ЗМІСТ

[1 РОЛІ КОРИСТУВАЧІВ В ІНФОРМАЦІЙНІЙ СИСТЕМІ 2](#_Toc470450044)

[2 USE CASE UML ДІАГРАМИ 5](#_Toc470450045)

[3 ДІАГРАМИ ПОСЛІДОВНОСТЕЙ 6](#_Toc470450046)

[4 IDEF3 11](#_Toc470450047)

[5 МОДЕЛІ ЖИТТЄВИХ ЦИКЛІВ РОЗРОБКИ ОСНОВНИХ ПРОЦЕСІВ СИСТЕМИ 14](#_Toc470450048)

[6 DFD 15](#_Toc470450049)

[7 ERD 16](#_Toc470450050)

[8 АРХІТЕКТУРИ СИСТЕМИ ДЛЯ ОСНОВНИХ ПРОЦЕСІВ ІС 17](#_Toc470450051)

# 1 РОЛІ КОРИСТУВАЧІВ В ІНФОРМАЦІЙНІЙ СИСТЕМІ

Усіх користувачів системи поділено на три категорії згідно з можливостями, які вони можуть виконувати, а також правами доступу до даних системи:

незареєстровані користувачі;

зареєстровані користувачі;

адміністратори даних.

Кожна з вищенаведених категорій має власну область видимості та може виконувати лише дії та , що передбачені їй системою.

Розподіл користувачів на підгрупи дозволяє системі працювати коректніше та надає змогу покращувати її функціонування.

# 2 USE CASE UML ДІАГРАМИ

Розроблені при виконанні даного етапу курсової роботи Use Case UML діаграми складаються з наступних елементів:

а) актор;

б) процес;

в) перехід.

Актор – це об’єкт ІС, що виконує первні процеси і поділяється на:

- master (адміністратори);

- slave (решта).

Для кожна з вищенаведених категорій побудовано UML-діаграми, які наглядно демонструють область видимості користувача та дії, передбачені системою, які він може виконувати.

UML-діаграми дозволяють підвищити супроводжуваність проекту і полегшують розробку документації..

# 3 ДІАГРАМИ ПОСЛІДОВНОСТЕЙ

Діаграма Побудовано Sequence діаграми взаємодій типу незареєстрований користувач-адміністратор та зареєстрований користувач-адміністратор.

Порядок взаємодії між об’єктами діаграми:

а) незареєстрований користувач – адміністратор. Користувач «Simple\_user», за допомогою інтерфейсу вибору дій «Browse interface » в програмі, обирає сторінку «ContactPage» , де знаходиться форма для відправки повідомлення адміністратору «Message interface» та друкує туди текст. Повідомлення надходить до кабінету адміністратора «Admin Cabinet», де користувач «Admin» може прочитати його.

б) зареєстрований користувач – адміністратор. Користувач «Active\_user», за допомогою інтерфейсу вибору дій «Browse interface » в програмі, обирає сторінку «Price list page» , де знаходиться сторінка вибору даних для замовлення «Order intarface». Обравши відповідні пункти замовлення, користувач переходить до «PaymentInterface», де обирає варіант оплати замовлення – за допомогою карти, чи готівкою. Якщо обрано перший варіант, то відбувається взаємодія з об’єктом «Credit Card Authorization» і при підтвердженні авторизації картки дані про оплачене замовлення відправляються до «Admin Cabinet»; якщо другий – дані про замовлення одразу відправляються до «Admin Cabinet». Користувач «Admin» може подивитися замовлення у «Admin Cabinet» та підтвердити замовлення. Підтвердження буде відображене у «Active\_user Cabinet», де користувач «Active\_user» побачить змінений статус свого замовлення.

# 4 IDEF3

Відповідно до обраної теми курсової роботи – «Організація мітингу» побудовано:

а) діаграму ієрархії процесів (основних). Головним процесом є «Meeting organization». Наступні 4 процеси є дочірніми: «View info page», «View price page», «View authorization page» та «View contacts page». Ці підпроцеси процесу «Meeting organization» також мають свої дочірні процеси. Ієрархія усіх процесів даної курсової роботи представлена на рисунку 2.1.

б) діаграми нотації станів всіх об’єктів системи, які беруть участь в основних процесах системи. Виділено наступні 3 об’єкти системи: «користувач» (рис. 3.1), «мітинг» (рис.3.2) та «форма» (рис. 3.3). Для даних об’єктів побудовано OSTN діаграми.

в) діаграми потоків основних процесів. Виділено наступні 4 основні процеси: «авторизація» (рис. 4.1), «перегляд інформації» (рис. 4.2), «контактування» (рис. 4.3) та «замовлення» (рис.4.4). На діаграмах зображено які дії може виконувати користувач при виборі одного із потоків системи.

# 5 МОДЕЛІ ЖИТТЄВИХ ЦИКЛІВ РОЗРОБКИ ОСНОВНИХ ПРОЦЕСІВ СИСТЕМИ

Система розроблюється за спіральною моделлю життєвого циклу. З кожним етапом, аналізуючи умови постановки завдання, виділяється чітка множина задач, що будуть реалізовані в процесі розробки програмного забезпечення.

Процес «Meeting organization» є основним процесом системи та розроблюється за спіральною моделлю життєвого цикла, оскільки його підпроцеси можуть змінюватися та може додаватися новий функціонал в ході розробки програмного продукту . Процеси «View info page», «View contacts page», «View general info», «View information about latest meeting», «View edition products catalog», «Edit price page» розроблюються також за спіральною моделлю життєвого циклу, оскільки можуть бути змінені та покращені в процесі розробки програмного забезпечення. Їх модифікація не призведе до глобальних змін в інформаційній системі.

Процеси «View price page», «View Authorization page», «To fill form», «Edit form», «Send message», «Drop form», «Order organization», «Order edition products», «Sign up», «Sign in», «Create temp user», «Drop user» розроблюються за каскадною моделлю життєвого циклу, адже від самого початку роботи програмного забезпечення повинні бути досконалими та працюючими. Від них залежить якість роботи програмного продукту.

Процес «Pay for meeting organization» розроблюється за каскадною з поверненням моделлю життєвого циклу. Вона забезпечує цьому процесу цілісність та повноту, бо від правильності його функціонування залежить не тільки робота програмного забезпечення, а й гроші користувачів.

# 6 DFD

У роботі описано бізнес-процес «Організація мітингу». За допомогою програмного забезпечення Power Designer було побудовано DFD діаграму, процесу «Організація мітингу».

DFD діаграма містить процеси, данні, потоки та сутності, розміщені на 3-х рівнях. Вони представлені та описані за допомогою DFD0, DFD1 та DFD2. Розглянемо далі детальніше опис, деталізацію та представлення процесів і даних у DFD діаграмах. Цей процес зображено за допомогою діаграм деталізії рівнів :

* DFD0 — деталізація 0-го рівня (один процес 1);
* DFD1 – деталізація першого рівня ( 4 підпроцеси 1.1, 1.2, 1.3, 1.4 для процесу 1);
* DFD2 – деталізація другого рівня ( підпроцеси 1.1.1, 1.1.2, 1.1.3, 1.2.1, 1.2.2, 1.3.1, 1.3.2, 1.3.3, 1.4.1, 1.4.2, 1.4.3, 1.4.4, 1.5.1, 1.5.2, 1.5.3, 1.5.4) для визначених 4 підпроцесів 1.1, 1.2, 1.3, 1.4 для процесу 1).

# 7 ERD

Розроблена ER-модель для п’яти сутностей відповідно до обраної теми курсової роботи – «Організація мітингу». Модель зображено на рисунку А.1.

Було обрано основні сутності – зареєстрований користувач системи (Active\_user), авторизований користувач системи (Auth\_user), адміністратор системи (Admin), замовлений мітинг (Meeteing\_ordering), система оплати замовленого мітингу (Payment\_system). Було встановлено зв’язки між сутностями, а також визначено ключі.

Сутності «Active\_user» та «Auth\_user» зв’язані між собою за допомогою зв’язку «Order of the meeting». Сутність «Active\_user» має ключовим атрибутом «user\_number», що передається даним зв’язком до сутності «Auth\_user».

Сутності «Auth\_user» та «Meeteing\_ordering» зв’язані між собою за допомогою зв’язку «User in system». Сутність «Auth\_user» має ключовим атрибутом «user\_status», що передається даним зв’язком до сутності «Meeteing\_ordering», а сутність «Meeteing\_ordering» має ключовим атрибутом «order\_meeteng\_status», що передається даним зв’язком до сутності «Auth\_user».

Сутності «Auth\_user» та «Payment\_system» зв’язані між собою за допомогою зв’язку «Payment for meeting». Сутність «Auth\_user» має ключовим атрибутом «user\_status», що передається даним зв’язком до сутності «Payment for meeting», а сутність «Payment for meeting» має ключовим атрибутом «payment\_status», що передається даним зв’язком до сутності «Auth\_user».

# 8 АРХІТЕКТУРИ СИСТЕМИ ДЛЯ ОСНОВНИХ ПРОЦЕСІВ ІС

В рамках Use case діаграма додатку складається із чотирьох основних блоків (Basic authorization, View info page, View contact page, View order page), що містять у собі інші блоки, які є доступними в залежності від ролі користувача в системі.

Basic authorization – процес авторизації користувачів в системі.

Для даного процесу доцільно використовувати архітектуру клієнт-серверної системи 1 – «Розподілене представлення даних». Доцільність використання:

a) на сторінці клієнта має відбуватися валідація введених даних, при цьому не має враховуватись можливість чи неможливість отримання результату операції;

b) сервер приймає рішення про валідність авторизації користувача;

c) реалізується високий рівень захисту системи, для захисту від взлому СУБД;

d) паролі шифруються під час передачі від клієнтської до серверної частини .

View info page – процес перегляду сторінок з інформацією про мітинги, включаючи коментарі та фотогалерею.

Для даного процесу доцільно використовувати архітектуру клієнт-серверної системи 4 – «Віддалений доступ до бази даних». Доцільність використання:

a) клієнт може додавати нові коментарі, відповідно на його сторінці має існувати даний функціонал;

b) клієнт, що зайшов в систему у ролі адміністратора системи може замінювати дані та редагувати інформацію, яка потім буде збережена на сервері.

View contact page – перегляд сторінки із контактами і відправки повідомлення.

Для даного процесу доцільно використовувати архітектуру клієнт-серверної системи 3 – «Розподілений застосунок». Доцільність використання:

a) на сторінці клієнта має відбуватися валідація введених даних, при цьому має враховуватись можливість чи неможливість отримання результату операції (заповнення форми), якщо форма повідомлення заповнена невірно (наприклад присутні порожні поля), функціонал клієнтської частини одразу виведе що є невірним та дасть змогу виправити помилку;

b) користувач, що авторизувався у системі у якості адміністратора може бачити всі дані (повідомлення) на клієнтській частині.

View order page – перегляд сторінки із цінами, додатковими товарами та замовлення.

Для даного процесу доцільно використовувати архітектуру клієнт-серверної системи 5 – «Розподілений сервер даних». Доцільність використання:

a) дані клієнта надсилаються до різних серверів, оскільки відбувається взаємодія не тільки із сервером системи, а й і з сервером банку – клієнт надсилає серверу системи сформоване замовлення, а серверу банку – номер карти. Сервер банку повертає серверу системи дані про оплату;

b) у разі знищення одного із серверів клієнтські дані про оплату замовлення будуть зберігатися на іншому, тому оплачене замовлення клієнта не має зникнути.