Лабораторна робота №2

Мета:Ознайомитися з поведінковими діаграмами UML, засвоїти їх призначення та особливості застосування для моделювання динамічних аспектів програмних систем. Набути практичних навичок побудови діаграм випадків використання, діаграм послідовностей, діаграм діяльності та діаграм станів для опису сценаріїв взаємодії користувача з системою та відображення внутрішніх процесів.

Завдання 1. Use Case Diagram (Real-case)
Розробити діаграму прецедентів (Use Case Diagram).

1. Оберіть **свій варіант** об’єкта зі списку (Додаток А).
2. Розробіть **UML Use case Diagram** для додатку.
3. Підготуйте **короткий текстовий опис**:

- перелік actors (з поясненням призначення кожного);
- перелік use cases і їх опис;

4. (Опціонально, +бали) Використайте inclusion, extension, generalization (actor, use case).
1. **Моделюйте зв’язки.**

- association між акторами та прецедентами;
- include для обов’язкових підпроцесів, що повторюються («Перевірити права», «Валідувати дані»);
- extend для варіативних/виняткових сценаріїв («Відновити пароль» як розширення «Увійти»);
- generalization між акторами/прецедентами (напр., «Клієнт» ⟶ «Преміум-клієнт»).

5. **Текстові специфікації.** Для **кожного** прецеденту — короткий опис (1–2 речення). Для **2–3 ключових** — повний опис.
6. **Узгодженість.** Переконайтеся, що кожен прецедент має хоча б одного актора й логічно входить у межі системи; уникайте дублювань.
7. **Оформлення.** Нотація UML 2.5+, читабельні назви, нумерація UC-0, UC-02, …, підписи на зв’язках <<include>>, <<extend>>.

					ЛР.ОК24.ПІ231.13.02		
Змн.	Арк.	№ докум.	Підпис	Дата			
Розроб.		Євчук М.В.			Behavioral UML-діаграми	Літ.	Арк. 1 / Аркушів 24
Перевір.		Жереб Д.В.					
Н. Контр.						ХПК	
Затверд.							

Завдання 2. Statechart Diagram (Lab Rat)
Розробити **діаграму станів (Statechart Diagram)**.

1. Оберіть **свій варіант** об’єкта зі списку (Додаток В).
2. Розробіть **UML Statechart Diagram** життєвого циклу об’єкта.
3. Підготуйте **короткий текстовий опис**:

- перелік станів (з поясненням призначення кожного);
- перелік подій/тригерів і можливих дій на переходах;

4. (Опціонально, +бали) Використайте **entry/exit/do** дії в ключових станах; за потреби — **композитні стани** або **вкладені підстани**.

Завдання 3. Statechart Diagram (Real-case)
Розробити **діаграму станів (Statechart Diagram):**

1. Оберіть **свій варіант** об’єкта зі списку (Додаток А).
2. Розробіть **UML Statechart Diagram** життєвого циклу об’єкта.
3. Підготуйте **короткий текстовий опис**:

- перелік станів (з поясненням призначення кожного);
- перелік подій/тригерів і можливих дій на переходах;

4. (Опціонально, +бали) Використайте **entry/exit/do** дії в ключових станах; за потреби — **композитні стани** або **вкладені підстани**.

Завдання 4. Activity Diagram (Lab Rat)
Розробити **діаграму активності (Activity Diagram):**

1. Оберіть **свій варіант** процесу зі списку (Додаток С).
2. Розробіть **UML Activity Diagram** виконання потоку дій.
3. Підготуйте **короткий текстовий опис**:

- перелік кроків виконання процесу;
- перелік всіх варіантів потоку роботи (подій);

4. (Опціонально, +бали) Використайте **Swimlanes** для учасників потоку дій; за потреби — **decision modes, fork&merge nodes, guards, states, time events, cycles**.

Завдання 5. Activity Diagram (Real-case)
Розробити **діаграму активності (Activity Diagram):**

1. Оберіть один процес відповідно **свому варіанту** зі списку (Додаток А).
2. Розробіть **UML Activity Diagram** виконання потоку дій.
3. Підготуйте **короткий текстовий опис**:

- перелік кроків виконання процесу;
- перелік всіх варіантів потоку роботи (подій);

4. (Опціонально, +бали) Використайте **Swimlanes** для учасників потоку дій; за потреби — **decision modes, fork&merge nodes, guards, states, time events, cycles**.

Завдання 6. Sequence Diagram (Lab Rat)
Розробити **діаграму послідовності (Sequence Diagram)**.

1. Оберіть один процес відповідно **свому варіанту** зі списку (Додаток D).
2. Розробіть **UML Sequence Diagram** послідовності процесу.
3. Підготуйте **короткий текстовий опис flow (послідовності)**:

- перелік учасників flow.
- перелік кроків виконання flow;

4. За потреби покажіть **create/destroy** життєвих об’єктів (інвойс, сесія тощо).

Завдання 7. Sequence Diagram (Real-case)

Розробити **діаграму послідовності (Sequence Diagram):**

1. Оберіть один flow відповідно **свому варіанту** зі списку (Додаток A).
2. Розробіть **UML Sequence Diagram** виконання flow.
3. Підготуйте **короткий текстовий опис flow (послідовності)**:

- перелік учасників flow.
- перелік кроків виконання flow;

Завдання 8. Collaboration Diagram (Lab Rat)

Розробити **діаграму колабарації (Collaboration Diagram):**

1. Оберіть один процес відповідно **свому варіанту** зі списку (Додаток D).
2. Розробіть **UML Collaboration Diagram** співпраці учасників процесу.
3. Підготуйте **короткий текстовий опис** співпраці учасників процесу:

- перелік учасників процесу.
- перелік зв’язків між учасниками та обмін повідомленнями;

4. Для зручності можна використати Sequence Diagram з завдання 6.

Завдання 9. Collaboration Diagram (Real-case)

Розробити **діаграму колобарації (Collaboration Diagram):**

1. Оберіть один процес відповідно **свому варіанту** зі списку (Додаток A).
2. Розробіть **UML Collaboration Diagram** співпраці учасників процесу.
3. Підготуйте **короткий текстовий опис** співпраці учасників процесу:

- перелік учасників процесу.
- перелік зв’язків між учасниками та обмін повідомленнями;

4. Для зручності можна використати Sequence Diagram з завдання 7.

Хід роботи

Завдання №1

Use Case Diagram (Real-case)

Додаток A

40. Онлайн-сервіс для обміну кулінарними рецептами з рейтингами

1. **Межі системи:**
   Що робить: Сервіс дозволяє користувачам переглядати, шукати, додавати рецепти, а також залишати оцінку та коментарі до потрібних рецептів.

Що не робить: Сервіс не може готувати їжу, не може здійснювати доставку продуктів.

2. **Актори:**

Актор	**Тип**	**Опис того, що повинен виконувати актор**
Гість	Первинний	Може переглядати рецепти, шукати їх, але не може додавати коментарі та оцінювати.
Зареєстрований користувач	Первинний	Може додавати рецепти, коментувати їх, також оцінювати, і редагувати свої ж рецепти.
Адміністратор	Первинний	Керує користувачами, видаляє невідповідний контент.
Система рейтингу	Вторинний	Буде обчислювати середній рейтинг того чи іншого рецепту.
Таймер	Вторинний	Буде автоматично перевіряти рецепти на актуальність та видаляти застарілу інформацію
Система аутентифікації	Зовнішня система	Дозволятиме реєстрацію через Google, Facebook, та інші, також забезпечує перевірку користувачів під час входу.

3. **Перелік прецедентів:**

№	**Назва**	**Хто виконує**
1	Переглянути рецепти	Користувач
2	Шукати рецепт	Користувач
3	Зареєструватися	Користувач
4	Увійти	Користувач
5	Відновити пароль	Користувач
6	Додати рецепт	Користувач
7	Редагувати/видалити рецепт	Користувач
8	Оцінити рецепт	Користувач
9	Коментувати рецепт	Користувач
10	Модерувати рецепти	Адміністратор
11	Блокувати користувача	Адміністратор
12	Розрахувати середню оцінку	Система рейтингу
13	Видалити застарілі рецепти	Таймер

4. **Моделювання зв’язків**

Association

- **Користувач ⇒** “Переглянути рецепти”, “Шукати рецепт”, “Зареєструватися”, “Увійти”, “Відновити пароль”, “Додати рецепт”, “Редагувати/видалити рецепт”
- **Адміністратор ⇒** “Модерувати рецепти”, “Блокувати користувача”
- **Система рейтингу ⇒** “Розрахувати середню оцінку”
- **Таймер ⇒** “Видалити застарілі рецепти”
- **Система аутентифікації ⇒** через <<include>> у “Увійти”

Include

- “Увійти” <<include>> “Перевірити облікові дані”
- “Оцінити рецепт” <<include>> “Розрахувати середню оцінку”

Extend

- “Відновити пароль” <<extend>> “Увійти”
- “Редагувати/видалити рецепт” <<extend>> “Додати рецепт”

Generalization

- Користувач ⇒ Гість та зареєстрований користувач

5. **Текстові специфікації:**

Переглянути рецепти

Користувач переглядає список доступних рецептів із бази даних.

Шукати рецепт

Користувач вводить ключове слово або параметр пошуку, а система відповідно повертає відповідний рецепт.

Зареєструватися

Актор: Користувач

Передумова: Користувач ще не має аккаунта.

Основний сценарій:

- Користувач відкриває форму реєстрації
- Вводить своє ім’я, email, пароль.
- Система робить валідацію даних.
- Система створює аккаунт та підтверджує реєстрацію.

Винятки:

- Якщо email вже використовується, то система видає повідомлення про це.
- Якщо введені дані некоректні, користувачу пропонується ввести їх знову.

Результат:

Аккаунт створено, користувач відповідно, стає “зареєстрований”.

Увійти

Актор: Користувач.

Передумова: Користувач вже має аккаунт.

Основний сценарій:

- Користувач вводить email та пароль.
- Система перевіряє дані на правильність.
- Якщо дані вірні, користувач отримує доступ до функцій.

Винятки:

- Якщо введений пароль був невірним, система видає повідомлення про помилку.
- Користувач може скористатись прецедентом **“Відновити пароль”**.

Результат:

Користувач увійшов в систему.

Відновити пароль

Якщо користувач забув свій пароль, він має можливість скинути його через email.

Додати рецепт

Актор: Користувач.

Передумова: Користувач аутентифікований у системі.

Основний сценарій:

- Користувач відкриває форму для додавання рецепту.
- Вводить назву рецепту, опис, та інгрідієнти, кроки приготування.
- Система перевіряє правильність введення даних.
- Система зберігає рецепт в базі даних.

Винятки:

Якщо поля заповнені некоректно, пропонується виправлення їх.

Результат:

Новий рецепт збережено та доступний для перегляду іншим користувачам.

Редагувати/видалити рецепт

Автор рецепту може оновити інформацію або видалити власний рецепт із системи.

Оцінити рецепт

Користувач виставляє оцінку рецепту, а система враховує її у загальний рейтинг.

Коментувати рецепт

Користувач залишає коментар до рецепту, який пізніше буде видно іншим.

Модерувати рецепти

Адміністратор переглядає нові або проблемні рецепти та приймає рішення про їх публікацію чи видалення.

Блокувати користувача

Адміністратор може обмежувати доступ користувача до системи у разі порушення ним правил.

Розрахувати середню оцінку

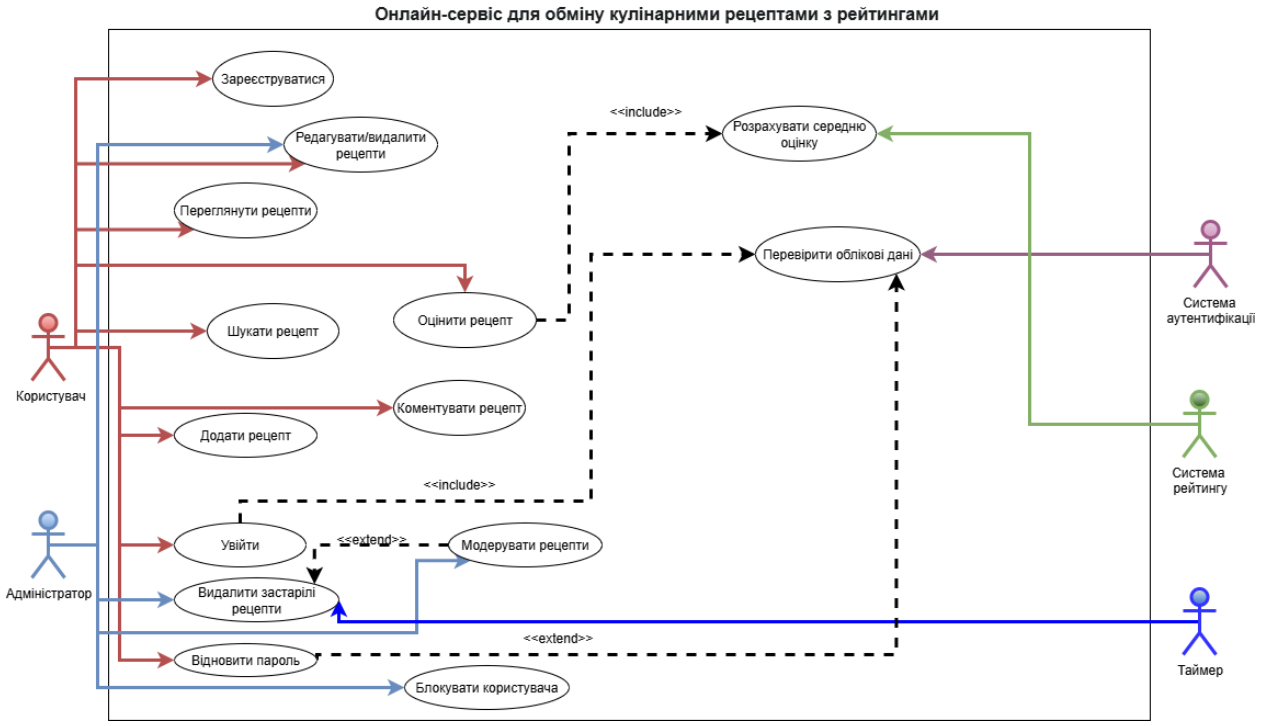
Система автоматично підраховує середню оцінку рецепту на основі голосів іншим користувачів.

Видалити застарілі рецепти

Таймер автоматично видаляє рецепти, які неакутальні та просрочені за визначеними правилами.

6. **Узгодженість**

Проаналізувавши наші дані, можемо сказати що кожен прецедент має хоча б одного актора, логічно входить у межі системи, дублікатів немає.



Завдання №2

Statechart Diagram (Lab rat)

Додаток В: 13. Доставка посилки

1. **Межі системи:**
   Моделюємо життєвий цикл **посилки** в системі доставки. Від моменту оформлення до вручення або повернення відправнику.
2. **Перелік станів:**

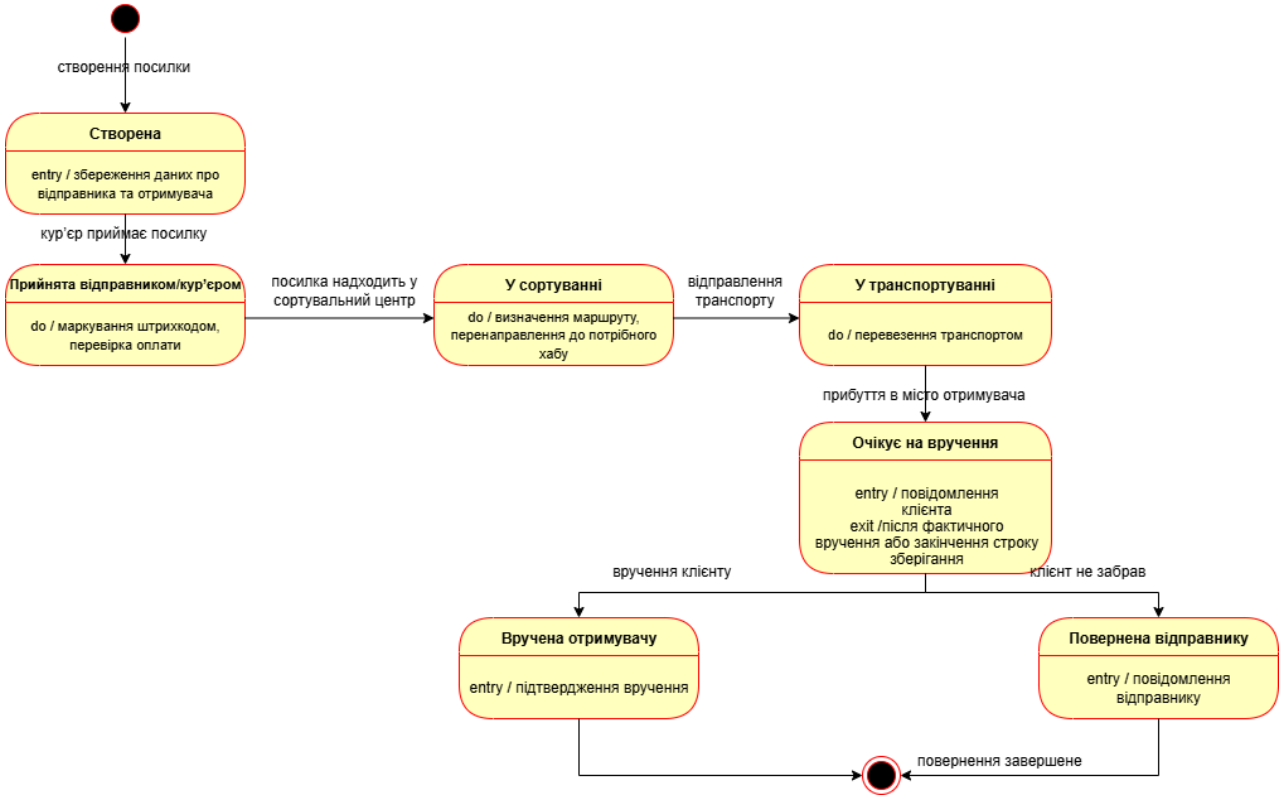
№	Стан	Опис	entry / do / exit дії
1	**Створена**	Посилка зареєстрована в системі відправником, але ще не прийнята кур’єром.	Entry / збереження даних про відправника та отримувача
2	**Прийнята відправником/кур’єром**	Посилка передана службі доставки.	Do / маркування штрихкодом, перевірка оплати
3	**У сортуванні**	Посилка знаходиться на сортувальному центрі.	Do / визначення маршруту, перенаправлення до потрібного хабу
4	**У транспортуванні**	Посилка їде між містами/хабами або доставляється локально кур’єром.	Do / перевезення транспортом
5	**Очікує на вручення**	Посилка прибула до відділення або до кур’єра в місті отримувача.	Entry / повідомлення клієнта Exit / після фактичного вручення або

			закінчення строку зберігання
6	**Вручена отримувачу**	Посилка видана адресату.	Entry / підтвердження вручення
7	**Повернена відправнику**	Якщо отримувач відмовився або не забрав у строк.	Entry / повідомлення відправнику

3. **Події/тригери переходів**
   - створення посилки ⇒ *Створена*
   - кур'єр приймає посилку ⇒ *Прийнята*
   - посилка надходить у сортувальний центр ⇒ *У сортуванні*
   - відправлення транспорту ⇒ *У транспортуванні*
   - прибуття в місто отримувача ⇒ *Очікує вручення*
   - вручення клієнту ⇒ *Вручена*
   - клієнт не забрав ⇒ *Final*
   - повернення завершене ⇒ *Final*

5. **Кількість переходів**

У нас є мінімум **8 переходів**:
Initial → Створена, Створена → Прийнята відправником/кур'єром, Прийнята відправником/кур'єром → У сортуванні, У сортуванні → У транспортуванні, У транспортуванні → Очікує на вручення, Очікує на вручення → Вручена отримувачу, Очікує на вручення → Повернена відправнику, Повернена відправнику → Final, Вручена отримувачу → Final.



Завдання №3

Statechart Diagram (Real-case)
Додаток А:
40. Онлайн-сервіс для обміну кулінарними рецептами з рейтингами

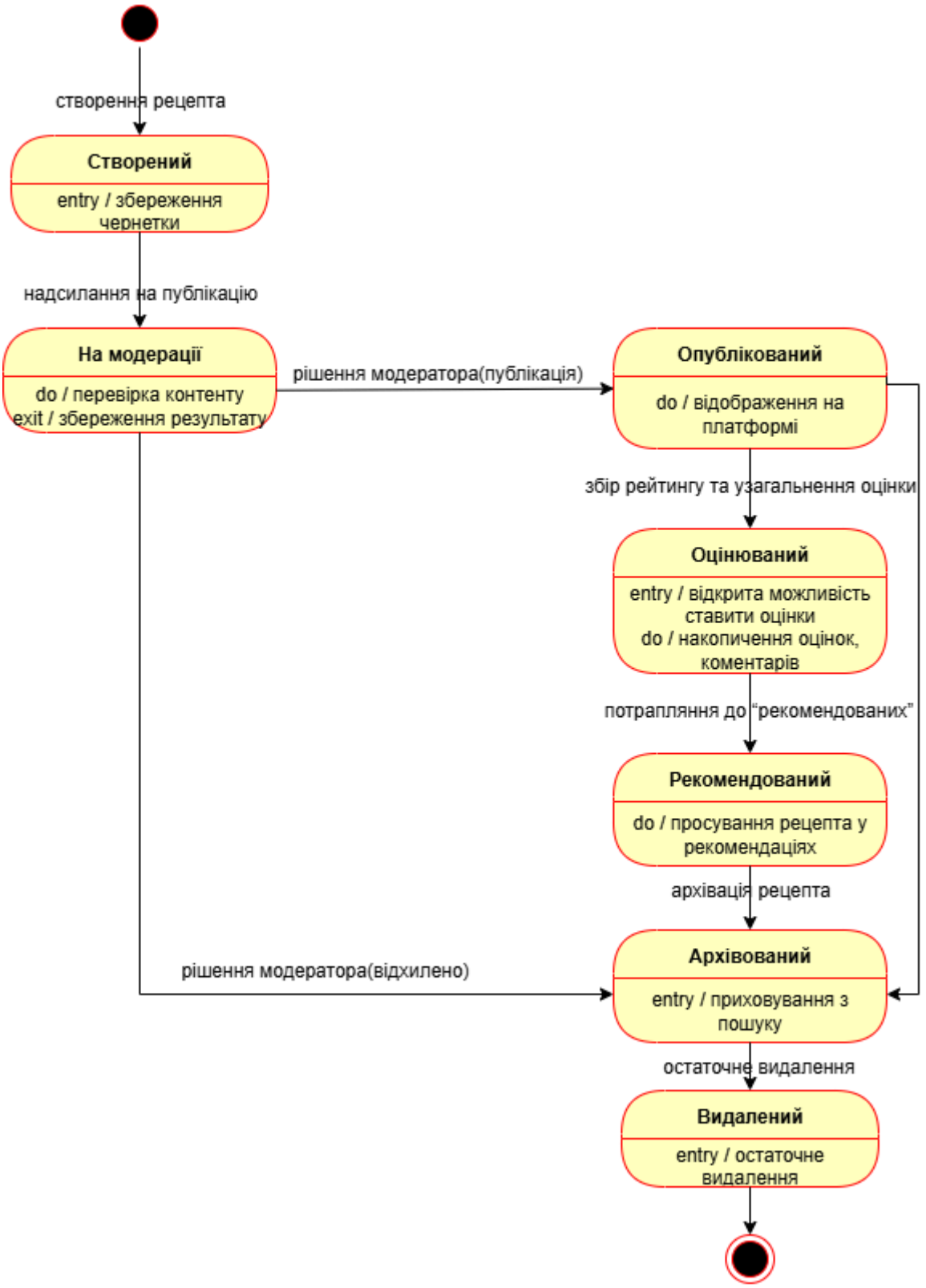
1. **Межі системи:**

Моделюється життєвий цикл рецепта у сервісі. Від моменту створення користувачем і до архівації/видалення.

2. **Перелік станів**

№	Стан	Опис	Entry/ do/ exit/ дії
1	Створений	Рецепт створено користувачем, але ще не опубліковано.	Entry / збереження чернетки
2	Опублікований	Рецепт доступний іншим користувачам для перегляду, оцінювання й коментування.	Do / відображення на платформі
3	На модерації	Якщо сервіс передбачає перевірку, рецепт потрапляє сюди.	Do / перевірка контенту на спам
4	Оцінюваний	Рецепт активно збирає відгуки/рейтинг.	Entry / відкрита можливість ставити оцінки Do / накопичення оцінок, коментарів
5	Рекомендований	Якщо рейтинг перевищив певний поріг — потрапляє в "рекомендовані".	Do / просування рецепта у рекомендаціях
6	Архівований	Рецепт знято з публікації — або користувачем, або автоматично за давністю чи порушення.	Entry / приховування з пошуку
7	Видалений	Рецепт остаточно видалений користувачем, або модератором.	Entry / остаточне виділення

3. **Події / тригери переходів**
   - створення рецепта.
   - надсилання на публікацію.
   - рішення модератора(публікація чи відхилено)
   - збір рейтингу та узагальнення оцінки.
   - потрапляння до “рекомендованих”.
   - архівація рецепта.
   - остаточне видалення.


створення рецепта
Створений
entry / збереження чернетки
надсилання на публікацію
На модерації
do / перевірка контенту
exit / збереження результату
рішення модератора(публікація)
Опублікований
do / відображення на платформі
збір рейтингу та узагальнення оцінки
Оцінюваний
entry / відкрита можливість ставити оцінки
do / накопичення оцінок, коментарів
потрапляння до "рекомендованих"
Рекомендований
do / просування рецепта у рекомендаціях
архівація рецепта
рішення модератора(відхилено)
Архівований
entry / приховування з пошуку
остаточне видалення
Видалений
entry / остаточне видалення

Завдання №4

Activity Diagram

Додаток С:

13. Доставка посилки клієнту

1. Межі процесу:

Старт: клієнт оформлює замовлення з доставкою.

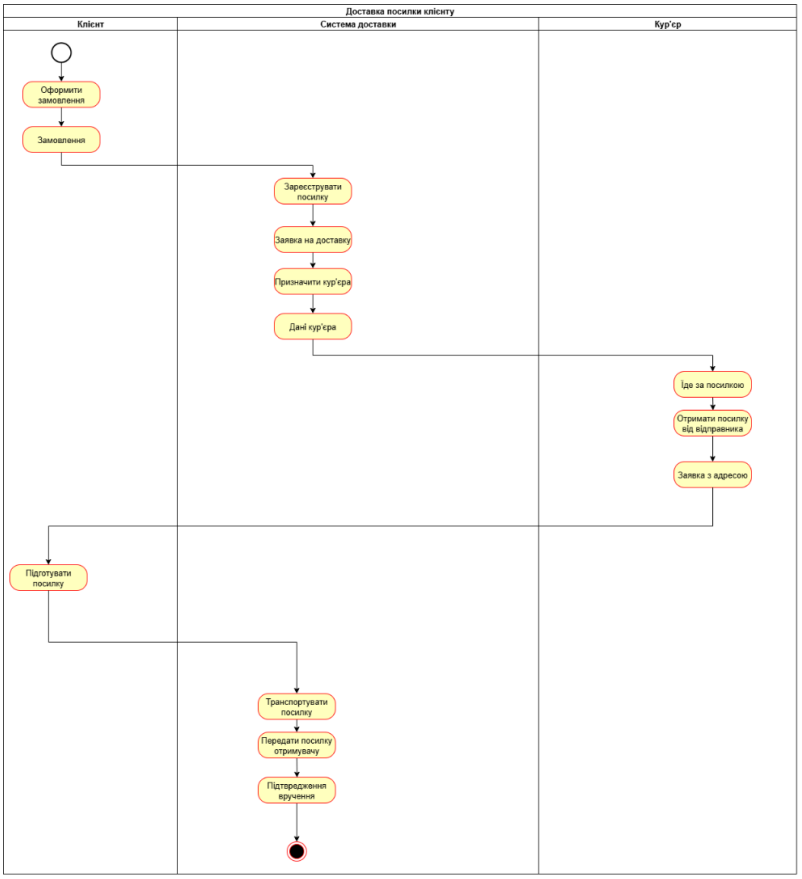
Завершення: посилка доставлена клієнту або повернута відправнику.

2. Учасники

- Клієнт
- Служба доставки
- Кур’єр

3. Потік дій

- **Клієнт оформлює замовлення** → об’єкт *Замовлення*.
- **Система реєструє посилку** та створює *Заявку на доставку*.
- **Система призначує кур'єра.**
- **Система видає дані про кур'єра.**
- **Система відмічає транспортування**
- **Передати посилку отримувачу.**
- **Підтвердження вручання.**
- **Final**



Завдання №5

Activity Diagram

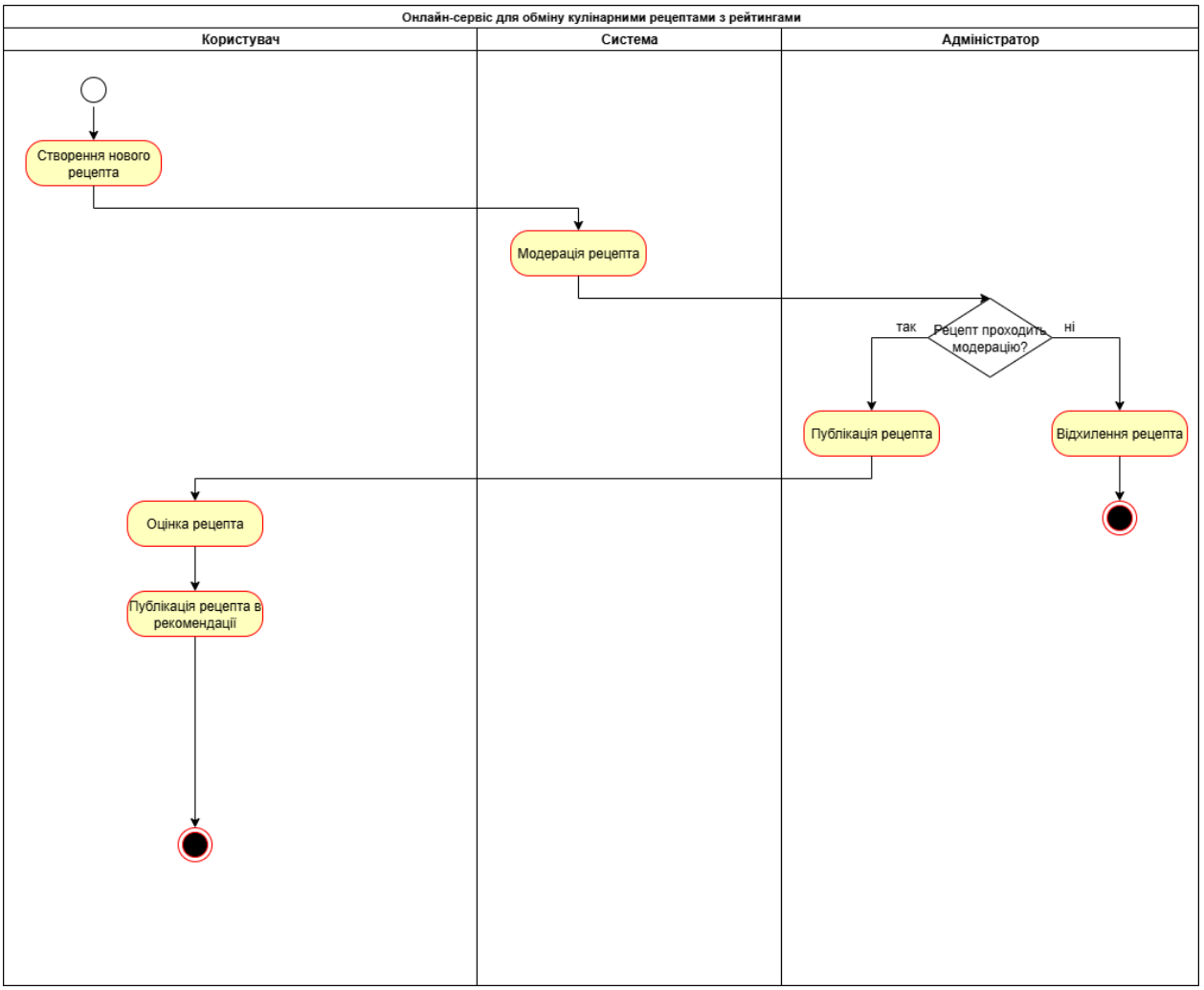
Додаток А: Онлайн-сервіс для обміну кулінарними рецептами з рейтингами

1. **Межі процесу:**
   Старт: Користувач створює новий рецепт на платформі.
   Завершення: Рецепт або публікується, або відхиляється, або архівується.
2. **Учасники:**

- **Користувач**: Створює рецепт, оцінює, коментує, додає в улюблені.
- **Система**: Обробляє рецепти, публікує їх, модерує, розраховує середні оцінки.
- **Адміністратор**: Модерує рецепти, видаляє невідповідні або порушуючі правила рецепти.

3. **Потік дій:**

- **Start node:** Користувач заходить на сайт і створює новий рецепт.
- **Створення рецепта:** Користувач заповнює форму для додавання рецепта.
- **Модерація:** Система перевіряє рецепт на відповідність вимогам (порушення або спам).
- Якщо **Так**, рецепт публікується.
- Якщо **Ні**, рецепт відхиляється.
- **Оцінка рецепта:** Користувач може оцінити рецепт.
- **Публікація рецепта в рекомендації** (за достатню к-сть оцінок)
- **Архівація:** Якщо рецепт неактуальний, він архівується або видаляється.
- **End node:** Завершення процесу.



Завдання №6

Sequence Diagram

Додаток D:

13. Клієнт – Сайт магазину – Кур’єр – Одержувач (*доставка посилки*).

1. **Учасники**

- **Клієнт —** <<actor>>: оформлює замовлення, підтверджує новий слот, передає посилку.
- **Сайт магазину —** <<boundary>>: приймає замовлення, призначає кур’єра, сповіщає сторони.
- **Кур’єр —** <<actor>>: приїзд на забір, повторний забір, доставка, передає .
- **Одержувач —** <<actor>>: приймає посилку, підтверджує отримання .

2. **Кроки виконання flow**

– **Клієнт → Сайт магазину:** *Оформити замовлення*

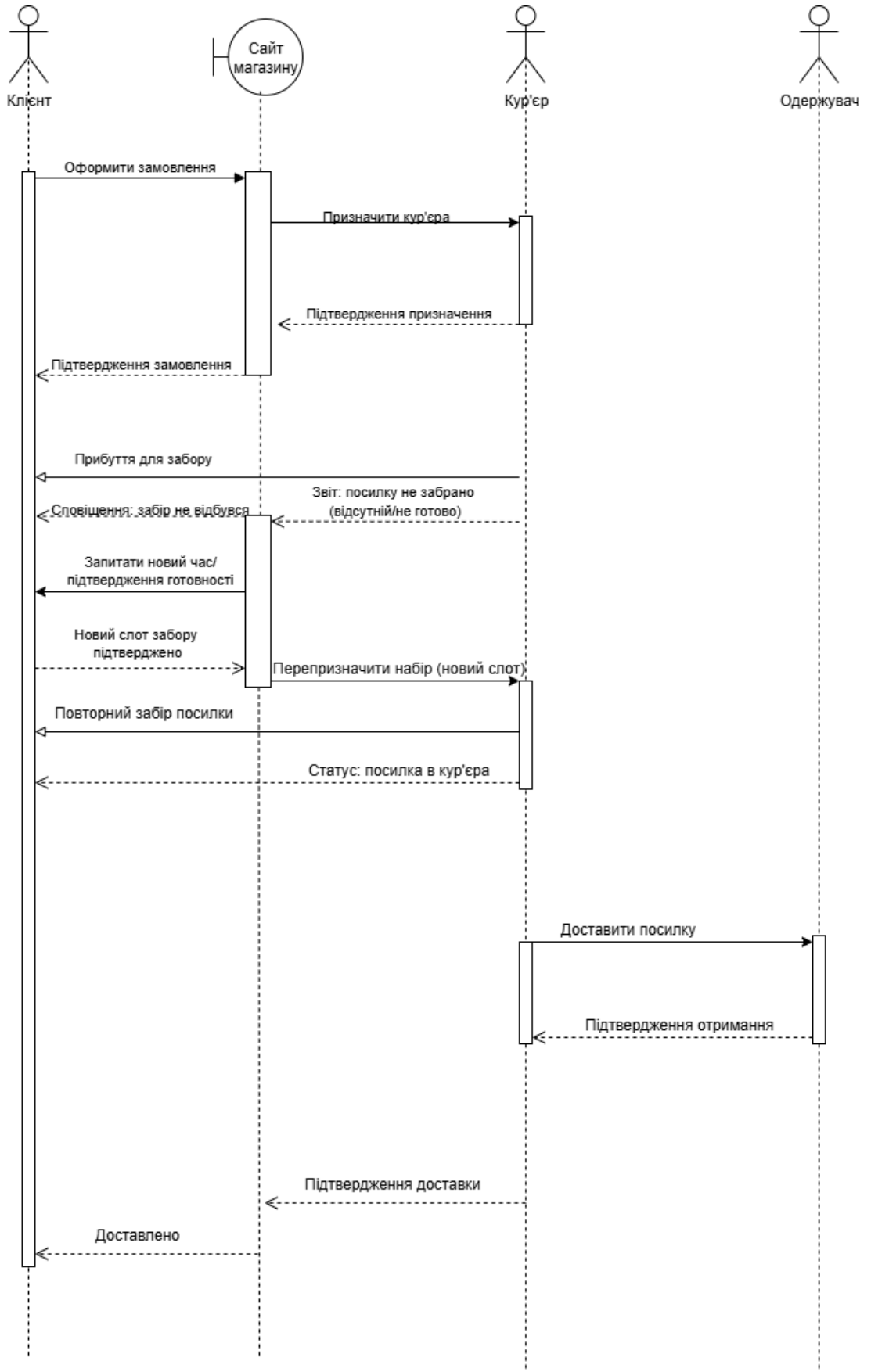
– **Сайт магазину → Кур’єр:**

– **Кур’єр → Сайт магазину (callback):** *Підтвердження призначення*
– Сайт магазину → Клієнт: *Підтвердження замовлення*

- **Кур’єр → Клієнт:** *Прибуття для забору (перша спроба)*
  – Кур’єр → Сайт магазину (звіт): *Посилку не забрано (відсутній/не готово)*

- **Сайт магазину → Клієнт:** *Сповіщення: забір не відбувся.*
  – Сайт магазину → Клієнт: *Запит нового часу/підтвердження готовності.*
  – Клієнт → Сайт магазину (callback): *Новий слот забору підтверджено.*
  – Сайт магазину → Кур’єр: *Перепризначити забір (новий слот).*

- **(Повторний забір) Кур’єр → Клієнт:** *Повторний забір посилки (успішний).*
  – Кур’єр → Сайт магазину (статус): *Посилка в кур’єра.*
- **Кур’єр → Одержувач:** *Доставити посилку.*
  – Одержувач → Кур’єр (відповідь): *Підтвердження отримання*
- **Кур’єр → Сайт магазину:** *Підтвердження доставки*
  – Сайт магазину → Клієнт: *Нотифікація: доставлено*



Змн.	Арк.	№ докум.	Підпис	Дата	ЛР.ОК24.ПІ231.13.02	Арк. PAG

Завдання №7

Sequence Diagram

Додаток А: Онлайн-сервіс для обміну кулінарними рецептами з рейтингами

1. **Короткий опис:**

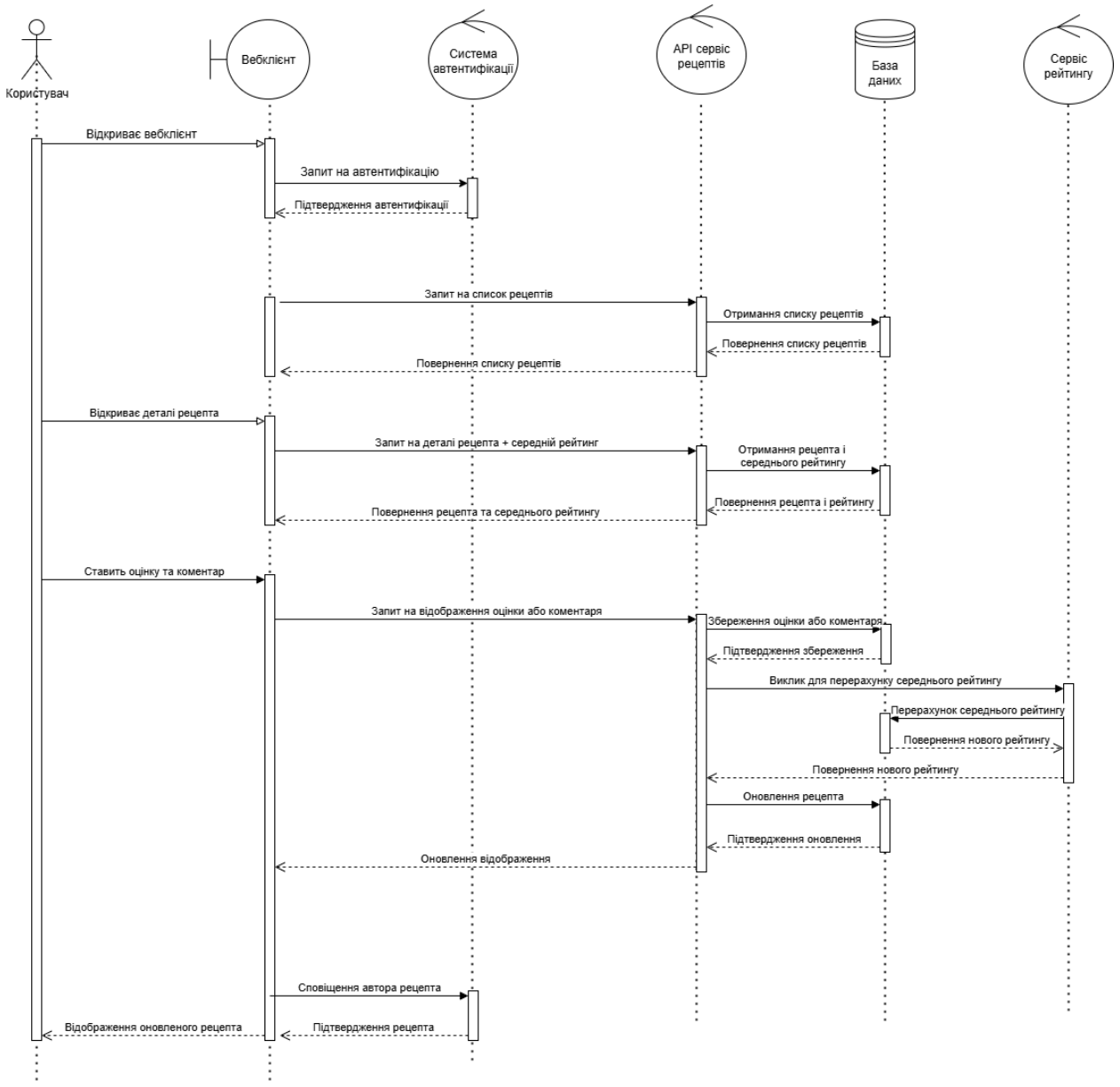
Суть: користувач заходить у сервіс, автентифікується, переглядає й фільтрує рецепти, відкриває карточку рецепта, ставить оцінку та коментар; система перераховує середній рейтинг і сповіщає автора/логіює події.
Фінал: користувач бачить оновлену оцінку/коментар на сторінці рецепта.

2. **Учасники**

Користувач — <<actor>>
Вебклієнт (WebApp) — <<boundary>>
API Сервіс рецептів — <<control>>
База даних — <<database>>
Сервіс рейтингу — <<control>>
Система автентифікації — <<control>>

3. **Послідовність кроків**

- Користувач відкриває вебклієнт, ініціюється сесія.
- Вхід: вебклієнт запитує токен в системи автентифікації, отримує підтвердження.
- Вебклієнт запитує список рецептів у API .
- Користувач відкриває деталі рецепта — API читає з БД й повертає рецепт + середній рейтинг.
- Користувач ставить оцінку й (за бажанням) коментар.
- API зберігає оцінку/коментар у БД; викликає Сервіс рейтингу для перерахунку середнього рейтингу; оновлює рецепт у БД.
- Вебклієнт оновлює відображення .



Завдання №8

Collaboration Diagram

Додаток D: Клієнт – Сайт магазину – Кур’єр – Одержувач (*доставка посилки*).

1. **Учасники процесу**
   - **Користувач <<user>>** — оформлює замовлення, передає посилку на забір, отримує сповіщення.
   - **Сайт магазину <<object>>** — приймає замовлення, призначає кур’єра, оновлює статуси.
   - **Кур’єр <<actor>>** — забирає, транспортує, вручає посилку.
   - **Одержувач <<actor>>** — приймає посилку, підтверджує вручення.
2. **Послідовність обміну повідомленнями (пронумерована)**
   - **клієнт → сайт:** Оформити замовлення (адреса, час).
     1. сайт → кур’єр: Створити заявку на забір.
     2. кур’єр → сайт: Підтвердження призначення.
     3. сайт → клієнт: Підтвердження замовлення.

- **кур’єр → клієнт:** Забрати посилку.
  1. кур’єр → сайт: Статус: посилка в кур’єра.
- **кур’єр → одержувач:** Доставити посилку.
  1. одержувач → кур’єр: Підтвердження отримання .
  2. кур’єр → сайт: Підтвердження доставки .
  3. сайт → клієнт: Сповіщення: доставлено.

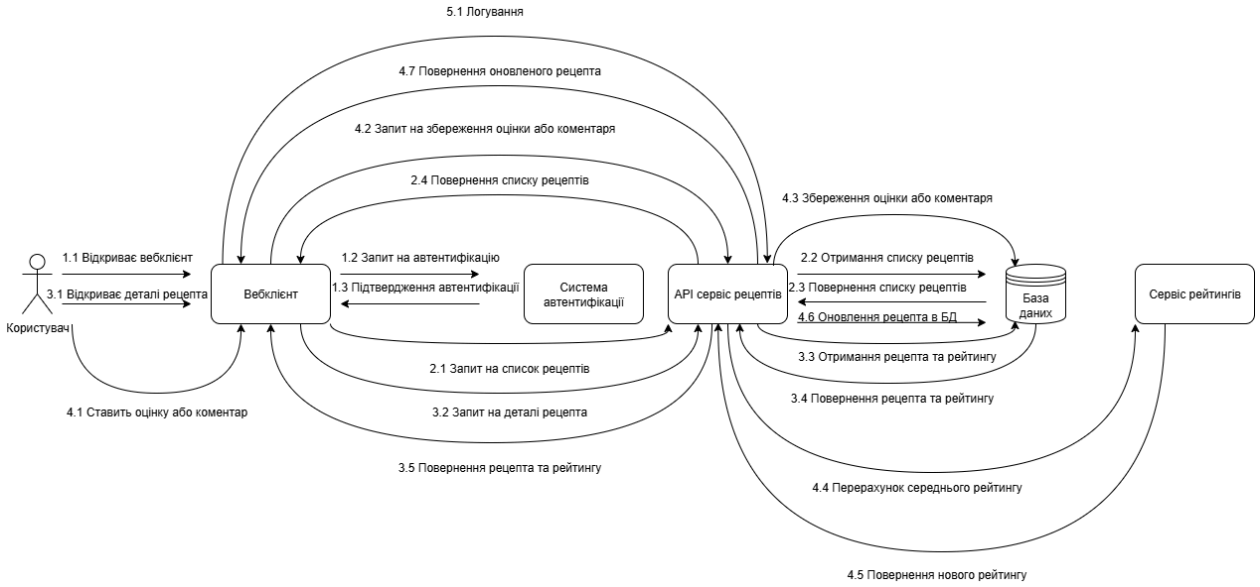
Завдання №9

Collaboration Diagram

Додаток А: ***Онлайн-сервіс для обміну кулінарними рецептами з рейтингами.***

Учасники процесу

1. **Користувач <<actor>>** — ініціює взаємодію (відкриває сервіс, переглядає рецепти, ставить оцінки та коментарі).
2. **Вебклієнт <<object>>** — інтерфейс, через який користувач працює із сервісом.
3. **API сервіс рецептів <<object>>** — основний бекенд, що приймає запити від UI, взаємодіє з іншими сервісами та БД.
4. **База даних <<database>>** — зберігає рецепти, коментарі, оцінки та інші дані.
5. **Сервіс рейтингів <<object>>** — виконує підрахунок і оновлення середнього рейтингу рецепта.
6. **Сервіс автентифікації <<object>>** — перевіряє користувачів та видає доступ.
4. **Послідовність кроків**
   - Користувач відкриває вебклієнт, ініціюється сесія.
   - Вхід: вебклієнт запитує токен в системи автентифікації, отримує підтвердження.
   - Вебклієнт запитує список рецептів у API .
   - Користувач відкриває деталі рецепта — API читає з БД й повертає рецепт + середній рейтинг.
   - Користувач ставить оцінку й (за бажанням) коментар.
   - API зберігає оцінку/коментар у БД; викликає Сервіс рейтингу для перерахунку середнього рейтингу; оновлює рецепт у БД.
   - Вебклієнт оновлює відображення . Паралельно йдуть логування/сповіщення .



Висновок: Під час виконання лабораторної роботи я ознайомився з поведінковими діаграмами UML, засвоїв їх призначення та особливості застосування для моделювання динамічних аспектів програмних систем.