Специфікація вимог до програмного забезпечення для додатку «Go ORM SQL Efficiency»

Затверджено версією 1.0

Підготовлено Ягнюковим Андрієм Юрійовичем

Харківський національний університет радіоелектроніки

10.06.2025

1. Вступ

1.1. Мета

Цей документ визначає функціональні та нефункціональні вимоги до додатку «Go ORM SQL Efficiency».

1.2. Умовні позначення

Ключові компоненти виділяються жирним, приклади та змінні – курсивом, назви структур, функцій і змінних – у стилі CamelCase.

1.3. Цільова аудиторія

* розробники Go.

1.4. Область проєкту

Додаток призначено для порівняльного дослідження ефективності підходів доступу до бази даних у Go-додатках: **GORM**, **Ent** та **чистий SQL (database/sql)**. Для кожного підходу реалізуються однакові запити, що дозволяє об’єктивно порівняти їхню продуктивність та зручність використання.

1.5. Посилання

* IEEE Std 830-1998;
* ISO/IEC 25010:2011.

2. Загальний опис

2.1. Перспектива продукту

Додаток є консольною утилітою для вимірювання продуктивності різних підходів до доступу до СУБД PostgreSQL у Go.

2.2. Основні функції

* створення нового продукту;
* отримання інформації про користувача за ідентифікаторам;
* оновлення ціни продукту за ідентифікаторам;
* видалення продукту за назвою;
* створення замовлення з товарами для конкретного користувача (транзакція);
* отримання статистики по замовленнях та витратах клієнта (агрегація);
* отримання інформації про продажі топ продуктів (складний JOIN);
* вимірювання середньої затримки, 95-й перцентиль затримки, пропускної здатності, середнього споживання оперативної пам’яті, часу пауз GC;
* вивід результатів у консоль у табличному вигляді.

2.3. Користувачі

Усі користувачі мають однаковий функціонал.

2.4. Оточення

* Go: >=1.24;
* СУБД: PostgreSQL;
* ОС: Linux, Windows, macOS.

2.5. Обмеження

* підтримка лише PostgreSQL;
* відсутність збереження або генерації результатів у файли;
* відсутній графічний інтерфейс (тільки CLI).

2.6. Документація

* README у репозиторії;
* Inline-коментарі коду.

2.7. Припущення

* користувач має досвід розробки на Go;
* налаштовано підключення до PostgreSQL.

3. Зовнішні інтерфейси

3.1. Інтерфейс користувача

* CLI-інтерфейс: запуск benchmark-сценаріїв;
* вивід результатів у консоль у зручному табличному вигляді.

3.2. Апаратні інтерфейси

* комп’ютер.

3.3. Програмні інтерфейси

* взаємодія з PostgreSQL через Docker-контейнер.

3.4. Комунікаційні інтерфейси

* локальне підключення до СУБД PostgreSQL.

4. Особливості системи

4.1. Модулі доступу до БД

* **GORM**: реалізація стандартних запитів;
* **Ent**: аналогічна реалізація;
* **SQL (database/sql)**: ті ж самі сценарії.

4.2. Проведення тестів

* запуск кожного підходу з однаковою кількістю ітерацій;
* фіксація метрик: середня затримка, 95-й перцентиль затримки, пропускна здатність, середнє споживання оперативної пам’яті, час пауз GC.

4.3. Вивід результатів

* таблиця у консоль.

4.4. Налаштування

* можливість налаштувати кількість ітерацій;
* конфігурація параметрів підключення до PostgreSQL.

5. Нефункціональні вимоги

5.1. Продуктивність

* точний вимір часу виконання та ресурсів для кожного підходу.

5.2. Безпека

* відсутність збереження чутливих даних;
* параметри підключення до БД не логуються.

5.3. Якість

* відповідність Go-стилю;
* документовані публічні API.

5.4. Надійність

* коректна обробка помилок підключення та виконання;
* вивід помилок у консоль.

5.5. Гнучкість

* легке додавання нових сценаріїв/запитів;
* додавання підтримки інших ORM у майбутньому.

Appendix A: Глосарій

* ORM – Object-Relational Mapping, технологія для роботи з БД через об'єкти;
* GORM та Ent – популярні ORM для Go;
* database/sql – стандартна бібліотека Go для роботи з SQL.

Appendix B: Моделі аналізу

* діаграма пакетів (див. додатки до дипломної роботи);
* ER-діаграма (див. додаток до дипломної роботи).

Appendix C: TBD

* додавання підтримки інших ORM;
* можливість генерації графіків за межами застосунку.