ДОДАТОК В

Специфікація програмного забезпечення

1. Специфікація вимог користувачів

Веб-додаток Quizzy розробляється як інструмент для створення та проходження онлайн-вікторин, призначений для використання в освіті, розвагах і корпоративному навчанні. Користувачі, включаючи вчителів, учнів, корпоративних тренерів і любителів інтерактивних ігор, очікують від системи простоту у створенні власних вікторин, зручний доступ до них у реальному часі та захист своїх персональних даних. Система має бути інтуїтивно зрозумілою для людей із різним рівнем технічної підготовки, дозволяючи створювати вікторини без глибоких знань програмування. Важливо, щоб додаток забезпечував швидке завантаження даних, наприклад, списків вікторин чи результатів, навіть при одночасному підключенні великої кількості користувачів. Користувачі також вимагають безпечного входу в систему з можливістю відновлення доступу та захисту від несанкціонованого доступу до їхніх даних, таких як імена, електронні адреси та результати проходження. Додатково передбачається можливість інтеграції з іншими платформами, що дозволить використовувати Quizzy у складі більших навчальних систем. Очікується, що система буде доступна через веб-інтерфейс і зможе працювати стабільно на різних пристроях, включаючи комп’ютери та смартфони.

2. Специфікація системних вимог

Quizzy реалізується як клієнт-серверна система з розподіленими компонентами, що включають серверну частину, базу даних і фронтенд, розроблений іншою особою. Серверна інфраструктура базується на хмарній платформі Render, яка забезпечує автоматичне масштабування та розгортання. Система має бути здатною обробляти до 200 одночасних користувачів із середнім часом відповіді 50 мілісекунд, а при зростанні навантаження — адаптуватися до додаткових ресурсів. База даних MongoDB, розміщена на MongoDB Atlas, використовується для зберігання даних користувачів, вікторин і результатів із підтримкою індексації ключових полів для оптимізації запитів. Система включає RESTful API як інтерфейс для взаємодії між сервером і фронтендом, із забезпеченням стандартних HTTP-методів (GET, POST, PUT, DELETE) і формату відповідей JSON. Безпека системи підтримується через шифрування даних за допомогою HTTPS, хешування паролів із бібліотекою bcrypt і автентифікацію за допомогою JWT через Passport.js. Система має бути стійкою до помилок, включаючи обробку некоректних даних і серверних збоїв, із глобальним обробником помилок, що повертає статуси 400 або 500. Для моніторингу працездатності передбачено інтеграцію з логами Render, а для майбутнього розвитку планується додавання кешування через Redis для зменшення навантаження на базу даних.

3. Специфікація програмних вимог

3.1 Функціональні вимоги

Серверна частина Quizzy виконує наступні функції, узгоджені між розробником і потенційними користувачами:

* обробка HTTP-запитів через маршрути, такі як /api/quizzes для створення, редагування та видалення вікторин, і /api/users для управління обліковими записами.
* реалізація асинхронних контролерів для бізнес-логіки, включаючи створення вікторин, реєстрацію користувачів і перевірку відповідей.
* забезпечення валідації вхідних даних, наприклад, перевірки формату email і мінімальної довжини пароля (8 символів).
* підтримка автентифікації з генерацією JWT-токенів після успішного логіну та їхньою перевіркою на захищених маршрутах.
* зберігання даних у MongoDB із схемами для колекцій User (id, username, password, email, createdAt) і Quiz (id, title, questions, creatorId, createdAt, updatedAt).
* індексація полів id, creatorId і title у MongoDB для прискорення пошуку даних.
* документування API через Swagger із чітким описом ендпойнтів і форматів відповідей.
* депloyment на Render із використанням Dockerfile і змінних середовища (MONGO\_URI, JWT\_SECRET, PORT).

Функції, які не підтримуються:

* локальне кешування даних на стороні сервера (планується додавання Redis у майбутньому).
* автоматичне резервне копіювання даних (поточна версія залежить від базового бекапу MongoDB Atlas).
* обробка офлайн-режиму, оскільки система повністю орієнтована на онлайн-доступ.

3.2 Вимоги до якості

* продуктивність: сервер має витримувати до 150 запитів за секунду з середнім часом відповіді 50 мілісекунд, що підтверджено тестами за допомогою Locust.
* надійність: система стабільно обробляє помилки, повертаючи відповідні статуси (400 для некоректних даних, 500 для серверних збоїв), із витривалістю до 80% навантаження.
* безпека: захист від XSS-атак забезпечується через валідацію вхідних даних, а паролі хешуються bcrypt; перевірка OWASP ZAP не виявила критичних уразливостей.
* масштабованість: поточна конфігурація підтримує до 200 користувачів, із можливістю розширення через оновлення плану Render.

3.3 Процедури перевірки

* тестування продуктивності за допомогою Locust із симуляцією 150–200 запитів за секунду.
* ручне тестування API через Postman для перевірки ендпойнтів /api/quizzes і /api/users.
* аналіз вразливостей за допомогою OWASP ZAP для виявлення потенційних загроз.
* моніторинг логів Render після деплою протягом тижня для оцінки стабільності.

3.4 Використані програмні засоби та технології

* мова програмування: JavaScript (Node.js).
* серверний фреймворк: Express.js для створення маршрутів і middleware.
* база даних: MongoDB із драйвером Mongoose для моделювання даних.
* автентифікація: Passport.js із підтримкою JWT.
* платформа деплою: Render із підтримкою Docker.
* інструменти тестування: Jest для модульного тестування, Locust для тестування навантаження, Postman для ручного тестування.
* документація: Swagger для API-документації.
* усі матеріали ККП виконуються українською мовою.

3.5 Обмеження та застереження

* система не підтримує більше 200 одночасних користувачів без оновлення тарифного плану Render.
* відсутність кешування може призвести до уповільнення при великій кількості запитів (час відповіді до 80 мс при 150 запитах/с).
* деталізація помилок у глобальному обробнику обмежена для безпеки, що ускладнює дебагінг.
* резервне копіювання залежить від базового механізму MongoDB Atlas і не налаштовано вручну.