ARQUIVO DE ANOTACOES REFERENTE AOS MODULOS DE REACT.

Diferenças entre React / ReactJS / React Native

React: Biblioteca de construção de interfaces e componentizacao.

ReactJS: Estamos no refererindo no comportamento do react diretamente no browser - quando o react se junta com o reactDOM (controla DOM do browser) vira o react JS.

React Native: soma do react + outra biblioteca que cuida de elementos nativos.

Criar uma pasta para o novo modulo e executar o yarn init -y
Acesse essa pasta pelo vscode e vamos instalar as bibliotecas para trabalhar com o react, execute o codigo abaixo para instalar tudo como dependencia de desenvolvedor:

“yarn add @babel/core @babel/preset-env @babel/preset-react webpack webpack-cli -D”

@babel/preset-env: responsavel por alterar funcionalidades do JS que o navegador ainda não entende;

@babel/preset-react: responsavel por transformar as coisas que o navegador não entende do react

agora execute :

“yarn add react react-dom”

crie um arquivo babel.config.js e configure o export dos presets
após crie um arquivo webpack.config.js, que nele sera exportado as configuracoes do webpack.
Na pasta src fica os arquivos de desenvolvimento com as funcionalidades que utilizamos, crie uma pasta public onde sera armazenado os arquivos já transpilados para a linguagemq que o JS e o navegador entendam.

Execute o comando abaixo para instalar o babel loader na aplicação e continue configurando o webpack.config.js,

“yarn add babel-loader -D”

Apos configurar o webpack, va no package.json, crie um novo script e dentro dele crie um script build, isso ira pegar o arquivo JS e iram converter para uma maneira que o navegador entenda.

Para executar esse script utilize a sintaxe:
“yarn build”
Ela vai criar o nosso arquivo bundle dentro da pasta public, nele estarao todos os codigos já transpilados para forma que todos os browsers entendam
após isso crie um arquiivo index.html na pasta public escreva html e seleciona a opção do html:5

dentro do body crie um script com src do bundle.js e faça algumas alteracoes no codigo html

Para ativar a funcionalidade de sempre que mudarmos algumas coisa no codigo o servidor atualizar automaticamente iremos instalar uma funcionalidade pelo seguinte codigo.

“yarn add webpack-dev-server -D”
depois disso va no webpack config e crie uma nova propriedade chamada devServer e configure. Depois disso va no package.json e crie um novo script chamado dev para webpack-dev-server, isso ira rodar nossa aplicacao

em um localhost que sera mostrado no terminal, com isso sempre que algo for atualizado e salvo no arquivo de js a pagina ser atualizada.
A questao do –mode development ele traz um bundle mais legivel e de entendimento mais facil porem dessa forma ele fica mais pesado, se colocarmos –-mode production ele ira gerar um bundle reduzido e menos legivel porem sera muito mais leve, ou seja build>producao / dev>desenvolvimento

Iremos começar a utilizar o react na aplicação

Nao e necessario fazer nenhuma config para que o webpack e o babel entendam pois já fizemos isso anteriormente no arquivo babel.config.js

Iremos mexer no arquivo index.js e criar um arquivo App.js que ficara contido o componente principal da nossa aplicação em react.

Iremos instalar os loaders de CSS para podermos utrilizar CSS na nossa aplicação:

“yarn add style-loader css-loader -D”

Vá no webpack config e crie uma nova rule, para testar arquivos css

Agora iremos instalar o loader de carregamento de imagens
“yarn add file-loader -D”

Agora precisamos criar uma pasta components dentro da pastta src, que ira armazenar a maioria dos componentes menos o App que sera o componente principal.

Para o babel entender as declarações resumidas de state precisamos instalar o plugin abaixo:
“yarn add @babel/plugin-proposal-class-properties -D”

Va no babel config e crie uma nova propriedade chamada plugins.

Default props exxplicacao no arquivo trechlistOlá

Prop-types serve para deixar visivel caso o dev esteja informando coisas erradas no codigo, onde deveria ser função esta um texto…
para isso instale a biblioteca abaixo:
“yarn add prop-types”

Ciclo de vida do componente:
os principais metodos do componente abaixo

```
  // Executado assim que o componente aparece em tela
 componentDidMount(){

 }

 //Executado sempre que houver alteracoes nas props ou est
ado
 componentDidUpdate(prevProps, prevState) {
  //this.props, this.state - apos o update caso seja neces
sariompodemos pegar os dados de props ou state antes de se
rem atualizados e comparar com os dados de agora.
 }


 // Executado quando o componente deixa de existir.
 componentWillUnmount(){

 }
```