

Специфікація вимог до програмного забезпечення (SRS)

1. Вступ

1.1 Призначення

Цей документ визначає функціональні та нефункціональні вимоги до веб-системи управління розкладом навчального закладу. Система призначена для автоматизації процесу створення, управління та оновлення розкладу занять. Вона дозволяє адміністраторам легко планувати розклад, викладачам – переглядати та коригувати свої заняття, а студентам – отримувати актуальну інформацію про навчальний процес.

1.2 Область застосування

Система буде використовуватись:

- Адміністрацією навчального закладу для управління розкладом.
- Викладачами для перегляду та редагування розкладу занять.
- Студентами для отримання актуального розкладу.

Система має бути доступною через веб-браузер і мобільні пристрої. Вона буде інтегрована з внутрішніми інформаційними системами навчального закладу для обміну даними.

1.3 Визначення, аббревіатури та аббревіатури

SRS - Специфікація вимог до програмного забезпечення

API - Інтерфейс прикладного програмування

UI - Користувацький інтерфейс

DB - База даних

1.4 Посилання

<https://github.com/orgs/pavelko-pletelia/repositories>

2. Загальний опис продукту

2.1 Перспективи продукту

Ця веб-система розробляється з метою покращення процесу управління розкладом у навчальному закладі. Традиційні методи створення та розповсюдження розкладу часто пов'язані з ручною роботою, помилками та затримками. Це призводить до незручностей для студентів, викладачів та адміністрації.

Основні переваги, які надає система:

➤ Автоматизація:

Система автоматизує створення, редагування та публікацію розкладу, зменшуючи ручну роботу та ймовірність помилок.

Автоматичне формування звітів та розсилка повідомлень економить час та ресурси.

➤ Централізоване управління:

Вся інформація про розклад зберігається в єдиній базі даних, що забезпечує її актуальність та доступність.

Адміністратори мають повний контроль над розкладом та користувачами.

➤ Зручний доступ:

Студенти, викладачі та адміністрація можуть отримати доступ до розкладу з будь-якого пристрою, підключеного до Інтернету.

Інтуїтивно зрозумілий інтерфейс забезпечує легкість використання системи.

➤ Підвищення ефективності:

Система допомагає оптимізувати використання аудиторій та ресурсів.

Своєчасне інформування про зміни в розкладі зменшує кількість пропущених занять.

➤ Інтеграція:

Можливість інтеграції з іншими системами навчального закладу (наприклад, системою обліку студентів) дозволяє створити єдиний інформаційний простір.

2.2 Функції продукту

- Створення та редагування розкладу занять.
- Публікація розкладу на веб-сайті навчального закладу.
- Доступ до розкладу для студентів, викладачів та адміністрації.
- Генерація звітів про розклад.
- Управління аудиторіями та викладачами.
- Інтеграція з іншими системами навчального закладу (за потреби).

2.3 Обмеження продукту

- Необхідність підключення до Інтернету
- Обмеження на кількість одночасних користувачів
- Залежність від сторонніх сервісів
- Обмеження авторизації
- Бюджет

2.4. Аудиторія

Адміністратори – створюють та коригують розклад, керують користувачами системи, генерують звіти.

Викладачі – переглядають свій розклад, отримують сповіщення про зміни, редагують інформацію про певні заняття.

Студенти – переглядають розклад своєї групи та отримують оновлення про зміни.

3. Вимоги

3.1 Функціональні вимоги

1. Створення розкладу: Адміністратор повинен мати можливість створювати розклад занять для навчального закладу. Він повинен вказати інформацію про предмет, викладача, групу студентів, дату, час і аудиторію проведення заняття.

2. Перегляд розкладу: Користувачі (студенти, викладачі) повинні мати можливість переглядати свій розклад на певний період.

3. Редагування розкладу (для адміністраторів): Адміністратори повинні мати можливість змінювати розклад (додавати заняття, змінювати час і місце проведення).

4. Видалення заняття з розкладу: Адміністратор повинен мати можливість видаляти заняття з розкладу. Після видалення система має сповістити всіх зацікавлених користувачів про зміни.

5. Аутентифікація: Користувачі повинні бути здатні зареєструватися та увійти в систему за допомогою логіна та пароля.

6. Сповіщення про зміни розкладу: Користувачі повинні отримувати сповіщення (через email) про будь-які зміни в розкладі.

7. Додавання додаткових заміток: Викладачі повинні мати можливість додавати примітки до конкретних занять для конкретних груп (наприклад, вказівки або особливі вимоги).

8. Можливість фільтрації: Користувачі повинні мати можливість фільтрувати розклад за викладачем, групою, навчальним предметом.

9. Автоматичне оновлення розкладу: Система повинна автоматично оновлювати розклад у разі зміни.

10. Звіти та аналітика: Адміністратори повинні мати можливість генерувати звіти щодо розкладу викладачів, використання аудиторій.

3.2 Нефункціональні вимоги

1. Продуктивність: Система повинна забезпечувати швидку обробку запитів, оновлення розкладу не повинно займати більше 2 секунд.

2. Доступність: Система повинна бути доступною 24/7 з мінімальним часом простою не більше 1% на місяць.

3. Безпека: Доступ до даних має бути захищений аутентифікацією та авторизацією. Паролі користувачів повинні зберігатися в зашифрованому вигляді. Передача даних повинна здійснюватися через HTTPS.

4. Масштабованість: Система повинна бути здатна масштабуватися для підтримки додаткових функцій та користувачів (як для малих, так і великих навчальних закладів).

5. Зручність користування: Інтерфейс системи має бути інтуїтивно зрозумілим, а новий користувач повинен освоювати базові функції за 5 хвилин.

6. Кросплатформеність: Система повинна підтримувати різні браузеры та мобільні платформи (Android, iOS).

3.3 Архітектура високого рівня

Веб-система управління розкладом для навчального закладу буде побудована за архітектурою клієнт-сервер, де сервер обробляє запити від клієнтів. Архітектура включає:

1. Клієнтський рівень: веб-браузер або мобільний додаток, через який користувачі взаємодіють із системою. Клієнтський додаток здійснює запити до сервера для отримання даних про розклад.

2. Серверний рівень: сервер (наприклад, Node.js або Python/Django), який обробляє запити, взаємодіє з базою даних і повертає результати на клієнтський рівень.

3. База даних: PostgreSQL для зберігання розкладів, інформації про викладачів, студентів, аудиторії, примітки та інших даних.

4. Система аутентифікації та авторизації: механізм для забезпечення безпеки даних, перевірка прав доступу для різних ролей користувачів (студент, викладач, адміністратор).

5. Інтерфейси API: для забезпечення взаємодії з іншими системами навчального закладу (наприклад, електронний журнал).

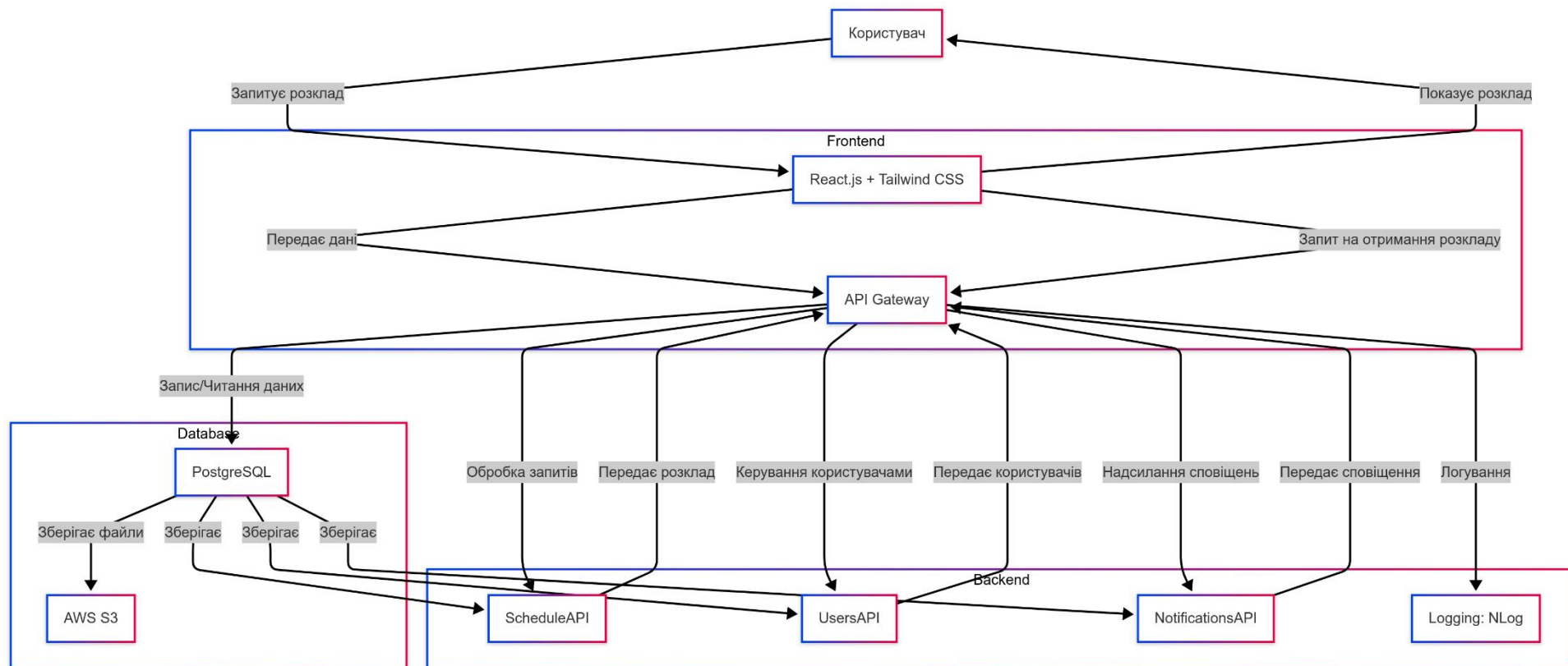
3.4. Архитектура

1. Фронтенд:

- **React.js** – JavaScript-бібліотека для створення інтерфейсу користувача. Вона забезпечує швидку взаємодію без перезавантаження сторінки.
- **Tailwind CSS** – фреймворк для стилізації, що дозволяє створювати адаптивні інтерфейси з мінімумом кастомного CSS.

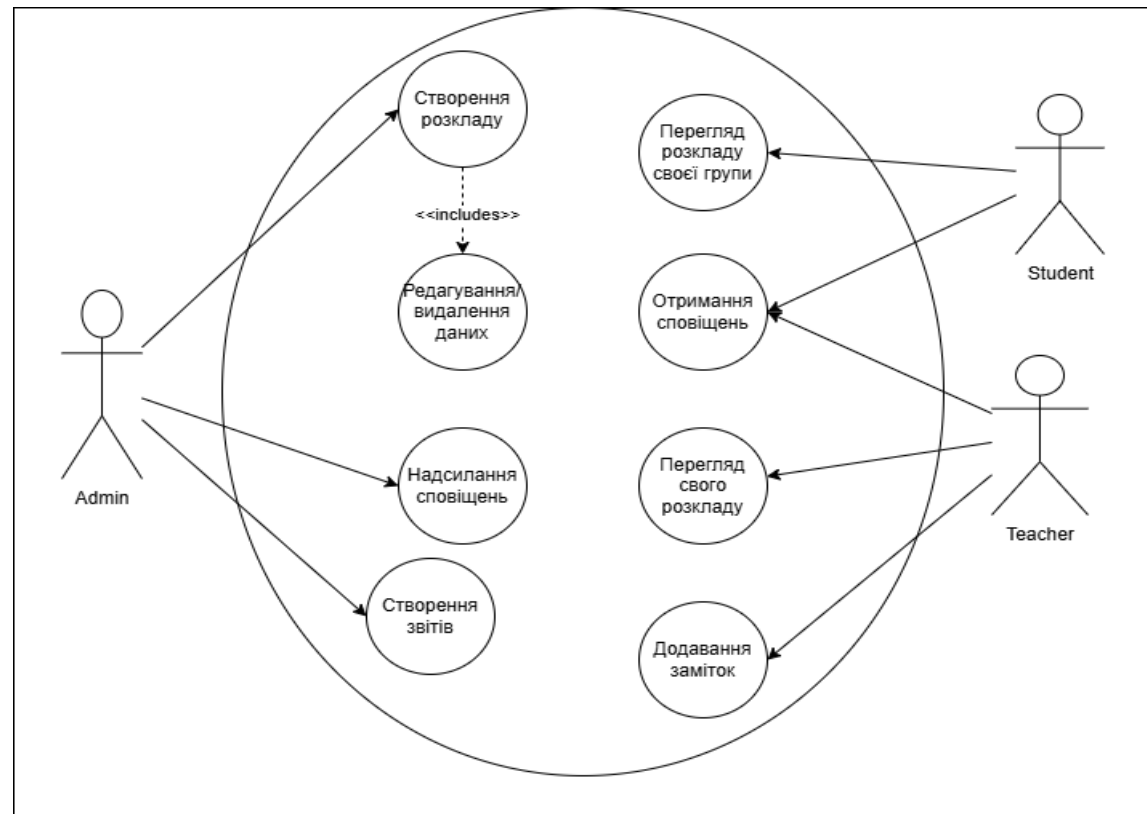
2. Бекенд:

- **Node.js (NestJS)** – серверний фреймворк, який використовує архітектуру REST або GraphQL API для обробки запитів. NestJS дозволяє створювати модульну і масштабовану систему.
- **TypeScript** – використовується для забезпечення строгої типізації та покращення якості коду.
- **PostgreSQL** – основна реляційна база даних, яка зберігає інформацію про користувачів, розклади та повідомлення.
- **AWS S3** – хмарне сховище для файлів (наприклад, додаткові матеріали, збережені користувачами).

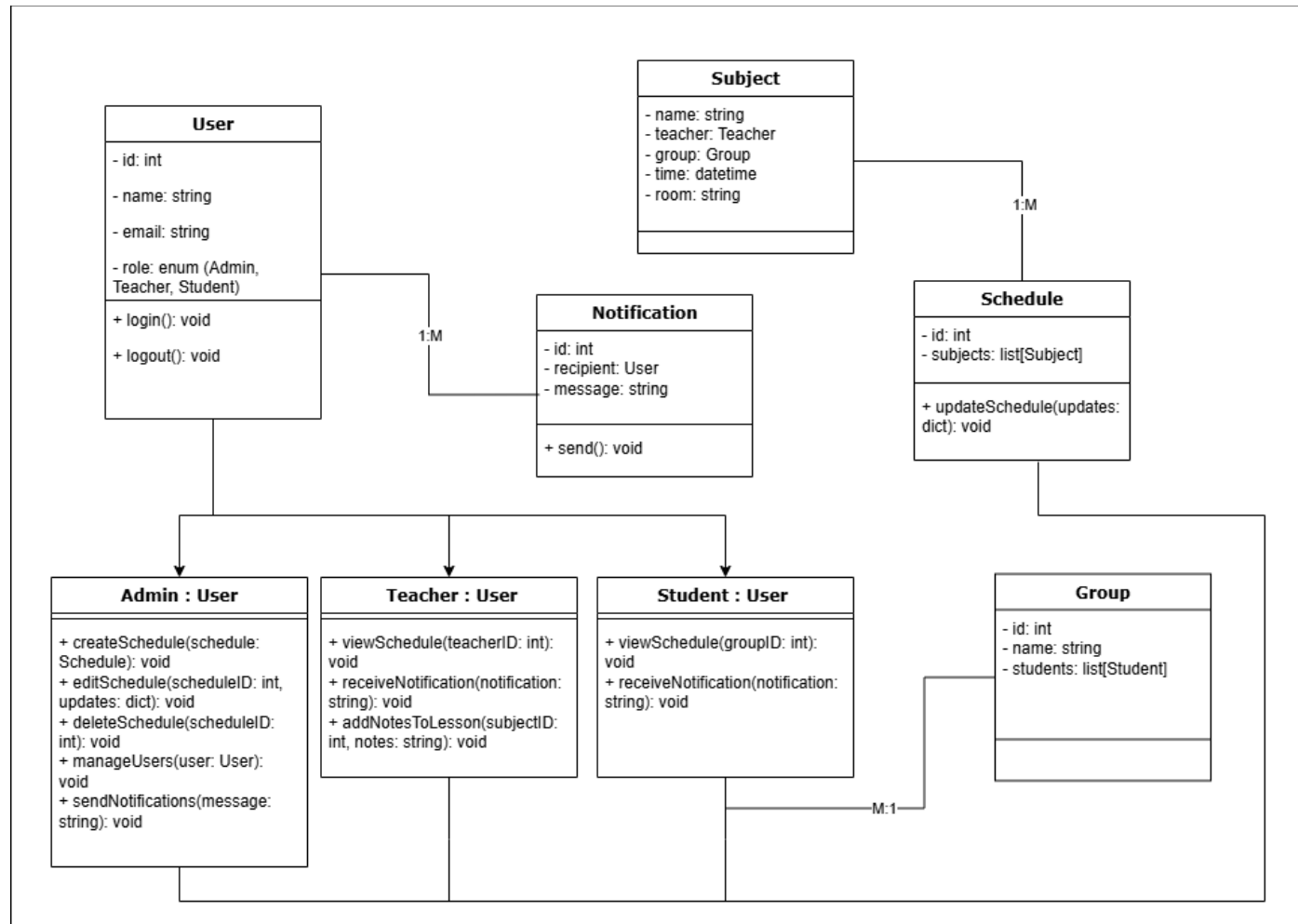


4. Concurrency Patterns usage

4.1 UML

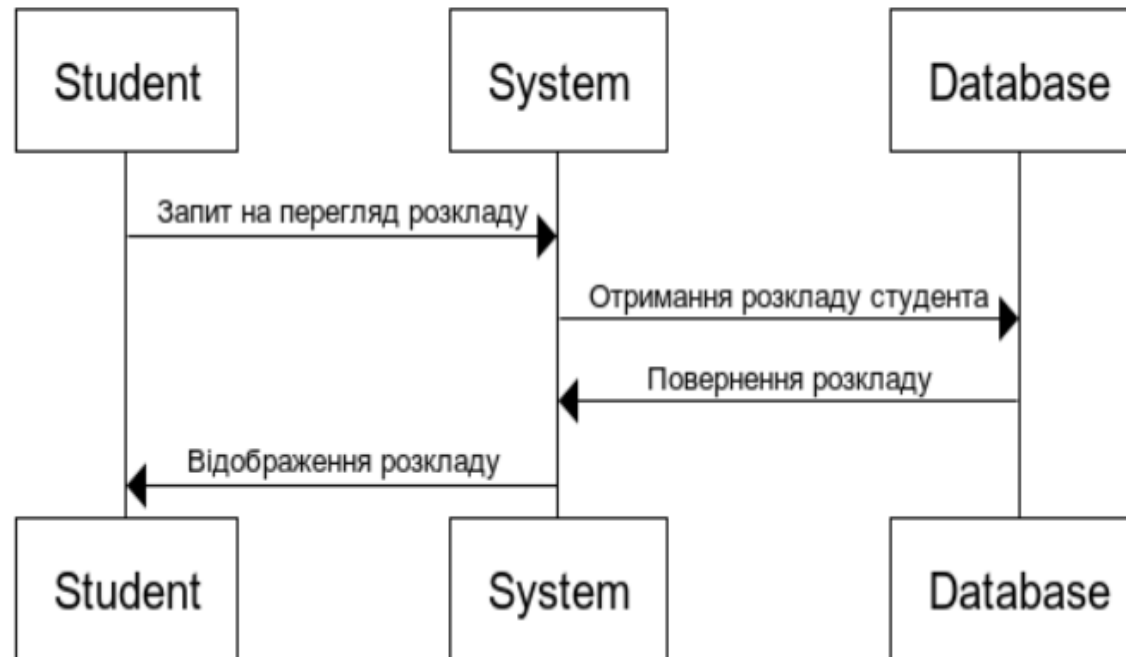


4.2 Class Diagram

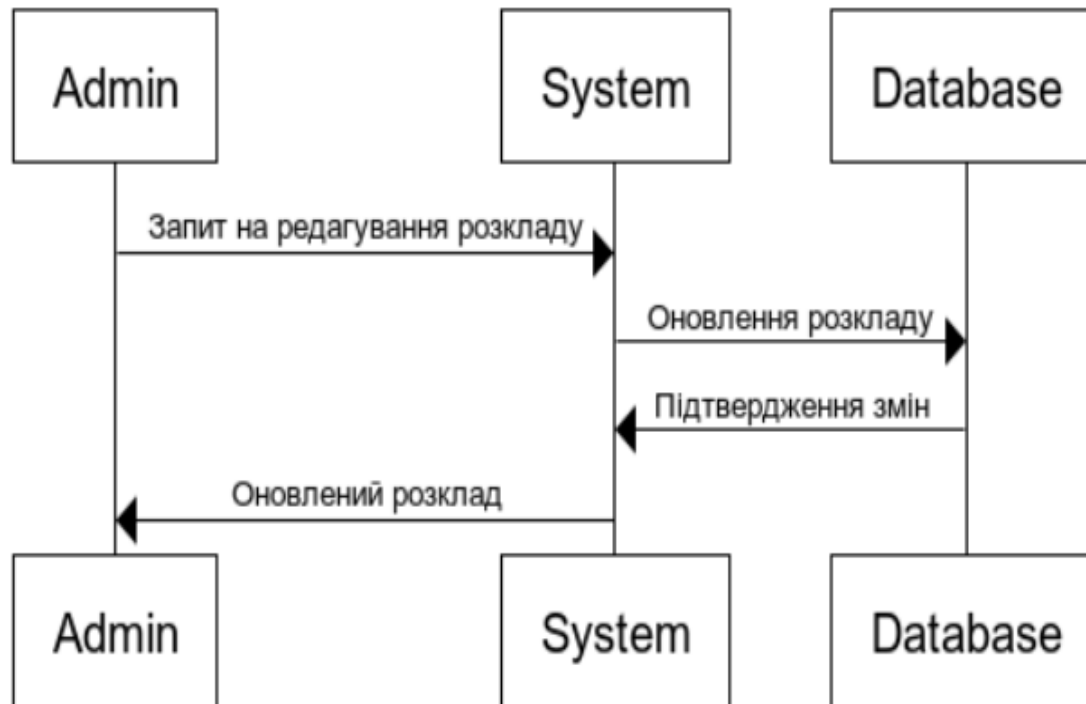


4.3 Sequence diagrams

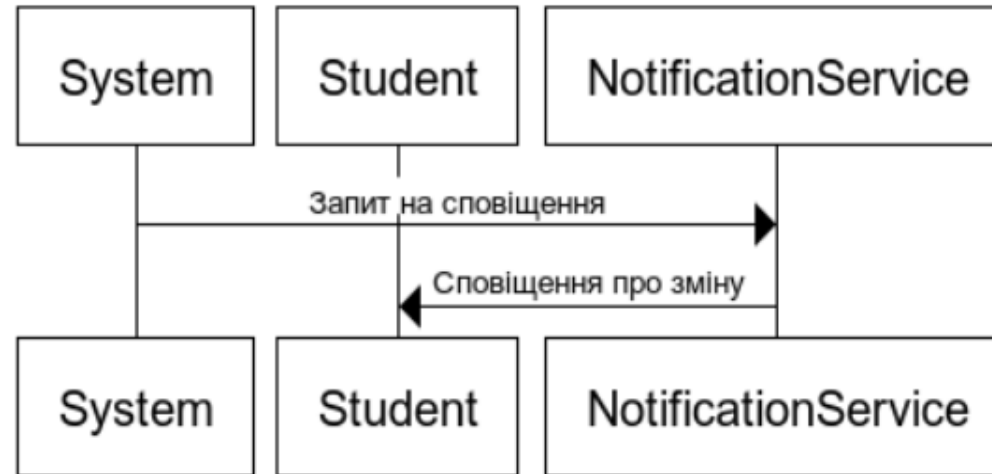
Перегляд розкладу студентом



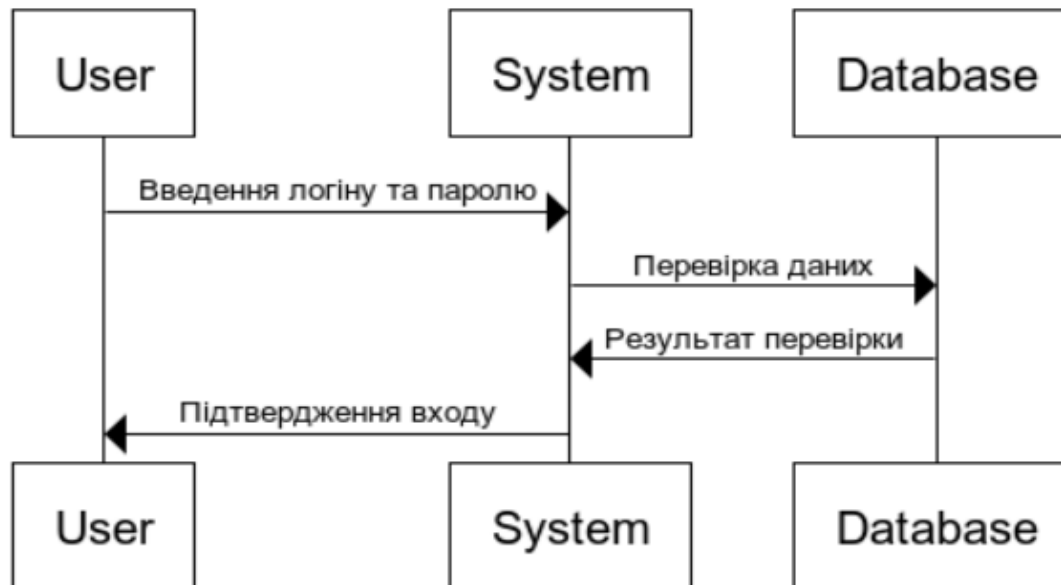
Редагування розкладу адміністратором



Сповіщення студенту про зміну в розкладі



Аутентифікація користувача



Генерація звіту адміністратором

