VLibras WikiLibras

DEV Guide

fev/2021

I. Introdução

O WikiLibras é uma plataforma para a construção e evolução colaborativa dos sinais do VLibras com a participação de uma comunidade de usuários através da Internet. Esta plataforma permite que uma rede de colaboradores possa criar, revisar, animar e avaliar sinais em Libras, e caso esses sinais sejam aprovados por especialistas, eles passarão a compor/complementar o dicionário de sinais da Suíte VLibras.

Este manual tem como objetivo apresentar os procedimentos para a instalação da ferramenta WikiLibras.

II. Tecnologias

O *Front End* WikiLibras foi construído utilizando o framework ReactJS. 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.

Nota: A versão do ReactJS utilizada 16.9.0 já possui suporte aos hooks.

Na parte do *Back End*, utilizamos o framework AdonisJS na sua versão 4.

Nota: A nova versão do Adonis NÃO possui suporte a versões anteriores, caso seja necessário instalar o *Back End*, certifique-se que a versão é a correta.

III. Bibliotecas

As principais bibliotecas utilizadas no *Front End* do WikiLibras possuem uma documentação completa e uma comunidade bastante ativa, evitando assim problemas futuros por questões de suporte.

Segue a lista de algumas bibliotecas essências utilizadas no projeto:

- Material UI - É uma biblioteca de componentes React para um desenvolvimento ágil e fácil.

- Axios - Axios é um cliente HTTP baseado em Promises para fazer requisições. Pode ser utilizado tanto no navegador quando no Node.js.
- Redux - O Redux é uma biblioteca para gerenciamento de estado que segue os princípios da arquitetura flux.
- Redux Saga - é uma biblioteca que visa tornar os efeitos colaterais do aplicativo mais fáceis de gerenciar e mais eficientes de executar.

> **Nota:** É necessário conhecimento básico das bibliotecas listadas acima para manutenção do código.

IV. Preparando ambiente

Nesta seção será descrito o passo a passo para preparar e executar o projeto *WikiLibras Front End*.

Sistemas Operacionais

Para o desenvolvimento e/ou manutenção do WikiLibras você pode usar qualquer um dos principais SOs no mercado (Windows, Linux ou MacOS).

> **Dica:** Recomendo alguma distro Linux pela simplicidade da instalação das dependências a seguir.

Browser

Você precisa ter um navegador instalado para executar o WikiLibras. Recomendo o Google Chrome ou Mozilla Firefox.

> **Nota:** Alguns navegadores (principalmente o Safari) podem apresentar problemas em algumas funcionalidades do WikiLibras. Por isso opte pelos recomendados acima.

Pré-requisitos

Antes de executar o projeto WikiLibras *Front End* precisamos de alguns pré-requisitos, são eles:

- NodeJS - Node.js é um software de código aberto, multiplataforma, que executa códigos JavaScript no backend/servidor e frontend/interface.
- Yarn - O Yarn é um gerenciador de pacotes para aplicar comandos prontos ao código de uma aplicação.

Dica: Recomendo utilizar o nvm para instalação do NodeJS. Com ele é possível instalar várias versões do Node e trocar de acordo com o projeto.

Instalando

Depois de ter feito o clone do repositório vlibras-wikilibras-frontend. Vá para o diretório do projeto:

```
cd vlibras-wikilibras-frontend
```

Agora instale todas as dependências do projeto com o comando:

```
yarn install
```

ou simplesmente

```
yarn
```

Se tudo foi executado corretamente você já está apto para executar o WikiLibras.

V. Preparando ambiente

Para executar em modo desenvolvimento, crie na raiz do projeto um arquivo chamado .env.development.local, dentro desse arquivo coloque as seguintes variáveis de ambiente:

```
REACT_APP_API_URL=

REACT_APP_REDIRECT_URI=

REACT_APP_CLIENT_ID=
```

REACT_APP_LOGIN_UNICO=

> **Nota:** Verifique quais são os valores dessas variáveis de ambiente no modo desenvolvimento.

Agora basta executar o comando:

```
yarn start
```

O seu navegador deverá abrir automaticamente com a seguinte tela:

VI. Preparando ambiente

Para executar em modo desenvolvimento, crie na raiz do projeto um arquivo chamado .env.production.local, dentro desse arquivo coloque as seguintes variáveis de ambiente:

REACT_APP_API_URL=

REACT_APP_REDIRECT_URI=

REACT_APP_CLIENT_ID=

REACT_APP_LOGIN_UNICO=

> **Nota:** Verifique quais são os valores dessas variáveis de ambiente no modo produção.

Agora basta executar o comando:

```
yarn build
```

Esse comando deverá gerar uma pasta chamada *build*, agora basta usar algum servidor para exportar essa pasta.

Você pode utilizar o pacote serve para testar sua build. Instale o pacote *serve* utilizando o comando:

```
yarn global add serve
```

Depois de instalado basta executar:

```
serve -s build
```

VII. Estrutura de Pastas

Nesta seção será descrito a estrutura de pastas do WikiLibras e seus principais arquivos.

- **`src`** - Diretório contendo todos os arquivos da aplicação, é criado um diretório src para que o código da aplicação possa ser isolado em um diretório e facilmente portado para outros projetos, se necessário;
    - **`assets`** - Diretório para armazenar arquivos em geral que possam ser utilizadas na aplicação (Ex. fontes, imagens etc.);
    - **`components`** - Diretório onde ficam os componentes da aplicação, como forma de padronização e boas práticas todo componente fica dentro de um diretório com seu nome;

- **`core`** - Diretório onde ficam dados/arquivos compartilhados com *Back End*;

    - **`consts.js`** - Arquivo com as constantes utilizadas no *Back End*. Caso seja necessário modificar esse arquivo certifique-se sempre com Back End se as informações estão sincronizadas;
- **`mock`** - Diretório onde ficam salvos dados estáticos para teste da nossa aplicação (Alguns dados estão desatualizados, verifique sempre a versão de desenvolvimento da API);

- **`layouts`** - Diretório onde ficam salvos os layouts a serem reutilizados nas páginas. No momento existem apenas 3 (três) layouts para reuso (Provavelmente não será necessário criar um novo *layout*);

- **`pages`** - Diretório onde ficam as páginas (telas) da aplicação, como forma de padronização e boas práticas toda página fica dentro de um diretório com seu nome;

- **`routes`** - Diretório onde ficam todas as rotas da aplicação (Mapeamento das páginas para uma *url*);

- **`services`** - Diretório onde serão criados os arquivos relacionados a serviços utilizados na aplicação, por exemplo, requisições HTTP, autenticação ou qualquer outro serviço que for utilizado;

    - **`VlibrasService.js`** - Arquivo com a configuração da biblioteca Axios para envio de requisições HTTP. Todas as rotas e chamadas estão presente nesse arquivo (Atenção!! Como o sistema é construído em *JavaScript* verifique quais os dados (e tipos) serão retornados pela API);

- **`store`** - Diretório onde ficam toda parte de gerenciamento de estado da aplicação (*reducers/actions/sagas*);

    - **`modules`** - Diretório onde ficam todos os estados separados por módulos, cada módulo deve possuir um arquivo de *reducer*, *action* e se necessário *saga*.

        - **`rootReducer.js`** - Arquivo responsável por juntar todos os *reducers* de cada módulo;
        - **`rootSaga.js`** - Arquivo responsável por juntar todos os *sagas* de cada módulo;
    - **`index.js`** - Arquivo responsável por exportar todos os *reducers/sagas* e criação do estado global da aplicação;

- **`styles`** - Diretório onde ficam todos estilos globais da aplicação (Ex. paleta de cores, espaçamentos etc.);

- **`utils`** - Diretório onde ficam todos os arquivos utilitários utilizados na aplicação (Ex. formatação de data, configuração de tabelas etc.)

- **`App.js`** - Arquivo responsável por centralizar o código do diretório src, nele são chamadas as rotas tal como qualquer outra configuração que precise ser executada na inicialização da aplicação, ele é como um *Entry Point* do diretório src;

- **`index.js`** - Arquivo raiz da aplicação, também chamado de *Entry Point*, é o primeiro arquivo chamado no momento do build e execução da aplicação;

- **`.editorconfig`** - Arquivo destinado à configuração do plugin Editor Config, que padroniza algumas configurações para o editor em diferentes ambientes;

- **`.eslintrc.js`** - Arquivo de configuração do ESLint, é nele que são inseridas as regras e configurações de Linting do projeto, tal como a configuração do Resolver para o Babel Plugin Root Import;

- **`jsconfig.json`** - Arquivo de configuração do JavaScript no Editor, ele é o responsável por ativar o Auto Complete de códigos JavaScript na aplicação;

- **`package.json`** - Arquivo com todas as dependências utilizadas no projeto;

Dicas

Para ReactJS/React Native sempre recomendo cursos e conteúdos da Rocketseat, eles disponibilizam diversos conteúdos grátis e uma comunidade bastante ativa nas redes sociais.
