображається перелік закріплених за ним відкритих та закритих залів із вказанням кількості активних учасників у кожній.

### 3. Навігаційна та операційна структура

Ліва бічна панель реалізує логіку управління, розподілену за функціональними доменами, що корелює з рівнями представлення (View) в архітектурі системи:

- Блок «Головне» включає модулі для перегляду учасників, конференцій та серверів.

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
|      |      |          |        |      |                         | 70   |
| Змн. | Арк. | № докум. | Підпис | Дата |                         |      |

- Блок «Адміністрування»: містить інструментарій для управління доступом та ведення журналу подій.

- Профіль адміністратора: нижня частина панелі ідентифікує поточного користувача з роллю Admin, що відповідає моделі автентифікації закритих систем.

#### 4. Списки активних сутностей

Нижня частина інтерфейсу містить деталізовані реєстри поточних процесів:

- Конференц-кімнати: перелік активних залів із зазначенням типу доступу (закрита, відкрита) та прив'язки до локальних серверів.

- Активні учасники: список авторизованих суб'єктів із вказанням їхніх ролей (Administrator, Participant) та конкретних локацій (кімнат) їхнього перебування.

Даний інтерфейс демонструє ефективну інтеграцію засобів візуалізації складних ієрархічних структур, забезпечуючи адміністратору повний контроль над станом розподіленої системи онлайн-конференцій.

#### 4.4. Інтерфейс модуля управління сесіями в межах ієрархічної системи

Представлений графічний інтерфейс (рис. 4.10) є візуалізацією модуля управління сесіями в межах ієрархічної системи ConferenceHub. Цей екран реалізує функції моніторингу та оперативного керування поточними процесами, що безпосередньо корелює з архітектурними сутностями рівня представлення (View), описаними в об'єктній моделі системи.

Наведемо опис структурних компонентів інтерфейсу:

##### 1. Навігаційна та статусна інфраструктура

- Глобальний статус системи: у верхній частині лівої панелі відображається індикатор «Root server активний», що підтверджує

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                         | 71   |

працездатність кореневого вузла, необхідного для координації ієрархічної реплікації даних.

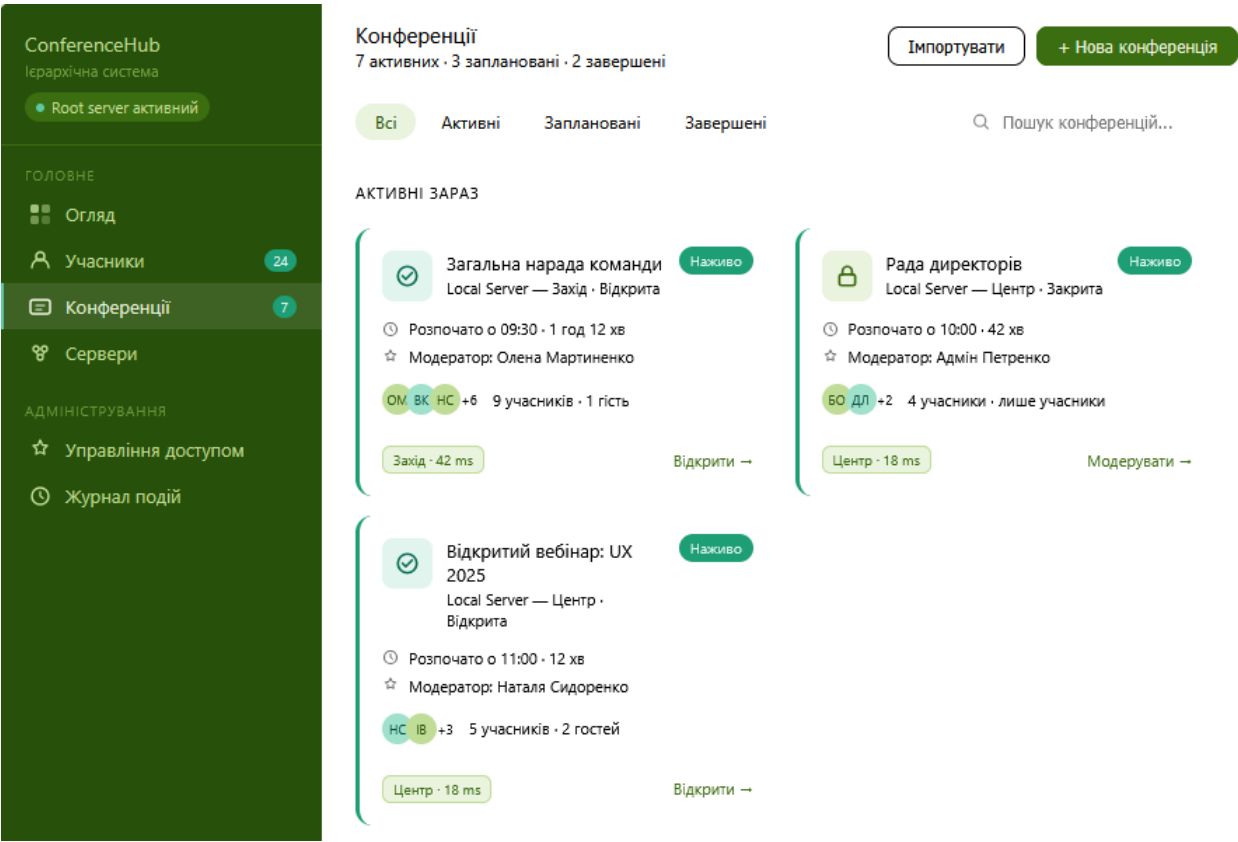


Рисунок 4.10 – Модуль управління сесіями (активні)

2. Панель аналітики та фільтрації

Статистичні метрики: Верхня секція головного вікна надає агреговані дані про стан системи: 7 активних, 3 заплановані та 2 завершені конференції.

Інструменти управління потоками: Реалізовано механізми фільтрації за станом сесій (Всі, Активні тощо) та пошуковий інтерфейс для швидкої ідентифікації конкретних подій у розподіленому середовищі.

3. Об'єктна візуалізація конференц-залів

Центральна область містить картки активних сесій, які є програмною імплементацією класів OpenConferenceRoom та ClosedConferenceRoom:

Типологія залів:

- Відкриті сесії (наприклад, «Загальна нарада команди») позначені іконкою перевірки та доступні для широкого кола учасників.

- Закриті сесії (наприклад, «Рада директорів») мають візуальний індикатор замка, що вказує на обмеження доступу та необхідність автентифікації через клас ClosedRoomConferee.

- Технічні параметри зв'язку: Кожна картка відображає прив'язку до конкретного Local Server (Захід, Центр) та показник мережевої затримки (42 ms, 18 ms), що критично для моніторингу якості зв'язку в ієрархічному режимі.

Вказано модераторів сесій та кількісний склад учасників і гостей, що дозволяє адміністратору оцінювати завантаженість серверів у реальному часі.

#### 4. Операційні виклики

Створення ресурсів: Кнопка «+ Нова конференція» ініціює процедуру алокації нових об'єктів у системі.

Управління доступом: Для закритих залів передбачена функція «Модерувати», яка дозволяє адміністратору оперативно втручатися в процеси управління правами користувачів.

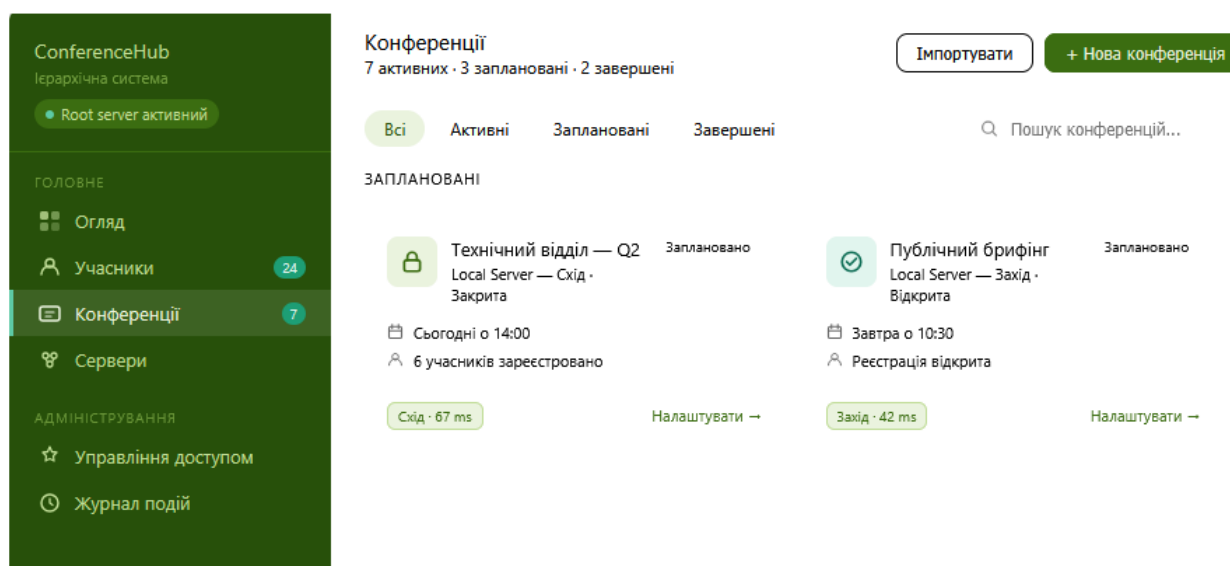


Рисунок 4.11 – Модуль управління сесіями (заплановані)

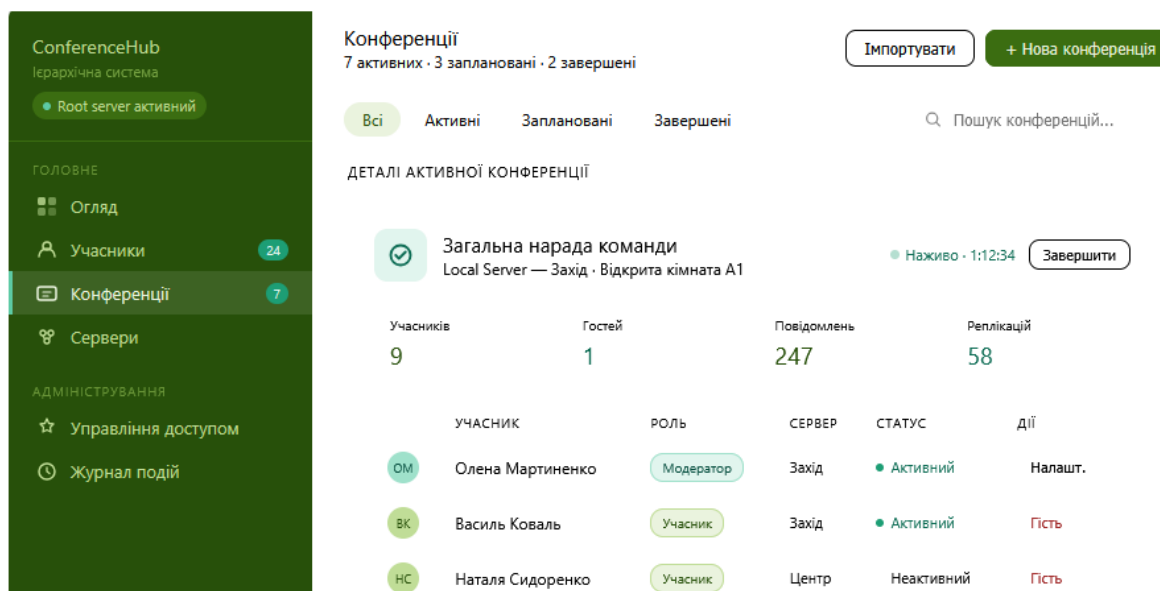


Рисунок 4.12 – Модуль управління сесіями (деталі вибраної конференції)

Цей інтерфейс демонструє високий рівень ергономічності, поєднуючи складні технічні метрики ієрархічної системи з інтуїтивно зрозумілими елементами управління, що забезпечує стабільну експлуатацію платформи онлайн-конференцій.

#### 4.5 Висновки по розділу

У четвертому розділі здійснено моделювання динамічних процесів взаємодії в системі за допомогою UML-діаграм. Побудовано діаграми послідовності та активності, що відображають ключові сценарії роботи користувачів і адміністратора. На основі отриманих моделей виконано алгоритмізацію основних робочих процесів системи. Реалізовано програмні модулі, зокрема модуль адміністрування та модуль управління сесіями, що забезпечують функціональну завершеність системи. Розроблені інтерфейсні рішення підтверджують практичну придатність запропонованої архітектури.

## ВИСНОВКИ

В бакалаврській роботі було проведено дослідження теоретичних, методологічних та практичних аспектів побудови сучасних систем онлайн-взаємодії, а також реалізовано програмне рішення, що враховує вимоги масштабованості, гнучкості та ефективного управління користувацькими сесіями.

У першому розділі здійснено ґрунтовний аналіз предметної області, зокрема розглянуто архітектурні принципи побудови веб-орієнтованих систем онлайн-конференцій та визначено основні сфери їх застосування. Встановлено, що сучасні системи розподіленої взаємодії користувачів характеризуються високими вимогами до надійності, доступності та масштабованості. Проведено класифікацію існуючих систем онлайн-конференцій за функціональними та архітектурними ознаками, що дозволило виділити ключові підходи до їх реалізації, серед яких домінують клієнт-серверна, мікросервісна та подієво-орієнтована архітектури. Особливу увагу приділено аналізу ієрархічних систем, які забезпечують структуроване управління користувачами та ресурсами, що є критично важливим для організації складних багаторівневих взаємодій.

У другому розділі сформовано архітектурну модель ієрархічної системи онлайн-конференцій, що базується на поєднанні сучасних програмних технологій та принципів модульності. Обґрунтовано вибір програмного базису, який забезпечує ефективну реалізацію серверної та клієнтської частин системи. Розроблено детальний опис робочих процесів і функціональних сценаріїв управління сесіями, що дозволяє забезпечити контроль доступу, керування ролями користувачів та оптимізацію взаємодії в реальному часі. Впровадження архітектурного шаблону Model-View-Controller забезпечило чітке розмежування рівнів представлення, бізнес-

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
|      |      |          |        |      |                         | 75   |
| Змн. | Арк. | № докум. | Підпис | Дата |                         |      |

логіки та доступу до даних, що сприяє підвищенню підтримуваності, розширюваності та повторного використання компонентів системи.

У третьому розділі розроблено методологію проектування ієрархічної системи, що базується на принципах системного аналізу та об'єктно-орієнтованого підходу. Проведено функціональне моделювання сценаріїв взаємодії користувачів із системою із використанням прецедентного підходу, що дозволило чітко визначити ролі основних акторів, зокрема адміністратора та учасників різних типів конференцій. Здійснено об'єктно-орієнтовану декомпозицію системи, у результаті якої виділено три основні рівні архітектури: рівень керування, рівень бізнес-логіки та даних, а також рівень представлення.

У четвертому розділі виконано моделювання динамічної поведінки системи за допомогою діаграм послідовності та активності, що дозволило формалізувати процеси взаємодії користувачів і системних компонентів. На основі отриманих моделей здійснено алгоритмізацію ключових робочих процесів, включаючи сценарії доступу учасників та адміністративного управління. Реалізовано програмні модулі системи, зокрема модуль адміністрування та модуль управління сесіями, що забезпечують функціональну повноту системи. Розроблений інтерфейс користувача відповідає сучасним вимогам зручності та інтуїтивності, що сприяє підвищенню ефективності взаємодії користувачів.

Узагальнюючи результати виконаної роботи, слід зазначити, що поставлена мета досягнута, а всі визначені завдання успішно вирішені. Запропонована ієрархічна система розподіленої взаємодії користувачів демонструє високий рівень адаптивності до змінних умов експлуатації, забезпечує ефективне управління користувацькими ролями та сесіями, а також може бути використана як основа для подальшого розвитку та інтеграції з іншими інформаційними системами.

|      |      |          |        |      |                         |      |
|------|------|----------|--------|------|-------------------------|------|
|      |      |          |        |      | БР.ІП – 78.00.00.000 ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                         | 76   |

## ПЕРЕЛІК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Mahmoud H., Abozariba R. A systematic review on WebRTC for potential applications and challenges beyond audio video streaming // Multimedia Tools and Applications. – Cham: Springer, 2025. – Vol. 84, No. 6. – P. 2909–2946.
2. Abukari A. M., Bankas E. K., Iddrisu M. M. A Secured Video Conferencing System Architecture Using a Hybrid of Two Homomorphic Encryption Schemes: A Case of Zoom // International Journal of Engineering Research & Technology. – Noida: IJERT, 2020. – Vol. 9, Issue 8. – P. 123–130.
3. Joskowicz J. Video Conferencing Technologies: Past, Present and Future // TechRxiv Preprint. – IEEE, 2023. – P. 1–15.
4. Wu M., Shen X., Zhang L., Liu X., Zhang H. A heuristic design toolkit for web conferencing software based on self-determination theory // Interacting with Computers. – Oxford: Oxford University Press, 2023. – Vol. 35, No. 4. – P. 455–470.
5. Zhang H., Wang J., Li Z., Li J. Design and Implementation of Immersive Audio and Video Communication Systems Based on Virtual Reality // Electronics. – Basel: MDPI, 2023. – Vol. 12, No. 5. – P. 1134.
6. Singh Y., Shishodia K. Revolutionizing Communication: Building a Full Stack Web-Based Video Conferencing App // International Journal for Multidisciplinary Research. – 2024. – Vol. 6, Issue 3. – P. 45–52.
7. Chan M. K. Y., Burke D. Review of Empirical Studies on Video Conferencing Platforms // Proceedings of the European Conference on e-Learning (ECEL 2025). – UK: Academic Conferences International, 2025. – P. 35–42.
8. Forbes P., West D., Hall T. Virtual conference design: features and obstacles // Multimedia Tools and Applications. – Springer, 2022. – Vol. 81. – P. 16901–16919.

|      |      |          |        |      |                           |      |
|------|------|----------|--------|------|---------------------------|------|
|      |      |          |        |      | БР.ІІІ – 78.00.00.000 ІІЗ | Арк. |
|      |      |          |        |      |                           | 77   |
| Змн. | Арк. | № докум. | Підпис | Дата |                           |      |

9. Seeling P. Web Conferencing Traffic Analysis using DimDim // IEEE ICC. – IEEE, 2010. – P. 1–6.
10. Hove D. P., Watson B. The shortcomings of video conferencing technology and emerging XR solutions // arXiv. – 2025. – P. 1–18.
11. Fielding R. Architectural Styles and the Design of Network-based Software Architectures // Doctoral Dissertation. – University of California, Irvine, 2000. – 162 p.
12. Gamma E., Helm R., Johnson R., Vlissides J. Design Patterns: Elements of Reusable Object-Oriented Software // Addison-Wesley. – Boston, 1995. – 395 p.
13. Bass L., Clements P., Kazman R. Software Architecture in Practice // Addison-Wesley. – Boston, 2013. – 624 p.
14. Sommerville I. Software Engineering // Pearson. – London, 2016. – 816 p.
15. Pressman R., Maxim B. Software Engineering: A Practitioner's Approach // McGraw-Hill. – New York, 2020. – 970 p.
16. Tanenbaum A., Van Steen M. Distributed Systems: Principles and Paradigms // Pearson. – New Jersey, 2017. – 686 p.
17. Coulouris G., Dollimore J., Kindberg T., Blair G. Distributed Systems: Concepts and Design // Addison-Wesley. – 2012. – 1024 p.
18. Newman S. Building Microservices // O'Reilly Media. – 2021. – 617 p.
19. Richardson C. Microservices Patterns // Manning Publications. – 2019. – 520 p.
20. Pautasso C., Zimmermann O., Leymann F. RESTful Web Services vs. Big Web Services // WWW Conference. – ACM, 2008. – P. 805–814.
21. Berners-Lee T., Fielding R., Masinter L. Uniform Resource Identifier (URI): Generic Syntax // IETF RFC 3986. – 2005.

|      |      |          |        |      |                           |      |
|------|------|----------|--------|------|---------------------------|------|
|      |      |          |        |      | БР.ІІІ – 78.00.00.000 ІІЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                           | 78   |

22. Rescorla E. The Transport Layer Security (TLS) Protocol Version 1.3 // IETF RFC 8446. – 2018.
23. Jennings C., Hardie T., Westerlund M. Real-Time Communication with WebRTC // IEEE Communications Magazine. – 2013. – Vol. 51. – P. 20–26.
24. Grigorik I. High Performance Browser Networking // O'Reilly Media. – 2013. – 400 p.
25. Tilkov S., Vinoski S. Node.js: Using JavaScript to Build High-Performance Network Programs // IEEE Internet Computing. – 2010. – Vol. 14. – P. 80–83.
26. Fowler M. Patterns of Enterprise Application Architecture // Addison-Wesley. – 2003. – 533 p.
27. Kruchten P. The 4+1 View Model of Architecture // IEEE Software. – 1995. – Vol. 12. – P. 42–50.
28. Booch G., Rumbaugh J., Jacobson I. The Unified Modeling Language User Guide // Addison-Wesley. – 2005. – 496 p.
29. Eriksson H., Penker M. UML Toolkit // Wiley. – 2004. – 432 p.
30. Schmidt D. Model-Driven Engineering // IEEE Computer. – 2006. – Vol. 39. – P. 25–31.
31. Papazoglou M., van den Heuvel W. Service-Oriented Architectures: Approaches, Technologies and Research Issues // VLDB Journal. – 2007. – Vol. 16. – P. 389–415.
32. Erl T. Service-Oriented Architecture: Concepts, Technology, and Design // Prentice Hall. – 2005. – 792 p.
33. Dean J., Ghemawat S. MapReduce: Simplified Data Processing on Large Clusters // OSDI. – 2004. – P. 137–150.
34. Kreps J., Narkhede N., Rao J. Kafka: A Distributed Messaging System for Log Processing // NetDB. – 2011. – P. 1–7.

|      |      |          |        |      |                           |      |
|------|------|----------|--------|------|---------------------------|------|
|      |      |          |        |      | БР.ІІІ – 78.00.00.000 ІІЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                           | 79   |

## **ДОДАТКИ**

## Додаток А

А.1. Компонент картки конференції (ConferenceCard) - реалізує візуалізацію відкритої або закритої конференції

```
import React from 'react';
import { Lock, CheckCircle, Users, Signal } from 'lucide-react';

const ConferenceCard = ({ room }) => {
  const isClosed = room.type === 'closed';

  return (
    <div className="bg-white p-6 rounded-xl border border-gray-100 shadow-sm hover:shadow-md">
      <div className="flex justify-between items-start mb-4">
        <div className="flex items-center gap-3">
          <div className={`p-2 rounded-lg ${isClosed ? 'bg-yellow-50 text-yellow-500' : 'bg-white text-gray-900'}`}>
            {isClosed ? <Lock size={20} /> : <CheckCircle size={20} />}
          </div>
          <div>
            <h3 className="font-semibold text-gray-900">{room.title}</h3>
            <p className="text-sm text-gray-500">{room.server} – {isClosed ? 'Закрито' : 'Відкрито'}</p>
          </div>
          <span className="px-3 py-1 bg-teal-500 text-white text-xs font-medium rounded">
            {room.status}
          </span>
        </div>
        <div className="space-y-3 mb-6">
          <div className="flex items-center text-sm text-gray-600 gap-2">
            <span className="text-gray-400"><Clock /></span> Розпочато о {room.startTime}
          </div>
          <div className="flex items-center text-sm text-gray-600 gap-2">
            <span className="text-gray-400"><Star /></span> Модератор: {room.moderator}
          </div>
        </div>
        <div className="flex justify-between items-center pt-4 border-t border-gray-100">
          <div className="flex items-center gap-2 text-sm text-gray-500">
            <Users size={16} />
            <span>{room.participantsCount} учасників</span>
          </div>
          <div className="flex items-center gap-1 text-xs font-medium text-green-600">
            <Signal size={12} />
            {room.latency} ms
          </div>
        </div>
        <button className="w-full mt-4 py-2 text-sm font-medium text-teal-600 border rounded">
          {isClosed ? 'Модерувати' : 'Відкрити'} →
        </button>
      </div>
    </div>
  );
};

export default ConferenceCard;
```

## 2. Об'єктна структура даних (Model Layer)

Для наповнення інтерфейсу використовується структура даних, що відповідає класам моделей `OpenConferenceRoom` та `ClosedConferenceRoom`.

```
const activeConferences = [  
  {  
    id: "room-a1",  
    title: "Загальна нарада команди",  
    type: "open",  
    server: "Local Server – Захід",  
    startTime: "09:30",  
    moderator: "Олена Мартиненко",  
    participantsCount: 9,  
    latency: 42  
  },  
  {  
    id: "room-b2",  
    title: "Рада директорів",  
    type: "closed",  
    server: "Local Server – Центр",  
    startTime: "10:00",  
    moderator: "Адмін Петренко",  
    participantsCount: 4,  
    latency: 18  
  }  
];
```

### A.3. Компонент ієрархії серверів (ServerHierarchy)

Фрагмент коду для візуалізації ієрархічного режиму, де Root Server координує Local Servers.

```
const ServerNode = ({ name, location, latency, status }) => (  
  <div className="flex items-center justify-between p-3 bg-gray-50 rounded-lg ml-  
    <div>  
      <p className="text-sm font-medium text-gray-800">{name} – {location}</p>  
      <p className="text-xs text-gray-500">{latency} ms затримка</p>  
    </div>  
    <div className="flex items-center gap-2 text-xs text-green-600 font-bold">  
      <span className="w-2 h-2 bg-green-500 rounded-full animate-pulse"></span>  
      {status}</div>  
  </div>  
  
const HierarchySection = () => (  
  <div className="bg-white rounded-xl shadow-sm border border-gray-100 overflow-l  
    <div className="bg-green-900 text-white p-4 flex justify-between items-center  
      <span className="font-semibold text-sm">Root Server (Корінь) – Київ</span>  
      <span className="text-xs opacity-80">148 реплікацій/хв</span>  
    </div>  
    <div className="p-4">  
      <ServerNode name="Local Server" location="Захід" latency={42} status="Onlin  
      <ServerNode name="Local Server" location="Центр" latency={18} status="Onlin  
      <ServerNode name="Local Server" location="Схід" latency={67} status="Online  
    </div>  
  </div>  

```

## **БІБЛІОГРАФІЧНА ДОВІДКА**

**Тема бакалаврської роботи:** “Веб-базована ієрархічна система розподіленої взаємодії користувачів”

Обсяг пояснювальної записки: 79 аркушів.

Дата закінчення роботи: 9 червня 2026 р.

Підпис студента \_\_\_\_\_