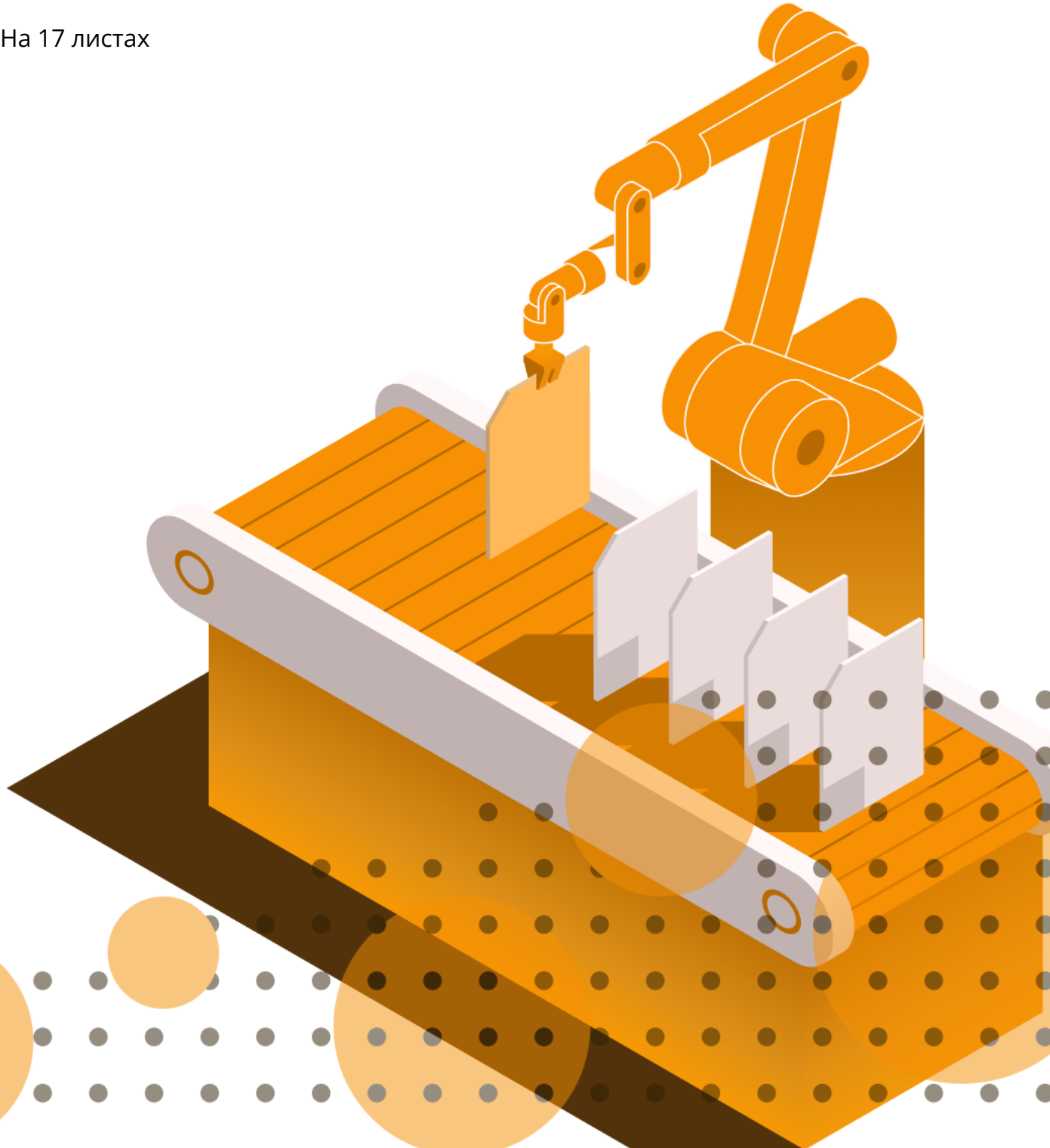
ООО "АППОМАРТ" / ИНН: 7811772730 / ОГРН: 1227800030765
Телефон: +7 (911) 833-33-84 / E-mail: hello@appomart.com

APPOBOX

Руководство пользователя и функциональные возможности

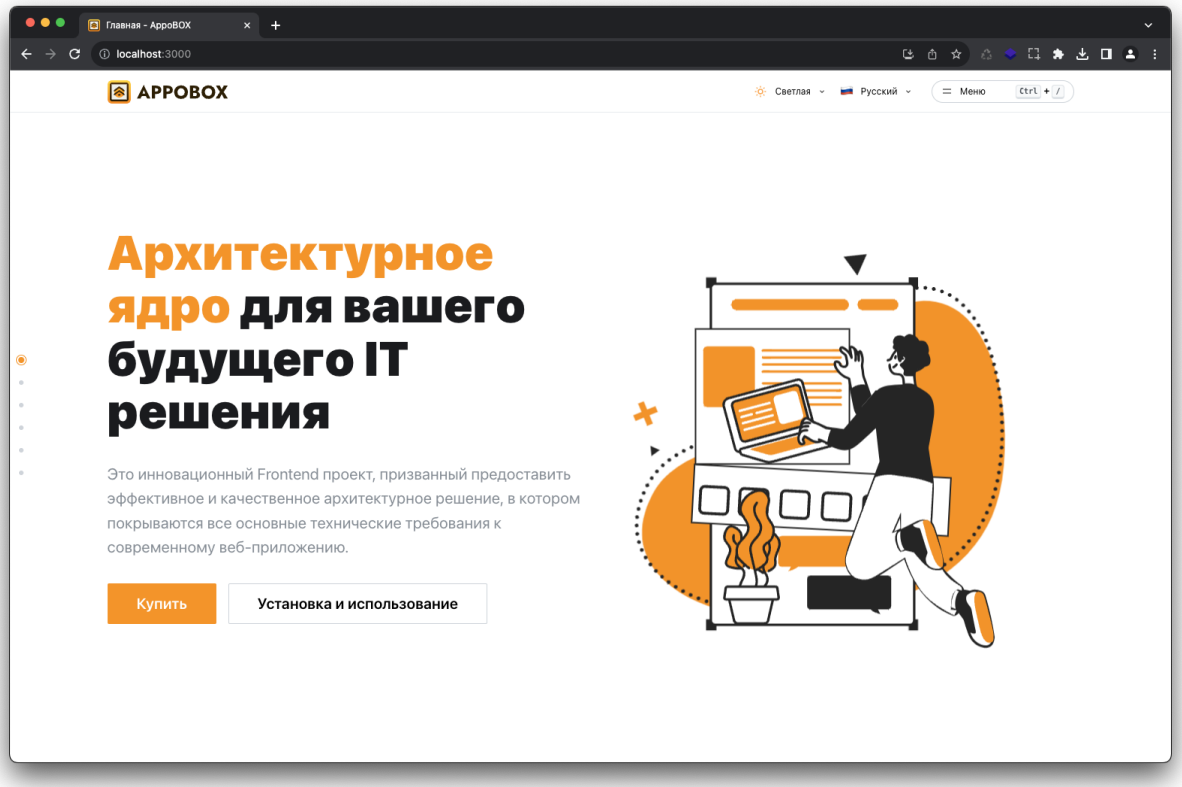
На 17 листах



Оглавление

Аннотация

Данный документ описывает функциональные особенности и основные технические аспекты программного обеспечения APPOBOX.



Функциональные возможности

Проект призван предоставить эффективное и качественное архитектурное решение, в котором покрываются все основные функционально-технические требования к современному веб-приложению. Набор программного кода и инструментов для разработчика, которые упрощают процесс разработки и ускоряют его:

1. Многостраничность - возможность создания приложения с более сложной структурой через поддержку нескольких страниц (включая динамические страны), обеспечивая логичную навигацию;

2. Интернационализация - проработанная гибкость и легкость адаптации приложения под различные языки, если приложение требует

мультиязычность. Учет «дробления» локализации на постраничные файлы и оптимизированная загрузка;

3. Внутреннее SEO - оптимизированная структура приложения, Server Side Rendering контента, Progressive Web App режим, поддержка HEAD заголовков для социальных сетей. Генерация sitemap.xml, robots.txt с удобным масштабированием через маршрутизацию;

4. Шаблонизация - проработанный механизм для создания и использования шаблонов, упрощая процесс создания структурированных страниц. Поддержка динамических параметров для условного рендера составных частей шаблона;

5. UI компоненты - Mantine компоненты, а также готовые базовые компоненты, например, баннер Cookie, модальное окно Captcha, Progress индикатор страницы, кнопки Share в социальные сети, и другие;

6. Темная тема - проработанный механизм смены темы веб-приложения, с возможностью выбора системной темы. Поддержка регидратации и отсутствие бликов при инициализации веб-приложения;

7. Сборка JSON контента - проработанная базовая логика для сборки статического JSON контента, если требуется абстракция объемных данных без реализации API сервера;

8. Централизованный API - проработанная интеграция библиотеки Redux Toolkit с поддержкой Queries, что позволяет значительно упростить взаимодействие с API и получить централизованное хранилище данных;

9. Анимация - поддержка Framer Motion для интеграции продвинутых анимаций и жестов. Поддержка анимированных иллюстраций Lottie.

Технические особенности

Проработаны следующие технические аспекты, призванные помочь вам в процессе разработки и при публикации вашего проекта:

1. Поддержка трех видов среды окружения - DEV, STAGE и PRODUCTION;

2. Монолитный архитектурный стиль для масштабирования проекта;

3. Поддержка валидации и форматирования кода через ESLint и Prettier;
4. Поддержка модульного тестирования через Jest.

Структура программного обеспечения

Параметры окружения

Базовые ENV параметры (устанавливаются в файле `.env`):

1. `NEXT_PUBLIC_DEFAULT_LOCALE` - базовый код языка (см. раздел - Локализация);
2. `NEXT_PUBLIC_TITLE` - v-название проекта. Добавляется к HEAD заголовкам страницы (см. раздел - Компоненты / Head заголовки);
3. `NEXT_PUBLIC_DESCRIPTION` - описание проекта. Выставляется по умолчанию для HEAD описаний, если отсутствует установка описания на странице (см. раздел - Компоненты / Head заголовки);

Базовые ENV параметры (устанавливаются в `.env.{mode}` файлах для режимов):

1. `NEXT_PUBLIC_BASE_URL` - базовый URL сайта;
2. `FORM_WEBHOOK_URL` - API маршрут для отправки данных с форм сайта.

PDF файлы политик

В `public/docs` каталоге располагаются файлы:

1. Политика использования файлов Cookie
2. Политика конфиденциальности

При публикации вашего проекта необходимо обязательно заменить данные файлы в соответствии с вашими политиками. Проконсультируйтесь с юристом, чтобы составить данные документы.

Шаблоны дизайна

Расположение шаблонов - `src/extra/layouts`
Расположение общих компонентов - `src/extra/layouts/extra`

Список компонентов:

1. Переключатель языка
2. Переключатель темы
3. Кнопка открытия поиска
4. Установщик Head заголовков темы

Основной шаблон

Рабочее название - `Default`

Это главный шаблон, который выставляется по умолчанию для всех реализуемых страниц. Данный шаблон включает Header навигацию и Footer подвал сайта.

Без подвала

Рабочее название - `WithoutFooter`

Упрощенный шаблон базового шаблона, который исключает подвал в нижней части. Пример использования - Страница не найдена (404).

Локализация

Для работы с файлами локализации используется библиотека `next-i18next`. На верхнем уровне проекта имеется файл с настройками библиотеки, где перечисляются языки в стандарте `ISO-639-alpha-2`.

Файлы локализации располагаются в каталоге src`/locales`, где каждый каталог - это код языка в стандарте `ISO-639-alpha-2`, а вложенные JSON файлы делятся на страницы (исключением является `common.json`, где описываются базовые локализации).

Список языков (для типизации) задается для `enum LanguageCode` в файле `src/locales/ILanguage.ts`. Этот список также должен быть продублирован в файл `next-i18next.config.js`. Код базового языка указывается в файле `.env` файле (см. раздел - Параметры окружения).

Подгрузка локализации выполняется в шаблонном обработчике `serverSideProps`, который переопределяет методы `getServerSideProps` и `getStaticProps` (см. примеры страниц).

Для интеграции текстов локализации необходимо использовать хук `useTranslate` (см. раздел - Хуки).

Для каждой из страниц можно задать принудительные языки отображения (в случаях, когда, например, динамический контент доступен только на части языков). Для этого нужно установить параметр `languages` при выполнении обработки `serverSideProps`, описывающий массив кодов `LangaugeCode`.

Статический JSON контент

При работе со статическим контентом, для облегчения разработки без API прослойки, проработана сборка контента из JSON файлов внутри самого проекта.

Для JSON контента определяется каталог `data`, в котором описываются необходимые файлы. Данный каталог будет дублироваться и модифицироваться через обработчик (см. ниже) в каталог `src/public/data`.

За сборку отвечает обработчик `src/builder`. В нем прорабатывается логика последовательность сборки контента. Для запуска сборки необходимо выполнить `yarn pre:build`. Определение типизации данных необходимо описывать в каталоге `src/interfaces`.

Для примера реализована сборка и обработка статических данных `Articles`:

1. Добавлен пример структуры данных в папку `data`;
2. Добавлен пример сборки контента в `builder/build-articles.ts`;
3. Добавлен пример обработки API контента в `src/api/article.ts`.

Обработка API

Вся обработка запросов выполняется через RTK Query, управление которым доступно в каталоге `src/api`. Для него подготовлена логика подстановки собственных HTTP заголовков.

Проработаны следующие маршруты:

1. Получение списка статей (статический JSON контент);
2. Получение детализации статьи (статический JSON контент);
3. Отправка формы на обратную связь;
4. Отправка формы на запрос покупки продукта.

Хранилище состояния

Для создания централизованного хранилища используется Redux Toolkit. Вся логика обработки хранилища располагается в каталоге `src/store`. Для получения доступа к хранилищу реализованы типизированные хуки (см. раздел - Хуки).

Хуки

Хуки хранилища

Для работы с хранилищем состояния Redux проработаны типизированные хуки `useDispatch` и `useSelector`.

Хук генерации Captcha

Для создания Canvas изображения и его последующей валидации реализован хук `useCaptcha`. Данный хук используется в модальном окне Captcha (см. раздел - Компоненты).

Хук технических данных

Для сбора технических данных и сборке и браузере пользователя реализован хук `useTechnicalInfo`. Данный хук используется в отправке формы Feedback (см. раздел - Компоненты).

Хук распознавателя жестов

Для создания события отслеживания движений мыши по DOM элементу реализован хук `useSwipeHandler`. Данный хук используется в компоненте презентационной карточки (см. раздел - Компоненты).

Хук перевода

Переопределяемый стандартный хук библиотеки `next-i18next`, необходимый, чтобы оптимизировать рендер данных при анимациях и упростить подстановку параметров в текст перевода.

Глобальные ссылки

Для глобального доступа к тем или иным компонента введены глобальные `ref` ссылки для:

1. Модального окна Captcha (см. раздел - Компоненты);
2. Модального окна видео (см. раздел - Компоненты). Чтобы дать доступ к вызову данных компонентов с любой точки жизненного цикла кода.

Компоненты

Базовые компоненты

Для работы доступны библиотеки Mantine в соответствии с установленными библиотеками от данного разработчика, указанные в файле `package.json`.

Графика и анимация

Для работы с графикой и анимацией доступны следующие библиотеки:

1. `lottie-react`
2. `framer-motion`
3. `react-simple-maps (d3-geo)`

Для последней библиотеки определены `topojson` файлы, доступные в каталоге `src/public/topojson`.

Head заголовки

Расположение - src/components/AppHead

Данный компонент описывает все необходимые заголовки:

1. SEO заголовки;

2. Socials заголовки;
3. Заголовки для alternate языков;
4. Share заголовки картинок.

Компонент необходимо импортировать в рендер страницы на верхнем уровне (см. примеры страниц).

Переключатель секций страницы

Расположение - `src/components/PageSwitcher`

Универсальный компонент, генерирующий вертикальное фиксированное меню в виде вертикальных точек, относительно списка переданных ID элементов DOM (см. пример интеграции на главной странице). Данный компонент отслеживает положение вертикального скролла и отображает текущий фокус на секции. Нажатие на точку проматывает пользователя до выбранной секции. Якорь синхронизируется с URI страницы, при клике на элемент меню.

Презентационная карточка

Расположение - `src/components/PresentationCard`

Компонент, который расширяет Paper компонент фреймворка Mantine, добавляя анимационные эффекты трансформации и теней при перемещении мыши. Пример интеграции данного компонента представлен на главной странице.

Кнопки “Поделиться”

Расположение - `src/components/ShareButtons`

Кнопки “Поделиться”, которые можно интегрировать в требуемое место. Автоматическое формирование ссылок с иконками (нужно передать искомый URL и текст) для российских и зарубежных социальных сетей, в том числе:

1. Вконтакте
2. Одноклассники
3. Telegram

Модальное окно Captcha

Расположение - `src/components/extra/CaptchaModal`

Данное модальное окно позволяет вызывать прохождение собственной проверки через Captcha через использование хука `useCaptcha` (см. раздел - Хуки). Вызов происходит через глобальную ссылку (см. раздел - Глобальные ссылки).

Данное модальное окно уже интегрировано и не нуждается в дополнительной инициализации.

Cookie баннер

Расположение - `src/components/extra/CookieBanner`

Баннер оповещение о том, что сайт использует файлы Cookie. Данное модальное окно уже интегрировано и не нуждается в дополнительной инициализации.

Индикатор прогресса страницы

Расположение - `src/components/extra/RouterProgress`

Полоса состояния прогрузки асинхронных данных. Уже интегрирована и не нуждается в дополнительной инициализации.

Кнопка “Вверх”

Расположение - `src/components/extra/ScrollToTop`

Выпадающая кнопка перелета до начала страницы. Уже интегрирована и не нуждается в дополнительной инициализации.

Модальное окно поиска

Расположение - `src/components/extra/SearchModal`

Данное модальное окно реализует Spotlight поиск на базе фреймворка Mantine. Уже интегрировано и не нуждается в дополнительной инициализации.

Модальное окно просмотра видео

Расположение - `src/components/extra/VideoModal`

Данное модальное окно позволяет вызывать глобальное модальное окно просмотра видео-потока). Вызов происходит через глобальную ссылку (см. раздел - Глобальные ссылки). Имеется поддержка frame-ссылок. Уже интегрировано и не нуждается в дополнительной инициализации.

SEO оптимизация

PWA поддержка

Сайт удовлетворяет все требования к PWA через использование библиотеки `next-pwa`. Для установки манифеста двух тем - проработана два JSON файла, один из которых устанавливается динамически.

Генерация sitemap.xml

Для проработки карты сайта используется библиотека `sitemap`. Карту определяется динамическая страница, где прорабатываются все требуемые маршруты, включая поддержку приоритета, частоты изменений и языкового параметра.

Генерация robots.txt

По аналогии с картой сайта - данный файл определяет динамическая страница, в которой могут быть проработаны те или иные условия доступа для индексации роботами.

Динамическая установка заголовков

Все страницы включают в себя поддержку компонента AppHead (см. раздел - Компоненты), который определяет все необходимые заголовки для индексации, публикации ссылки в социальных сетях или отправке ссылки в мессенджерах.

Параметры

Константы

Расположение `src/constants/index.ts`

В данном файле собраны воедино все необходимые постоянные параметры, которые используются в проекте.

Контакты

Расположение `src/constants/contacts.ts`

В данном файле определена удобная структура контактных данных для получения доступа к ним с любой точки рендера. Пример использования представлен на главной странице, на карте контактов.

Конфиг темы

Расположение `src/constants/theme.ts`

Данный файл определяет конфиг стилей и параметров для фреймворка Mantine. Также здесь указан ключ для сохранения состояния в куки и цветовая палитра, которая выставляется по-умолчанию.

Языковые стандарты

В проекте используются два стандарта:

1. Для языков - `ISO-639-1-alpha-2`
2. Для стран - `ISO-3166-1-alpha-2`

Оба стандарта собраны в соответствующие файлы:

1. Языки - `src/constants/languages.json`
2. Страны - `src/constants/countries.json`

в которых определены:

1. Названия страны для каждого кода на искомом языке страны;
2. Название языка для каждого кода языка;

3. ISO флаг языка для каждого кода языка.

Помощники

Расположение - `src/helpers`

В данном каталоге располагаются важные функции помощи при реализации тех или иных функций. Примеры:

1. Переопределенный `getServerSideProps` для интеграции в страницы;
2. Сборка JSON в FormData;
3. Валидаторы типов (почта);
4. Определить path для URL с учетом поддерживаемых локалей сайта.

Страницы

Главная страница

Маршрут страницы - `/{locale}`
Расположение страницы - `src/pages/index.tsx`
Расположение компонентов - `src/extra/pages/Home`

Быстрый старт

Маршрут страницы - `/{locale}/quick-start`
Расположение страницы - `src/pages/quick-start.tsx`
Расположение компонентов - `src/extra/pages/QuickStart`

Купить

Маршрут страницы - `/{locale}/buy`
Расположение страницы - `src/pages/buy.tsx`
Расположение компонентов - `src/extra/pages/Buy`

О разработчике

Маршрут страницы - `/{locale}/developer`
Расположение страницы - `src/pages/developer.tsx`
Расположение компонентов - `src/extra/pages/Developer`

Страница 404

Маршрут страницы - `любая несуществующая страница`
Расположение страницы - `src/pages/404.tsx`
Расположение компонентов - `src/extra/pages/NotFound`

Карта сайта

Маршрут страницы - `/sitemap.xml`
Расположение страницы - `src/pages/sitemap.xml.tsx`

Конфиг для робота поисковика

Маршрут страницы - `/robots.txt`
Расположение страницы - `src/pages/robots.txt.tsx`

API обработчик формы обратной связи

Маршрут запроса - `POST /api/feedback`
Расположение обработчика - `src/pages/api/feedback.ts`

API обработчик формы покупки продукта

Маршрут запроса - `POST /api/purchase`
Расположение обработчика - `src/pages/api/purchase.ts`

Сборка и запуск через Node.js

Собрать и развернуть проект можно для `Stage` и `Production` окружений.

Для работы имеются следующие Node команды:

1. `pre:build` - компиляция статического JSON контента;
2. `build:stage` - сборка проекта со `Stage` окружением;
3. `build:production` - сборка проекта с `Production` окружением;
4. `start:stage` - запуск проекта со `Stage` окружением;
5. `start:production` - запуск проекта с `Production` окружением.

Во время разработки данную команду выполнять необходимо при каждых модификациях сборки статического контента (см. раздел - Статический JSON контент).

Сборка Docker контейнера с выбором окружения

Для удобства работы проработана сборка Docker контейнера. Пример команды для сборки проекта в режиме `Production`.

```
docker build --target production -t {image-name}:production .
  --platform=linux/amd64
```

Сборка контейнера включает в себя сборку статического контента (см. раздел - Статический JSON контент).