# ЗМІСТ

ВСТУП ……………………………………………………………………… 4

РОЗДІЛ 1

АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ …..…………………………………… 5

1.1 Мета та призначення Веб-форуму……...………………...........….... 5

1.2 Користувачі Веб-форуму….….……………………………............... 7

1.3 Функціональні та нефункціональні можливості….……………….. 9

1.4 Аналіз існуючих Веб-форумів.……………………………............... 10

1.5 Висновок………………...….….…………………………….............. 13

РОЗДІЛ 2

ПОСТАНОВА ЗАДАЧІ….………………………………………………...... 13

2.1 Опис задачі….……………………………...........................…………14

2.2 Вимоги до програмного забезпечення………………………………15

2.3 Архітектура задачі………….………………………………............... 17

2.4 Висновок………….………………………………............................... 22

РОЗДІЛ 3

СТРУКТУРИ ДАНИХ ТА ДІАГРАМИ КЛАСІВ ………………………… 23

3.1 Загальна структура класів (моделей) бази даних..…………….……23

3.2 Структура класу (моделі) користувача……………………………....24

3.3 Структура класу (моделі) підфорумів та контенту..……………….. 25

3.4 Висновок..……………………………………………………………..27

РОЗДІЛ 4

СХЕМИ ВЗАЄМОДІЇ ОБ’ЄКТІВ ТА АЛГОРИТМИ ВИКОНАННЯ ЇХ МЕТОДІВ…………………………………………….………………………. 28

4.1 Загальна схема взаємодії з Веб-форумом………...…………….…… 28

4.2 Схема потоку аутентифікації ….……..…………………………….... 31

4.3 Схеми, пов’язані з підфорумами……..…………………………….... 33

4.4 Схеми, пов’язані з темами…………………...………………...…….. 41

4.5 Схеми, пов’язані з коментарями…………………...……………….. 48

РОЗДІЛ 5

ОПИС РОЗРОБЛЕНОЇ ПРОГРАМИ ………………………………………. 55

5.1 Опис архітектури Веб-форуму …………………...…………….…… 55

5.2 Опис реалізації основних функціональних можливостей…………. 58

5.3 Опис реалізації нефункціональних вимог ………....……………….. 61

5.4 Розробка та розгорнення програми …………………………………. 64

5.5 Використання сторонніх бібліотек та фреймворків………………… 65

РОЗДІЛ 6

ОПИС КОНТРОЛЬНОГО ПРИКЛАДУ …………………………………… 69

6.1 Користування веб-форумом (Frontend частина)…..…………….…... 70

6.2 Огляд бази даних активного форуму……….………………………... 75

ЛІТЕРАТУРА……………………………………………………………... 79

# Вступ

В сучасному університетському середовищі існує гостра потреба в ефективних каналах комунікації між студентами та викладачами. Традиційні методи, такі як лекції, семінари та лабораторні роботи, не завжди дають можливість для вільного обміну думками, ідеями та досвідом. Це може призводити до зниження зацікавленості студентів у навчанні, а також до труднощів у засвоєнні матеріалу.

Веб-форуми, подібні до Reddit, пропонують гнучку та зручну платформу для спілкування, яка може вирішити цю проблему. Їхні ключові переваги включають:

* Доступність: Користувачі можуть отримувати доступ до форуму з будь-якого пристрою з підключенням до Інтернету, що робить його доступним 24/7.
* Інтерактивність: Форум дає можливість студентам та викладачам вести спілкування, ставити питання, ділитися знаннями та досвідом.
* Різноманітність: Форум може містити різні підфоруми, присвячені різним темам, що дозволяє користувачам знаходити інформацію, яка відповідає їхнім інтересам.
* Спільнота: Форум може сприяти формуванню почуття спільноти серед студентів та викладачів, що може позитивно вплинути на їхню мотивацію та успішність навчання.

В даному курсовому проекті буде розроблена та реалізована об’єктно-орієнтована програмна система Web-форуму для університету, основана на ідеї Reddit. Ця система буде розроблена з використанням сучасних веб-технологій, таких як Next.js, TypeScript, Postgres, Prisma, Redis та Git.

Система буде мати наступні функціональні можливості:

* Реєстрація та авторизація користувачів
* Створення та редагування форумів
* Публікація та перегляд тем
* Безкінечна прокрутка та Оптимістичні оновлення
* Доступність для людей з обмеженими можливостями
* Можливість створення власних форумів з конкретними інтересами

Впровадження цієї системи у середовищі університету може призвести до:

* Покращення комунікації між студентами та викладачами
* Збільшення зацікавленості студентів у навчанні
* Підвищення ефективності засвоєння матеріалу
* Формування почуття спільноти серед студентів та викладачів
* Створення нового простору для самовираження та творчості

# Розділ 1 Аналіз предметної області

В даному розділі буде проведено детальний аналіз предметної області, а саме веб-форуму для університету, що буде адаптованою версією Reddit. Будуть визначені його мета, призначення, користувачі, функціональні та нефункціональні вимоги.

## 1.1 Мета та призначення Веб-форуму

Мета створення веб-форуму для університету полягає в наданні зручної та динамічної онлайн-платформи для спілкування, обміну знаннями та співпраці між студентами, викладачами та адміністрацією.

**Призначення** веб-форуму можна окреслити наступними пунктами:

* **Сприяння спілкуванню та обміну знаннями:** Форум має стати місцем, де студенти зможуть вільно спілкуватися один з одним, ставити запитання, ділитися ідеями та отримувати допомогу з навчальних предметів. Викладачі також зможуть використовувати форум для публікації оголошень, навчальних матеріалів та проведення онлайн-дискусій.
* **Підтримка навчального процесу:** Форум може слугувати додатковим інструментом для навчання, пропонуючи студентам доступ до різноманітних навчальних ресурсів, а також можливість отримати консультації та підтримку від викладачів та одногрупників.
* **Створення спільноти:** Форум може сприяти формуванню почуття спільноти серед студентів, викладачів та адміністрації університету, надаючи їм платформу для спільного обговорення актуальних питань, організації заходів та формування нових зв’язків.
* **Розширення можливостей досліджень:** Форум може використовуватися для проведення онлайн-досліджень, опитувань та збору зворотного зв’язку від студентів та викладачів.
* **Підтримка університетських ініціатив:** Форум може слугувати платформою для просування університетських ініціатив, оголошення про події та заходи, а також збирання зворотного зв’язку від спільноти.

Важливо зазначити, що веб-форум для університету має суттєві відмінності від комерційних аналогів, таких як Reddit. До цих відмінностей належать:

* **Цільова аудиторія:** Веб-форум для університету орієнтований на чітко визначену аудиторію - студентів, викладачів та адміністрацію конкретного навчального закладу.
* **Тематика:** Тематика веб-форуму для університету обмежена питаннями, пов’язаними з навчальним процесом, науковою діяльністю, життям університету та його поточними подіями.
* **Модерація:** Веб-форум для університету потребує більш жорсткої модерації для забезпечення дотримання правил поведінки та запобігання поширенню дезінформації та образливого контенту.
* **Інтеграція:** Веб-форум для університету має бути інтегрований з іншими університетськими системами, такими як система управління навчанням та система аутентифікації.

Очікується, що веб-форум для університету стане цінним інструментом для спілкування, співпраці та навчання, сприяючи покращенню навчального середовища та зміцненню зв’язків у межах університетської спільноти.

## 1.2 Користувачі Веб-форуму

Веб-форум для університету розрахований на три основні категорії користувачів:

**1. Студенти:**

* Основна цільова аудиторія форуму.
* Можуть використовувати форум для:
  + Спілкування з одногрупниками та студентами з інших курсів.
  + Обговорення навчальних предметів, отримання допомоги з домашніми завданнями та підготовки до іспитів.
  + Пошуку інформації про університетське життя, події та можливості.
  + Створення власних підфорумів для обговорення тем, що їх цікавлять.
  + Участі в онлайн-дискусіях та опитуваннях.

**2. Викладачі:**

* Можуть використовувати форум для:
  + Публікації оголошень, навчальних матеріалів та завдань.
  + Проведення онлайн-дискусій та консультацій зі студентами.
  + Спілкування з колегами та обмін досвідом викладання.
  + Участі в розробці навчальних програм та методичних матеріалів.
  + Відстеження успішності студентів та надання їм зворотного зв’язку.

**3. Адміністрація університету:**

* Може використовувати форум для:
  + Інформування студентів про важливі новини та події університету.
  + Збору зворотного зв’язку від студентів та викладачів щодо покращення навчального процесу.
  + Проведення онлайн-опитувань та досліджень.
  + Сприяння розвитку університетської спільноти.
  + Моніторингу настроїв студентів та викладачів.

Важливо зазначити, що доступ до певних функцій форуму може бути обмежений залежно від категорії користувача. Наприклад, студенти можуть не мати можливості створювати власні підфоруми, а заблоковані користувачі не можуть створювати новий контент (теми, коментарі, тощо.)

Окрім основних категорій користувачів, до форуму можуть також мати доступ зовнішні користувачі, такі як потенційні студенти, роботодавці та представники інших навчальних закладів.

## 1.3 Функціональні та нефункціональні можливості Веб-форуму

### Функціональні можливості

Веб-форум для університету буде мати широкий спектр функціональних можливостей, які можна розділити на наступні категорії:

**1. Базові функції:**

* **Реєстрація та аутентифікація користувачів:** Користувачі зможуть зареєструватися на форумі та увійти до системи за допомогою своїх університетських логінів або інших методів аутентифікації.
* **Створення та управління підфорумами:** Користувачі зможуть створювати власні підфоруми, присвячені певним темам, а також управляти їхніми налаштуваннями та доступом.
* **Публікація та редагування тем:** Користувачі зможуть публікувати текстові повідомлення, зображення, відео та інші типи контенту в підфорумах.
* **Голосування та коментування:** Користувачі зможуть голосувати за повідомлення та коментувати їх, щоб висловити свою думку та вступити в дискусію.
* **Підписки:** Користувачі зможуть підписуватися на підфоруми та отримувати інформації про нові теми у своїй стрічці.
* **Пошук:** Користувачі зможуть шукати інформацію на форумі за ключовими словами, тегами та іншими критеріями.

**2. Додаткові функції:**

* **Персоналізація:** Користувачі зможуть налаштовувати свій профіль, вибирати теми оформлення та інші налаштування інтерфейсу.
* **Інтеграція з соціальними мережами:** Користувачі зможуть ділитися повідомленнями та посиланнями з форуму в соціальних мережах.
* **Модерація:** Модератори форуму зможуть видаляти спам, образливий контент та інші порушення правил.
* **Статистика:** Користувачі та модератори зможуть отримувати статистику про активність на форумі.

### Нефункціональні можливості

Окрім функціональних можливостей, веб-форум для університету також повинен відповідати певним нефункціональним вимогам, таким як:

* **Продуктивність:** Форум має бути доступним та швидко завантажуватися навіть при великій кількості користувачів та повідомлень.
* **Масштабованість:** Форум має бути здатним до масштабування для підтримки зростаючої кількості користувачів та даних.
* **Надійність:** Форум має бути надійним та працювати без збоїв.
* **Безпека:** Форум має бути безпечним для захисту даних користувачів від несанкціонованого доступу та кібератак.
* **Доступність:** Форум має бути доступним для користувачів з обмеженими можливостями.
* **Ергономічність:** Інтерфейс форуму має бути зручним та простим у використанні.

## 1.4 Аналіз існуючих Веб-форумів

### Reddit

Reddit - це популярний веб-сайт соціальних новин, де користувачі можуть ділитися посиланнями, текстом та іншим контентом. Користувачі можуть голосувати за контент, щоб підвищити його рейтинг, а також коментувати його. Reddit розділений на тематичні підрозділи, які називаються “сабреддітами”.

**Переваги Reddit:**

* **Велика та активна спільнота:** Reddit має мільйони активних користувачів, що робить його чудовою платформою для спілкування з широкою аудиторією.
* **Широкий спектр тем:** На Reddit є сабреддіти на будь-яку тему, що робить його цінним ресурсом для інформації та дискусій.
* **Простота використання:** Reddit має простий та інтуїтивно зрозумілий інтерфейс, що робить його доступним для користувачів будь-якого рівня досвіду.

**Недоліки Reddit:**

* **Може бути токсичним:** Reddit відомий своєю токсичною культурою, де користувачі часто ображають та цькують один одного.
* **Складність пошуку інформації:** Через велику кількість контенту на Reddit може бути складно знайти інформацію, яку ви шукаєте.
* **Не підходить для приватних розмов:** Reddit не є приватною платформою, тому не підходить для приватних розмов.

### Facebook (коли він розпочинав як застосунок для студентів Гарвардського університету)

Facebook - це соціальна мережа, де користувачі можуть створювати профілі, спілкуватися з друзями та родиною, ділитися фотографіями та оновленнями. Facebook спочатку був запущений як застосунок для студентів Гарвардського університету, але згодом розширився, щоб охопити користувачів з усього світу.

**Переваги Facebook:**

* **Широка аудиторія:** Facebook має мільярди активних користувачів, що робить його чудовою платформою для спілкування з великою аудиторією.
* **Простий та інтуїтивно зрозумілий інтерфейс:** Facebook має простий та інтуїтивно зрозумілий інтерфейс, що робить його доступним для користувачів будь-якого рівня досвіду.
* **Можливість створення груп та сторінок:** Facebook дозволяє користувачам створювати групи та сторінки для спілкування за інтересами.

**Недоліки Facebook:**

* **Проблеми з конфіденційністю:** Facebook неодноразово критикували за проблеми з конфіденційністю даних користувачів.
* **Поширення дезінформації:** Facebook використовується для поширення дезінформації та фейкових новин.
* **Залежність:** Facebook може викликати звикання, через ботів та контент-ферми, тощо. Це може негативно вплинути на психічне здоров’я користувачів.

Reddit та Facebook - це дві популярні платформи, які можна використовувати як основу для створення веб-форуму для університету. Кожна платформа має свої переваги та недоліки, тому важливо ретельно їх оцінити, перш ніж приймати рішення.

## 1.5 Висновок

У цьому розділі було проведено детальний аналіз предметної області, а саме веб-форуму для університету, який буде адаптованою версією Reddit. Було визначено мету та призначення форуму, його користувачів, а також функціональні та нефункціональні вимоги.

Результати дослідження показали, що веб-форум для університету може стати цінним інструментом для спілкування, співпраці та навчання, сприяючи покращенню навчального середовища та зміцненню зв’язків у межах університетської спільноти.

Важливо зазначити, що успішне впровадження веб-форуму для університету потребуватиме ретельного планування, розробки та тестування, а також постійної підтримки та модерації.

Я впевнений, що веб-форум для університету, створений на основі результатів цього дослідження, стане успішним проектом, який принесе користь студентам, викладачам та адміністрації університету.

# Розділ 2 Постановка задачі

На основі попереднього розділу буде чітко сформульована задача, яку потрібно вирішити, а саме розробка веб-форуму для університету з певними функціональними можливостями та відповідністю до нефункціональних вимог.

## 2.1 Опис задачі

Створення веб-форуму для університету, який буде адаптованою версією Reddit

**Мета:**

* Надати зручну та динамічну онлайн-платформу для спілкування, обміну знаннями та співпраці між студентами, викладачами та адміністрацією університету.
* Сприяти покращенню навчального середовища та зміцненню зв’язків у межах університетської спільноти.

**Функціональні вимоги:**

* Реєстрація та аутентифікація користувачів.
* Створення та управління підфорумами.
* Публікація та редагування тем.
* Голосування та коментування.
* Підписки та сповіщення.
* Пошук.
* Інтеграція з соціальними мережами.
* Модерація.
* Статистика.

**Нефункціональні вимоги:**

* Продуктивність.
* Масштабованість.
* Надійність.
* Безпека.
* Доступність.
* Ергономічність.

**Користувачі:**

* Студенти
* Викладачі
* Адміністрація університету

**Технології:**

* Next.js
* TypeScript
* Prisma
* Git
* Redis
* Zod

**Обмеження:**

* Веб-форум буде функціонально доступний лише для студентів, викладачів та адміністрації університету.
* Деякі функціональні можливості можуть бути обмежені залежно від категорії користувача.
* Веб-форум буде відповідати політиці та правилам університету.

## 2.2 Вимоги до програмного забезпечення

**Мова програмування та фреймворк:**

**Next.js:** - це вебфреймворк з відкритим вихідним кодом, створений приватною компанією Vercel, що дозволяє розробляти веб-додатки на основі React із рендерингом на стороні сервера та генерацією статичних веб-сайтів. Next.js буде використовуватися для розробки front-end та back-end частини веб-форуму. Цей фреймворк обрано завдяки його простоті використання, високій продуктивності та можливості серверного рендерингу, що важливо для SEO та покращення завантаження сторінок.

**TypeScript:** TypeScript - TypeScript — мова програмування, представлена Microsoft восени 2012; позиціонується як засіб розробки вебзастосунків, що розширює можливості JavaScript. TypeScript буде використовуватися для покращення читабельності та надійності коду, а також для запобігання помилкам під час розробки.

**База даних:**

* **PostgreSQL** PostgreSQL - об’єктно-реляційна система керування базами даних (СКБД). Є альтернативою як комерційним СКБД (Oracle Database, Microsoft SQL Server, IBM DB2 та інші), так і СКБД з відкритим кодом (MySQL, Firebird, SQLite). PostgreSQL буде використовуватися для збереження данних у вигляді таблиці.
* **Prisma:** Prisma - це ORM з відкритим вихідним кодом для Node.js та TypeScript. Вона використовується як альтернатива написанню простого SQL або використанню інших інструментів доступу до баз даних, таких як конструктори SQL-запитів (наприклад, knex.js) або ORM (наприклад, TypeORM і Sequelize).
* **Redis:** Redis - розподілене сховище пар ключ-значення, які зберігаються в оперативній пам’яті, з можливістю забезпечувати довговічність зберігання на бажання користувача. Redis буде використовуватися для кешування даних веб-форуму, щоб покращити його продуктивність.

**Інші інструменти та бібліотеки:**

**ESLint:** ESLint - це інструмент статичного аналізу коду для JavaScript. ESLint буде використовуватися для забезпечення дотримання стандартів кодування та запобігання помилкам.

**dotEnv:** Dotenv - це модуль з нульовою залежністю, який завантажує змінні оточення з файлу .env в process.env. Зберігання конфігурації в оточенні окремо від коду базується на методології The Twelve-Factor App.

**Git:** Git - розподілена система керування версіями файлів та спільної роботи. Git є однією з найефективніших, надійних і високопродуктивних систем керування версіями, що надає гнучкі засоби нелінійної розробки, що базуються на відгалуженні та злитті гілок. Для забезпечення цілісності історії та стійкості до змін заднім числом використовуються криптографічні методи, також можлива прив’язка цифрових підписів розробників до тегів і комітів. Git буде використовуватися для автоматичного розгортання змін до коду веб-форуму на production сервер.

**Zod:** Zod - це бібліотека для оголошення та перевірки типів даних на мові TypeScript. Zod розроблено так, щоб бути максимально зручним для розробників. Метою є усунення дублюючих оголошень типів. У Zod ви оголошуєте валідатор один раз, і він автоматично виводить статичний тип TypeScript. Це дозволяє легко компонувати простіші типи у складні структури даних.

## 2.3 Архітектура задачі

**Модель сутностей зв’язків (ER-модель)**

Для формалізації задачі розробки веб-форуму для університету буде використана модель сутностей зв’язків (ER-модель). ER-модель дозволяє візуально представити сутності системи, їхні атрибути та зв’язки між ними.

**Сутності (моделі Prisma):**

* **Account**
  + ID запису (ключ)
  + ID користувача
  + Тип облікового запису
  + Провайдер облікового запису
  + ID облікового запису провайдера
  + Токен оновлення
  + Токен доступа
  + Дата закінчення токену
  + Тип токену
  + Дозволи токену
  + ID токену
  + Статус сеансу
  + Користувач (зв’язок з сутністю User)

Прошу зазначити, що ця модель розроблена за специфікацією NextAuth.js, тож має підтримувати поля, які не будуть використані в цьому проєкті (тип токену, ID токену і т.д.). Але це дає нам змогу додати інших провайдерів даних користувачів, окрім Google.

* **Session:**
  + ID запису (ключ)
  + Токен сеансу
  + ID користувача
  + Дата закінчення
  + Користувач (зв’язок з сутністю User)
* **User:**
  + ID запису (ключ)
  + Ім’я
  + Пошта
  + Підтвердження пошти
  + Псевдонім
  + Аватар
  + Роль
  + Облікові записи (зв’язок з сутністю Account)
  + Сеанси (зв’язок з сутністю Session)
  + теми (зв’язок з сутністю Thread)
  + Коментарі (зв’язок з сутністю Comment)
  + Проголошені коментарі (зв’язок з сутністю CommentVote)
  + Проголошені теми (зв’язок з сутністю Vote)
  + Створені підфоруми (зв’язок з сутністю Subforum)
  + Підписки (зв’язок з сутністю Subscription)
* **Subforum:**
  + ID запису (ключ)
  + Назва
  + Титульна назва
  + Опис
  + Дата створення
  + Дата оновлення
  + теми (зв’язок з сутністю Thread)
  + ID творця
  + Творец (зв’язок з сутністю User)
  + ID модераторів
  + Підписники (зв’язок з сутністю Subsctiption)
* **Subscription:**
  + ID створюється автоматично зв’язкою полей userId та subforumId
  + Користувач (зв’язок з сутністю User)
  + ID користувача
  + підфорум
  + ID підфоруму
* **Thread:**
  + ID запису (ключ)
  + Заголовок
  + Контент
  + Дата створення
  + Дата оновлення
  + підфорум (зв’язок з сутністю Subforum)
  + ID підфоруму
  + Автор (зв’язок з сутністю User)
  + ID автора
  + Коментарі (зв’язок з сутністю Comment)
  + Голоса (зв’язок з сутністю Vote)
* **Comment:**
  + ID запису (ключ)
  + ID коментаря
  + Текст
  + Дата створення
  + Дата оновлення
  + Автор (зв’язок з сутністю User)
  + ID автора
  + Тема (зв’язок з сутністю Thread)
  + ID теми
  + Відповідь до ID
  + Відповідь до (зв’язок з сутністю Comment)
  + Відповіді (зв’язок з сутністю Comment)
  + Голоса (зв’язок з сутністю CommentVote)
* **Vote:**
  + ID створюється автоматично зв’язкою полей userId та threadId
  + Користувач (зв’язок з сутністю User)
  + ID користувача
  + Тема (зв’язок з сутністю Thread)
  + ID теми
  + Тип
* **CommentVote:**
  + ID створюється автоматично зв’язкою полей userId та commentId
  + Користувач (зв’язок з сутністю User)
  + ID користуача
  + Коментар (зв’язок з сутністю Comment)
  + ID коментаря
  + Тип

**Сценарії використання**  
**1: Реєстрація та аутентифікація користувача**

1. Користувач переходить на сторінку авторизації веб-форуму.
2. Користувач надає свої дані через сервіс Google.
3. Якщо дані введені правильно, система створює нового користувача та автоматично аутентифікує користуача.
4. Сисиема перенаправляє користувача на попередню сторінку веб-форуму.

**2: Створення та управління підфорумом**

1. Користувач переходить на сторінку створення нового підфоруму.
2. Користувач вводить назву, опис та тему підфоруму.
3. Система створює новий підфорум та надає користувачу права адміністратора підфоруму.
4. Адміністратор підфоруму може редагувати інформацію про підфорум, видаляти повідомлення та коментарі, а також додавати нових модераторів.

**3: Публікація тем**

1. Користувач переходить на сторінку підфоруму.
2. Користувач натискає кнопку “Створити тему”.
3. Користувач вводить заголовок та контент теми.
4. Система публікує повідомлення на підфорумі.

**4: Голосування та коментування**

1. Користувач переходить на сторінку повідомлення.
2. Користувач може проголосувати за повідомлення, натиснувши кнопку “Подобається” або “Не подобається”.
3. Система автоматично відображає нову кількість голосів.

## 2.4 Висновок

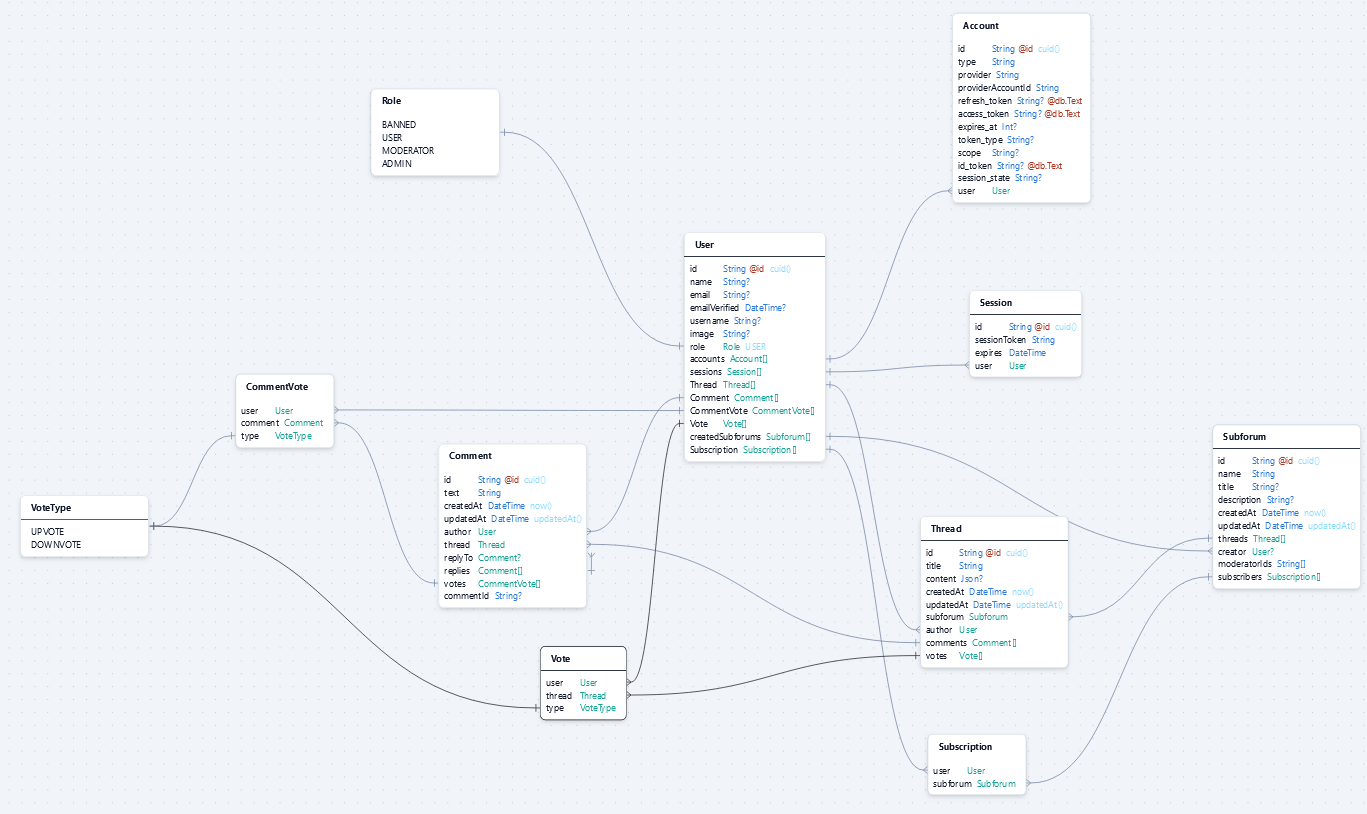
У цьому розділі було описано постановку задачі розробки веб-форуму для університету. Було формалізовано задачу за допомогою ER-моделі та сценаріїв використання, а також описано вимоги до програмного забезпечення, включаючи функціональні, нефункціональні, продуктивності, безпеки, доступності та пріоритети вимог. Було також представлено план реалізації проекту.

# Розділ 3 Структури даних та діаграми класів

В цьому розділі буде описано структури даних, які будуть використовуватися для зберігання інформації у веб-форумі, а також буде представлена діаграма класів, яка відображає взаємозв’язки між цими структурами.

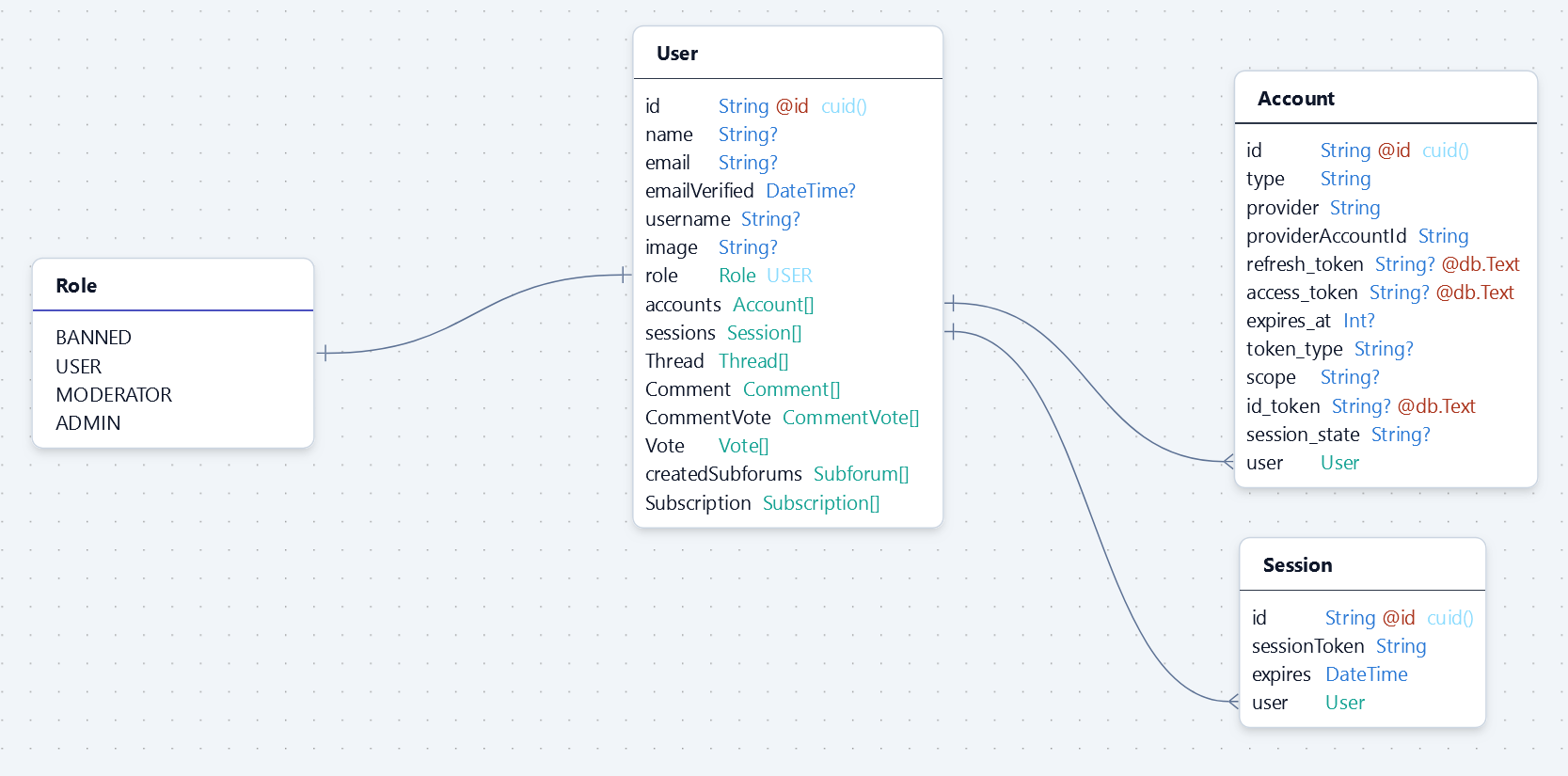
## 3.1 Загальна струкура класів (моделей) бази даних

Ця діаграма показує загальну структуру всіх класів (моделей), використаних у базі даних Веб-форуму.



## 3.2 Структура класу (моделі) користувача

Ця діаграма показує взаємодію між користувачами та їхніми обліковими записами.



**Користувач:** Це сутність, яка представляє користувача системи. Користувач має такі атрибути:

* id: Унікальний ідентифікатор користувача.
* name: Ім’я користувача.
* email: Адреса електронної пошти користувача.
* emailVerified: Дата й час підтвердження адреси електронної пошти користувача.
* username: Псевдонім користувача.
* image: URL-адреса зображення профілю користувача.

**Обліковий запис:** Це сутність, яка представляє обліковий запис користувача в системі. Обліковий запис має такі атрибути:

* id: Унікальний ідентифікатор облікового запису.
* type: Тип облікового запису (наприклад, “Google”, “Facebook” тощо).
* provider: Постачальник облікового запису (наприклад, “Google”, “Facebook” тощо).
* providerAccountId: Ідентифікатор облікового запису в системі постачальника.
* refresh\_token: Токен оновлення для облікового запису.
* access\_token: Токен доступу для облікового запису.

**Роль:** Це сутність, яка представляє роль користувача в системі. Роль є типом ENUM (енумератор) та не має атрибутів. Замість цього цю сутність можна використовувати як тип даних та значення.

**Сеанс:** Це сутність, яка представляє сеанс користувача в системі. Сеанс має такі атрибути:

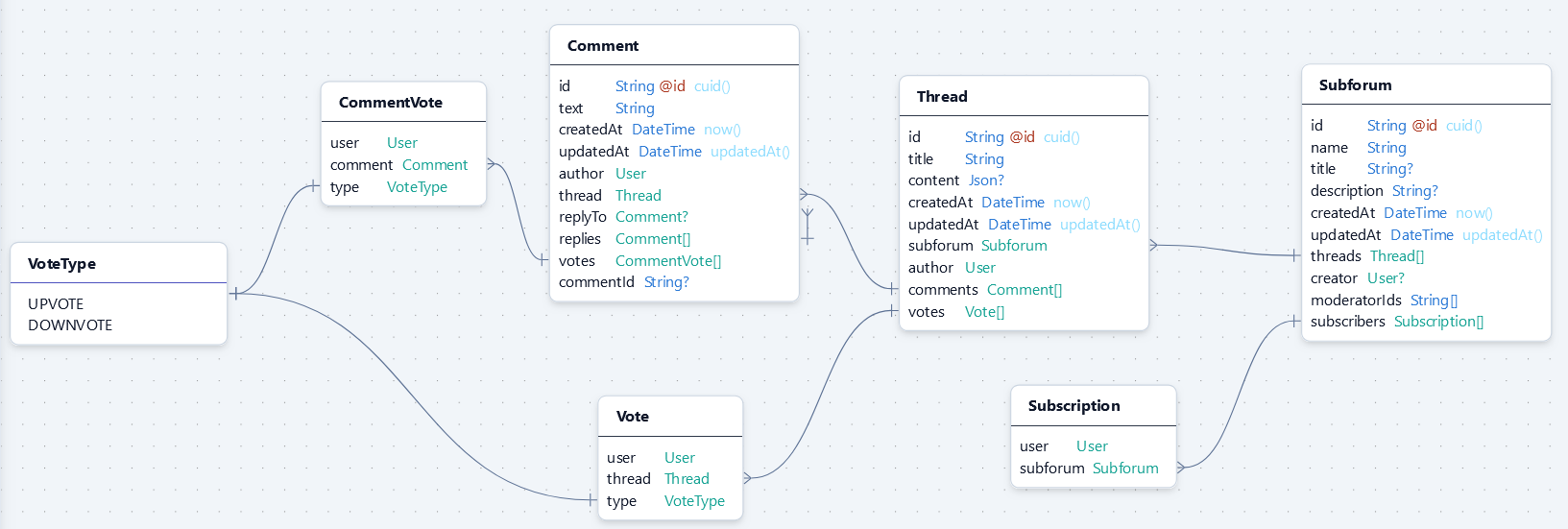
* id: Унікальний ідентифікатор сеансу.
* token: Токен сеансу.
* expires: Дата й час закінчення терміну дії сеансу.
* user: Користувач, до якого належить сеанс.

**Відносини**

* Користувач має Роль
* Користувач може мати багато Облікових записів
* Користувач може мати багато Сеансів

## 3.2 Структура класу (моделі) підфорумів та контенту

Ця діаграма показує взаємодію серед контенту, що наповнює підфоруми.



**Підфорум:**

* id: Унікальний ідентифікатор підфоруму.
* name: Коротка назва підфоруму.
* title: Титульна назва підфоруму
* description: Опис підфоруму.

**Тема:**

* id: Унікальний ідентифікатор теми.
* title: Заголовок теми.
* content: Контент теми.
* author: Користувач, який створив тему.
* subforum: Підфорум, до якого належить тема.
* comments: Коментарі, що належать до цієї теми.
* votes: Перелік голосів, що належать до цієї теми.

**Коментар:**

* id: Унікальний ідентифікатор коментаря.
* text: Текст коментаря.
* author: Користувач, який написав коментар.
* thread: Тема, до якої належить коментар.
* replyTo: Коментар, до якого належить відповідь.
* replies: Відповіді, які належать до цього коментаря.
* votes: Перелік голосів, що належать до цього коментаря.

**Тип голосу:** Це сутність, яка представляє всі типи голосів (UPVOTE, DOWNVOTE). Ця сутність є типом ENUM (енумератор) та не має атрибутів. Замість цього цю сутність можна використовувати як тип даних та значення.

**Голос:**

* user: Користувач, який проголосував.
* thread: Тема, за яку проголосували.
* type: Тип голосу

**Голос до коментаря:**

* user: Користувач, який проголосував.
* comment: Коментар, за який проголосували.
* type: Тип голосу

**Відносини**

* Користувач може мати багато тем.
* Користувач може мати багато коментарів.
* Підфорум може мати багато тем.
* Тема може мати багато коментарів.
* Тема може мати багато голосів.
* Коментар може мати багато голосів.

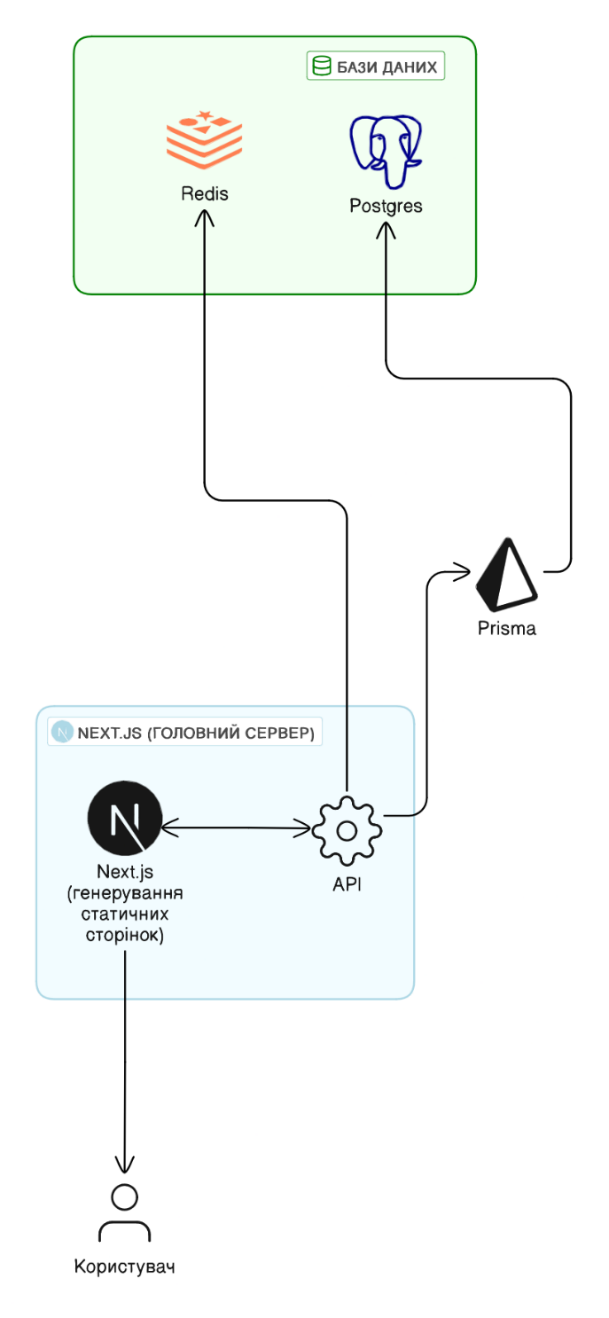
## 3.3 Висновок

У цьому розділі було подано у якості діаграм та описано структуру даних усього Веб-форуму. Були описани основні атрибути моделей, використанних у базі данних, а також проаналізовані відносини між моделями та їх атрибутами.

# Розділ 4 Схеми взаємодії об’єктів та алгоритми виконання їх методів

В цьому розділі буде представлена діаграма взаємодії об’єктів, яка показує, як об’єкти системи взаємодіють один з одним для виконання певних функцій. Також буде описано алгоритми виконання методів класів, які реалізують функціональні можливості веб-форуму.

## 4.1 Загальна схема взаємодії з Веб-форумом



**1. Бази даних:**

* **Postgres:** використовується для зберігання даних веб-застосунку.
* **Redis:** використовується для кешування даних, щоб покращити продуктивність.

**2. Next.js:**  
використовується для обробки HTTP-запитів та відповідей, а також для маршрутизації до API або сервера генерування статичних сторінок.

* **API:** використовується для обробки API-запитів та відповідей.
* **Next.js (генерування статичних сторінок):** використовується для генерування статичних HTML-сторінок.

**3. Користувач:**

* Користувач взаємодіє з веб-застосунком через браузер.

**4. Взаємодія між компонентами:**

* **Next.js** ↔ **API:** Ці два компоненти взаємодіють один з одним для обробки запитів, які потребують генерування динамічного контенту.
* **API** → **Prisma** → **Postgres:** API використовує Prisma для доступу та керування даними в Postgres.
* **API** → **Redis:** API використовує Redis для кешування даних, щоб покращити продуктивність.
* **Користувач** ↔ **Next.js:** Next.js генерує статичні HTML-сторінки, які надсилаються користувачеві через браузер.

## 4.2 Схема потоку аутентифікації

**Учасники:**

* **Browser:** Браузер користувача.
* **Server:** Веб-сервер.
* **Auth Server:** Сервер автентифікації Google.

**1. Ініціалізація:**

* Користувач в браузері надсилає GET-запит на сервер за адресою /api/auth/signin.
* Сервер перенаправляє користувача на сторінку входу Google.

**2. Аутентифікація Google:**

* Користувач вводить свої дані для входу в Google.
* Google перевіряє правильність введених даних.
* Google створює одноразовий код та перенаправляє користувача на callback URL сервера, який містить код.

**3. Обмін кодом на токен доступу:**

* Сервер зберігає отриманий код.
* Сервер надсилає POST-запит на Google з кодом, щоб отримати токен доступу.
* Google перевіряє код та генерує токен доступу.
* Google надсилає токен доступу серверу.

**4. Створення токена сеансу:**

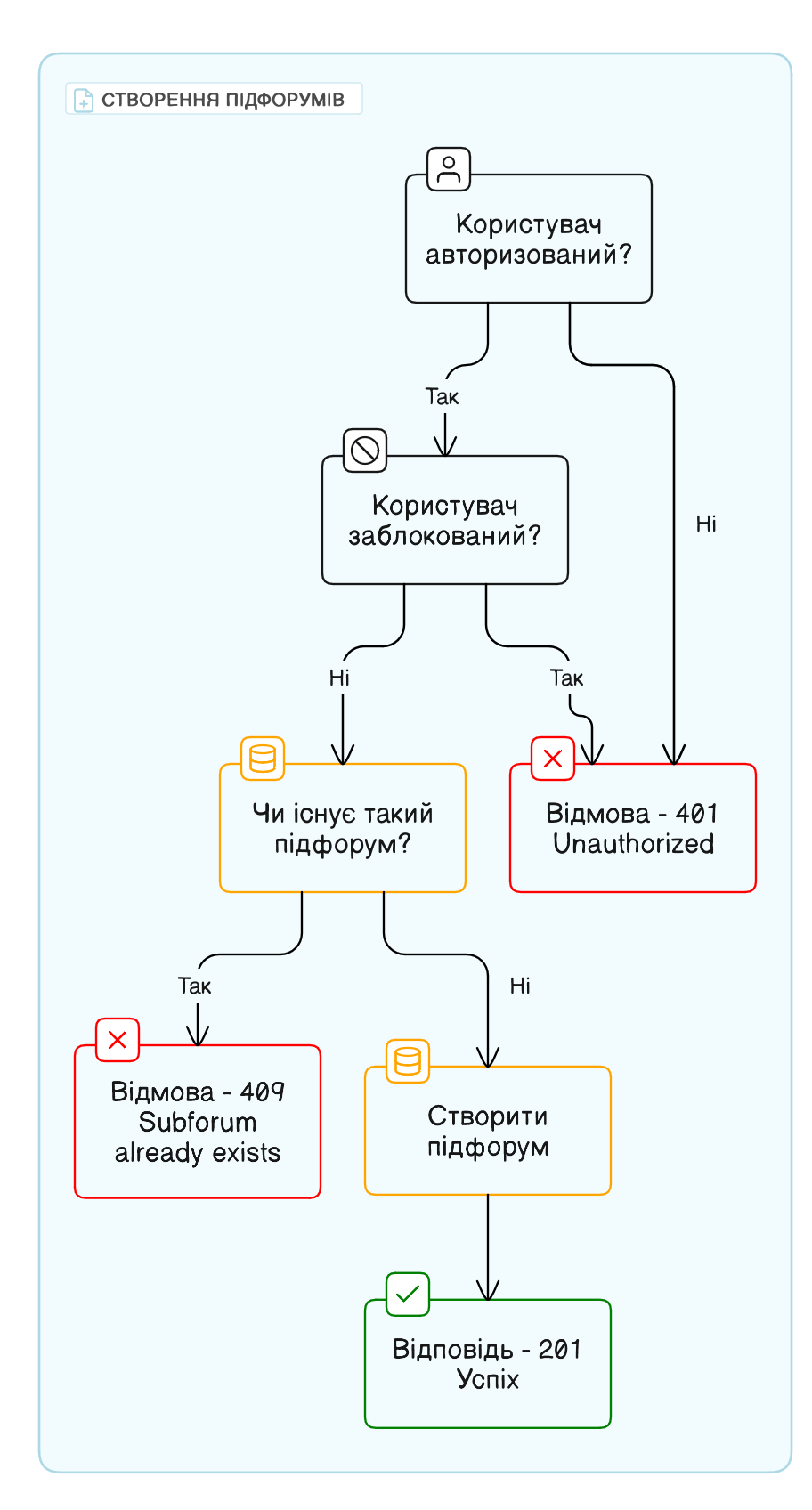
* Сервер генерує токен сеансу та зберігає його.
* Сервер надсилає токен сеансу браузеру.

**5. Авторизація користувача:**

* Браузер зберігає токен сеансу.
* Користувач авторизований на сервері.

## 4.3 Схеми, пов’язані з підфорумами

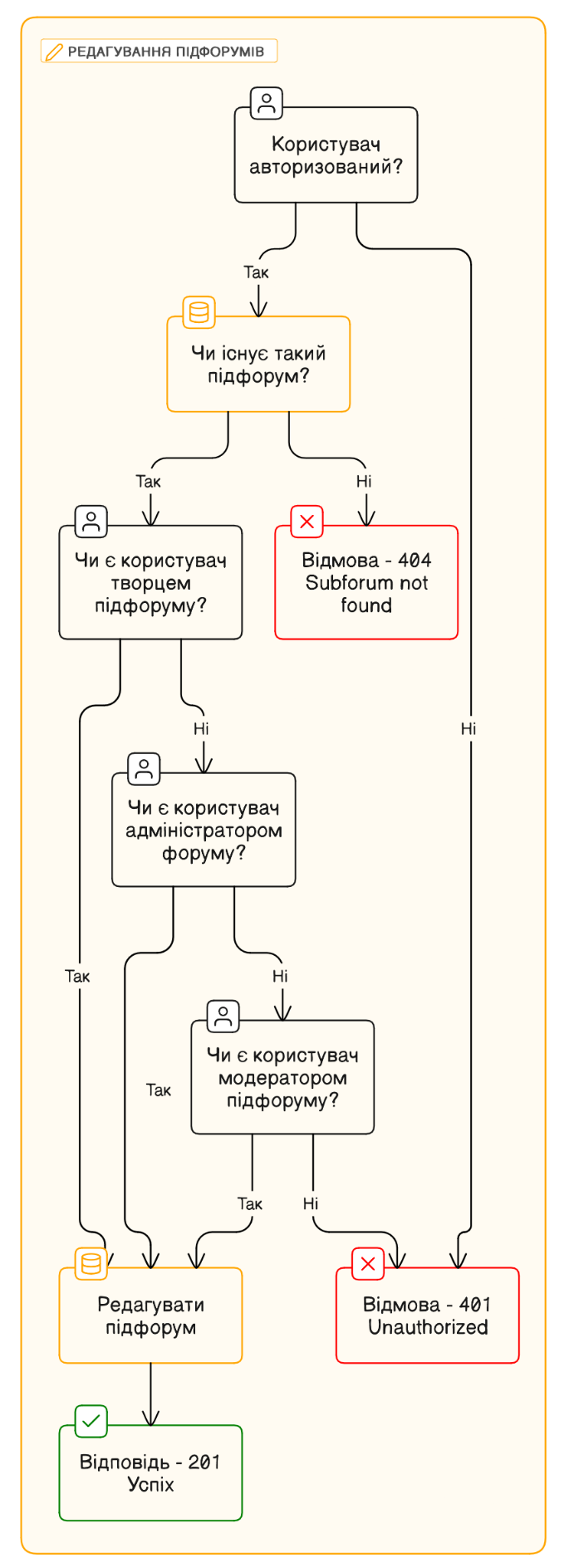
### Створення підфорума



**Процес:**

1. Користувач намагається створити новий підфорум.
2. Система перевіряє, чи авторизований користувач.
   * **Так:** Перехід до кроку 3.
   * **Ні:** Система повертає помилку “401 Unauthorized”.
3. Система перевіряє, чи заблокований користувач.
   * **Так:** Система повертає помилку "401 Unauthorized.
   * **Ні:** Перехід до кроку 4.
4. Система перевіряє, чи існує вже підфорум з такою ж назвою.
   * **Так:** Система повертає помилку “409 Subforum already exists”
   * **Ні:** Перехід до кроку 5.
5. Система створює новий підфорум.
6. Система повертає повідомлення про успішне створення підфоруму “201 Success”.

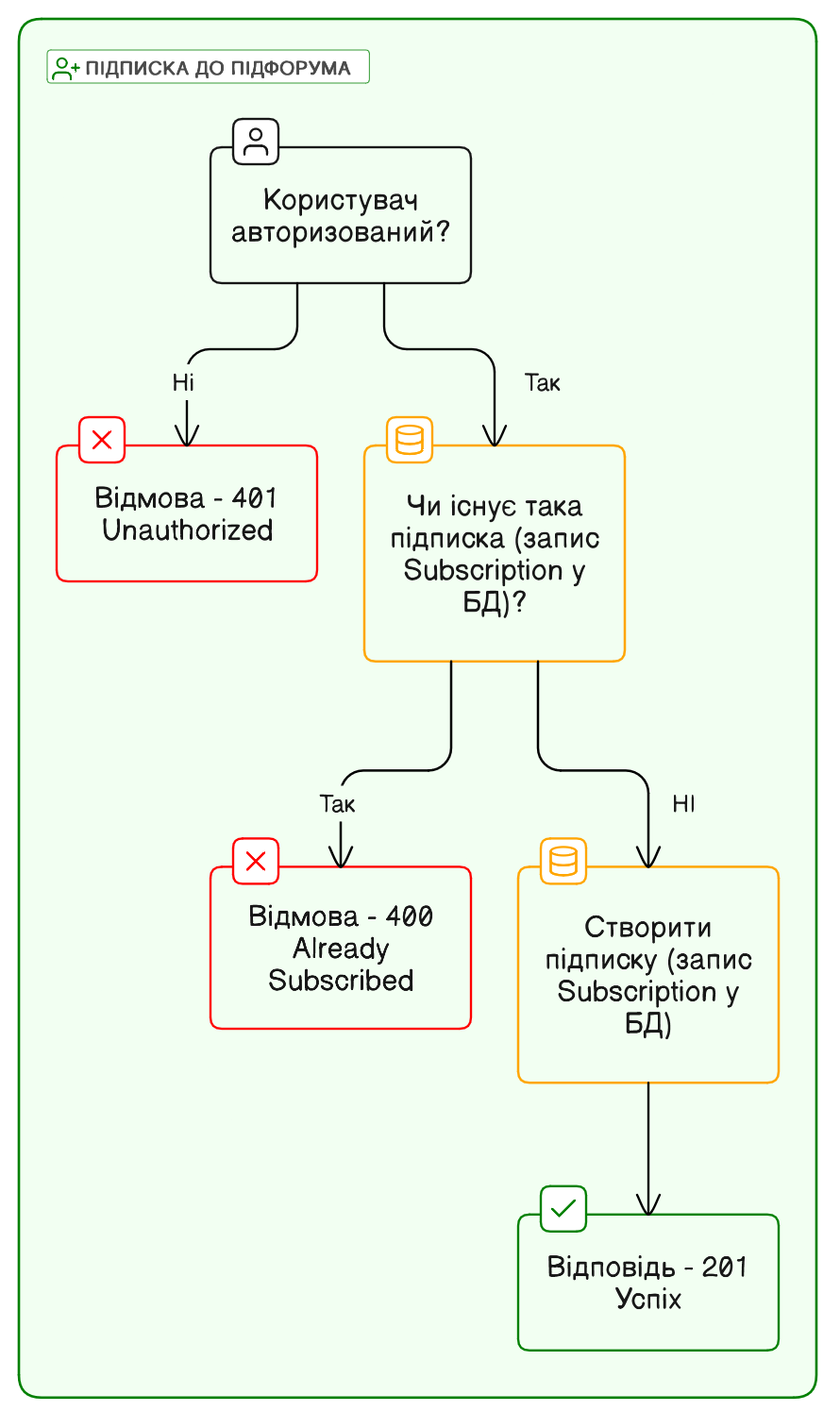
### Редагування підфорума



**Процес:**

1. Користувач намагається редагувати підфорум.
2. Система перевіряє, чи авторизований користувач.
   * **Так:** Перехід до кроку 3.
   * **Ні:** Система повертає помилку “401 Unauthorized”.
3. Система перевіряє, чи існує підфорум, який користувач хоче редагувати.
   * **Так:** Перехід до кроку 4.
   * **Ні:** Система повертає помилку “404 Subforum not found”.
4. Система перевіряє, чи є користувач творцем підфоруму.
   * **Так:** Перехід до кроку 5.
   * **Ні:** Система перевіряє, чи є користувач адміністратором форуму.
     + **Так:** Перехід до кроку 5.
     + **Ні:** Система перевіряє, чи є користувач модератором підфоруму.
       - **Так:** Перехід до кроку 5.
       - **Ні:** Система повертає помилку “401 Unauthorized”.
5. Система редагує підфорум.
6. Система повертає повідомлення про успішне редагування підфоруму “201 Success”.

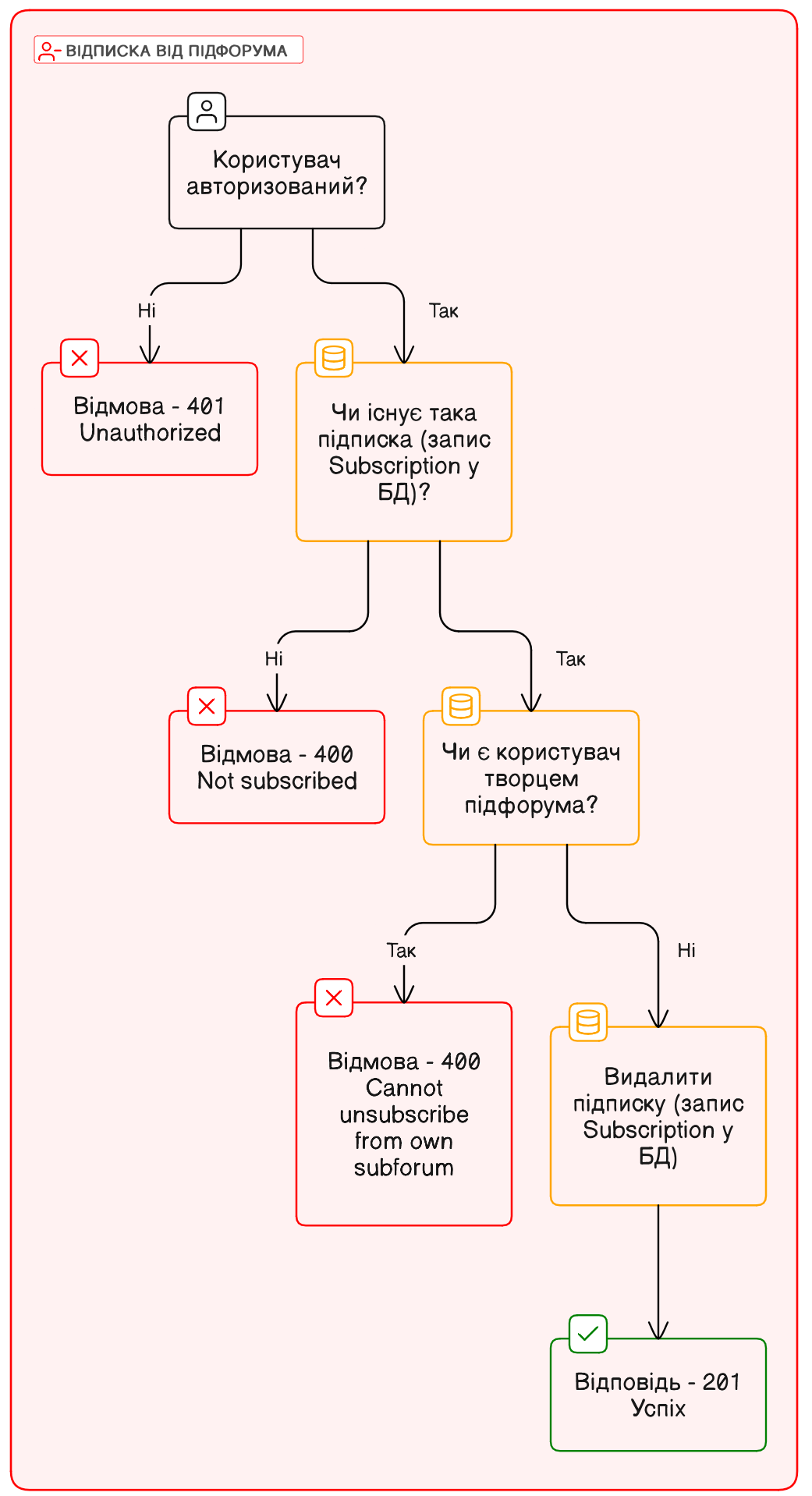
### Підписка до підфорума



**Процес:**

1. Користувач намагається підписатися на підфорум.
2. Система перевіряє, чи авторизований користувач.
   * **Так:** Перехід до кроку 3.
   * **Ні:** Система повертає помилку “401 Unauthorized”.
3. Система перевіряє, чи вже користувач підписаний на даний підфорум.
   * **Так:** Система перевіряє, чи існує запис про підписку в базі даних.
     + **Так:** Система повертає помилку “400 Already Subscribed”.
     + **Ні:** Система створює запис про підписку в базі даних.
   * **Ні:** Система створює запис про підписку в базі даних.
4. Система повертає повідомлення про успішну підписку “201 Success”.

### Відписка від підфорума

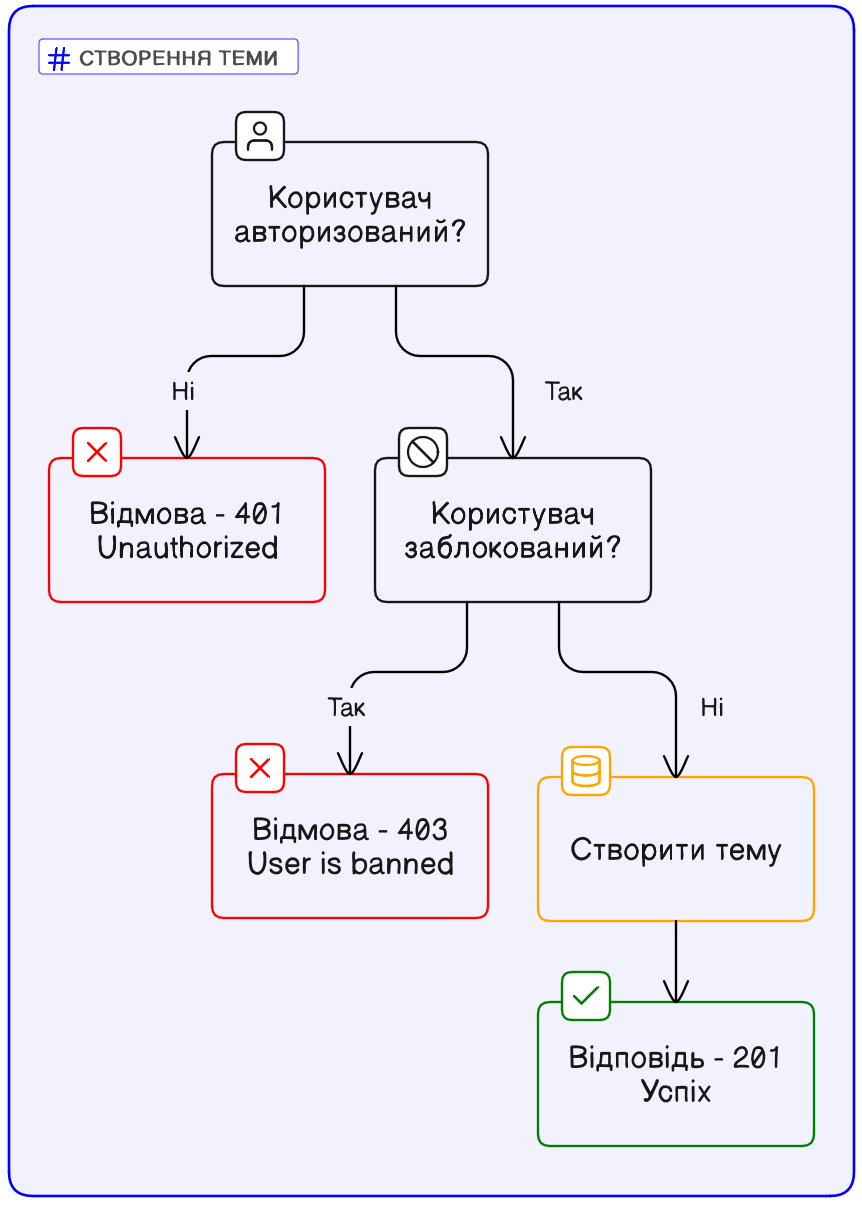


**Процес:**

1. Користувач намагається відписатися від підфоруму.
2. Система перевіряє, чи авторизований користувач.
   * **Так:** Перехід до кроку 3.
   * **Ні:** Система повертає помилку “401 Unauthorized”.
3. Система перевіряє, чи користувач вже підписаний на даний підфорум.
   * **Так:** Перехід до кроку 4.
   * **Ні:** Система повертає помилку “400 Not subscribed”.
4. Система перевіряє, чи є користувач творцем підфоруму.
   * **Так:** Система повертає помилку “400 Cannot unsubscribe from own subforum”.
   * **Ні:** Система видаляє запис про підписку з бази даних.
5. Система повертає повідомлення про успішну відписку “201 Success”.

## 4.4 Схеми, пов’язані з темами

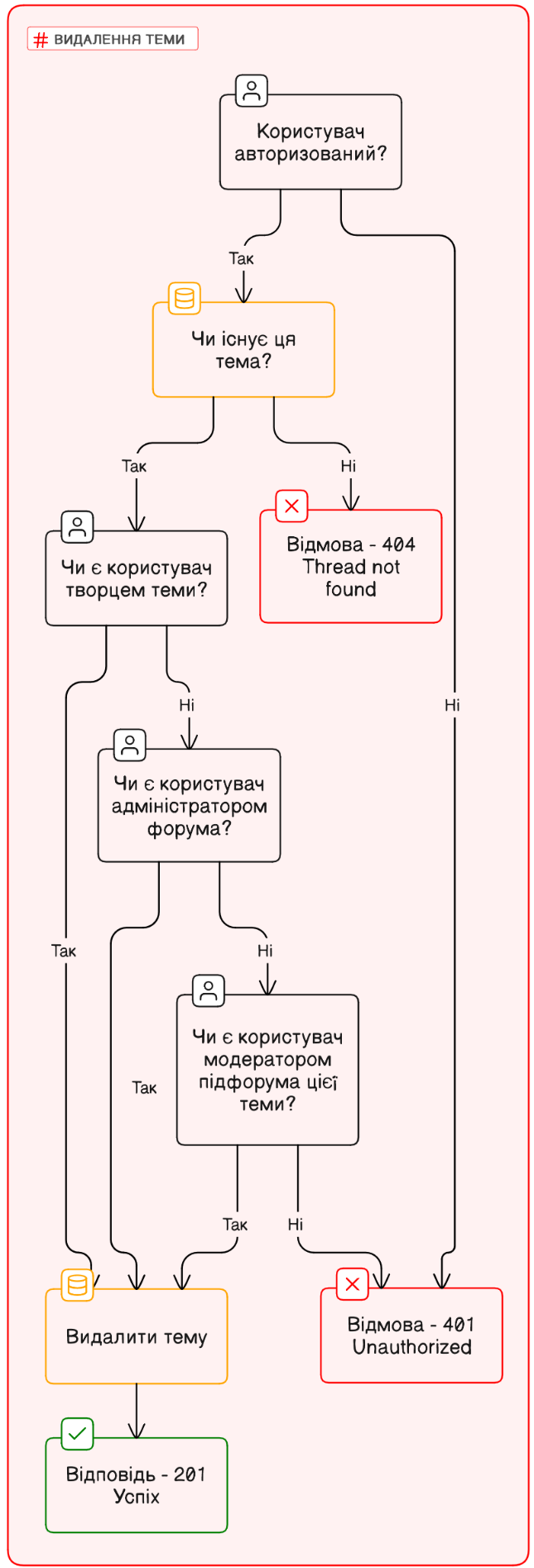
### Створення теми



**Процес:**

1. Користувач намагається створити тему.
2. Система перевіряє, чи авторизований користувач.
   * **Так:** Перехід до кроку 3.
   * **Ні:** Система повертає помилку “401 Unauthorized”.
3. Система перевіряє, чи заблокований користувач.
   * **Так:** Система повертає помилку “403 User is banned”.
   * **Ні:** Система створює тему на форумі.
4. Система повертає повідомлення про успішне створення теми “201 Success”.

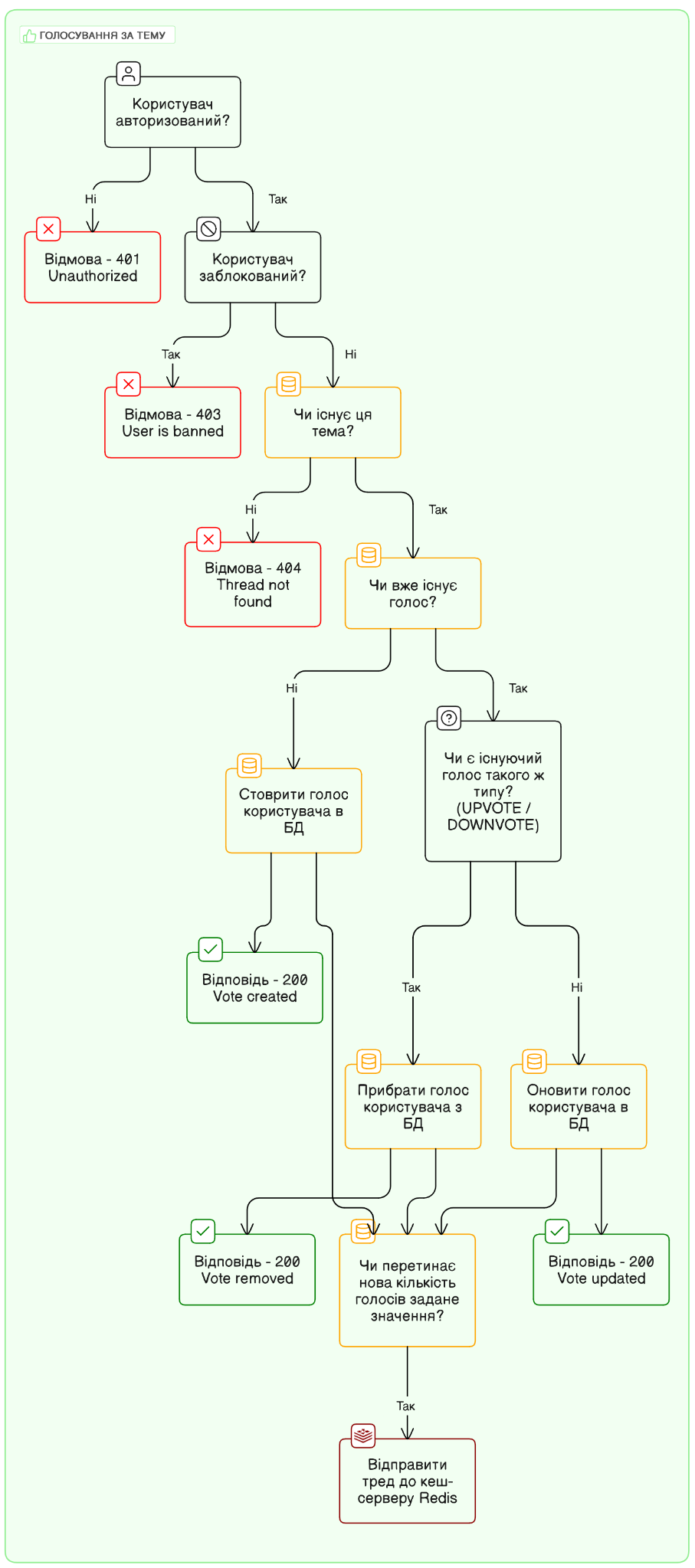
### Видалення теми



**Процес:**

1. Користувач намагається видалити тему.
2. Система перевіряє, чи авторизований користувач.
   * **Так:** Перехід до кроку 3.
   * **Ні:** Система повертає помилку “401 Unauthorized”.
3. Система перевіряє, чи існує тема, яку хоче видалити користувач.
   * **Так:** Перехід до кроку 4.
   * **Ні:** Система повертає помилку “404 Thread not found”.
4. Система перевіряє, чи є користувач творцем теми.
   * **Так:** Перехід до кроку 5.
   * **Ні:** Система перевіряє, чи є користувач адміністратором форуму.
     + **Так:** Перехід до кроку 5.
     + **Ні:** Система перевіряє, чи є користувач модератором підфоруму, де розміщена тема.
       - **Так:** Перехід до кроку 5.
       - **Ні:** Система повертає помилку “401 Unauthorized”.
5. Система видаляє тему з форуму.
6. Система повертає повідомлення про успішне видалення теми “201 Success”.

### Голос за тему

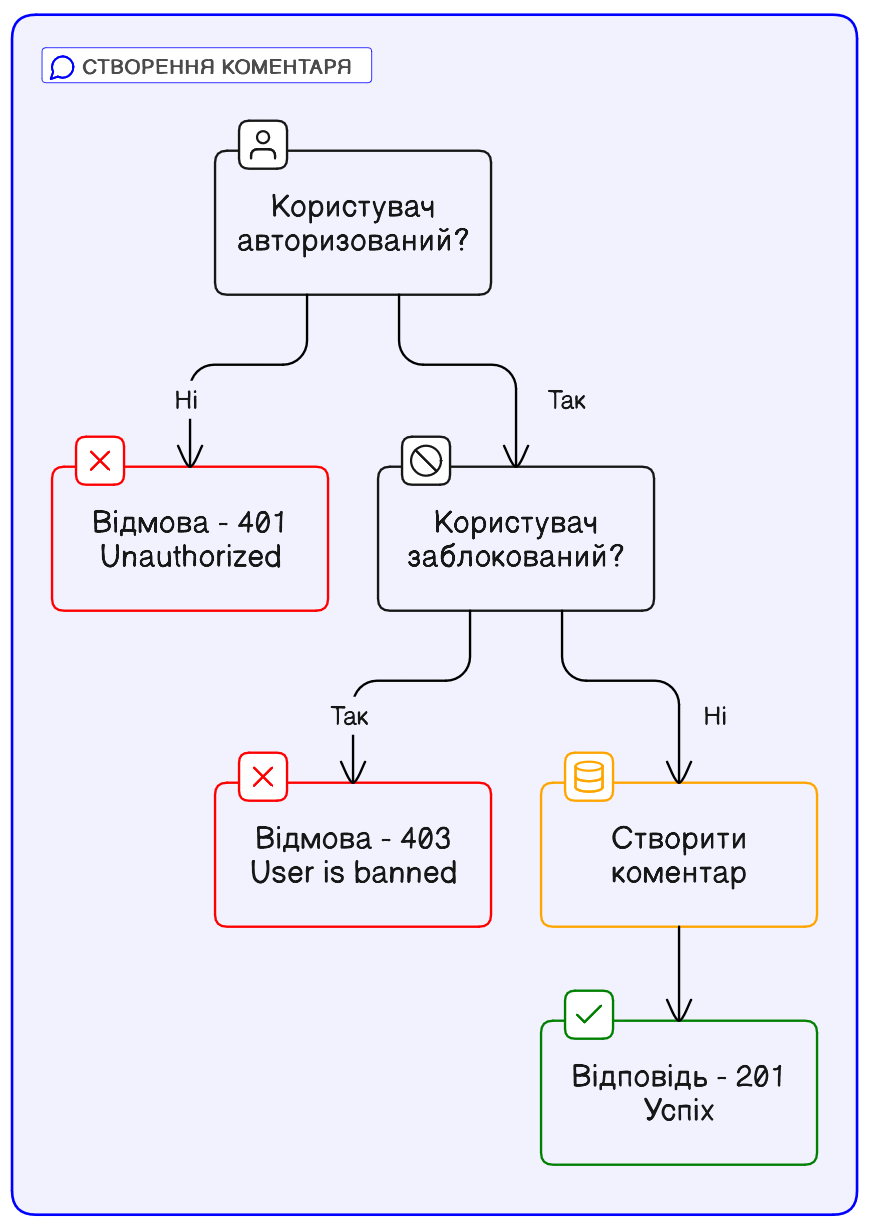


**Процес:**

1. Користувач намагається проголосувати за тему.
2. Система перевіряє, чи авторизований користувач.
   * **Так:** Перехід до кроку 3.
   * **Ні:** Система повертає помилку “401 Unauthorized”.
3. Система перевіряє, чи заблокований користувач.
   * **Так:** Система повертає помилку “403 User is banned”.
   * **Ні:** Перехід до кроку 4.
4. Система перевіряє, чи існує тема, за яку хоче проголосувати користувач.
   * **Так:** Перехід до кроку 5.
   * **Ні:** Система повертає помилку “404 Thread not found”.
5. Система перевіряє, чи вже голосував користувач за цю тему.
   * **Так:** Перехід до кроку 6.
   * **Ні:** Перехід до кроку 15.
6. Система перевіряє, чи новий голос має той самий тип (UPVOTE / DOWNVOTE), що й існуючий.
   * **Так:** Перехід до кроку 7.
   * **Ні:** Перехід до кроку 12.
7. Система видаляє голос користувача з бази даних.
8. Система повертає повідомлення “200 Vote removed”.
9. Перевірка “IsVoteTreshold” не виконується.
10. Система не надсилає тему до кеш-серверу Redis.
11. Перехід до кроку 14.
12. Система оновлює голос користувача в базі даних.
13. Система повертає повідомлення “200 Vote updated”.
14. Перевірка “IsVoteTreshold” не виконується.
15. Система створює запис про голос користувача в базі даних.
16. Система повертає повідомлення “200 Vote created”.
17. Система перевіряє, чи нова кількість голосів перетинає задане значення.
    * **Так:** Система відправляє тему до кеш-серверу Redis.
    * **Ні:** Система не робить нічого.

## 4.5 Схеми, пов’язані з коментарями

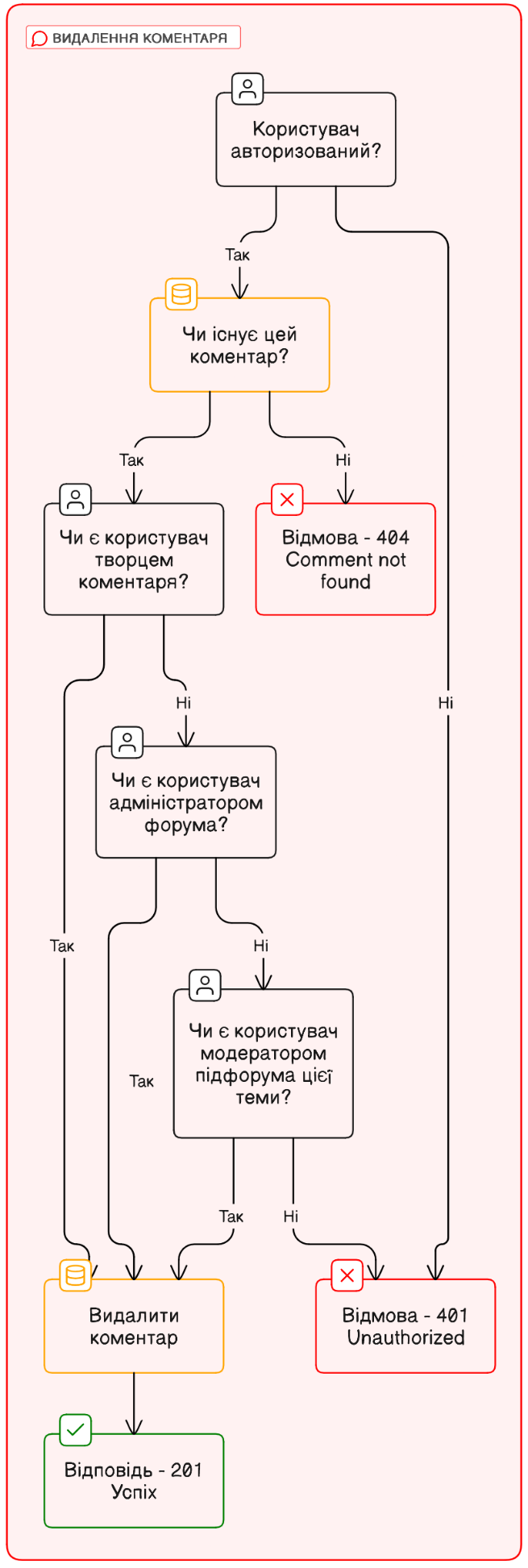
### Створення коментаря



**Процес:**

1. Користувач намагається створити коментар
2. Система перевіряє, чи авторизований користувач.
   * **Так:** Перехід до кроку 3.
   * **Ні:** Система повертає помилку “401 Unauthorized”.
3. Система перевіряє, чи заблокований користувач.
   * **Так:** Система повертає помилку “403 User is banned”.
   * **Ні:** Система створює коментар.
4. Система створює коментар у БД.
5. Система повертає повідомлення про успішне створення коментаря “201 Success”.

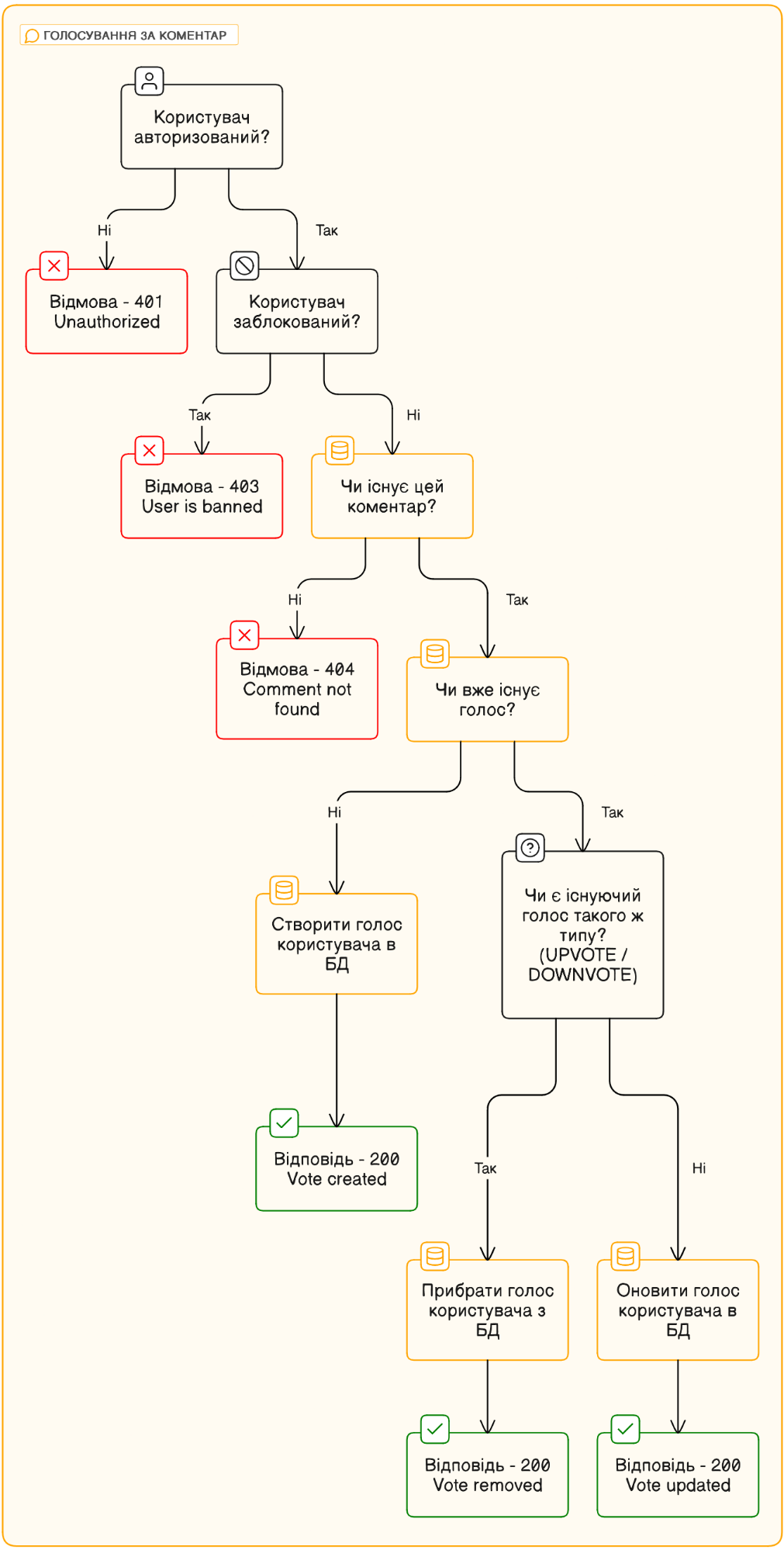
### Видалення коментаря



**Процес:**

1. Користувач намагається видалити коментар.
2. Система перевіряє, чи авторизований користувач.
   * **Так**: Перехід до кроку 3.
   * **Ні**: Система повертає помилку “401 Unauthorized”.
3. Система перевіряє, чи існує коментар, який хоче видалити користувач.
   * **Так**: Перехід до кроку 4.
   * **Ні**: Система повертає помилку “404 Comment not found”.
4. Система перевіряє, чи є користувач творцем коментаря.
   * **Так**: Перехід до кроку 5.
   * **Ні**: Система перевіряє, чи є користувач адміністратором форуму.
     + **Так**: Перехід до кроку 5.
     + **Ні**: Система перевіряє, чи є користувач модератором підфоруму цієї теми.
       - **Так**: Перехід до кроку 5.
       - **Ні**: Система повертає помилку “401 Unauthorized”.
5. Система видаляє коментар.
6. Система повертає повідомлення про успішне видалення коментаря “201 Success”.

### Голос за коментар



**Процес:**

1. Користувач намагається проголосувати за коментар.
2. Система перевіряє, чи авторизований користувач.
   * **Так**: Перехід до кроку 3.
   * **Ні**: Система повертає помилку “401 Unauthorized”.
3. Система перевіряє, чи заблокований користувач.
   * **Так**: Система повертає помилку “403 User is banned”.
   * **Ні**: Перехід до кроку 4.
4. Система перевіряє, чи існує коментар, за який хоче проголосувати користувач.
   * **Так**: Перехід до кроку 5.
   * **Ні**: Система повертає помилку “404 Comment not found”.
5. Система перевіряє, чи вже голосував користувач за цей коментар.
   * **Так**: Перехід до кроку 6.
   * **Ні**: Перехід до кроку 15.
6. Система перевіряє, чи новий голос має той самий тип.
   * **Так**: Перехід до кроку 7.
   * **Ні**: Перехід до кроку 12.
7. Система видаляє голос користувача з бази даних.
8. Система повертає повідомлення “200 Vote removed”.
9. Перевірка “IsVoteTreshold” не виконується.
10. Система не надсилає коментар до кеш-серверу Redis.
11. Перехід до кроку 14.
12. Система оновлює голос користувача в базі даних.
13. Система повертає повідомлення “200 Vote updated”.
14. Перевірка “IsVoteTreshold” не виконується.
15. Система створює запис про голос користувача в базі даних.
16. Система повертає повідомлення “200 Vote created”.

4.4 Висновок

У цьому розділі було показано та описано логіку backend-частини мого Веб-форуму: авторизація та аутентифікація користувачів, створення, редагування та видалення контенту - підфорумів, тем та коментарів.

# Розділ 5 Опис розробленої програми

В цьому розділі буде детально описана розроблена програма веб-форуму, включаючи її архітектуру, реалізацію основних функціональних можливостей, відповідність нефункціональним вимогам, тестування та використання сторонніх бібліотек та фреймворків.

## 5.1 Опис архітектури Веб-форуму

Веб-форум побудований на сучасних технологіях та принципах об’єктно-орієнтованого програмування, що забезпечує його масштабованість, надійність та зручність використання. Архітектура веб-форуму складається з наступних компонентів:

* **Next.js:** Використовується як фреймворк для розробки серверної частини веб-додатку. Next.js забезпечує серверну рендеринг сторінок, що покращує SEO та початкову загрузку сторінок, а також генерує статичні сайти, що робить їх більш швидкими та безпечними. Next.js також надає API-маршрути для обробки запитів від клієнта, що робить веб-додаток більш динамічним та функціональним.
* **TypeScript:** Використовується як мова програмування для розробки веб-додатку. TypeScript додає статичну типізацію до JavaScript, що покращує читабельність, надійність та керованість коду. TypeScript також має високий рівень інтеграції з IDE, що полегшує розробку та налагодження коду.
* **Prisma:** Використовується як ORM (Object-Relational Mapper) для взаємодії з базами даних. Prisma спрощує роботу з базами даних, надаючи API на основі JavaScript, що є більш зручним та лаконічним, ніж SQL-запити.
* **Бази даних:**
  + **Postgres:** Використовується як реляційна база даних для зберігання основних даних веб-форуму, таких як пости, коментарі, користувачі та підфоруми.
  + **Redis:** Використовується як кеш-сервер для зберігання часто використовуваних даних, таких як метадані тем, список користувачів, які перебувають онлайн, та інше. Це покращує продуктивність веб-додатку та зменшує навантаження на Postgres.

### 2. Взаємодія компонентів

* **Next.js:** Next.js генерує статичні HTML-сторінки, що відповідають за основний інтерфейс користувача веб-форуму. Ці сторінки надсилаються користувачеві при першому запиті, що забезпечує швидке завантаження.
* **Next.js (API):** API Next.js використовується для обробки динамічних запитів від користувача, таких як публікація постів, коментарів, голосування, авторизація та інше.
* **Prisma:** Prisma використовується для взаємодії з базами даних Postgres та Redis. Prisma перетворює запити API на SQL-запити та кешує дані в Redis, коли це можливо.
* **Бази даних:**
  + **Postgres:** Postgres зберігає основні дані веб-форуму, такі як пости, коментарі, користувачі та підфоруми.
  + **Redis:** Redis зберігає кешовані дані, такі як метадані тем, список користувачів, які перебувають онлайн, та інше.

### 3. Переваги архітектури

* **Масштабованість:** Архітектура веб-форуму може бути легко масштабована для обслуговування великої кількості користувачів та трафіку. Це досягається за рахунок горизонтального масштабування серверів Next.js та Redis.
* **Надійність:** Архітектура веб-форуму розроблена з урахуванням надійності. Next.js та Redis мають високу доступність, а Postgres використовує резервне копіювання та відновлення даних.
* **Зручність використання:** Архітектура веб-форуму проста та зрозуміла, що робить її зручною для розробки та підтримки. Next.js, TypeScript та Prisma – це добре документовані та популярні технології, що полегшує пошук
* **Продуктивність:** Використання кешування Redis значно покращує продуктивність веб-додатку. Часто використовувані дані зберігаються в Redis, що зменшує навантаження на Postgres та пришвидшує обробку запитів користувачів.
* **Безпека:** Next.js та Prisma мають вбудовані функції безпеки, які допомагають захистити веб-додаток від різних загроз. Крім того, використання реляційної бази даних Postgres та кеш-сервера Redis забезпечує додатковий рівень безпеки даних.
* **SEO-оптимізація:** Серверна рендеринг сторінок Next.js покращує SEO веб-форуму, оскільки пошукові системи можуть легко індексувати контент сторінок.

## 5.2 Опис реалізаціїї основних функціональних можливостей

### Реєстрація та аутентифікація користувачів

Користувачі можуть зареєструватися на форумі за допомогою своїх університетських логінів.

* Для реєстрації використовується API-endpoint /api/auth/signIn.
* Цей endpoint повністю оброблюється бібліотекою **NextAuth.js**
* Після успішної авторизації користувачу видається JWT-токен.
* JWT-токен використовується для автентифікації користувача при доступі до інших API-endpointів.

### Створення, управління та підписки підфорумів

Користувачі можуть створювати власні підфоруми.

* Для створення підфоруму використовується API-endpoint /api/subforum (POST).
* Цей endpoint приймає наступні параметри:
  + name: Назва підфоруму
  + title: Титульна назва підфоруму
  + description: Опис підфоруму
  + creatorId: ID користувача, вказується автоматично сервером

Користувачі можуть управляти налаштуваннями своїх підфорумів.

* Для цього використовується API-endpoint /api/subforum (PATCH).
* Цей endpoint приймає наступні параметри:
  + name: Назва підфоруму
  + title: Титульна назва підфоруму (опціонально)
  + description: Опис підфоруму (опціонально)

Користувачі можуть підписуватись на створені підфоруми.

* Для цього використовується API-endpoint /api/subforum/subscribe
* Цей endpoint приймає наступні параметри:
  + subforumid: ID підфоруму
  + userId: ID користувача, вказується автоматично сервером

Користувачі можуть відписуватися від підфорумів.

* Для цього використовується API-endpoint /api/subforum/unsubscribe
* Цей endpoint приймає наступні параметри:
  + subforumid: ID підфоруму
  + userId: ID користувача, вказується автоматично сервером

### Публікація, голосування та видалення тем

Користувачі можуть публікувати текстові повідомлення, зображення, відео та інші типи контенту в підфорумах.

* Для публікації теми використовується API-endpoint /api/subforum/thread/create
* Цей endpoint приймає наступні параметри:
  + subforumId: ID підфоруму
  + title: Назва теми
  + content: Контент теми (може містити текст, зображення, відео та інші типи контенту. Передається у форматі JSON))
  + authorId: ID користувача, вказується автоматично сервером

Користувачі, модератори та адміністрація можуть видаляти свої або чужі теми.

* Для видалення теми використовується API-endpoint /api/subforum/thread/delete
* Цей endpoint приймає наступні параметри:
  + id: ID теми

Користувачі можуть голосувати за повідомлення.

* Для голосування за повідомлення використовується API-endpoint /api/subforum/thread/vote
* Цей endpoint приймає наступні параметри:
  + threadId: ID теми
  + voteType: Тип голосу (UPVOTE або DOWNVOTE)

### Публікація, голосування та видалення коментарів

Користувачі можуть коментувати теми.

* Для коментування теми використовується API-endpoint /api/subforum/thread/comment.
* Цей endpoint приймає наступні параметри:
  + threadId: ID теми
  + content: Контент коментаря

Користувачі можуть голосувати за коментарі.

* Для голосування за коментар використовується API-endpoint /api/subforum/thread/comment/vote
* Цей endpoint приймає наступні параметри:
  + commetId: ID коментаря
  + voteType: Тип голосу (UPVOTE або DOWNVOTE)

Користувачі, модератори та адміністрація можуть видаляти свої або чужі коментарі.

* Для видалення теми використовується API-endpoint /api/subforum/thread/comment/delete
* Цей endpoint приймає наступні параметри:
  + id: ID коментаря

### Персоналізація

Користувачі можуть налаштовувати свій профіль.

* Для цього використовується API-endpoint /api/settings.
* Цей endpoint приймає наступні параметри:
  + username: Новий псевдонім
  + id: ID користувача, вказується автоматично сервером

Користувачі можуть вибирати між світлою та темною темами. Для автоматизації цього процесу була використана бібліотека **NextThemes.js**, яка використовуює браузерну технологію Local Storage (інформація про тему буде збережена та оброблена повністю на клієнтський частині)

### Інше

Користувачі можуть шукати підфоруми за допомогою Пошукової строки.

* Для цього використовується API-endpoint /api/search
* Цей endpoint приймає наступні параметри:
  + q: Пошукова строка (query). Цю строку буде використано як пошуковий запит до усіх назв підфорумів. Спеціально для цього, в БД було індексовано усі назви підфорумів, щоб пошвидшити цей процес.

Користувачі можуть додавати поширені посилання на сайти при створенні тем.

* Для цього використовується API-endpoint /api/crawl
* Цей endpoint приймає наступні параметри:
  + url: Пряме посилання на сайт. Сервер відправить запит до цього сайту, та завантажить мета-тегі, які використовуються спеціально для цього: title, description, og:image, theme-color.

Користувачі можуть додавати зображеня при створенні тем.

* Для цього використовується API-endpoint /api/uploadthing та сервіс **uploadthing**.
* Цей endpoint повністю оброблюється бібліотекою **uploadthing/next**
* Після успішного завантаження зображення до серверу uploadthing, користувач отримає пряме посилання до свого зображення.

## 5.3 Опис реалізаціїї нефункціональних вимог

### Продуктивність

* **Кешування:** Використовується кешування Redis для зберігання часто використовуваних даних. Це значно покращує час завантаження сторінок та зменшує навантаження на базу даних Postgres.
* **Оптимізація зображень:** Зображення стискаються та оптимізуються перед завантаженням сторінки, що зменшує час завантаження та економить пропускну здатність.
* **Серверний рендеринг сторінок:** Next.js використовується для серверної рендерингу сторінок, що покращує SEO та початкову загрузку сторінок.

### Масштабованість

* **Горизонтальне масштабування:** Форум може бути легко масштабований для обслуговування великої кількості користувачів та трафіку за рахунок горизонтального масштабування серверів Next.js та Redis.
* **Балансування навантаження:** Балансувальник навантаження може використовуватися для розподілу трафіку між декількома серверами Next.js та Redis.

### Надійність

* **Резервне копіювання та відновлення:** Регулярно створюються резервні копії бази даних Postgres та кешу Redis. У разі збою система може бути відновлена з резервної копії.
* **Кластеризація:** Redis може бути розгорнутий у кластері для забезпечення високої доступності та стійкості до відмов.
* **Моніторинг:** Система моніториться за допомогою інструментів моніторингу Vercel, які дозволяють виявляти та вирішувати проблеми до того, як вони вплинуть на користувачів.

### Безпека

* **Аутентифікація та авторизація:** Користувачі повинні пройти аутентифікацію та авторизацію перед доступом до форуму.
* **Контроль доступу:** Різні рівні доступу до форуму дозволяють контролювати, хто може публікувати повідомлення, редагувати контент та виконувати інші дії.
* **Захист від XSS та CSRF:** Застосовуються заходи безпеки для захисту форуму від атак XSS та CSRF.
* **Регулярні оновлення:** Система регулярно оновлюється для усунення вразливостей та випуску нових функцій безпеки.

### Доступність

* **Веб-стандарти:** Форум розроблений відповідно до веб-стандартів W3C, що робить його доступним для користувачів з різними браузерами та пристроями.
* **Доступний дизайн:** Інтерфейс форуму розроблений з урахуванням доступності, що робить його більш зручним для користувачів з обмеженими можливостями.
* **Альтернативний текст:** Зображення мають альтернативний текст, який описує їхній вміст для користувачів, які не можуть їх бачити.
* **Клавіатурна навігація:** Користувачі можуть переходити по форуму за допомогою клавіатури.

### Ергономічність

* **Інтуїтивно зрозумілий інтерфейс:** Інтерфейс форуму розроблений таким чином, щоб він був інтуїтивно зрозумілим та простим у використанні.
* **Чітка структура:** Форум має чітку структуру, що полегшує користувачам пошук потрібної інформації.

## 5.4 Розробка та розгорення програми

### Розробка

* **Фронтенд:** Фронтенд форуму розроблений з використанням Next.js, React, TypeScript та TailwindCSS. Next.js використовується для серверної рендерингу сторінок, що покращує SEO та початкову загрузку сторінок. React використовується для створення інтерфейсу користувача. TypeScript використовується для статичної перевірки типів коду, що робить його більш надійним та легким у підтримці. TailwindCSS використовується для пришвидшення написання CSS-коду.
* **Git:** Код форуму зберігається в репозиторії Git. Git дозволяє відстежувати зміни коду та співпрацювати з іншими розробниками.

### Розгорення

* **CI/CD:** Використовується платформа CI/CD, а саме GitHub Actions, для автоматизації процесу розгорення.
* **Vercel:** Vercel використовується для розгорення форуму в Інтернет. Vercel - це американська хмарна платформа як сервісна компанія (PaaS). Компанія підтримує фреймворк веб-розробки Next.js. Архітектура Vercel побудована на основі композитної архітектури, а розгортання здійснюється через репозиторії Git.

**Процес розгортки:**

1. Код змінюється в репозиторії Git.
2. Платформа CI/CD запускає, тестує та збирає код.
3. Код компілюється та оптимізується.
4. Форум розгортається на Vercel.
5. Форум становиться доступний за новою адресою URL (або ж за старою, якщо це production-збірка).

## 5.5 Використання сторонніх бібліотек та фреймворків

* **Editor.js:** Editor.js - це модульний редактор для створення багатого контенту в веб-застосунках. Він пропонує набір плагінів для різних типів контенту, таких як текст, зображення, код, таблиці та багато іншого.
* **Hook Form:** Hook Form - це бібліотека React для створення форм з підтримкою валідації та управління станом. Вона пропонує простий та декларативний API, який робить створення форм легким та зручним.
* **Mantine:** Mantine - це бібліотека React, яка фокусується на створенні красивих та доступних компонентів. В цьому проєкті були використани тільки React-хуки від Mantine, які спрощують написання логіки на frontend частині додатку.
* **Next Auth:** Next Auth - це офіційна бібліотека автентифікації для Next.js. Вона спрощує процес автентифікації користувачів та надає ряд функцій, таких як підтримка OAuth, відновлення пароля та захист електронної пошти.
* **Prisma:** Prisma - це ORM (Object-Relational Mapper) для JavaScript, який спрощує взаємодію з базами даних. Він пропонує потужний API для створення, читання, оновлення та видалення даних, а також генерує типи TypeScript для ваших моделей даних.
* **Radix UI:** Radix UI - це бібліотека React UI з акцентом на доступність та гнучкість. Вона пропонує набір першокласних компонентів, які можна легко налаштувати відповідно до ваших потреб. В цьому проєкті був використаний форк Radix UI під назвою Shadcn UI.
* **Tanstack:** Tanstack - це набір бібліотек React для створення сучасних веб-застосунків. Він включає в себе компоненти UI, бібліотеки даних, інструменти розробки та багато іншого. В цьому проєкті були використани тільки React-хуки від Tanstack, які спрощують написання логіки на frontend частині додатку.
* **UploadThing:** Upload Thing - це бібліотека React для завантаження файлів. Вона пропонує простий та зручний API для завантаження файлів на сервер.
* **Axios:** Axios - це популярна бібліотека JavaScript для HTTP-запитів. Вона пропонує простий та зручний API для надсилання та отримання даних з серверів.
* **Cheerio:** Cheerio - це бібліотека JavaScript для роботи з HTML-документами. Вона пропонує API, подібний до jQuery, для вибору, маніпулювання та аналізу HTML-елементів.
* **Clsx:** Clsx - це бібліотека JavaScript для складання імен CSS-класів. Вона спрощує процес написання CSS-коду та робить його більш лаконічним.
* **Date-FNS:** Date-FNS - це бібліотека JavaScript для роботи з датами та часом. Вона пропонує широкий спектр функцій для форматування дат, обчислення часових інтервалів та багато іншого.
* **Lucide React:** Lucide React - це бібліотека React для використання іконок Lucide. Вона пропонує компоненти React для відображення іконок Lucide у ваших веб-застосунках.
* **Next:** Next.js - це фреймворк React для створення серверних та статичних веб-застосунків. Він пропонує ряд функцій, таких як маршрутизація, серверна рендеринг та автоматична оптимізація для SEO.
* **Next Themes:** Next Themes - це офіційна бібліотека тем для Next.js. Вона дозволяє вам легко створювати та налаштовувати теми для ваших веб-застосунків Next.js.
* **React:** React - це JavaScript-бібліотека для декларативного створення інтерфейсів користувача. Вона пропонує простий та зручний API для створення компонентів, які можна використовувати для побудови складних інтерфейсів користувача.
* **React Hook Form:** React Hook Form - це бібліотека React для створення форм з підтримкою валідації та управління станом. Вона є альтернативою Hook Form, що згадувалась раніше, і пропонує схожий функціонал.
* **React Textarea Autosize:** React Textarea Autosize - це бібліотека React для автоматичного регулювання висоти текстової області (textarea) в залежності від введеного контенту. Вона робить роботу з текстовими полями більш зручною для користувачів.
* **Redis:** Redis - це сховище даних типу “key-value”, яке часто використовується для кешування даних та повідомлень. Воно може покращити продуктивність вашої програми шляхом зменшення кількості запитів до бази даних.
* **Server Only:** Server Only - це бібліотека React для позначення компонентів, які повинні бути рендеровані лише на сервері. Це корисно для компонентів, які потребують доступу до даних сервера або виконують дії на сервері перед відображенням на клієнті.
* **Zod:** Zod - це бібліотека TypeScript для валідації даних. Вона дозволяє вам легко визначати схеми даних та перевіряти, чи дані відповідають цим схемам.
* **Autoprefixer:** Autoprefixer - це інструмент для автоматичного додавання вендорних префіксів до CSS-коду. Він забезпечує сумісність вашого CSS-коду з різними браузерами.
* **Eslint:** Eslint - це інструмент для статичного аналізу коду JavaScript. Він допомагає вам писати чистий та лаконічний код, а також виявляти потенційні помилки.
* **PostCSS:** PostCSS - це інструмент для трансформації CSS-коду. Він дозволяє вам використовувати сучасні функції CSS, такі як змінні та міксини, і потім трансформувати їх у код, який сумісний зі старішими браузерами.
* **Prettier:** Prettier - це інструмент для форматування коду JavaScript. Він допомагає вам підтримувати узгоджений стиль кодування у вашому проекті.
* **Tailwind CSS:** Tailwind CSS - це бібліотека утилітних класів CSS. Вона дозволяє вам швидко створювати інтерфейси користувача за допомогою класів, які вже визначені в бібліотеці.
* **Typescript:** Typescript - це “надмножина” JavaScript, яка додає статичну типізацію до мови. Це означає, що ви можете явно вказувати типи змінних, функцій та інших елементів коду, що робить його більш надійним та легким у підтримці.

# Розділ 6. Опис контрольного прикладу

Цей розділ містить вихідні дані та їх аналіз; дані, одержані в результаті роботи програми, результати роботи програми у вигляді роздруківок вмісту екрану, вікон тощо.

## 6.1 Користування веб-форумом (Frontend частина)

### Головна сторінка

A screenshot of a computer

Description automatically generated

На головній сторінці програми представлено стрічку з темами користувачів. Кожна тема містить заголовок, опис, ім’я автора, час публікації та кількість коментарів.

У верхній частині головної сторінки розташовано дві кнопки:

* “Пошук серед форумів” - відкриває сторінку пошуку, де користувачі можуть шукати теми за ключовими словами.
* “Увійти” - відкриває сторінку авторизації, де користувачі можуть увійти до свого акаунту або створити новий.

Також можна побачити повідомлення про отримання персоналізованої сторінки, після авторизації.

### Авторизація

A screenshot of a computer

Description automatically generated

Основні деталі:

* Веб-сторінка має простий і зрозумілий дизайн.
* Інструкції щодо входу чітко написані.
* Форма входу компактна та зручна для використання.
* Посилання на реєстрацію легко знайти.
* Логотип Google дає зрозуміти, що веб-форум використовує технології Google.

### Перегляд підфоруму

A screenshot of a computer

Description automatically generated

Сторінка підфоруму нагадує головну сторінку: представлено стрічку з темами користувачів. Кожна тема містить заголовок, опис, ім’я автора, час публікації та кількість коментарів. Але стрічка має теми тільки зі свого сабфоруму. Окрім цього можна побачити панель керування для Модераторів та Адміністраторів, а також кнопки “Приєднатися” та “Створити тему”.

### Створення теми

A screenshot of a computer

Description automatically generated

Основні деталі:

* Веб-сторінка має простий і зрозумілий дизайн.
* Форма написання тексту компактна та зручна для використання.
* Знак “+” дає чіткий опис розширеного функціоналу редактору WSWYG
* Розширений функціонал редактору містить:
  + Звичайний текст
  + Заголовок
  + Посилання на інтернет сторінку
  + Зображення
  + Список
  + Блок коду

### Перегляд, створення коментар

A screenshot of a video game

Description automatically generated

Під кожою темою можна переглядати, створювати та реагувати на коментарі.

Основні деталі:

* Веб-сторінка має простий і зрозумілий дизайн.
* Функціонал коментарів чітко пояснено знаками “⬆”, “⬇” та “🗨”
* Зміщення коментарів чітко пояснює який коментар є відповіддю на інший коментар
* Кількість голосів зручно розташовано між кнопками голосів

## 6.2 Огляд бази даних активного форуму

### Таблиця Account

A screenshot of a computer

Description automatically generated

### Таблиця Comment

A screenshot of a computer

Description automatically generated

### Таблиця CommentVote

A screenshot of a computer

Description automatically generated

### Таблиця Subforum

A screenshot of a computer

Description automatically generated

### Таблиця Subsctibtion

A screenshot of a computer

Description automatically generated

### Таблиця Thread

A screenshot of a computer

Description automatically generated

### Таблиця User

A screenshot of a computer

Description automatically generated

### Таблиця Vote

A screenshot of a computer

Description automatically generated

# ЛІТЕРАТУРА

1. МЕТОДИ АНАЛІЗУ ЕФЕКТИВНОСТІ ВЕБ-ФОРУМІВ: Ю.О. Сєров, Р.Б. Кравець, А.М. Пелещишин, 2009
2. Reddit для медіа: як, навіщо та кому це треба: Тартачний Олександр, 2023
3. Refactoring UI: Adam Wathan, Steve Schoger, 2019

# СПИСОК ДЖЕРЕЛ

1. <https://nextjs.org/docs/getting-started/project-structure>
2. <https://www.typescriptlang.org/>
3. <https://www.prisma.io/>
4. <https://www.postgresql.org/>
5. <https://redis.io/>
6. <https://react.dev/>
7. <https://legacy.reactjs.org/>
8. <https://editorjs.io/>
9. <https://react-hook-form.com/>
10. <https://mantine.dev/>
11. <https://next-auth.js.org/>
12. <https://www.radix-ui.com/>
13. <https://ui.shadcn.com/>
14. <https://tanstack.com/query/latest>
15. <https://tanstack.com/form/latest>
16. <https://uploadthing.com/>
17. <https://axios-http.com/docs/intro>
18. <https://cheerio.js.org/>
19. <https://lucide.dev/guide/packages/lucide-react>
20. <https://nextjs.org/docs/app/building-your-application/routing>
21. <https://vitejs.dev/>
22. <https://code.visualstudio.com/docs>
23. <https://date-fns.org/>
24. <https://www.npmjs.com/package/clsx>
25. <https://mui.com/material-ui/react-textarea-autosize/>
26. <https://github.com/pacocoursey/next-themes>
27. <https://zod.dev/>
28. <https://github.com/postcss/autoprefixer>
29. <https://eslint.org/>
30. <https://postcss.org/>
31. <https://prettier.io/>
32. <https://tailwindcss.com/>
33. <https://github.com/>
34. <https://swc.rs/>
35. <https://vercel.com/docs>
36. <https://www.prisma.io/react-server-components>
37. <https://github.com/ploxxxy/match-game>
38. <https://docs.npmjs.com/>
39. <https://nextjs-forum.com/>
40. <https://nextjs.org/docs/community>