UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
UNIDADE ACADÊMICA DE SISTEMAS E COMPUTAÇÃO
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

HEBERT LUCAS DE LIMA MORAIS

SHAPE-SORTER:

UMA FERRAMENTA DE LINHA DE COMANDO PARA AUTOMAÇÃO DE CRIAÇÃO E MANUTENÇÃO DE MICRO FRONTENDS

CAMPINA GRANDE - PB

2021

HEBERT LUCAS DE LIMA MORAIS

SHAPE-SORTER:

UMA FERRAMENTA DE LINHA DE COMANDO PARA AUTOMAÇÃO DE CRIAÇÃO E MANUTENÇÃO DE MICRO FRONTENDS

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Professor Dr. João Arthur Brunet Monteiro.

CAMPINA GRANDE - PB

2021

M827s Morais, Hebert Lucas de Lima.
Shape Sorter: uma ferramenta de linha de comando para a automação de criação e manutenção de micro frontends. / Hebert Lucas de Lima Morais. – 2021.

12 f.

Orientador: Prof. Dr. João Arthur Brunet Monteiro.
Trabalho de Conclusão de Curso - Artigo (Curso de Bacharelado em Ciência da Computação) - Universidade Federal de Campina Grande; Centro de Engenharia Elétrica e Informática.

1. Micro frontends. 2. Frontend. 3. Desenvolvimento web. 4. Webpack. 5. Modules federation. 6. NPM – Node JavaScript. 7. Interface de linha de comando – CLI. 8. Ferramenta de CLI. 9. Shape Sorter. 10. Desenvolvedores de software. I. Monteiro, João Arthur Brunet. II. Título.

CDU:004.41(045)

Elaboração da Ficha Catalográfica:

Johnny Rodrigues Barbosa
Bibliotecário-Documentalista
CRB-15/626

HEBERT LUCAS DE LIMA MORAIS

SHAPE-SORTER:

UMA FERRAMENTA DE LINHA DE COMANDO PARA AUTOMAÇÃO DE CRIAÇÃO E MANUTENÇÃO DE MICRO FRONTENDS

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

BANCA EXAMINADORA:

Professor Dr. João Arthur Brunet Monteiro
Orientador – UASC/CEEI/UFCG

Professor Dr. Dalton Dario Serey Guerrero
Examinador – UASC/CEEI/UFCG

Professor Dr. Tiago Lima Massoni
Professor da Disciplina TCC – UASC/CEEI/UFCG

Trabalho aprovado em: 24 de maio de 2021.

CAMPINA GRANDE – PB

Shape Sorter: uma ferramenta de linha de comando para a automação de criação e manutenção de micro frontends

Trabalho de Conclusão de Curso

Hebert Lucas de Lima Morais

Orientador: João Arthur Brunet

Universidade Federal de Campina Grande

Campina Grande, Paraíba, Brasil

hebert.morais@ccc.ufcg.edu.br

Resumo

Visto que houve uma crescente popularização na arquitetura de microsserviços, que são majoritariamente implementados no backend, observou-se a possibilidade da sua implementação no frontend, concebendo-se o termo micro frontends. Apesar de micro frontends ser um termo conhecido, sua proposta ainda está sendo difundida entre a comunidade e o mercado. Sendo assim, ainda não existem ferramentas de fácil utilização que auxiliem os desenvolvedores a implementarem a infraestrutura de micro frontends de forma otimizada. Para solucionar esse problema foi criado uma ferramenta de CLI que busca auxiliar os desenvolvedores a criarem e manterem seus micro frontends de forma interativa em seus terminais, criando componentes compartilhados de forma automática, e permitindo que as configurações sejam criadas sem que haja uma pesquisa aprofundada ou um conhecimento técnico específico sobre o tema.

Repositório:

https://github.com/hebertmorais/shape-sorter

Palavras-chave:

micro frontends, cli, frontend, desenvolvimento web, webpack, modules federation

1. Introdução

O termo micro frontend surgiu pela primeira vez no ThoughtWorks Technology Radar no final de 2016. Semestralmente, a ThoughtWorks, empresa especializada em consultoria global de software, publica a ThoughtWorks Technology Radar, que é um documento que define as mudanças que são consideradas atualmente como relevantes no âmbito de desenvolvimento de software. Esse documento reflete a opinião particular de um grupo de tecnólogos experientes e é baseado no trabalho e experiências do dia-a-dia da própria ThoughtWorks. [1]

A formulação da ideia de micro frontends estende conceitos de microsserviços para o mundo do frontend. A tendência atual é construir um aplicação web poderosa e rica em recursos, que se baseia em uma arquitetura de microsserviços. Com o tempo, a camada de frontend, geralmente desenvolvida por uma equipe separada, cresce e fica mais difícil de manter. Isso é chamado de monólito de frontend (imagem 1.1).

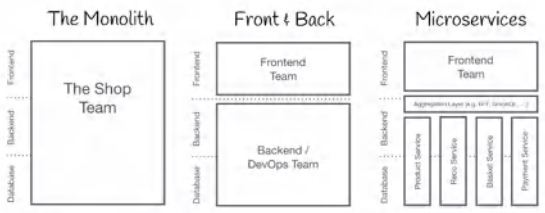


Imagem 1.1

https://tautorn.github.io/micro-frontends/ressources/diagrams/organisational/monolith-frontback-microservices.png

A ideia por trás de Micro Frontends é pensar um website, ou aplicação web, como uma composição de funcionalidades que são propriedades de equipes independentes. Cada equipe tem uma área distinta de negócios, ou missão, do qual se preocupam em especializar-se. Uma equipe é *cross functional* e desenvolvem suas funcionalidades fim-a-fim (imagem 1.2), desde o banco de dados até a interface para o usuário. (Michael Geers, Micro Frontends [2])

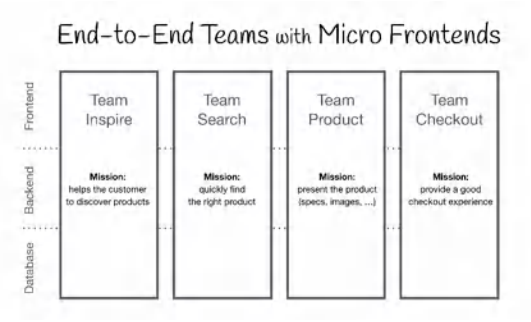


Imagem 1.2
https://tautorn.github.io/micro-frontends/ressources/diagrams/organisational/verticals-headline.png

Segundo Martin Fowler, Chief Scientist na ThoughtWork e um dos fundadores da ideia de micro frontends, alguns dos principais benefícios que são vistos na implementação da arquitetura de micro frontends são: códigos menores, mais coesos e mais fáceis de manter, permitir que organizações sejam mais escaláveis com equipes independentes e autônomas, além de prover uma capacidade de melhorar, atualizar ou mesmo reescrever partes do frontend de uma forma mais incremental do que era possível anteriormente. [3]

Apesar de ser um termo utilizado conhecido há algum tempo, sua implementação entre as empresas ainda está pouco consolidada e sua proposta ainda está sendo difundida entre a comunidade. Sendo assim, ainda não existem ferramentas de fácil utilização que auxiliem os desenvolvedores a implementarem a infraestrutura de micro frontend, sendo sempre necessário a implementação de toda uma nova estrutura, e como não existe uma gama variada de material disponível para pesquisa, se torna um trabalho bem laborioso.

2. Solução

Visto que há um grande trabalho cada vez que um desenvolvedor necessita criar a infraestrutura de um micro frontend, esse trabalho, chamado *Shape Sorter*, visa propor o desenvolvimento de uma ferramenta CLI (Command Line Interface) que auxilia os desenvolvedores a criarem e manterem seus micro frontends de forma interativa em seus terminais, criando componentes compartilhados de forma automática, e permitindo que as configurações sejam criadas sem que haja uma pesquisa aprofundada ou um conhecimento técnico específico em micro frontends.

2.1 Descrição

O Shape Sorter auxilia o desenvolvedor que necessita criar uma infraestrutura de micro frontend, fornecendo, através de templates, o projeto inicial junto com um arquivo de configuração. Esse arquivo de configuração é onde todas as configurações referentes ao micro frontend serão adicionadas e manipuladas. A ferramenta de linha de comando permite que o desenvolvedor:

- Crie um projeto apto a integrar, no caso de um host, ou ser integrado, no caso de um módulo, em um micro frontend, junto com um projeto em React.js;
- Adicione ao host novos módulos remotos;
- Exponha componentes específicos de um módulo;
- Adicione novas dependências a serem compartilhadas.

Nessa versão do *Shape Sorter*, o template criado é de um projeto em React.js, que foi escolhido por ser uma ferramenta popular entre a comunidade, de fácil utilização, e no caso da ferramenta, permite que os componentes dos módulos sejam facilmente integrados ao host.

2.2 Funcionalidades

Para avaliar a viabilidade da ferramenta, foi criado um projeto semelhante ao demonstrado por Michael Geers em [2]. A página do produto de uma loja de modelos de tratores vai servir como base para os exemplos a seguir.

Imagem 2.1
https://tautorn.github.io/micro-frontends/ressources/screen/three-teams.png

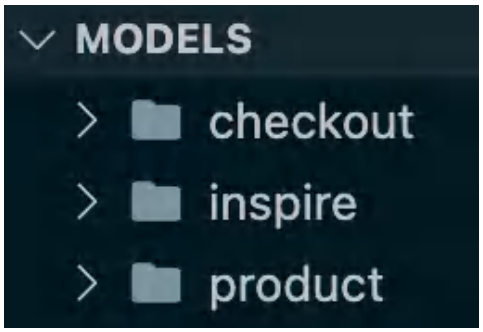
2.2.1 Inicialização

A instalação da ferramenta é feita através do comando:

$ npm install -g shape-sorter

Esse comando permite instalar o Shape Sorter e utilizar o CLI em qualquer projeto de forma global.

Após instalar a ferramenta, é criada uma pasta do projeto, com as subpastas equivalentes a cada micro frontend. No caso do exemplo, foi criada a pasta Models, referente ao projeto como todo, e como pastas filhas os projetos relacionados aos times *product*, *checkout* e *inspire*, no qual são sugeridos no exemplo referente à imagem 2.1.



Após isso é feita a criação dos templates em cada projeto.

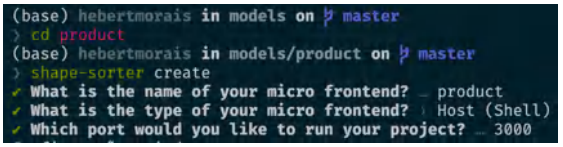
2.2.2 Comando de criação

A criação do projeto da página *product*, que é um host (página que engloba todas as outras), é feito através do comando:

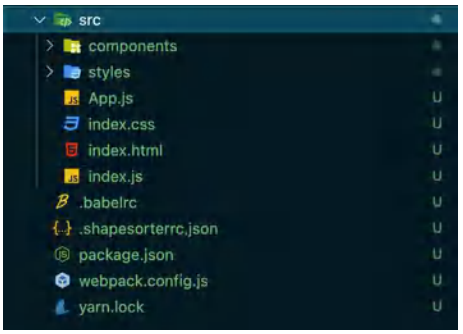
$ shape-sorter create

Esse comando mostra algumas perguntas para a criação do projeto, que são:

- Qual o nome do projeto?
- Qual o tipo do micro frontend? (Host ou módulo)
- Qual porta gostaria de executar o projeto?



Quando o comando é efetivado e as perguntas respondidas, é criado um projeto em React.js com o arquivo *.shapesorterrc.json* contendo todas as variáveis para utilização dos micro frontends.



Para iniciar o projeto e visualizar no browser, basta inserir os comandos:

$ npm install // ou yarn
$ npm run start // ou yarn start

Após isso o projeto já estará sendo executado no browser, podendo ser acessado como endereço localhost e a porta inserida na criação.

Após a inserção dos arquivos, a fim que esteja como o exemplo, o projeto *products* se apresentará dessa forma:

Para a criação dos projetos *checkout* e *inspire*, que são módulos inseridos em *products,* segue-se o mesmo comando de criação citado anteriormente. Após a inserção do nome do módulo na pergunta feita na criação, ambos projetos serão selecionados como *Module*, e executados respectivamente nas portas 3001 e 3002, no caso do exemplo.

2.2.3 Exposição de componentes de um módulo

O módulo *checkout* irá expor dois componentes: o *Basket,* que indica o número de itens no carrinho e *CheckoutButton*, que é o botão para finalizar a compra.

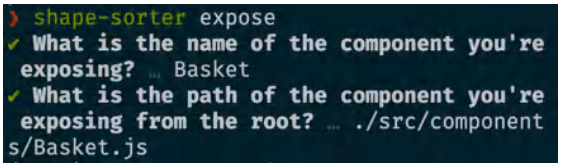
Acima os dois componentes relacionados ao projeto *checkout, Basket* e *CheckoutButton*

Para expor os componentes de *Basket* e *CheckButton* para a página de *product*, é inserido o seguinte comando:

$ shape-sorter expose

Esse comando irá perguntar ao desenvolvedor:

- Com qual nome irá expor esse componente?
- Qual o caminho que esse componente está localizado?



O exemplo acima é relativo ao componente *Basket*. Após isso o mesmo é feito para o componente *CheckoutButton*. Ao fim ambos os componentes estarão aptos a serem utilizados pelo host *products*.

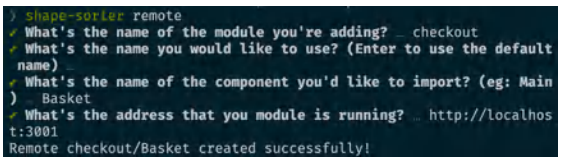
Assim como feito no projeto de *checkout*, o mesmo é feito no projeto *inspire*; o componente *RelatedProducts* é exposto, seguindo o mesmo comando de expor, citado anteriormente.

Acima imagem do componente de *RelatedProducts* no projeto *checkout*

2.2.4 Importação de um módulo

Para que todos os micro frontends sejam integrados no projeto final, os componentes expostos anteriormente são anexados como módulos remotos no host *products* através do seguinte comando:

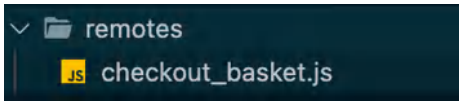
$ shape-sorter remote



Esse comando faz algumas perguntas como:

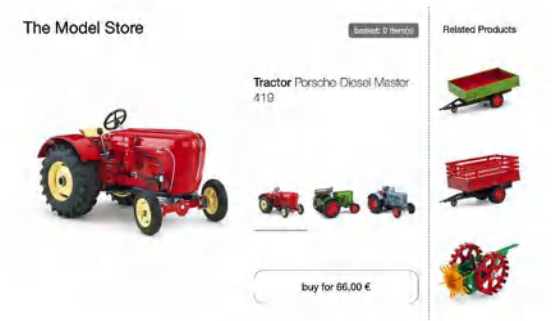
- Qual o nome do módulo cujo componente está sendo exportado?
- Com qual nome gostaria de usar o módulo (opcional)
- Qual o nome do componente que está sendo importado?
- Qual o endereço que o módulo está sendo executado?

Após esse comando o endereço do componente anexado é integrado à lista de módulos remotos do host, possibilitando utilizá-lo em tempo de execução. Esse comando também cria um arquivo referente ao componente importado, possibilitando sua utilização de forma assíncrona.



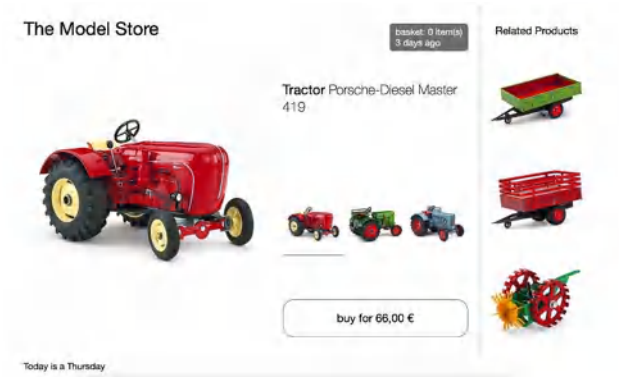
Assim como o componente *Basket*, do projeto de *checkout,* foi importado, o mesmo é feito com o *CheckoutButton*, também do projeto *checkout*, e além disso é importado o componente *RelatedProducts* do projeto *inspire*.

Como os arquivos criados na pasta de remotes já permitem sua utilização, os componentes são adicionados à página do host de produtos, fazendo que a página final se assemelhe àquela demonstrada inicialmente, simbolizada na imagem 2.1.



2.2.5 Inserção de dependências

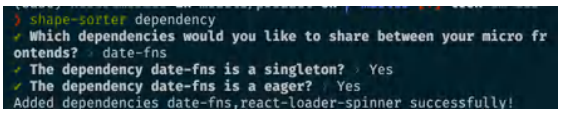
A fim de demonstrar o compartilhamento de dependências entre micro frontends foi adicionado uma biblioteca, em *projects* e em *checkout*, chamada date-fns para auxiliar na manipulação de datas. Em *projects,* no final da página ao lado esquerdo, é apresentado qual dia da semana é hoje, e em *checkout*, indicada no componente de *Basket,* a data de três dias atrás formatada.



Para compartilhar a biblioteca entre projetos, com o propósito de que não haja redundância de código ou divergência entre verões, o desenvolvedor pode inserir quais dependências gostaria de compartilhar entre os projetos com o seguinte comando:

$ shape-sorter dependency

O comando lista todas as dependências que ainda não foram compartilhadas e o desenvolvedor é apto a escolher aquelas que deseja compartilhar.



O comando exibe uma lista de dependências, e o desenvolvedor pode selecionar as que deseja adicionar. Após a seleção as seguintes perguntas são feitas, a cada dependência selecionada:

- A dependência é um singleton? (uma versão única)
- A dependência é eager? (é carregada no início do projeto)

2.3 Tecnologias utilizadas

2.3.1 Webpack

Como base para o funcionamento do micro frontend, foi utilizado o Webpack, que é em sua essência um empacotador de módulo estático para aplicativos JavaScript modernos. Quando o Webpack processa uma aplicação, ele constrói internamente um grafo de dependência que mapeia todos os módulos de que o projeto precisa e gera um ou mais pacotes (bundles). [4]

Atualmente, a forma mais factível de trabalhar com a estruturação de micro frontends é com Webpack Modules Federation. Esse plugin foi adicionado no Webpack 5, versão publicada em outubro de 2020. O Modules Federation permite disponibilizar módulos (projetos ou componentes de um projeto) de forma local ou remota. Módulos locais fazem parte da *build* atual de um projeto. Já os módulos remotos não fazem parte da build atual e são carregados dentro de um container em tempo de execução [5].

O Modules Federation permite que os deploys dos micro frontends sejam feitos de forma independente, sejam eles de repositórios, hosts, até de frameworks diferentes, e ainda assim permite o compartilhamento de dependências para que não haja duplicidade de código.

2.3.2 Prompts

O Prompts é uma ferramenta que auxilia na criação de prompts de comando de forma interativa [6]. Ela tem as principais características:

- Simples: os prompts não têm grandes dependências nem são divididos em uma dúzia de pequenos módulos que funcionam bem juntos;
- Fácil de usar: o prompt usa layout e cores para criar interfaces de CLI com uma melhor estética;
- Assíncrono: usa promises e async / await.
- Flexível: todos os prompts são independentes e podem ser usados individualmente.
- Testável: fornece uma maneira de enviar respostas de forma programática.
- Unificado: experiência consistente em todos os prompts.

2.3.3 Node.js

Node.js é um software de código aberto, multiplataforma, que executa códigos JavaScript no backend/servidor e frontend/interface, baseado no interpretador V8 de JavaScript em C++ do Google, mantido pela fundação Node.js em parceria com a Linux Foundation [7].

Uma das razões pelas quais o Node.js se tornou tão popular é o rico ecossistema de pacotes com mais de 900.000 pacotes no registro npm. Ao escrever CLIs em Node.js, é possível explorar esse ecossistema, incluindo sua grande quantidade de pacotes com foco em CLI.

2.3.4 React.js

O React é uma biblioteca JavaScript de código aberto com foco em criar interfaces de usuário em páginas web. É mantido pelo Facebook, Instagram, outras empresas e uma comunidade de desenvolvedores individuais. É utilizado nos sites da Netflix, Imgur, Feedly, Airbnb, SeatGeek, HelloSign, Walmart e outros [8].

O React.js foi utilizado como o template inicial do projeto. Permitindo que os componentes sejam criados facilmente e compartilhados entre vários projetos.

2.3.5 NPM

NPM é o gerenciador de pacotes para a plataforma Node JavaScript. Ele coloca módulos no lugar para que o node.js possa localizá-los e gerenciar conflitos de dependência de maneira inteligente.
É extremamente configurável para suportar uma variedade de casos de uso. Mais comumente, é utilizado para publicar,

descobrir, instalar e desenvolver programas em node.js. [9]

Com o NPM é possível criar um *link* no projeto, permitindo que possa ser acessado localmente em qualquer local na máquina que foi instalado.

Para que o projeto seja acessível pela comunidade, através do NPM, é necessário criar uma conta no site https://npmjs.com e com as credenciais realizar a autenticação na linha de comando. Após isso é possível fazer o upload para que seja usado de forma pública, sem necessitar de um *link* para utilização local.

3. Avaliação e Resultados

3.1 Coleta de dados

A ferramenta foi apresentada de forma síncrona, através de reuniões, e assíncrona através de emails, a um grupo de desenvolvedores, que foi tido como cliente. A instalação e utilização foram demonstradas por meio de reuniões e da documentação encontrada no repositório, e ao fim da experiência foi entregue a cada um dos participantes um formulário contendo as seguintes perguntas:

- Você teve alguma dificuldade ao instalar a ferramenta? Quais?
- Você conseguiu utilizar os comandos propostos pela ferramenta? Se não, por qual motivo?
- De 0 a 5, o quanto você acredita que a ferramenta auxilia na criação e manutenção de micro frontends?
- Quais os benefícios você acredita que estão sendo entregues com a ferramenta?
- Se você acredita que a ferramenta não será útil, qual seria o motivo?
- Quais suas sugestões para a ferramenta?

3.2 Análise das respostas

O grupo, que é composto por cinco desenvolvedores frontend, que variam em níveis de experiência, respondeu o formulário e suas respostas às perguntas citadas anteriormente são mostradas no seguinte documento: https://docs.google.com/spreadsheets/d/1PDeAAF8cxUzvzfpLBLtwUOT3mwIJqpb7krj7g_mDras/edit?usp=sharing .

Os desenvolvedores que não apresentaram experiência prévia com o conceito de micro frontends (20%) ou que conhecem o conceito mas nunca trabalharam em um projeto em produção (40%), conseguiram instalar e utilizar a ferramenta sem complicações, assim como os que já tinham experiência com o conceito e já tinham trabalhado em um projeto em produção.

Todos do grupo votaram 4 ou 5 (numa escala de 0 a 5), quando questionados de quanto a ferramenta auxilia na criação e manutenção de micro frontends. Mostrando que para esse grupo, que foi tomado como cliente, a ferramenta é relevante.

Quando questionados sobre os principais benefícios da ferramenta, todos responderam que a ferramenta diminui o tempo de criação e manipulação de projeto, dando mais velocidade ao desenvolvimento, e além disso que facilita a criação de projetos sem que haja uma pesquisa técnica aprofundada sobre o tema

Além das respostas, houveram sugestões como: adição de novos comandos e respostas pré-selecionadas, assim como a possibilidade de utilizar outros framework; inserção da possibilidade da utilização da ferramenta em um projeto existente e um melhor tratamento de erros.

4. Experiências e trabalhos futuros

A priori, a iniciativa de desenvolvimento da ferramenta partiu de um grupo de desenvolvedores, de uma empresa privada no ramo financeiro, que tinha a necessidade de facilitar o processo de criação e manutenção de seus projetos de micro frontends. Esse grupo foi considerado como o cliente, e a

ferramenta construída segundo suas necessidades.

Através de respostas ao formulário da seção anterior foi possível ver que, apesar de alguns desenvolvedores serem experientes com o tema, e já terem iniciado o desenvolvimento de seus projetos em micro frontends previamente, houve uma certa dificuldade para realizar uma configuração inicial, visto que não existe um consenso entre o mercado e a comunidade sobre um padrão de configuração de micro frontends, o que demandou um maior esforço para encontrar uma literatura que fosse suficiente para conduzir os desenvolvedores como um todo a configurarem seus projetos de forma adequada.

O grupo relatou que depois de criar e configurar o projeto, a experiência de manutenção, que inclui: adicionar novos módulos, compartilhar dependências entre projetos, exportar e importar componentes de um módulo, foi tida como monótona, tediosa e repetitiva.

A ferramenta foi desenvolvida para sanar os empecilhos descritos anteriormente, que se resumem em:

1. Dificuldade de encontrar material específico sobre o tema;
2. Falta de padrão de configuração entre a comunidade e o mercado;
3. Manutenção monótona, tediosa e repetitiva

Até a data de publicação deste artigo não foram encontradas ferramentas que auxiliassem na manipulação do Modules Federation de forma fácil e rápida, sendo necessário que os desenvolvedores fizessem uma pesquisa árdua para encontrar a melhor forma de estruturar seus projetos, e ainda assim mantê-los de forma enfadonha.

O desenvolvimento foi feito tomando como base os principais elementos adicionados no plugin de Modules Federation, na configuração do Webpack. Apesar de existirem algumas configurações avançadas, a manipulação dos itens mais comuns foi automatizada pelo *Shape Sorter*.

Como trabalhos futuros serão adicionados novos parâmetros nos comandos para que também possa ser feito a remoção de alguns itens, visto que foi desenvolvido apenas comandos de criação. Além disso, serão feitos tratamentos de erros mais elaborados, e adicionado suporte a diferentes linguagens e frameworks.

Referências:

[1] ThoughtWorks Radar Micro Frontends https://www.thoughtworks.com/radar/techniques/micro-frontends

[2] Micro frontends https://tautorn.github.io/micro-frontends/

[3] Martin Fowler Micro frontends https://martinfowler.com/articles/micro-frontends.html

[4] Concepts of Webpack https://webpack.js.org/concepts/

[5] Webpack Modules Federation https://webpack.js.org/concepts/module-federation/

[6] Prompts: Lightweight, beautiful and user-friendly interactive prompts https://github.com/terkelg/prompts

[7] Node.js https://en.wikipedia.org/wiki/Node.js

[8] React.js https://en.wikipedia.org/wiki/React_(JavaScript_library)

[9] About NPM https://docs.npmjs.com/about-npm