УДК 004.774

РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ ВЕДЕНИЯ ФИНАНСОВОЙ СТАТИСТИКИ

С. Ю. Борисов

Белорусский государственный университет,
пр. Независимости, 4, 220030, г. Минск, Беларусь, mmf.borisovsy@bsu.by Научный руководитель: Е. В. Кремень, кандидат физико-математических наук, доцент

Разработано веб-приложение для ведение финансовой статистики. Для оптимизации работы клиентского приложения использована технология серверного рендеринга. Для ускорения выбора данных за определенный период времени использовано секционирование (partitioning). Дополнительно, во избежание дополнительных повторных расчетов, используется система кеширования на основе Redis.

Ключевые слова: серверный рендеринг; секционирование данных; система кеширования; финансовая аналитика.

Зачастую человеку нужен удобный инструмент для ведения своей финансовой статистики: расходов, доходов. Кому-то достаточно Excel-таблицы, а кому-то неудобно настраивать все, и он хочет видеть отчеты за долгое время в виде интерактивных графиков. По сути, приложений, предоставляющих такое достаточно много, однако им свойственно наличие различного рода рекламы и т. п.

После регистрации и авторизации в приложении пользователю предоставляется возможность записывать свои расходы/доходы (транзакции), заполняя всю необходимую информацию: валюту (реализована поддержка различных валют с использованием обменного курса на текущий момент времени), описание траты/дохода, категории траты/дохода и т. п. После пользователь может просматривать историю своих транзакций, обобщенную по периоду статистику в виде графика и дополнительных плиток, удалять транзакции.

Дополнительной возможностью является возможность использования событий для автоматического создания транзакций. Так, например, можно создать автоматический расход, например, на оплату какой-то ежемесячной (ежегодной, ежедневной) услуги, и в выбранный период транзакция будет создана в автоматическом режиме. Это позволяет не тратить лишнее время на рутинное заполнение информации о расходе, например, каждый день.

Для создания клиентской части приложения были выбраны React, Next.JS и Zustand для максимальной скорости работы, отзывчивости интерфейсов и удобства разработки и поддержки приложения. Использование Next.JS позволяет использовать механизм серверного рендеринга (а также статической генерации страниц), которое ускоряет загрузку приложения.

Для создания серверной части приложения была выбрана программная платформа Node.js, в качестве фреймворка для разработки сервера – Nest.js, в качестве базы данных – PostgreSQL. Кроме того, для обеспечения кеширования использована база данных «ключ-значение» Redis.

Приложение использует сторонний API (Национальный банк Республики Беларусь) для получения курсов валют, поэтому для создания расходов может быть использованы различные валюты: BYN, USD, EUR, что может быть полезно при учете в расходах, например, подписок на различные сервисы.

Стоит отметить механизм, использованный для обеспечения скорости выборок данных за различные периоды: в приложении можно выбрать произвольный период. Секционирование – метод разделения больших таблиц на много маленьких, происходящий прозрачным для приложения способом. В данном случае применено горизонтальное секционирование по дате. В

задачах данного приложения был использован подход с добавлением в таблицу базы данных дополнительного числового поля, которое будет выражать период, когда эта запись создана. Так, например, для даты 10 марта 2024 соответствующее значение столбца будет 20240310. В таком случае запрос к периоду 10 марта 2024 года – 17 марта 2024 года будет включать в себя те записи, где значение колонки (привычное название – partition) будет больше либо равно 20240310 и меньше либо равно 20240317, что гораздо быстрее, чем, например, сравнивать даты. К тому же, такой формат записи даты позволяет, например, проводить группировку: по дню (то есть по колонке partition), по месяцу (то есть по колонке, разделенной на 100), по году (то есть по колонке, разделенной на 10000).

Для сравнения, на базе, в которой есть 1 млн записей за разные даты, двумя разными способами: с помощью встроенного типа datetime и колонки created_at, с помощью дополнительной колонки partition:

1. Группировка по дню, сумма доходов
   a. Колонка created_at: 0.42 с.
   b. Колонка partition: 0.17 с.
2. Группировка по месяцу и году, сумма доходов
   a. Колонка created_at: 0.30 с.
   b. Колонка partition: 0.16 с.
3. Группировка по году, сумма доходов
   a. Колонка created_at: 0.22 с.
   b. Колонка partition: 0.16с.

Как можно понять из примеров производительности, в большинстве примеров добавление секционирования выигрывает по скорости. К тому же, так гораздо проще описывать необходимую выборку, ибо это более декларативно.

Система кеширования в приложении также позволяет грамотно расходовать ресурсы сервера. Например, если человек запросит статистику по заданному периоду, она будет добавлена в Redis, чтобы в течение следующих 30 минут (стандартный интервал) не было необходимости пересчета.

В дальнейшем приложение можно дополнить созданием команд (организаций) для объединения нескольких пользователей приложения (например, небольшой компании) с возможностью настройки доступов для создания доходов/расходов и т. п. Также можно улучшить систему кеширования, например, давние периоды можно не пересчитывать более долгий период времени, так как изменения в них могут случиться только в случае удаления транзакций.

Библиографические ссылки

1. Документация Next.JS [Электронный ресурс] / Vercel. Inc. URL: https://nextjs.org/docs (дата обращения: 05.04.2024).

2. Документация СУБД PostgreSQL [Электронный ресурс] / The PostgreSQL Global Development Group. URL: https://www.postgresql.org/docs (дата обращения: 05.04.2024).

3. Документация Nest.js [Электронный ресурс] / MIT by Kamil Mysliwiec.
URL: https://docs.nestjs.com (дата обращения: 05.04.2024).