15.11.2023

Zustand

Global state manager для React.

Используется внутри модуля генерации команд.

Причины выбора:

Используемый в клиентской части React Query, заменяющий средства работы с состояниями, не слишком подходит для модуля генерации команд, потому что в клиентской части большая часть state - данные с сервера, которые почти не нужно преобразовывать. В данном модуле же большая часть state генерируется самим модулем и все части состояния сильно связаны друг с другом. Zustand в этом случае решает проблему лишних ререндеров и делает код более читаемым и масштабируемым.

Рассмотренные альтернативы:

Redux - много шаблонного кода и более сложная работа с предоставляемым функционалом. Кажется, что overkill для такого сравнительно небольшого модуля.

MobX - не выбрал по причине отсутствия опыта работы с библиотекой

Strapi Design System

UI-kit для плагинов Strapi.

Используется внутри модуля генерации команд.

Причины выбора:

Предоставляет почти все необходимые для модуля компоненты. Модуль является частью админ-панели, поэтому компоненты, которые strapi использует в других частях админ-панели, выглядят уместно.

Styled Components

CSS-in-JS библиотека для React.

Используется внутри модуля генерации команд.

Причины выбора:

Поддерживается административной панелью strapi без дополнительной настройки, что особенно принципиально учитывая почти полное отсутствие официальной документации про разработку плагинов.

31.10.2023

Redis

In-memory средство для работы с данными формата: ключ-значение. Часто используется для кэширования запросов, здесь - как замена базы данных.

Используется для хранения refresh-токенов на стороне сервера.

Причины выбора:

* Скорость - хочется сделать процесс авторизации быстрым. Здесь скорость достигается тем, что все данные хранятся в памяти, а не на диске.
* Простота - для данной задачи нет необходимости в богатом функционале реляционных бд и их субд. Усложняться задача не будет, поэтому не вижу смысла закладывать более продвинутые средства для неё.
* Встроенная инвалидация значений - удобная возможность удалять токены с истёкшим сроком
* Снапшоты - решают проблемы хранения данных в памяти. В случае перезапуска сервера данные потеряются, но redis предоставляет возможность периодически сохранять все токены. Потеря токенов за последние 1-2 часа не принципиальна, в этом случае пользователям просто придётся ещё раз авторизоваться.

Если у этого же сервера возникнет необходимость иметь более сложную базу данных, возможно, и функционал хранения refresh-токенов переедет на неё.

29.09.23

Strapi

Headless CMS, которая будет использоваться для административной панели.

18.09.23

Frontend

Next.js

Фреймворк, основанный на React, предоставляющий функционал для создания production-ready приложений.

Для меня самым принципиальным моментом было наличие SSR и ISR, которые позволяют SPA стать SEO-friendly. Помимо этого фреймворк предоставляет много вещей, улучшающих DX и UX: специальный обработчик изображений с возможностью пререндера, отправки изображений нужного разрешения (меньшего, чем есть на сервере) для оптимизации скорости, встроенный маршрутизатор и многое другое.

Важный момент, что фреймворк использует именно React, самую используемую библиотеку JS, что значит, что работа над проектом будет доступна большему числу разработчиков.

Tanstack Query (React Query)

Библиотека для управления запросами. Предоставляет кэширование, управление ошибками и обновление данных на сервере. По факту заменяет global state менеджеры (redux, zustand).

Typescript

Значительно упрощает процесс разработки избежанием множества багов, частично может заменять документацию, а интеграция с редакторами кода улучшает DevEx.

Tailwind

В этой библиотеке мне больше всего нравится, что код для CSS-стилей остаётся читаемым с ходом времени. Часто при классическом подходе к написанию CSS-классов через семантический смысл элементов через время проблематично ориентироваться в коде, то же самое происходит и при чтении кода другого разработчика. Tailwind использует utility-классы, оторванные от бизнес-логики приложения, которые не меняются со временем и одинаковы для всех пользователей библиотеки.

React-chartjs-2 / Recharts / Tremor

Библиотеки для отрисовки метрик. Пока не уверен, какую из них буду использовать

Jest

Библиотека для написания unit- и integration-тестов. Хорошо работает в связке с React Testing Library, что позволяет быстро и удобно писать тесты. Также Jest будет использоваться и для бекенда.

Cypress

Библиотека для написания e2e-тестов.

И Jest, и Cypress, я выбрал из-за опыта использования.

Backend

Node.js / Express

Реальный выбор на момента начала работы над проектом стоял между JS (Node.js/Express) и PHP (Laravel). В итоге выбрал JS из-за возможности переиспользования наработок с прошлых проектов, большего реального опыта разработки и удобства использования одного и того же языка на фронте и беке (а также потенциальной возможности использования библиотек, рассчитанных на один и тот же язык во всём приложении, как, например tRPC).

PostgreSQL

Я не любитель использования нереляционных СУБД в больших проектах, а среди реляционных работал только с postgres и mysql. Принципиальной разницы за время работы не увидел, но знаю, что postgres лучше справляется со сложными запросами, что может быть полезно для эффективной работы с метриками. Переход к другой СУБД при необходимости не будет проблематичным из-за использования ORM, поддерживающей основные популярные СУБД.

Prisma

ORM для JS. Позволяет писать более надёжные и читаемые запросы к бд.

Passport.js

Библиотека для авторизации и аутентификации пользователей. С учётом использования Next.js + Express, выбор стоял между NextAuth и Passport.js. Остановился на passport из-за наличия опыта и наработок с прошлых проектов. Поддерживает классическую авторизацию через логин-пароль (для админ-панели), SSO (можно расширять своими провайдерами) и JWT-аутентификацию

CodeQL / ?SonarQube?

Для анализа качества кода. Пока не вникал в подробности, понял только, что, видимо, SonarQube для тех целей, которые нужны реестру, доступен только в виде платной версии. CodeQL предлагает достаточно трудоёмкую настройку для автоматизации работы, но выглядит вполне настраиваемым для целей, которые мне нужны.

?Socket.io?

Есть шанс, что будет использоваться для обновления метрики вне очереди из админ-панели. Возможно, ограничусь поллингом.

Postman

Для тестирования запросов к серверу.