Суть застосунку (коротко): Створити рейтингову систему для розробників. Відомо, що їм часто доводиться мати справу із чужим кодом. Застосунок відображає хто саме писав(редагував) цю частину коду, дозволяє оцінити її, додати коментар(видимий тільки автору коду). Для кожного розробника коментарі окремо групуються для кращого засвоєння. Також відображається рейтинг розробників, статистика(для всіх проєктів чи тільки поточного), скільки всього було написано, відредаговано коду, коментарів, в яких проєктах у розробника хороша, в яких погана оцінка.

Solution Requirements

Функціональні вимоги (Functional Requirements FR):

1. Аутентифікація та авторизація:

1.1 Користувачі можуть створювати облікові записи та авторизовуватись в системі. (Приклад у формі User story: Як *користувач*, я хочу мати можливість *створювати обліковий запис*, щоб *я міг авторизовуватись у системі*.)

1.2 Користувачі можуть управляти своїми профілями і змінювати особисту інформацію.(У формі User-story: Як *користувач*, я хочу *мати можливість змінювати свою особисту інформацію та опції профілю*, щоб мати можливість *керувати своїм профілем*.)

2. Оцінка та коментування коду:

2.1 Користувачі можуть оцінювати частини коду.

2.2 Користувачі можуть залишати коментарі до конкретних рядків або блоків коду.

3. Рейтингова система:

3.1 Додаток обчислює рейтинг розробників на основі отриманих оцінок.

3.2 Користувачі можуть переглядати рейтинги розробників та їх статистику.

4. Статистика проєктів та активності:

4.1 Додаток збирає та відображає статистичні дані про активність розробників, такі як кількість написаного та відредагованого коду.

4.2 Користувачі можуть переглядати статистику проєктів, включаючи якість коду та кількість оцінок та коментарів.

5. Управління користувачами та чатами:

5.1 Власники чатів можуть створювати та керувати чатами, включаючи налаштування та додавання/видалення учасників.

Нефункціональні вимоги (Non-Functional Requirements NFR):

1. Надійність (Reliability):

1.1 Середній час безвідмовної роботи системи повинен бути не менше 99,9%.

1.2 Система повинна автоматично відновлюватися після помилки протягом 5 секунд.

1.3 Максимальний час відновлення системи після серйозної помилки повинен бути не більше 1 години.

2. Продуктивність (Performance):

1.1 Середній час відповіді на запити користувачів повинен бути менше 500 мілісекунд.

1.2 Час виконання операцій повинен бути менше 2 секунд.

1.3 Система повинна забезпечувати обробку не менше 200 запитів на секунду.

3. Безпека (Security):

3.1 Доступ до системи повинен бути захищений паролем і двофакторною аутентифікацією.

3.2 Всі комунікації між клієнтом і сервером повинні бути шифровані з використанням протоколу TLS/SSL.

3.3 Доступ до конфіденційної інформації повинен бути обмежений і контрольований з використанням рівнів доступу та ролей.

3.4 Використання протоколу HTTPS для захищеної передачі даних між клієнтом та сервером.

3.5 Зберігання паролів користувачів у захешованому форматі з використанням солі.

4. Легкість використання (Usability):

4.1 Інтерфейс користувача повинен бути інтуїтивно зрозумілим для нових користувачів.

4.2 Необхідна документація повинна бути надана у вигляді докладних посібників та відеоуроків.

5. Масштабування (Scalability):

5.1 Система повинна бути готовою до масштабування для обробки зростаючого обсягу даних та користувачів.

5.2 Під час масштабування системи не повинно виникати падіння продуктивності більш ніж на 50%.

6. Доступність (Availability):

6.1 Зміни в системі повинні виконуватись без збоїв у роботі існуючих функцій.

6.2 Час відновлення після випадку неполадок повинен бути не більше 30 хвилин.

7. Ремонтопридатність (Maintainability)

7.1 Автоматичне резервне копіювання бази даних щоденно на окремий сервер.

7.2 Збереження резервних копій даних протягом не менше 30 днів.

Обмеження (Constraints):

1. Використання стеку LAMP при побудові веб-версії сервісу (обслуговуючий персонал не володіє іншими засобами).
2. Система повинна бути розроблена до 27 липня 2024 року (розробка робиться для презентації на виставці у цю дату).

(*Додаткові приклади: Використання бази даних IBM DB2, що впроваджена у організації; Використання зовнішньої платіжної системи, яка використовується для обробки платежів організації тощо*).