МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Российский государственный университет им. А.Н. Косыгина   
(Технологии. Дизайн. Искусство)»**

Институт ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И ЦИФРОВЫЕ ТРАНСФОРМАЦИИ

**Отчет по заданию**

**по дисциплине «Веб-разработка на стороне клиента»**

**Тема: «****Парсер динамических страниц»**

Выполнил: Сидоров Д. С., группа ИТС-123

Проверил: Кононова О. С.

Москва, 2025г

**Парсер динамических страниц**

Задание: разработать веб-парсер, который будет собирать и представлять сведения о составе и структуре веб-страниц для заданных пользователем сайтов. Парсер должен иметь удобный и функциональный пользовательский интерфейс.

**Требования к интерфейсу:**

1. Веб-страница с формой для ввода списка URL-адресов.
2. Возможность задавать параметры парсинга: выбор элементов, которые нужно анализировать (заголовки, ссылки, изображения, тексты) и установка глубины парсинга (как глубоко следовать по вложенным ссылкам).
3. Кнопка "Парсить", запускающая процесс парсинга.
4. Вывод результатов парсинга на выбор в графическом виде или json.
5. Возможность экспортировать результаты в форматы JSON или XML.

**Функционал:**

1. Скрипт на Python, библиотеки для работы с HTTP-запросами и для разбора HTML.
2. Проверка на ошибки, если сайт недоступен или адрес неверно введен.
3. Сбор указанных в форме данных:

- заголовки (H1, H2 …);

- внутренние и внешние ссылки;

- изображения;

- описание (мета-теги, ключевые слова) и текстовый контент

1. Опробовать возможность парсинга динамических страниц.
2. Опробовать возможности использования API DeepSeek (или др.) для анализа собранных данных, например, для поиска ключевых слов в текстах, а также для визуализации результатов обработки с помощью инфографики.
3. Хранение результатов парсинга и истории запросов в БД.

Отчёт по проекту Web Parser:

1. Использованные библиотеки:

- Backend:

- @nestjs/common/core - фреймворк для Node.js

- mongoose - для работы с MongoDB

- js-yaml - для конвертации в YAML

- xml-js - для конвертации в XML

- axios - для HTTP запросов

- cheerio - для парсинга HTML

- Puppeteer – для получения информации об хтмл страницы, поддерживающий динамический контент

- Frontend:

- React + TypeScript

- Vite - сборщик проекта

- axios - для API запросов

- react-json-view - для отображения JSON

- react-router-dom – для роутинга страниц

- effector – стм для логики

Инструкция по запуску проекта:

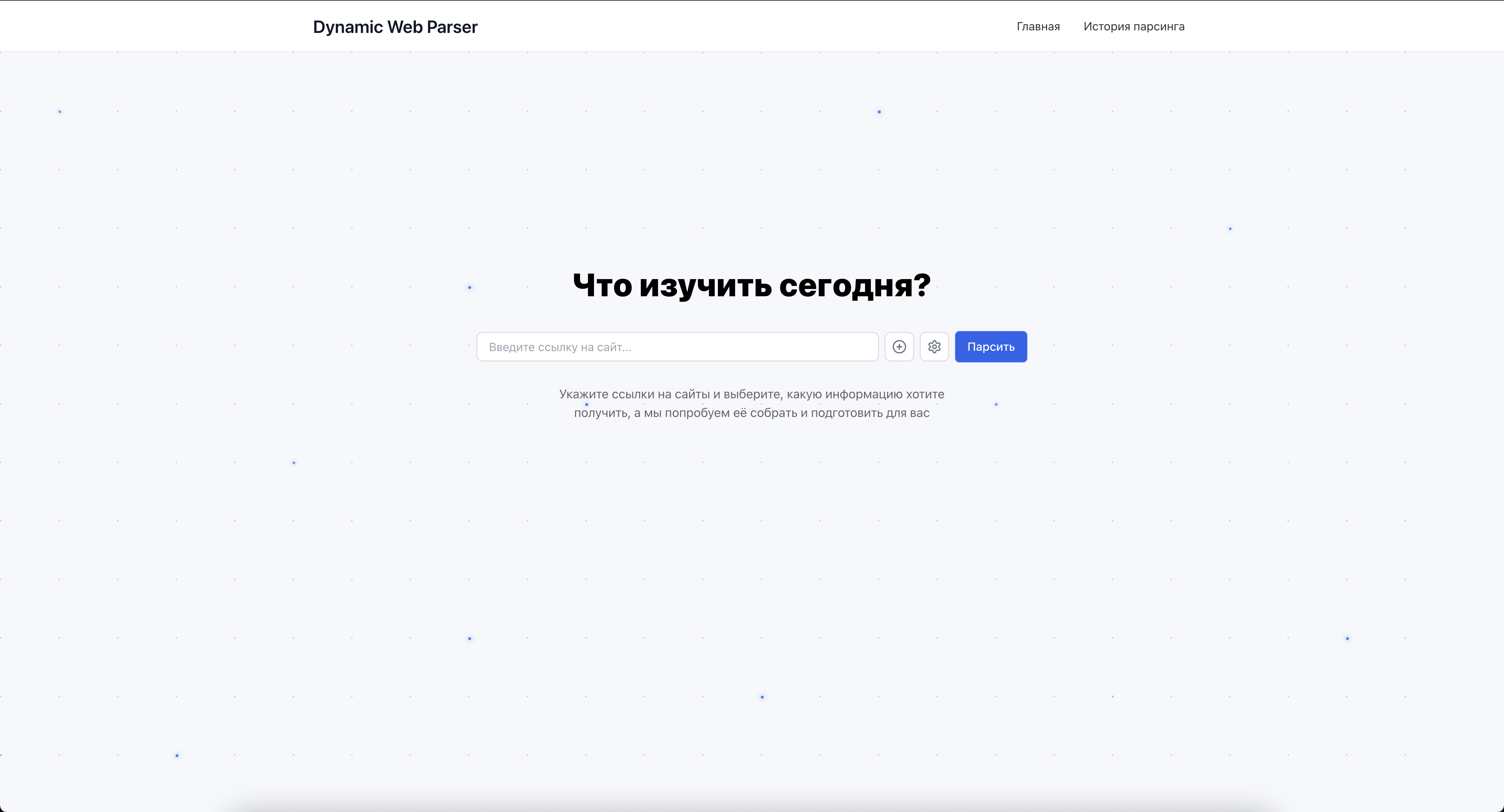
Заранее нужно установить на ПК менеджер пакетов npm и БД mongodb  
1. Глобально установить pnpm – менеджер пакетов, используемый в проекте

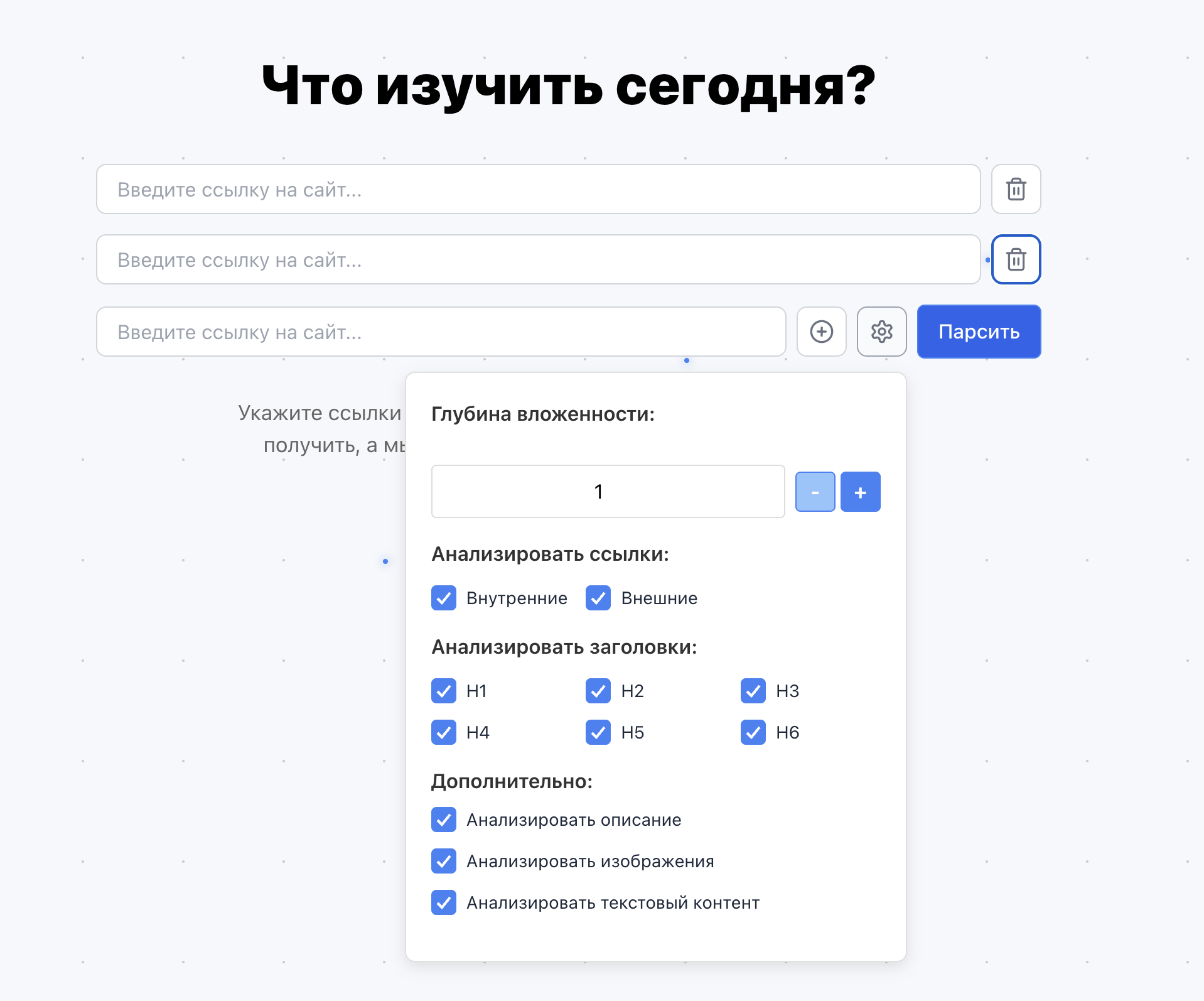
npm install -g pnpm

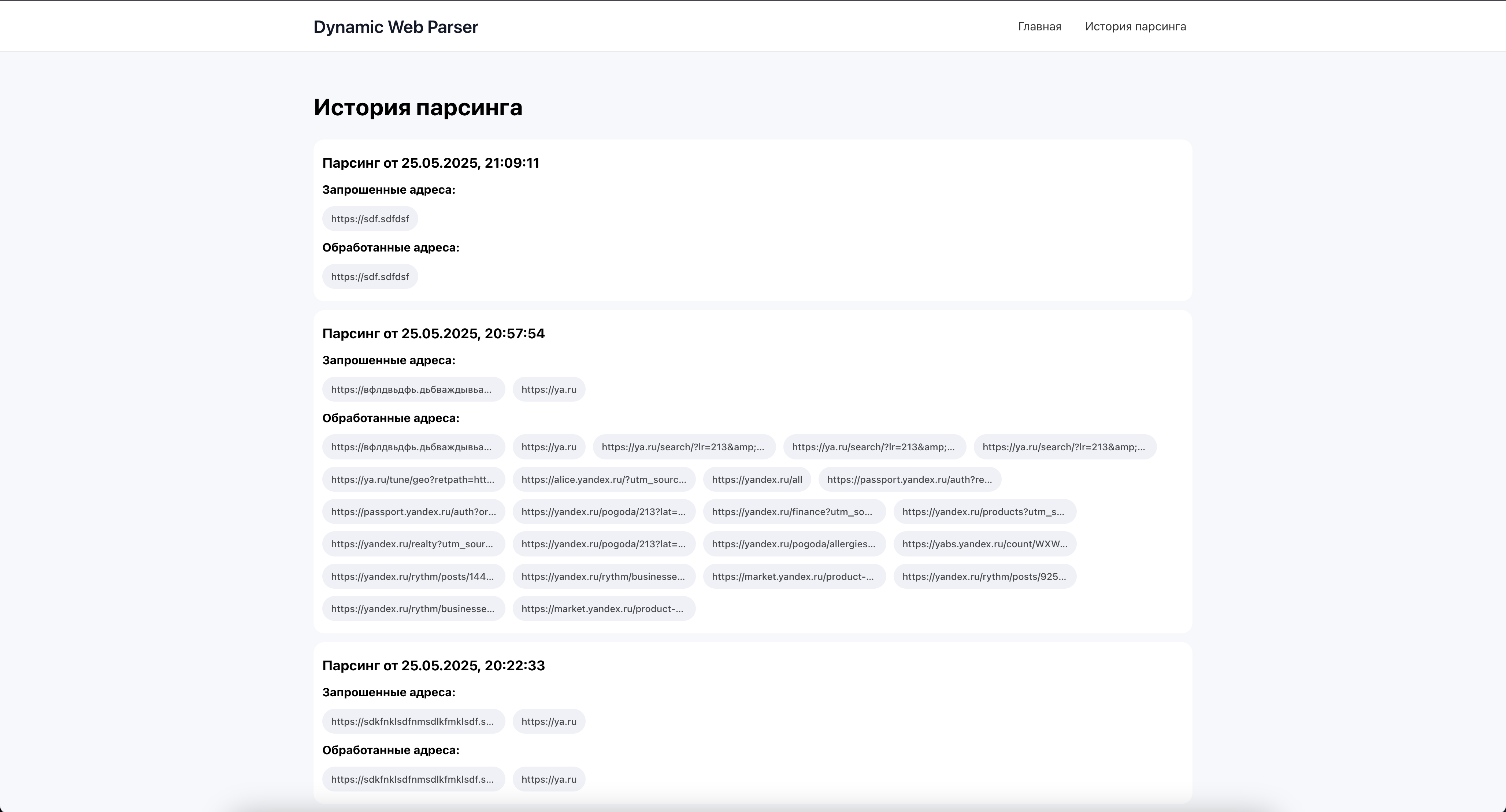
2. Установить зависимости в папках

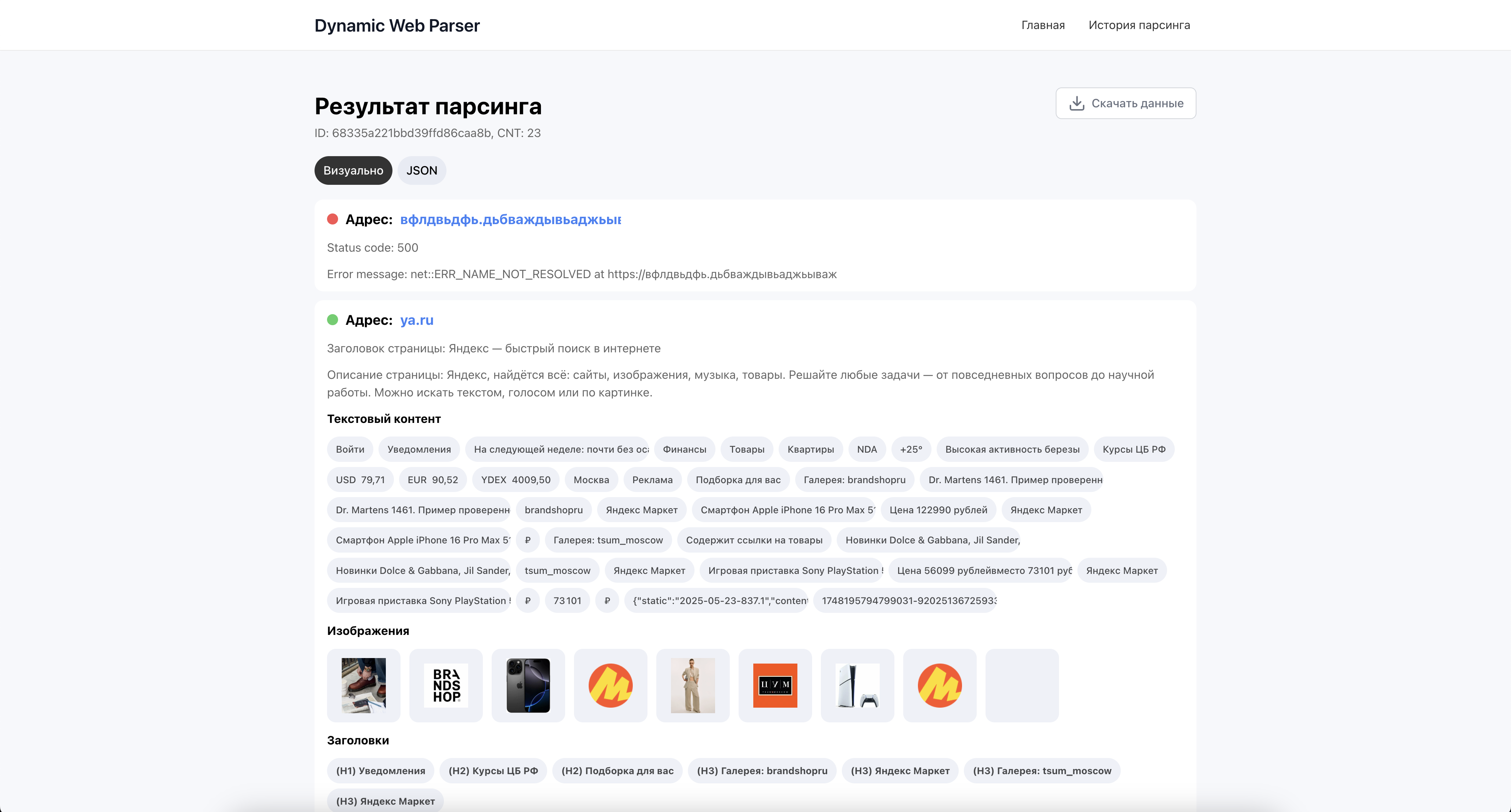
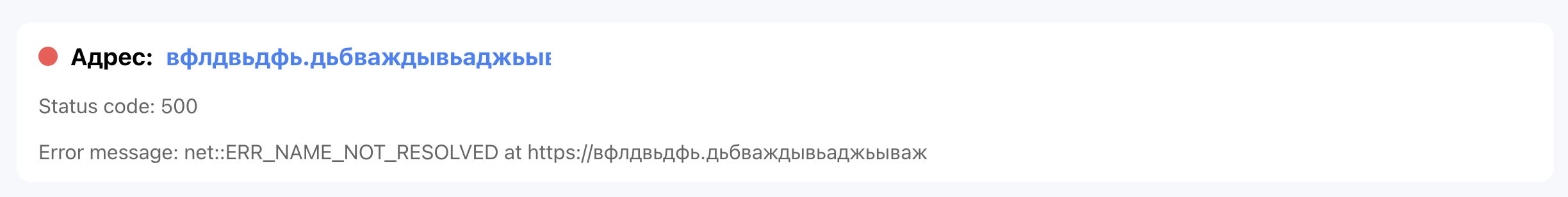
Pnpm i  
 cd backend && pnpm i  
 cd ../  
 cd frontend && pnpm i  
3. Запустить проект, для этого в главной папке выполнить pnpm start  
4. Перейти на <http://localhost:5173/>, там будет доступен полностью работающий парсер

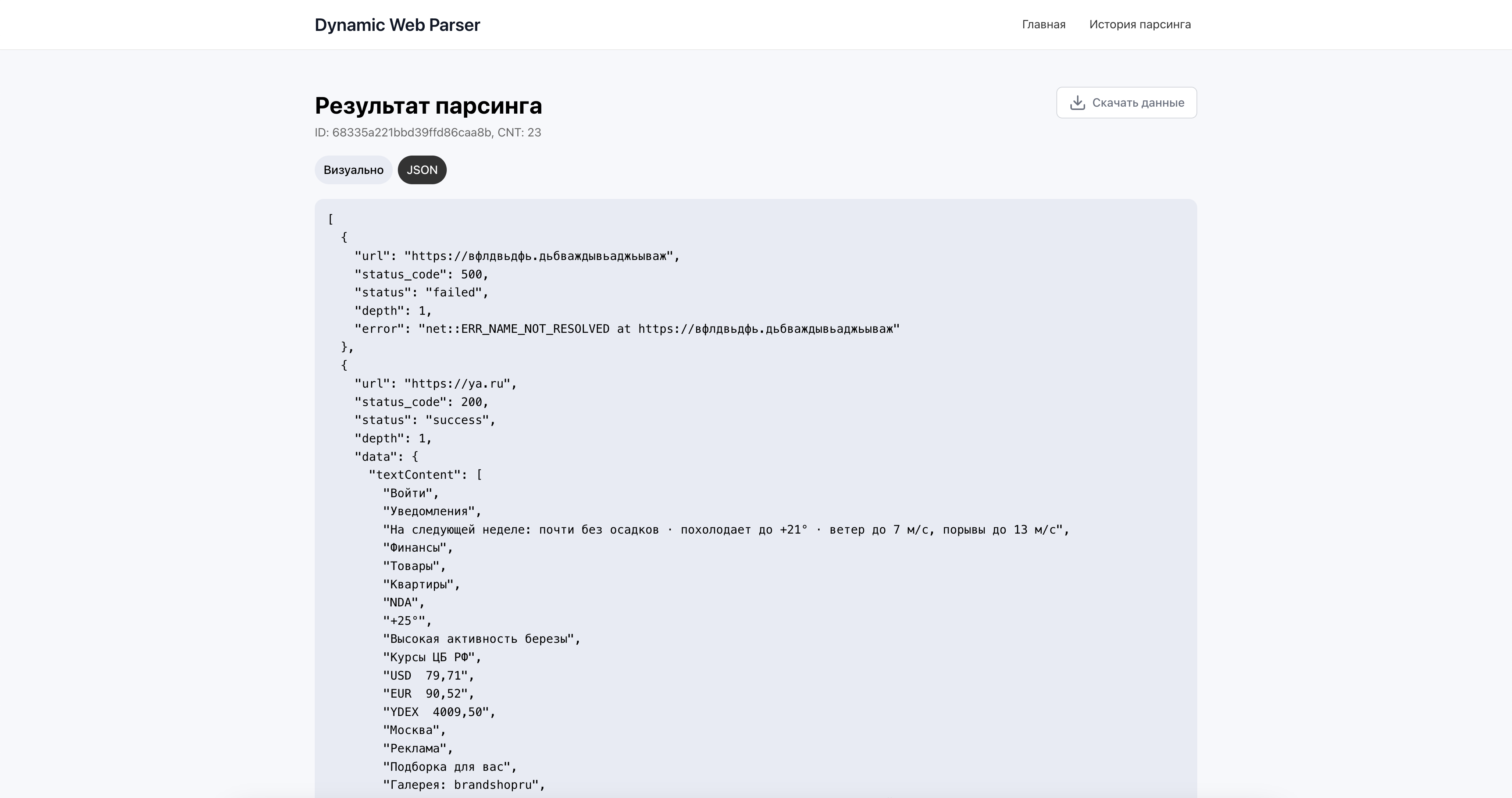
Скриншоты работы программы:

  
Рисунок 1. Главная страница

  
Рисунок 2. Возможность добавлять ссылки и настраивать парсинг

  
Рисунок 3. Просмотр истории парсинга.

  
Рисунок 4. Возможность просмотра результатов парсинга  
  
Рисунок 5. Просмотр страницы, загрузить которую не удалось

  
Рисунок 6. Возможность просмотреть результат парсинга в json

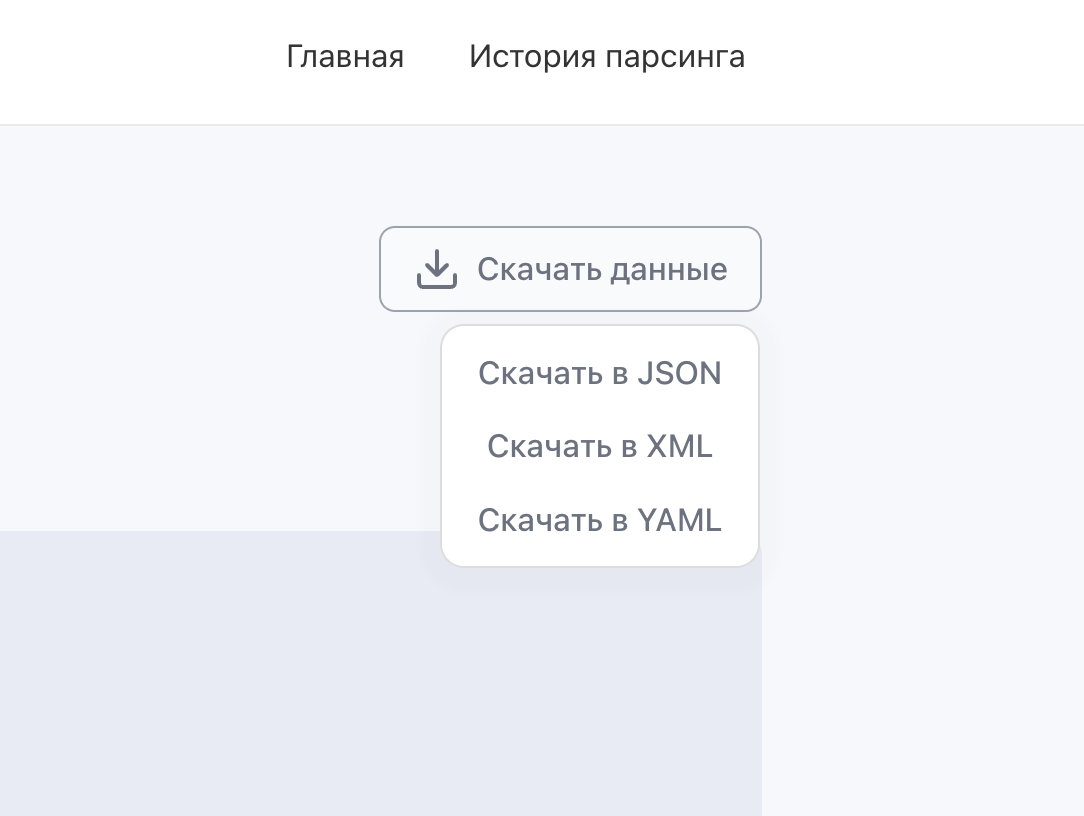


Рисунок 6. Возможность скачать результаты парса в 3х форматах

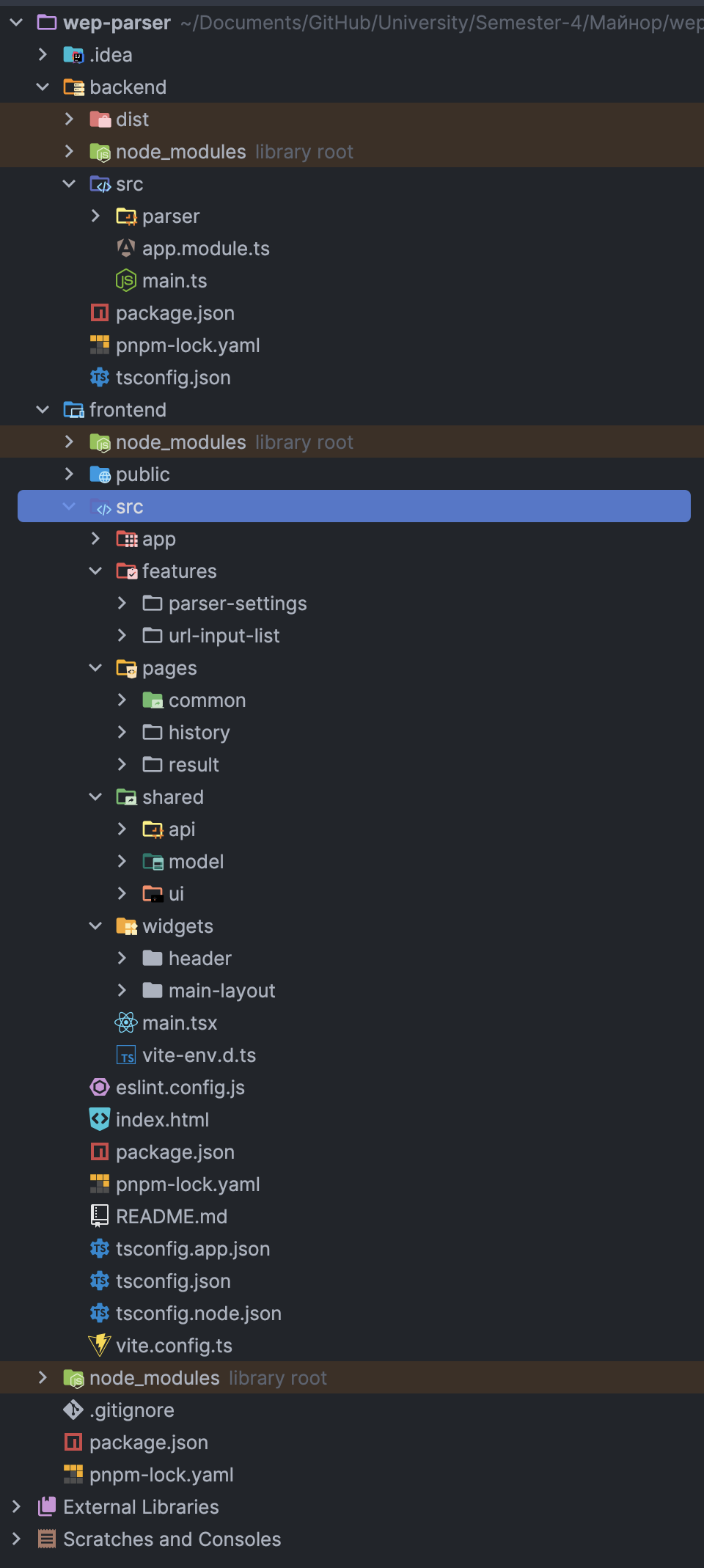


Рисунок 7. Структура проекта