УДК 004.415.2

ШАБЛОН ПРОЕКТИРОВАНИЯ ВЕБ-ПРИЛОЖЕНИЙ SHARED MODELS

А. Д. Захаренко

Белорусский государственный университет, Беларусь, Минск, arsen.zaharenko@gmail.com

Цель исследования — раскрыть особенности использования шаблона проектирования веб-приложений Shared Models. В статье дается определение и характеристика данного шаблона. Научная новизна исследования заключается в подходе к решению проблемы консистентности данных в межсервисном взаимодействии. В результате охарактеризован принцип работы шаблона и описан сценарий возможной реализации.

Ключевые слова: шаблон проектирования; межсервисное взаимодействие; асинхронная коммуникация; консистентность данных; распределенные модели; CQRS; Redis.

SHARED MODELS WEB APPLICATION DESIGN PATTERN

A. D. Zaharenko

Belarussian state university, Belarus, Minsk, arsen.zaharenko@gmail.com

The aim of the research is to explore the specifics of using the Shared Models design pattern in web applications. The article provides a definition and characteristics of this pattern. The scientific novelty of the research lies in the approach to solving the problem of data consistency in inter-service interaction. As a result, the working principle of the pattern is described, and a scenario for its possible realization is outlined.

Keywords: design pattern; inter-service interaction; asynchronous communication; data consistency; shared models; CQRS; Redis.

Введение

Шаблоны проектирования используются для решения типовых проблем, возникающих при разработке веб-приложений, а также для повышения их гибкости, масштабируемости и обслуживаемости, поэтому изучение шаблонов проектирования является актуальным.

В случае сложных веб-систем, состоящих из нескольких сервисов, взаимодействующих между собой, консистентность данных является од-

ной из главных проблем. Поскольку не существует четкого и стандартизированного решения для этой проблемы, предлагается рассмотреть применение шаблона *Shared Models* в качестве интересного подхода к ее решению.

Приведенные ниже источники содержат информацию о возможных проблемах при реализации шаблона, преимуществах и недостатках шаблона [1, с. 41], идейных компонентах шаблона [2-5], GitHub-репозиторий с полным примером реализации шаблона [6].

Определение и принцип работы шаблона

Shared Models (распределенные модели) — реализация информационного поля, основанная на моделях ORM, несущих в себе функционал передачи своих объектов на другие сервисы при их создании или изменении.

Распределенные модели представлены внутри специального shared-сабмодуля, который интегрируется в каждый сервис, использующий такие модели. Использование сабмодуля позволяет описать основные свойства моделей, а также базовый принцип передачи объектов в одном месте.

Шаблон *Shared Models* является разновидностью реализации CQRS архитектуры [2], т.к. операции записи и чтения распределенных моделей разделены.

Особенность такой реализации в том, что операция записи провоцирует передачу объекта с текущего сервиса на другие.

Принцип работы:

1. Для передачи данных используется резидентная база данных со своим шаблоном Pub/Sub [3], выступающая в роли брокера сообщений, например Redis [4];

2. Все сервисы имеют доступ к одной и той же базе данных Redis;

3. Каждый сервис, взаимодействующий с распределенными моделями, имеет свою объектно-реляционную базу данных;

4. Все изменения, записанные в базу данных, связанные с распределенными моделями, должны быть представлены в формате JSON и отправлены в Redis-очередь;

5. Все сервисы должны непрерывно прослушивать привязанные к ним Redis-каналы и записывать все изменения объектов распределенных моделей в свою базу данных.

Реализация шаблона

Рассмотрим веб-систему из трех сервисов: для продаж, администрирования и аналитики. На каждом из них есть общая модель заказа (Order), которую необходимо будет синхронизировать, т.е. каждый сервис может принимать новые и измененные заказы из Redis-канала и сохранять в свою базу данных.

Пусть сервис продаж дополнительно имеет возможность создавать новые заказы и добавлять их в Redis-каналы, а сервис администрирования — редактировать заказы и добавлять соответствующие дельты в Redis-каналы. Таким образом, все сервисы по умолчанию являются слушателями и могут иметь дополнительные роли (создатель, редактор).

Модель Order храниться в специальном shared-сабмодуле, т.к. она попадает под понятие распределенной. В проект каждого сервиса добавляется shared-сабмодуль, из которого импортируется распределенная модель для использования в логике соответствующего сервиса.

Реализация базируется на примере для E-Commerce приложения [5]. Помимо Redis, в качестве средств разработки используются язык программирования Python, веб-фреймворк Flask, ORM-библиотека SQLAlchemy, СУБД PostgreSQL.

Заключение

Целесообразность внедрения данного шаблона может быть обусловлена следующими причинами:

- увеличение объема трафика приложения;
- повышение пропускной способности и отказоустойчивости сервисов при высокой посещаемости;
- сокращение несоответствий данных;
- наличие общей модели на сервисах;
- упрощение внедрения общих моделей в будущем.

После проведения нагрузочного тестирования было установлено, что использование данного шаблона приводит к заметному увеличению производительности приложения по сравнению с использованием REST-подхода для синхронизации общих моделей.

Библиографические ссылки

1. Веб-программирование и интернет-технологии (WebConf2024) [Электронный ресурс] : материалы 6-й Междунар. науч.-практ. конф., Минск, 15–16 мая 2024 г. / Белорус. гос. ун-т ; редкол.: И. М. Галкин (гл. ред.) [и др.]. Минск : БГУ, 2024. URL:

https://elib.bsu.by/bitstream/123456789/313416/4/WebConf2024.pdf (date of access: 10.04.2025).

2. CQRS [Electronic resource]. URL: https://martinfowler.com/bliki/CQRS.html. (date of access: 13.03.2025).

3. Publish-subscribe pattern [Electronic resource]. URL: https://en.wikipedia.org/wiki/Publish-subscribe_pattern. (date of access: 13.03.2025).

4. Introduction to Redis [Electronic resource]. URL: https://redis.io/about. (date of access: 13.03.2025).

5. How to Build an E-Commerce App Using Redis with the CQRS Pattern [Electronic resource]. URL: https://redis.io/learn/howtos/solutions/microservices/cqrs. (date of access: 13.03.2025).

6. GitHub repository with Shared Models pattern realization example [Electronic resource]. URL: https://github.com/arsen-zaharenko/shared_models. (date of access: 13.03.2025).