**React js**

**React** é uma biblioteca JavaScript para construção de interfaces de usuário.

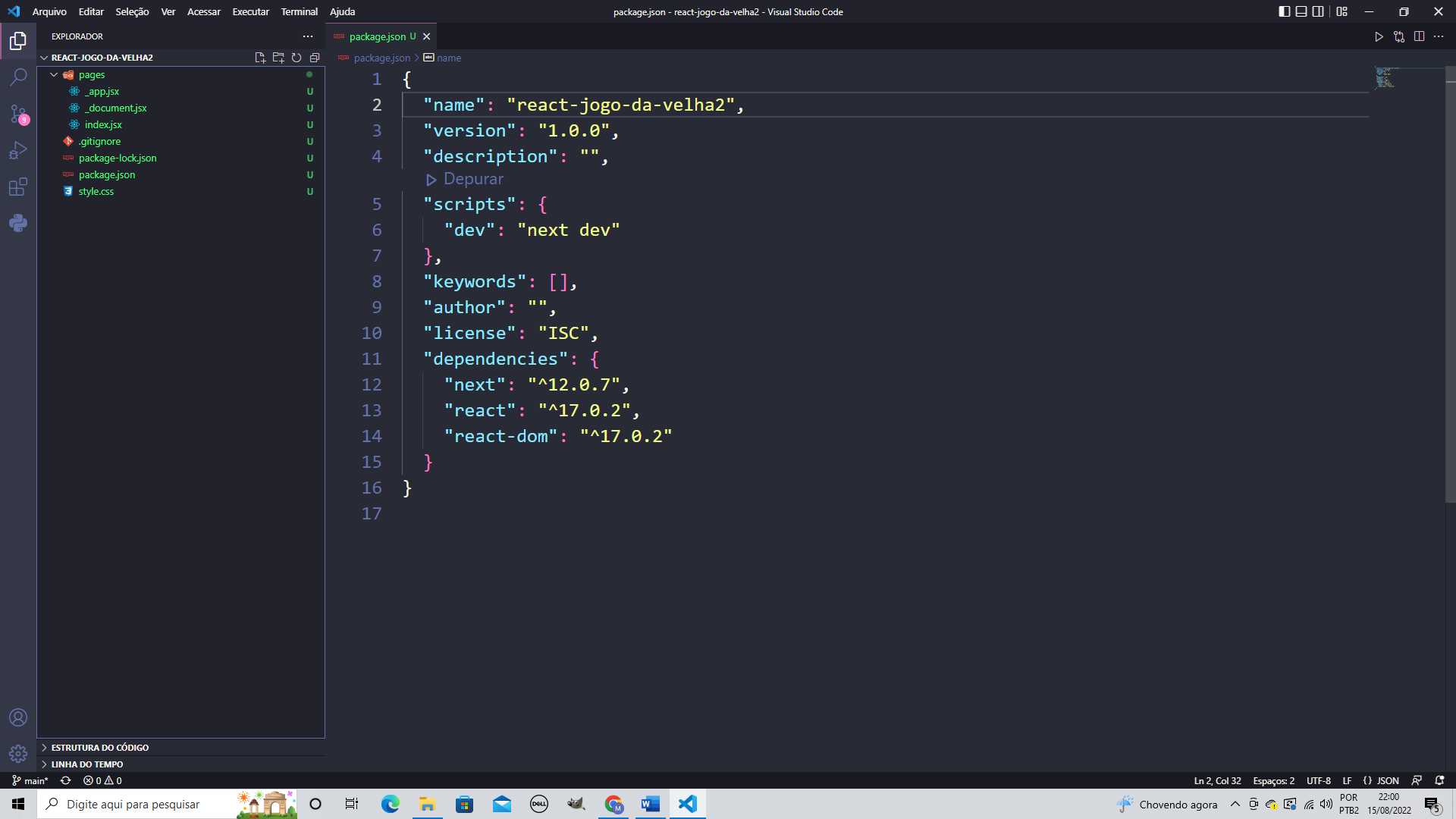
O **React** é uma biblioteca JavaScript declarativa, eficiente e flexível para criar interfaces com o usuário. Ele permite compor UIs complexas a partir de pequenos e isolados códigos chamados “components”.

**Projeto Jogo da velha**

1 – **Vídeo 02** – Iniciando o Projeto: Baixar o projeto do **github**. Clicar em **Code,** depois **Download ZIP**

2 – Após extrair os arquivos, renomear a pasta para o nome desejado e abrir no editor.

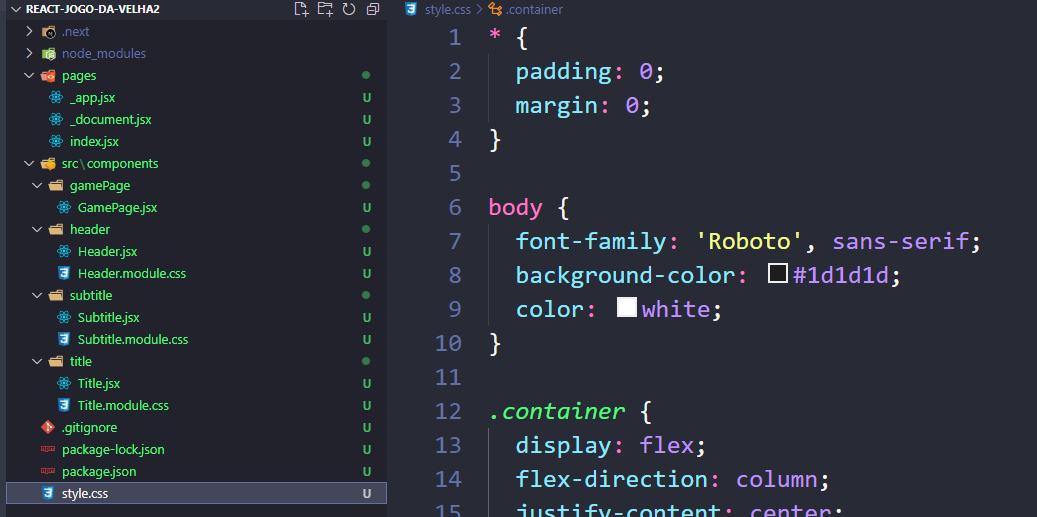
3 – O projeto inicial consta com apenas alguns arquivos:



4 – Baixar as bibliotecas do projeto. Sempre que começamos um projeto que não fomos nós que iniciamos, devemos baixar o arquivo **node\_modules**. Para isso devemos digitar no terminal: **npm i**

5 – **Crtl l**: limpa. E depois, abrir o site com o comando: **npm run dev**

6 – Trocar cor de fundo através de **background-color:** **#1d1d1d;** (preto). Devemos fazer isso no **.body**, em estilo global: **styles.css** Outras alterações são em **module.css**

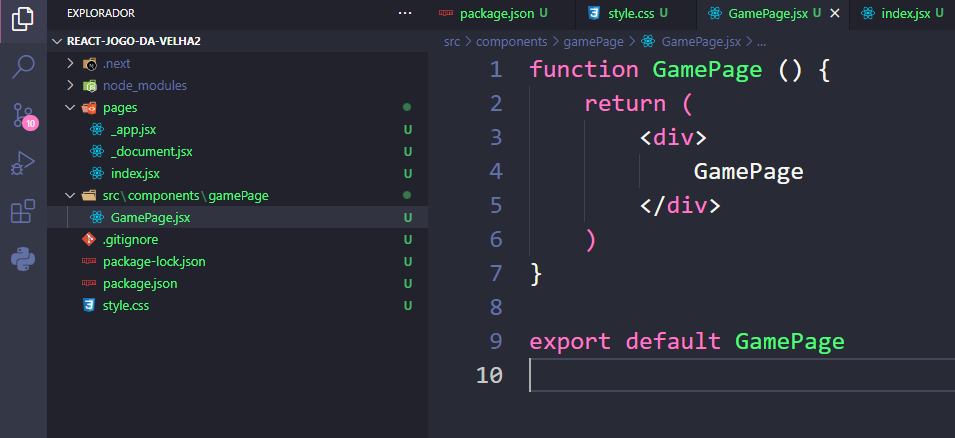
****

7 – Trocar cor da fonte, **color: White;**

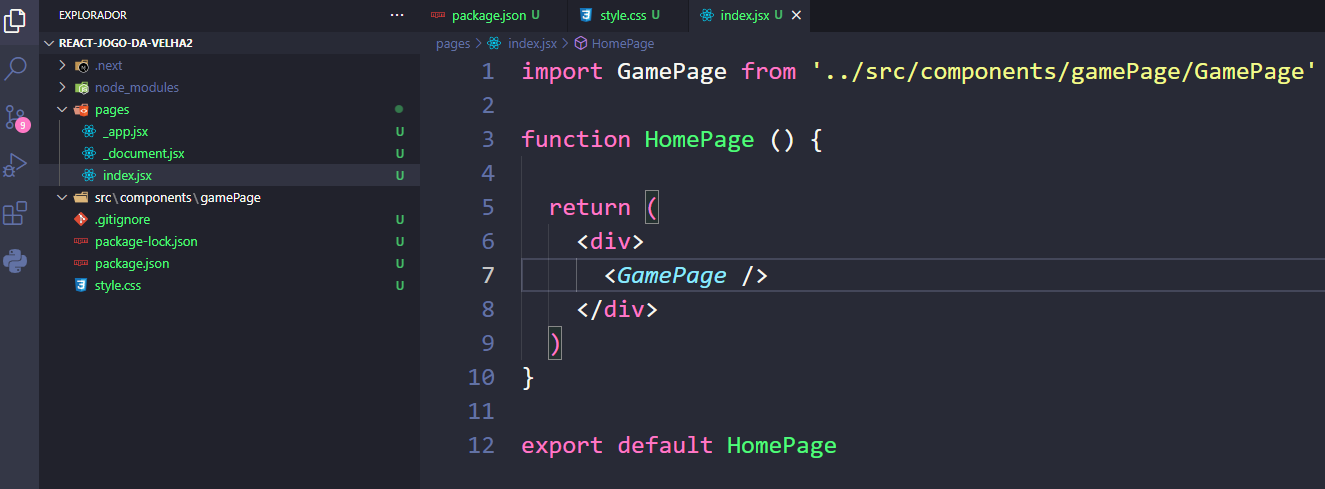
8 – Criar pastas: **src/components/gamePage**

9 – Criar arquivo: **GamePage.jsx** (jsx é quando temos um componente react).

10 – Dentro de **GamePage.jsx**:



11 – Em **index.jsx**, importar **GamePage.** Trocar **SejaDev** por **<GamePage />** (dentro da **div**)



12 – Em **GamePage.jsx** criar **h1** Jogo da velha e **parágrafo** Criado por Marcos Passos



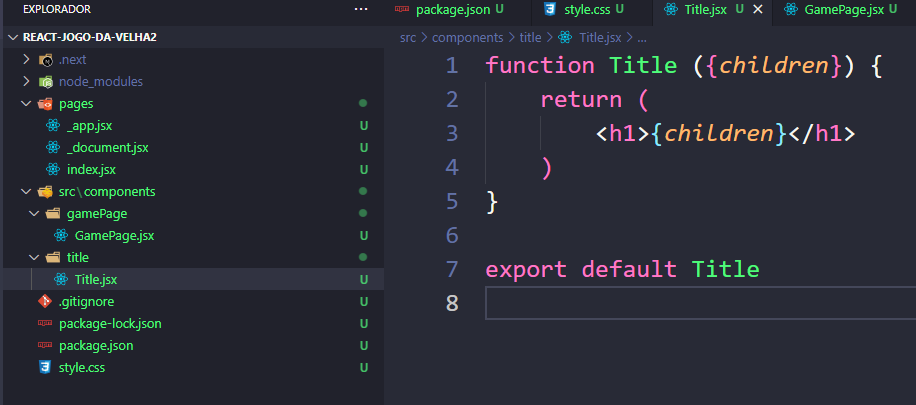
13 – Criar **className=”container”** na div do **gamePage.jsx**, trazendo essa configuração do estilo global **styles.css**



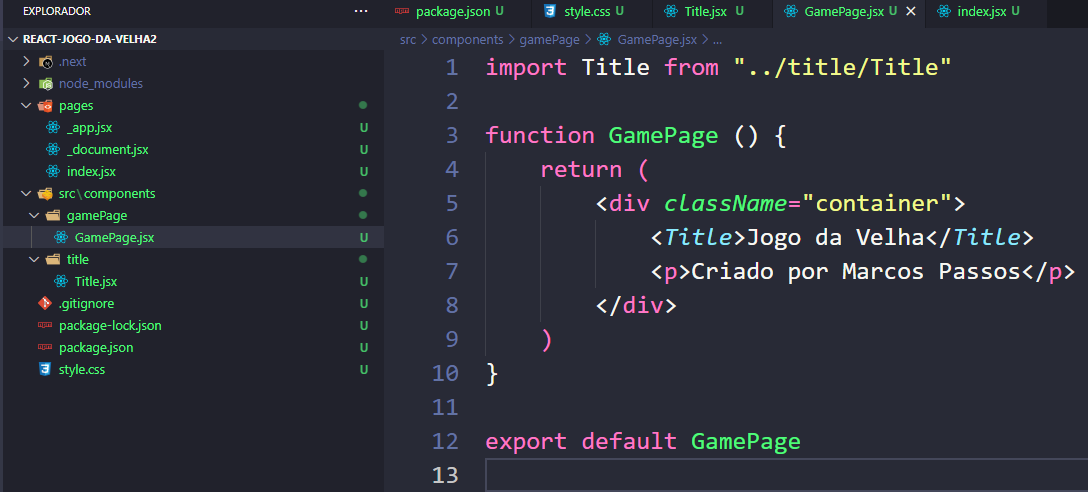
14 – Criar **.container** dentro de **styles.css** com **display flex, flex direction etc:**

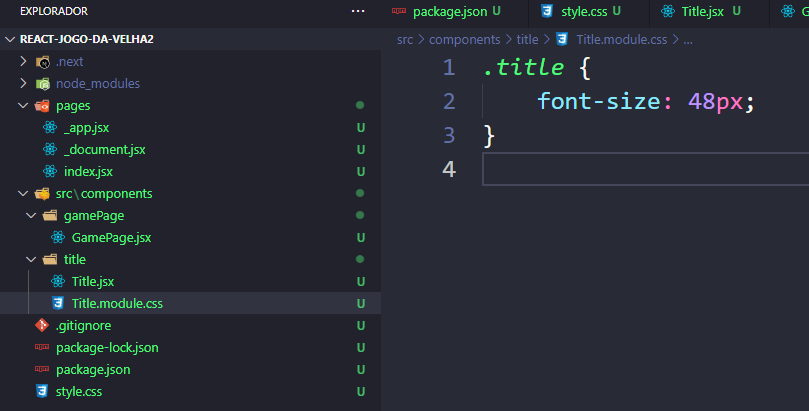


15 – Em **components,** criar pasta **title** e arquivo **Title.jsx.** Em **Title.jsx**, criar **children**:



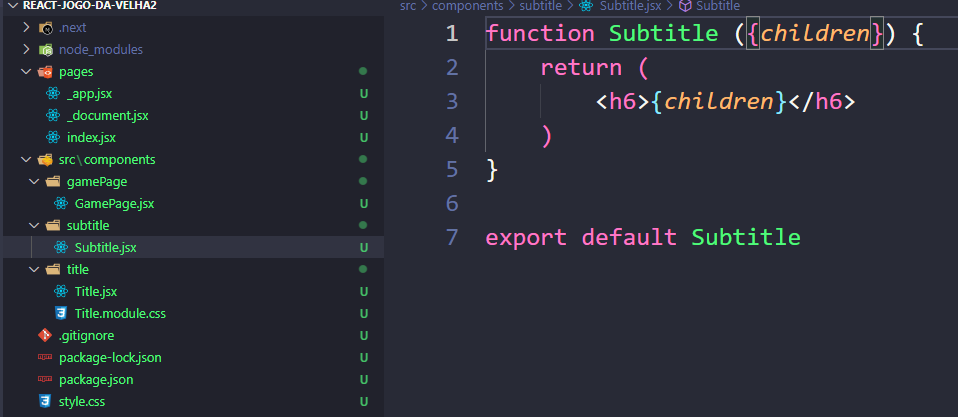
16 – Em GamePage.jsx importar o Title e tirar o h1 e colocar o Title:



17 – Dentro da pasta **title** criar arquivo **Title.module.css** com **font-size** de **48px**

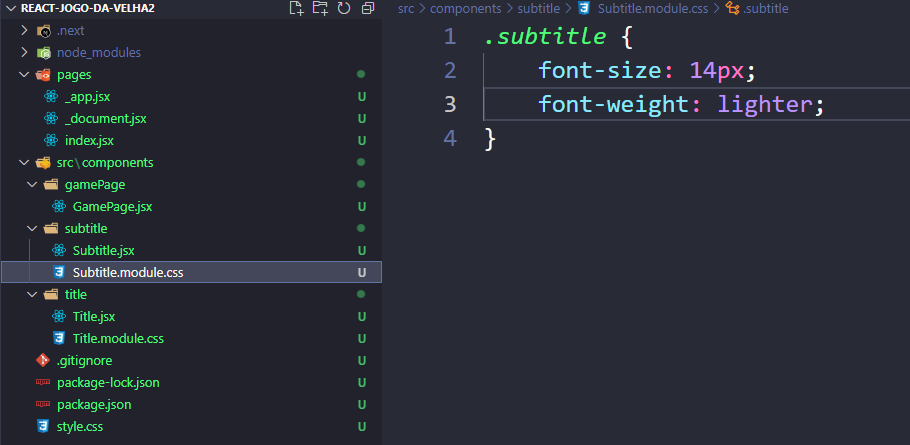
18 – Importar esse **.title** para dentro do **Title.jsx,** só que agora com **className** com chaves:



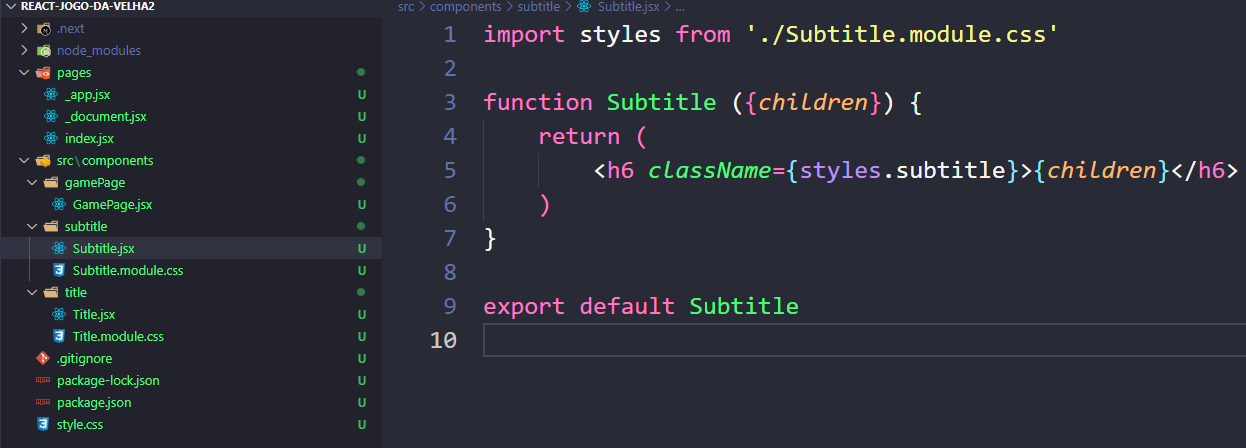
19 – 18:30 – Criar dentro de components a pasta **subtitle.** Depois criar arquivo **Subtitle.jsx**

20 – Importar o **Subtitle** dentro do **GamePage.jsx.** Trocar o **<p>** por **Subtitle.**

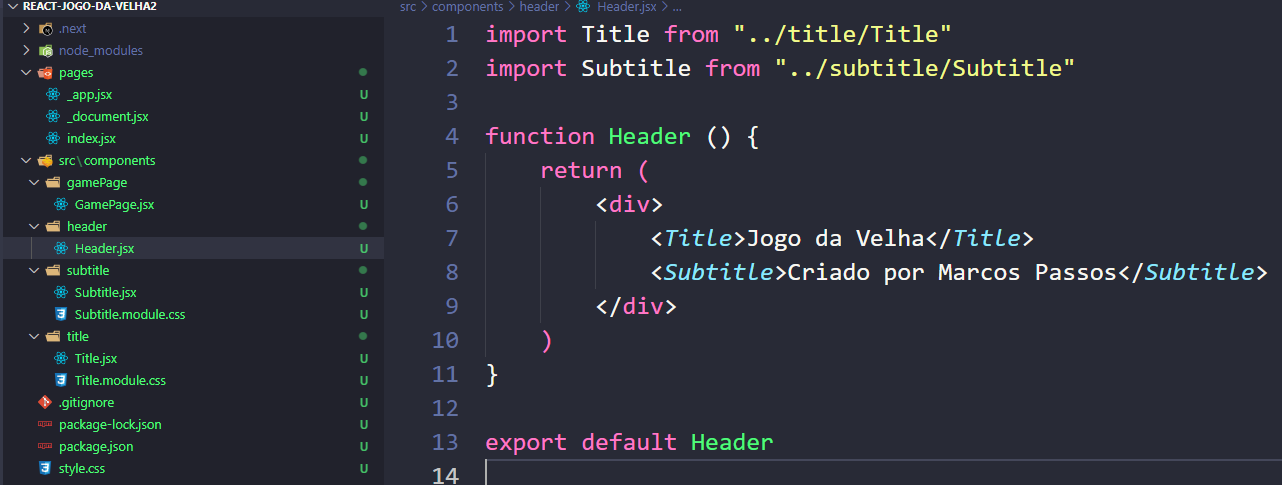
****

21 – Criar dentro da pasta **subtitle** o arquivo **Subtitle.module.css**

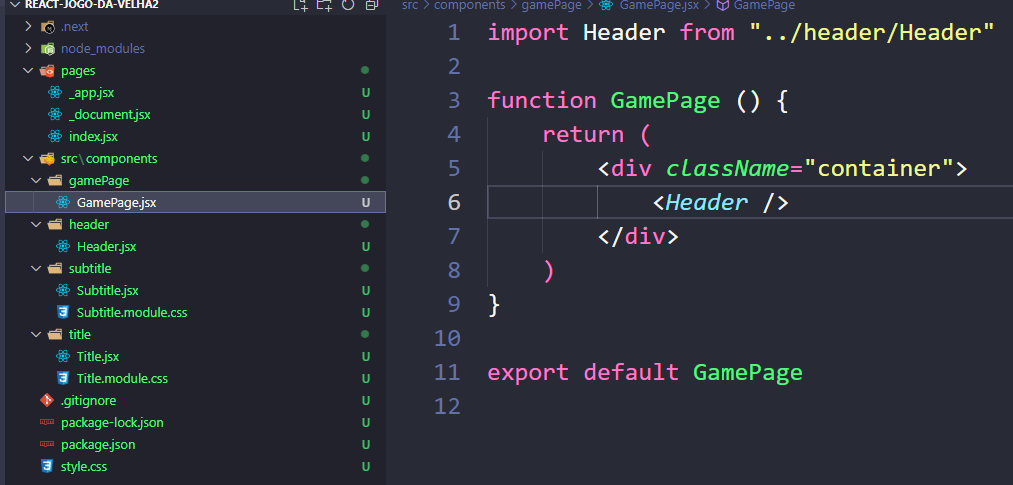
22 – Importar o **Subtitle.module.css** para dentro do Subtitle.jsx



23 – Em **compenents** criar pasta **header** e arquivo **Header.jsx.** Importarpara dentro de **Header.jsx, Title** e **Subtitle.**

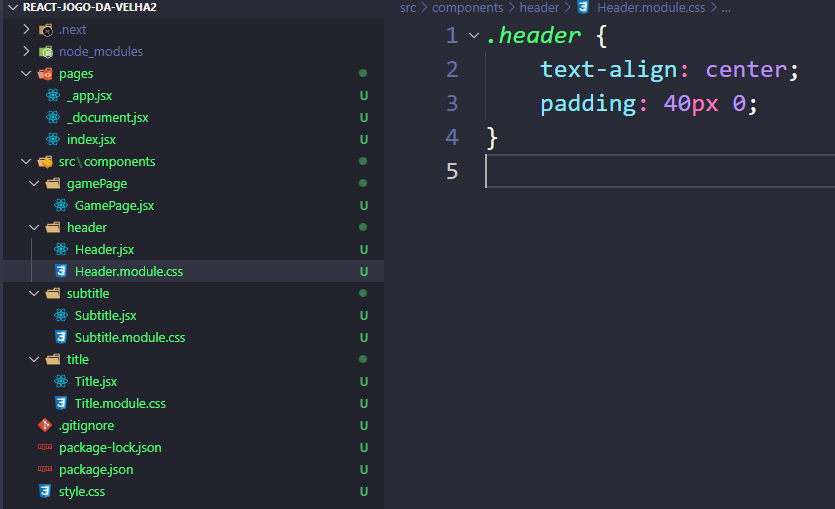
****

24 – Retirar de dentro de **GamePage.jsx** os import **Title** e **Subtitle** que foram passados para o **Header.jsx** e importar o **Header**.



25 – Na pasta **header**, criar arquivo **Header.module.css** e importa-lo para dentro de **Header.jsx**. Colocar na **div**, o **className.**



26 – Em **Header.module.css,** centralizar e alinhar o texto:

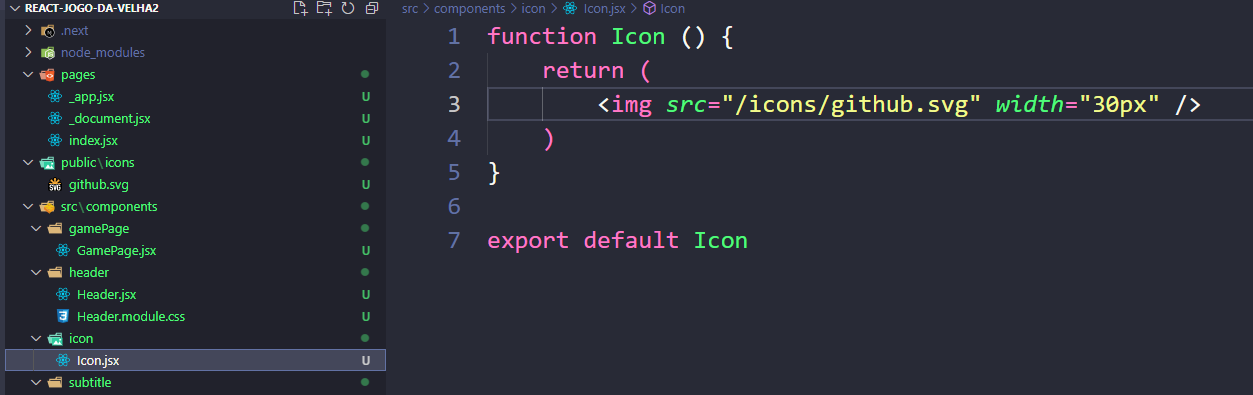
27 :

1. Iniciar projeto no **github** com o comando **git init.**
2. Colocar o nome do projeto (o ideal é o nome que está na pasta) e mandar para lá.
3. Trocar a **branch** de **master** para **main:** *git branch -M main*
4. Colocar o *git remote add origin https...(link)*
5. Fazer o **commit**, inicializando o projeto, colocando o nome do **commit**, adicionar os arquivos criados ou modificados (que pode ser clicando no maiszinho) ou digitando **git add .**
6. Clicar em **ctrl + enter,** depois clicar em **Publish branch** oudigitar **git push origin main**
7. 

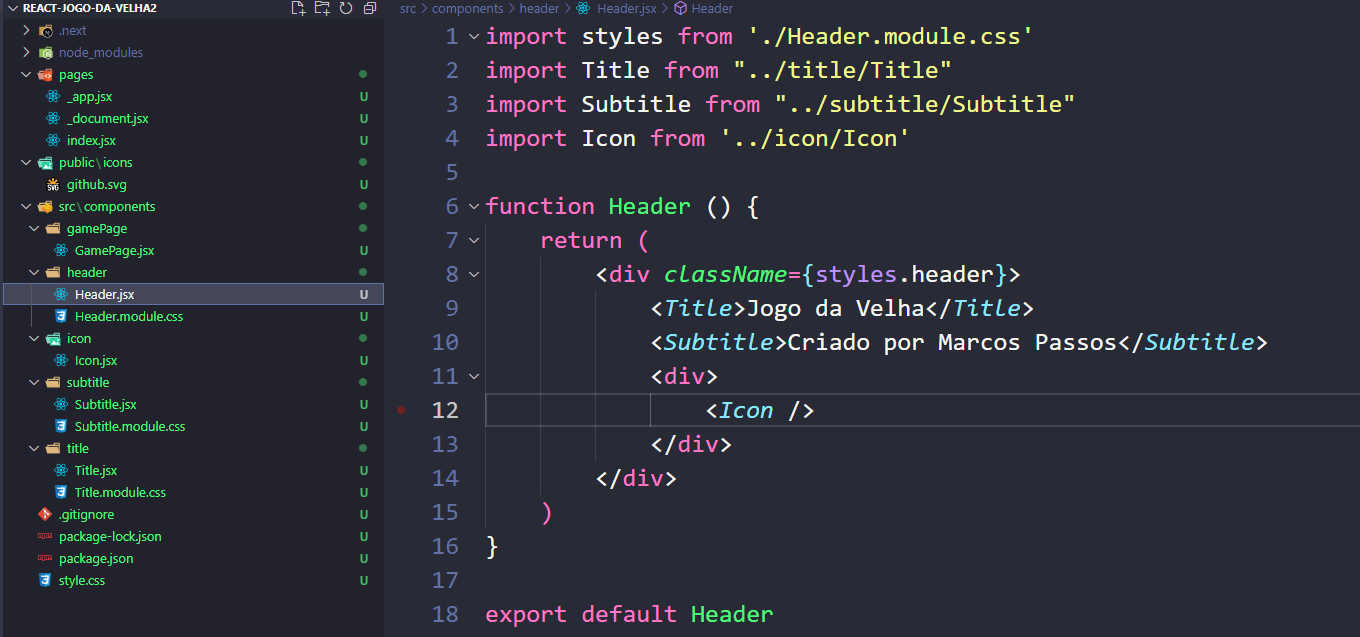
28 – **Vídeo 03 - Criando componente de ícone.** Agora vamos concluir o header. Baixar o ícone do github (conforme o link oferecido pelo curso) e salvar na pasta do projeto: github.svg.

29 – Depois que a imagem apareceu no **visual studio code**, criar pasta **public** e depois outra pasta **icon** e colocar a imagem dentro de **icon**.

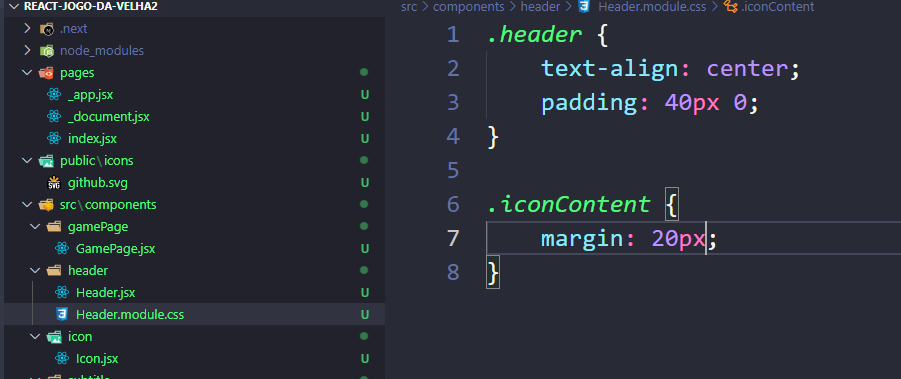
30 – Em components criar pasta icon, e dentro dela criar arquivo Icon.jsx:



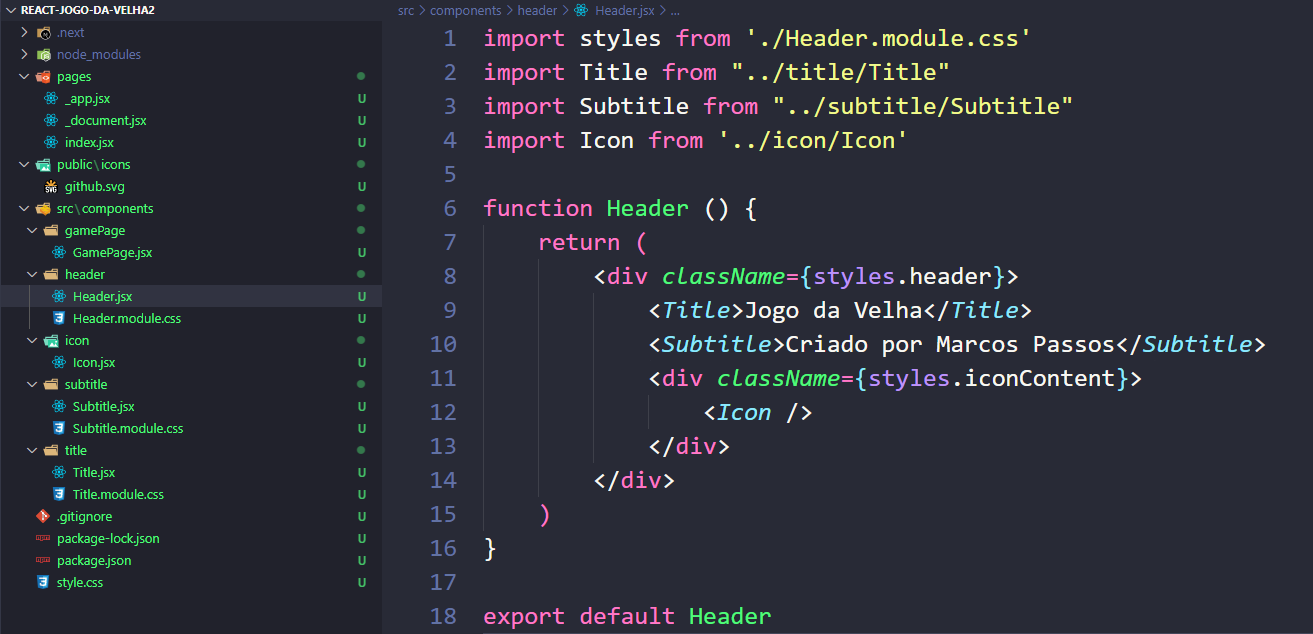
31 – Dentro de **Header.jsx**, importar o **Icon** e criar uma div com o **<Icon />**



32- Observar como está a página e ver se está tudo ok (**npm run dev**). Agora criar uma margem, uma distância entre o ícone e a frase do título. Para isso criar dentro do arquivo **Header.module.css, margin: 20px**

****

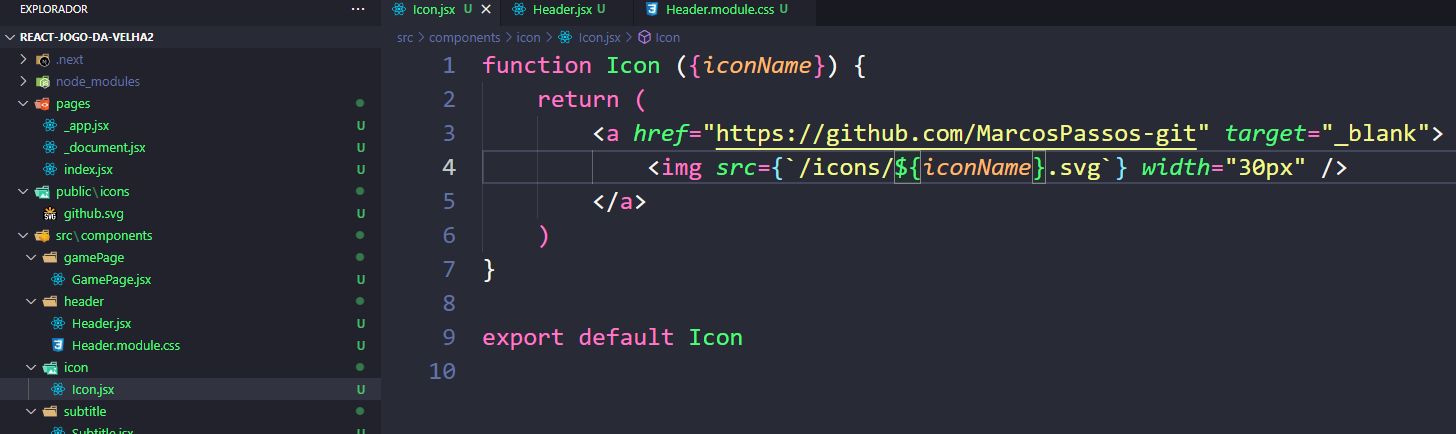
**33 –** Criar a **className: <div className={styles.iconContent}>** na **div** do **Icon** dentro do arquivo **Header.jsx:**

****

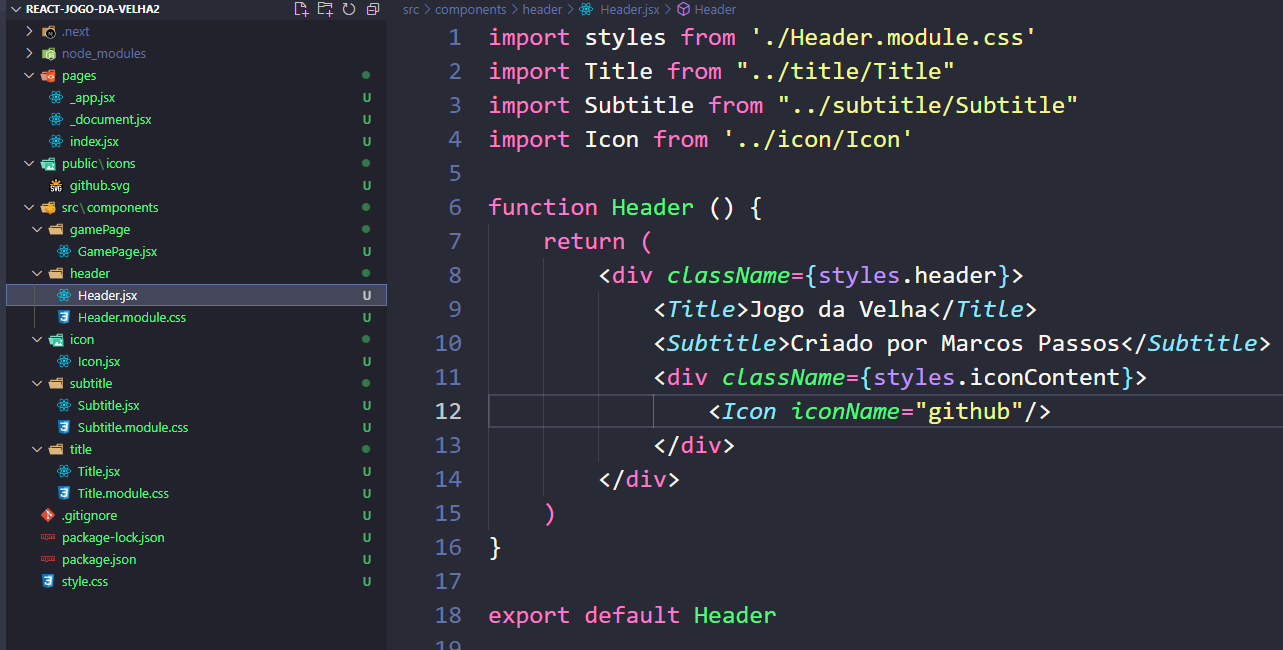
34 – Em Icon.jsx criar um link para o meu github. Para isso colocamos um “a” e colocar a imagem dentro dele. Digitar target para abrir em nova pagina.



35 – Dentro de **Icon.jsx**, passar uma propriedade para o componente criando o **destroctor** (ou uma **props.** alguma coisa), mudando toda a escrita que está dentro do src, com as crases, cifrão e chaves:



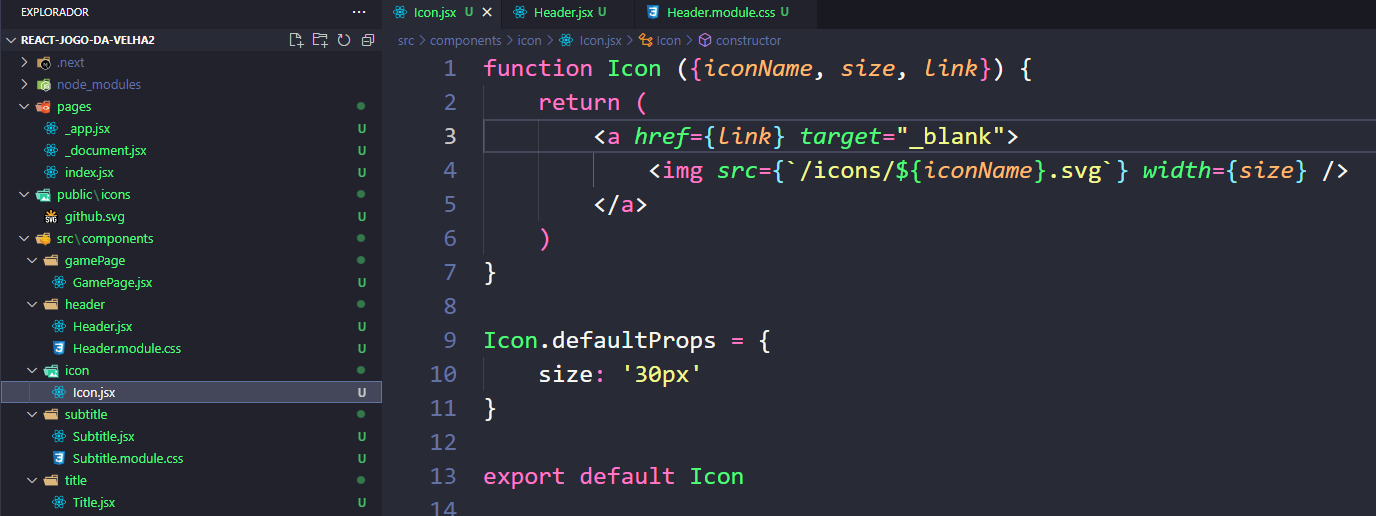
36 – Em **Header.jsx** colocar no **Icon** o **iconName:**

****

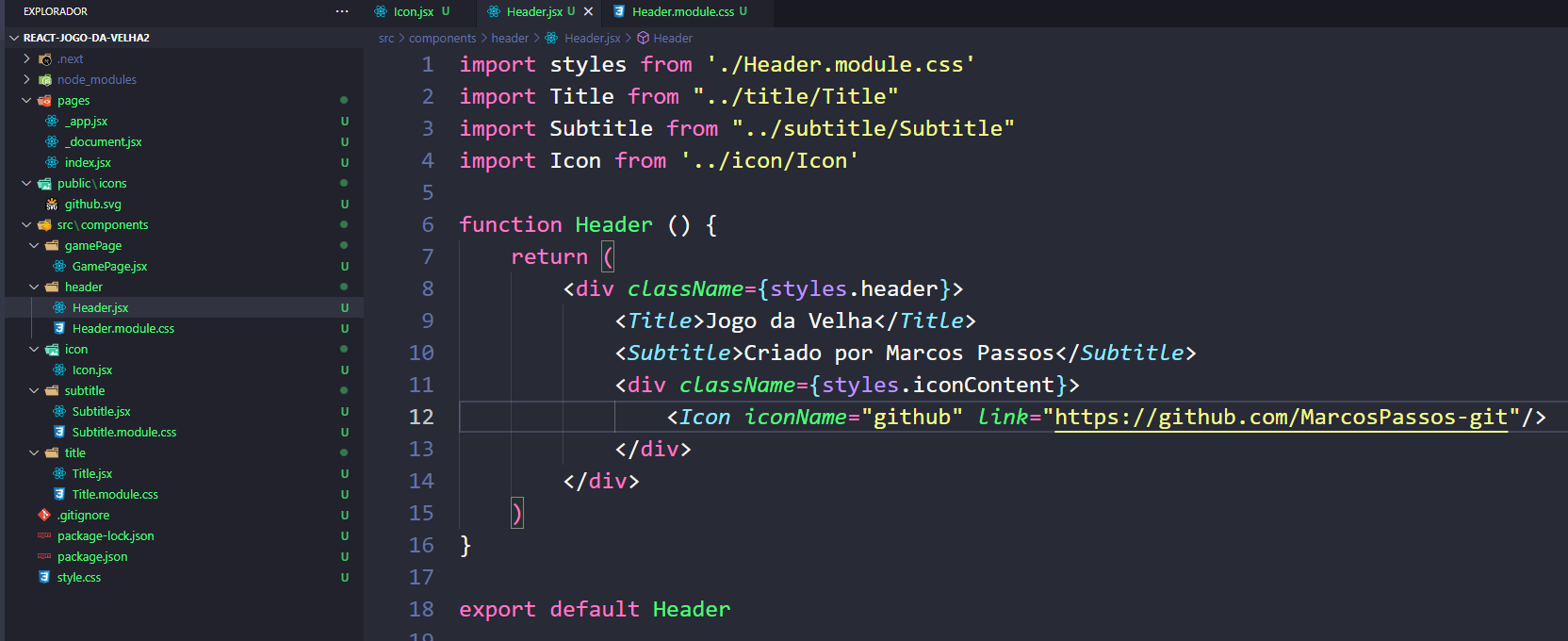
37 – Em **Icon.jsx** criar **props size** e **Icon.defaultProps = {}**

****

38 – Criar **props link** em **Icon.jsx:**

****

39 – Colocar **link** em **Header.jsx (no Icon):**

****

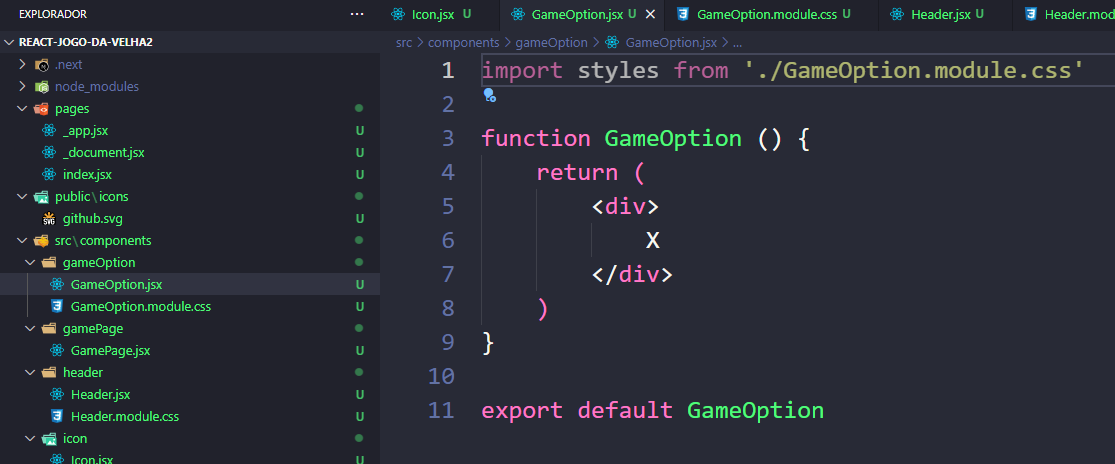
40 – Finalizar **commit** e enviar para o **github:** clicar em **controle de código**, clicar no **maiszinho**, renomear arquivo (Criado componente de ícone), clicar **ctrl enter** e depois **Publish branch** ou digitar **git push origin main:**



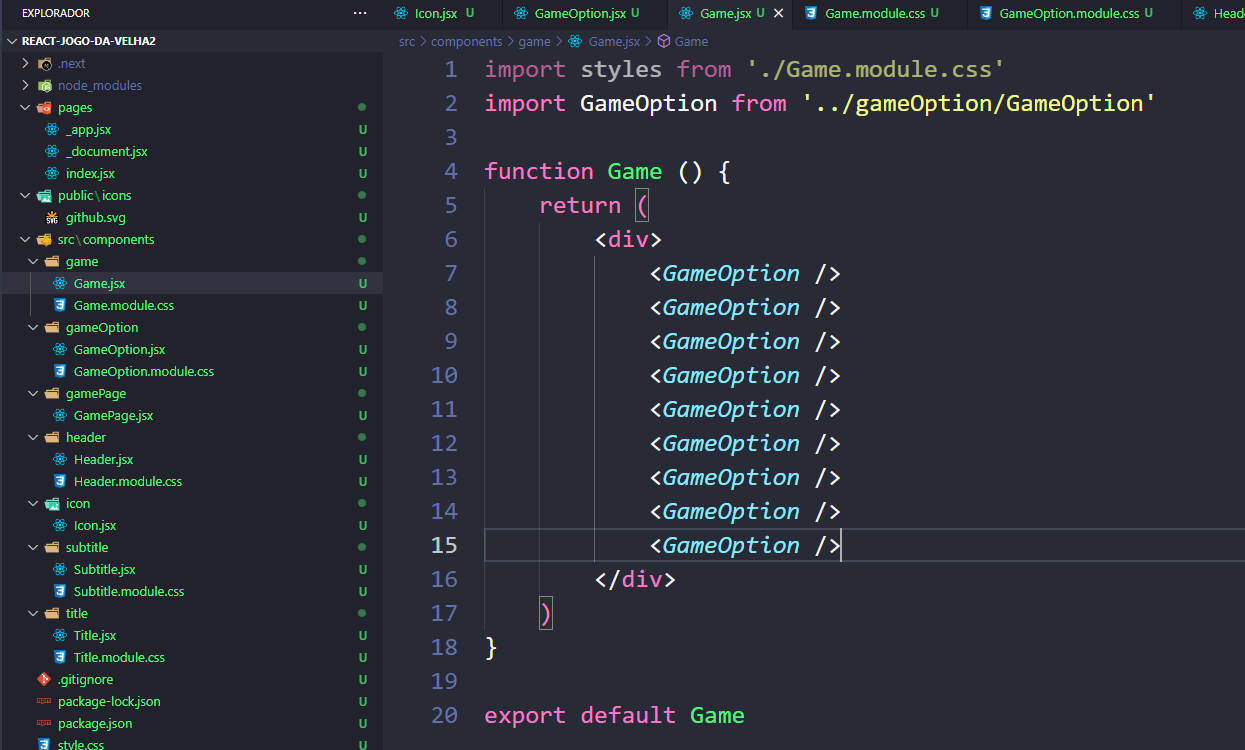
41 – **Vídeo 04 – Criando Board –** Em **components** criar pasta **gameOption**. Depois criar arquivo **GameOption.jsx**:



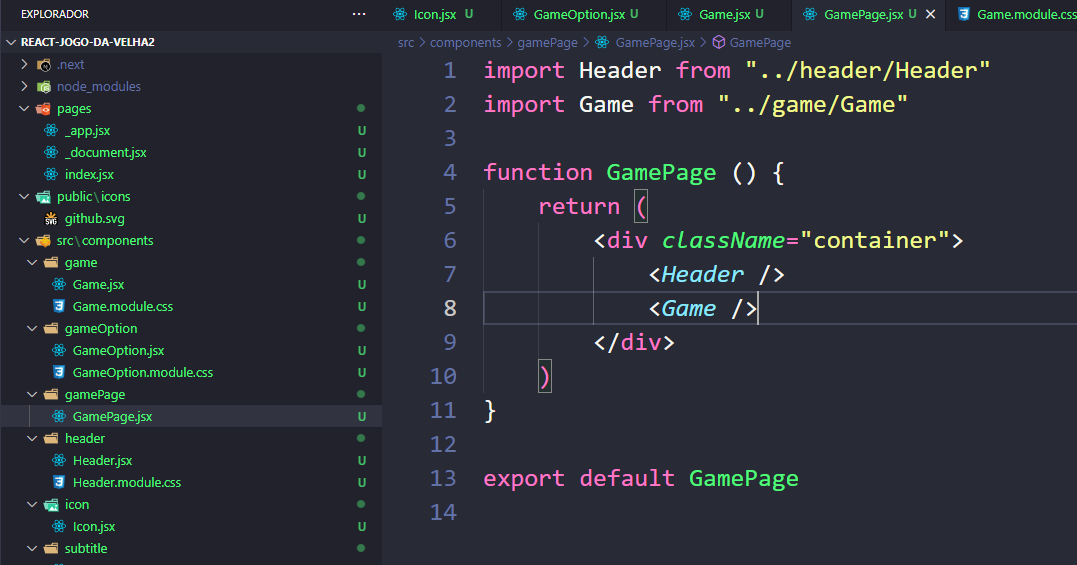
42 – Criar pasta **GameOption.module.css**. Depois importar para dentro de **GameOption.jsx**:



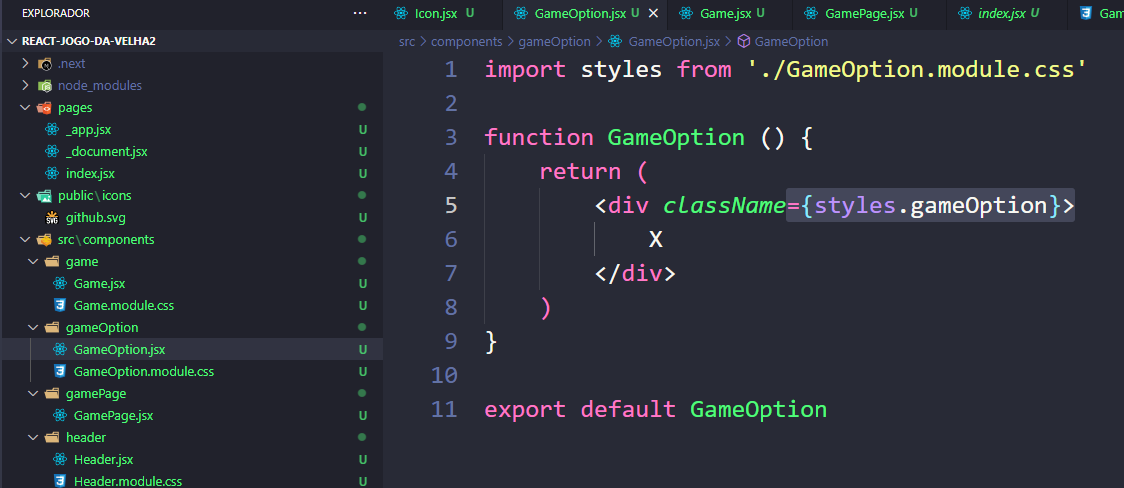
43 – Criar em **components** pasta **game**. Depois criar arquivo **Game.jsx**, e arquivo **Game.module.css**. depois importar o **GameOption** para dentro do **Game.jsx (**criando 9 **GameOption)**:



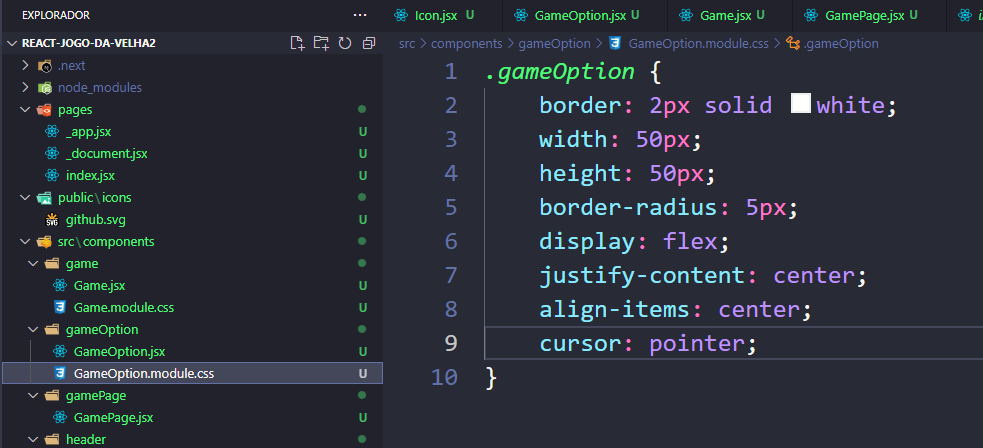
44 – Dentro do **GamePage.jsx**, importar o **Game.jsx**:



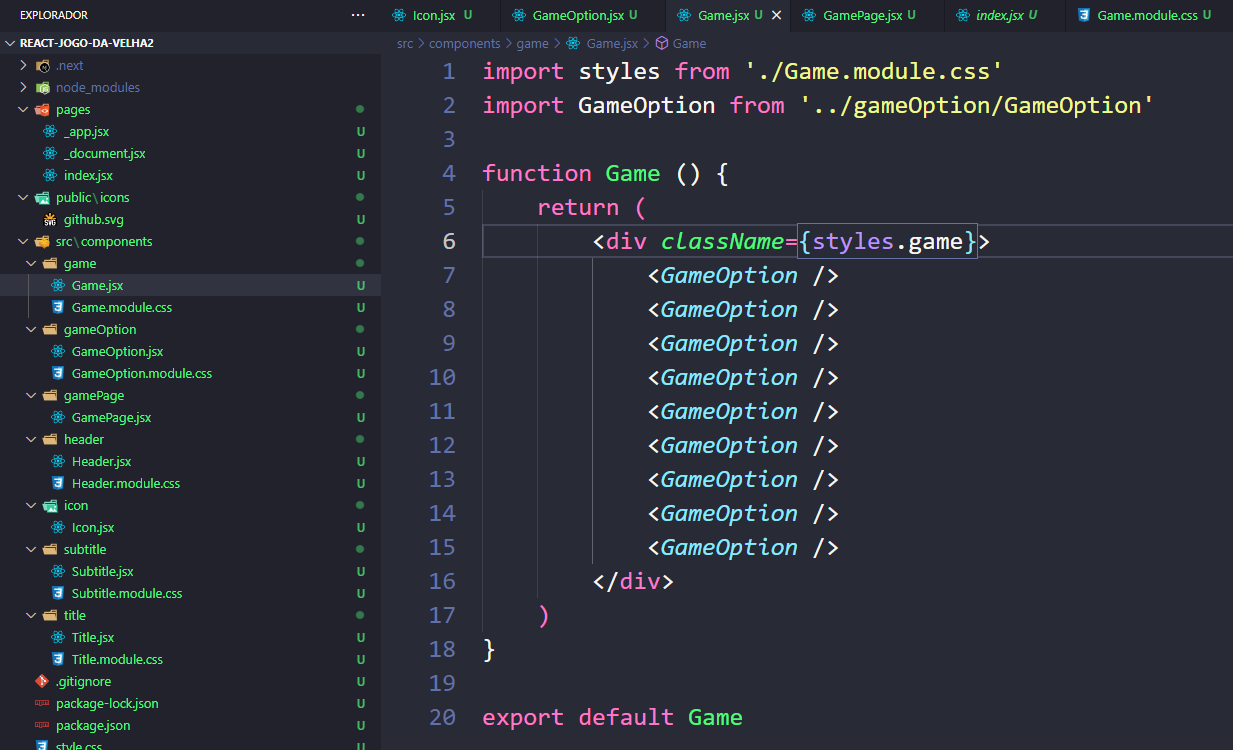
45 – Criar **className={styles.gameOption}>,** na **div** dentro do **GameOption:**



46 – Em GameOption.module.css, criar todas as configurações necessárias:



47 – Criar **className={styles.game}** na **div** dentro de **Game.jsx**



48 – Em Game.module.css criar todas as configurações necessárias:



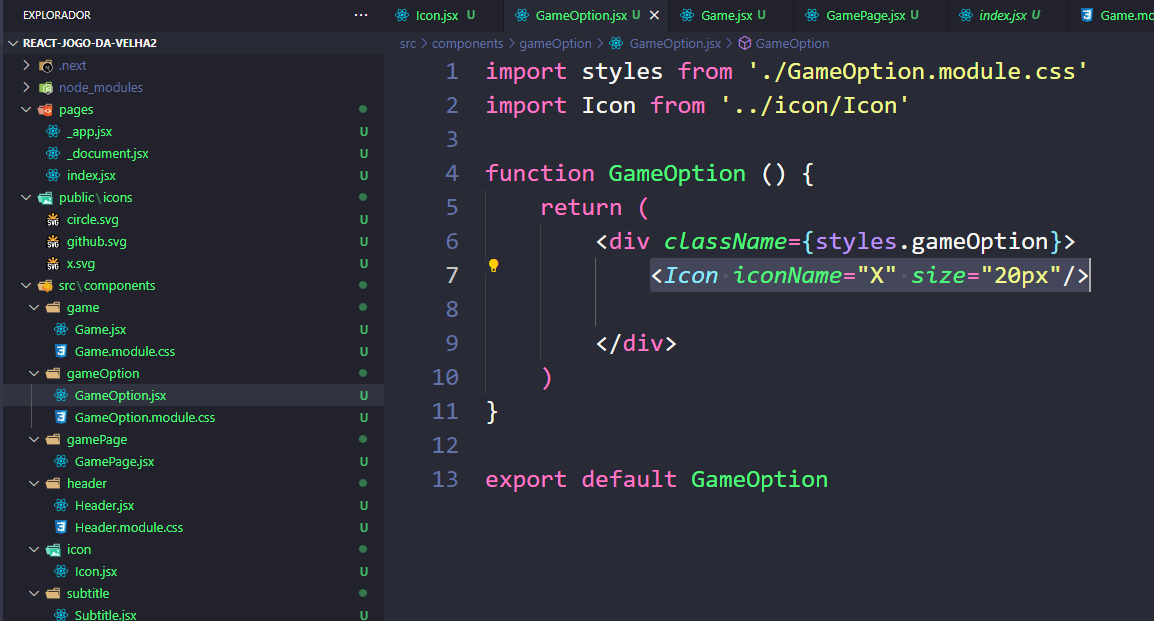
49 - Finalizar **commit** e enviar para o **github:** clicar em **controle de código**, clicar no **maiszinho**, renomear arquivo (Criado componente Game), clicar **ctrl enter** e depois **Publish branch** ou digitar **git push origin main:**



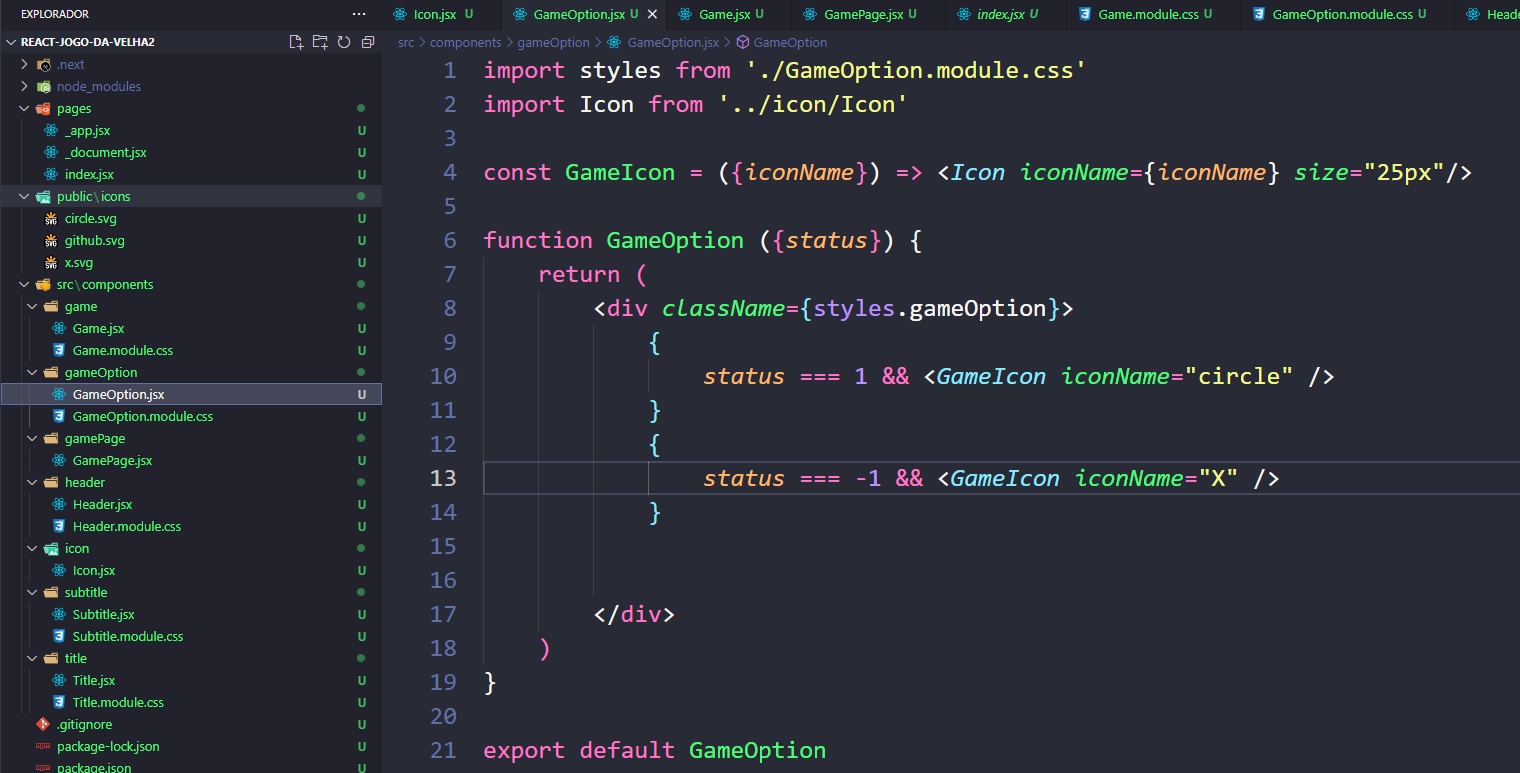
50 – **Vídeo 05 – Trabalhando no gameOption –** baixar os ícones de círculo e xis, fornecidos pelo curso.

51 – Em **gameOption.jsx** importar o Icon e dentro da **div** colocar:

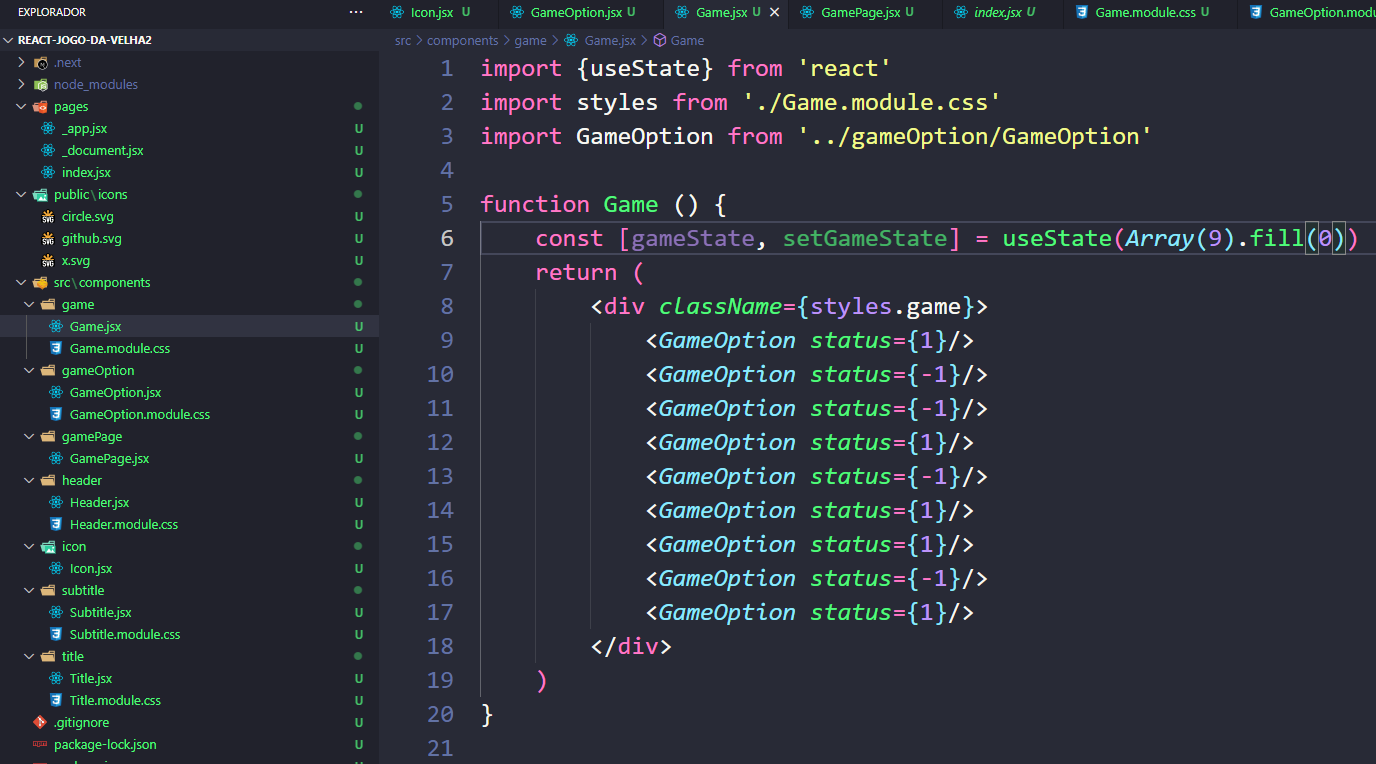
**<Icon *iconName*="X" size="20px"/>**



52 – Criar uma **const,** para não precisar escrever o tamanho (size) para todos os elementos que forem escritos. Sempre devemos fazer isso quando há situações que se repetem muito:



53 – Em **Game.jsx** importar o **{useState}** do ‘**react**’, depois criar a **const** com o **Array**:



54 – Depois da **div** **className** em **Game.jsx**, criar **gameState.map** entre chaves com o **value** e **pos**:



55 - Finalizar **commit** e enviar para o **github:** clicar em **controle de código**, clicar no **maiszinho**, renomear arquivo (Criado estados/status no GameOption), clicar **ctrl enter** e depois **Publish branch** ou digitar **git push origin main:**



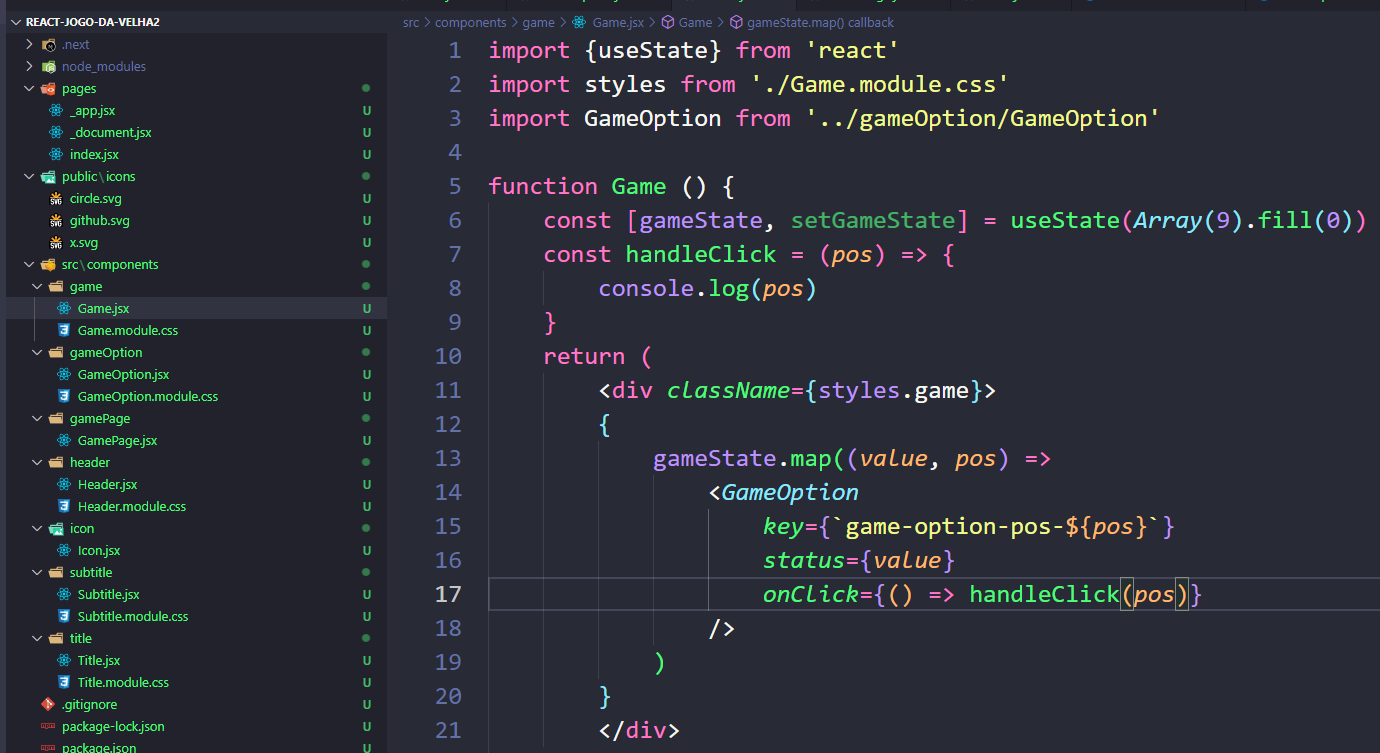
56 – **Vídeo 06 A Lógica do Jogo** – Em GameOption.jsx criar o onClick:



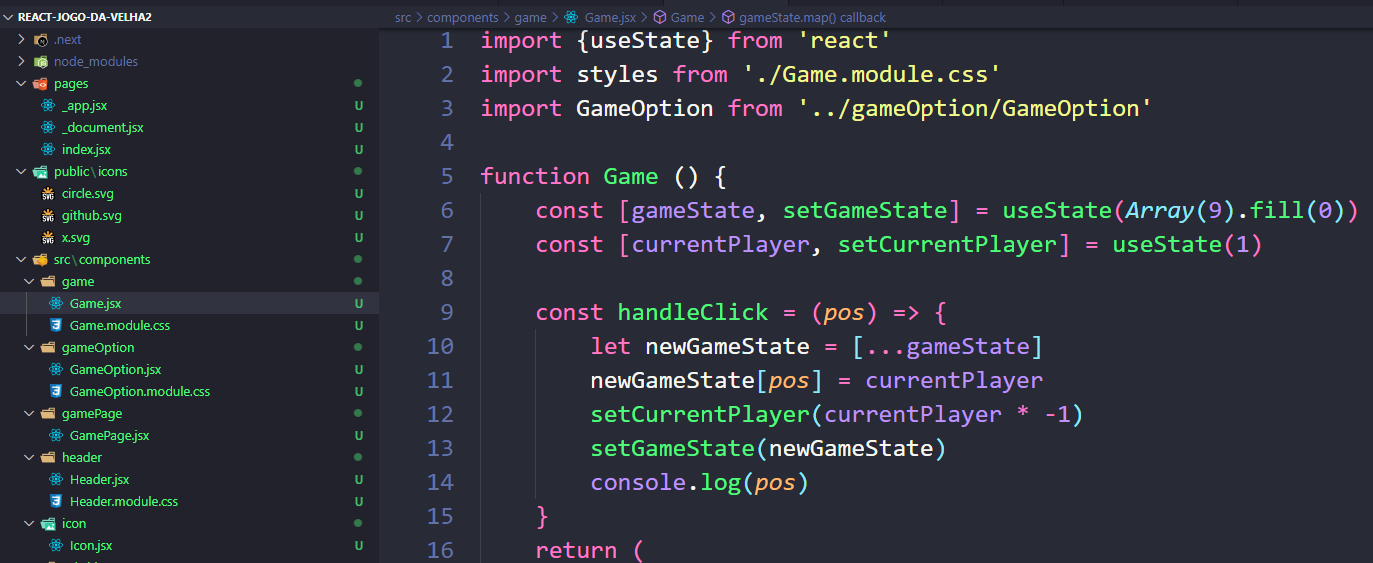
57 – Chamar dentro de Game.jsx o onClick. Porém, ainda não sabemos de qual posição este clique está vindo.



58 – Criar o const handleClick e passar os parâmetros para o onClick:



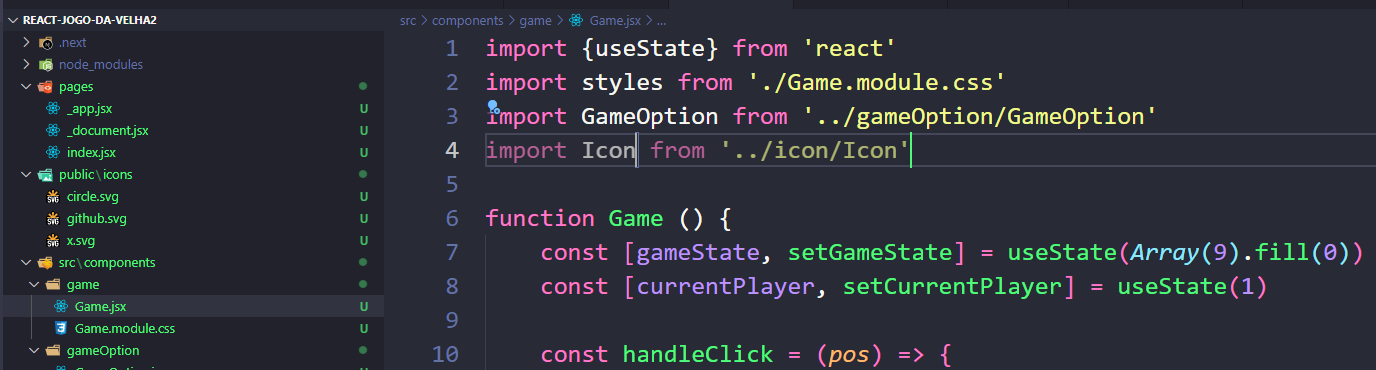
59 – Na aerow function do handleClick, criar o let newGameState e criar a const currentPlayer conforme abaixo:



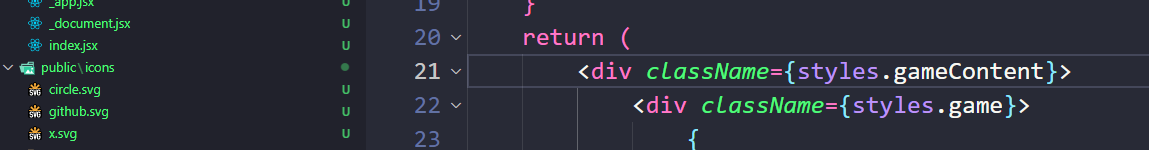
60 – Para que não haja clicks duplos devemos permitir o click somente se a opção estiver sem o circulo ou o X. Devemos então colocar tudo dentro do if:

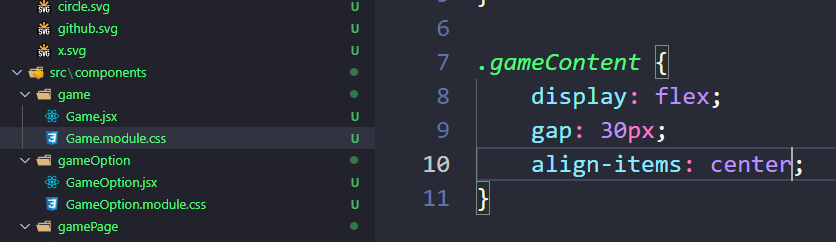


61 – Dentro do Game.jsx, importar o Icon:

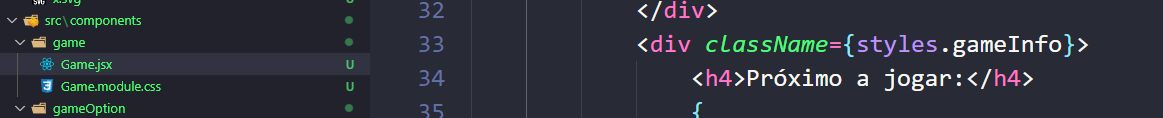


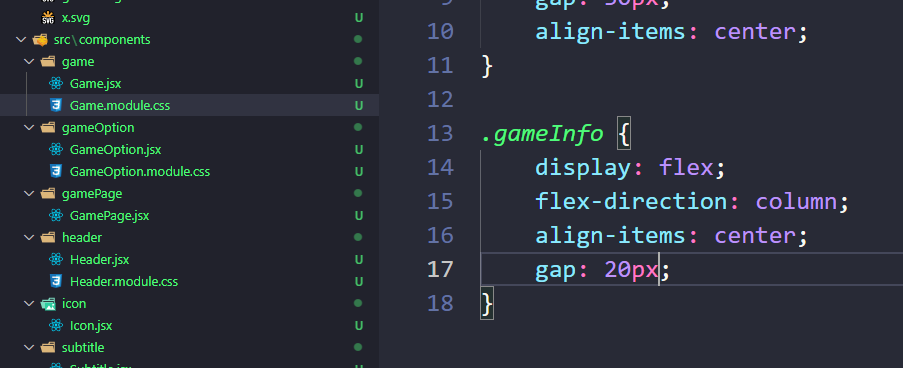
62 – Na primeira **div** do **return** (em **Game.jsx**), criar **className** **styles.gameContent**. Depois no **Game.module.css** criar o **display: flex**:



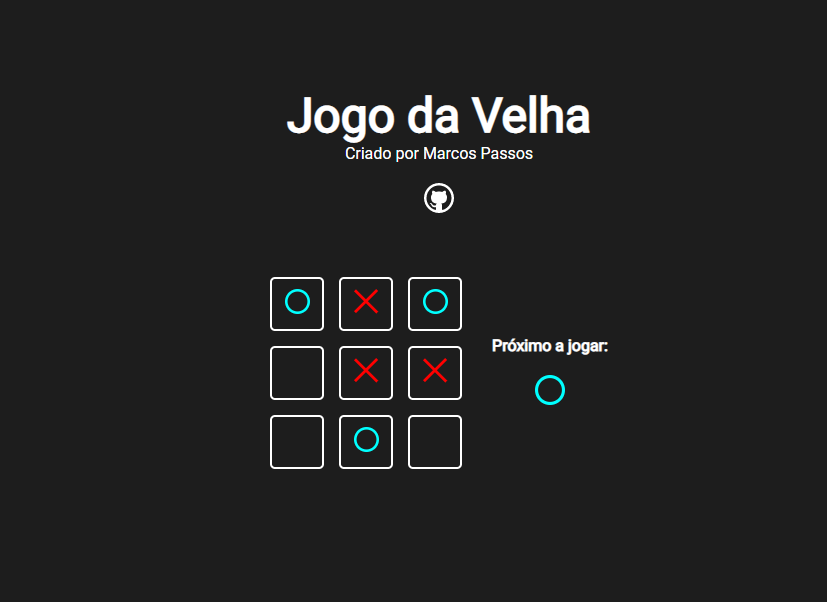


63 – Criar o **className gameInfo** na **div** do **h4** (Próximo a jogar, no **Game.jsx**):





64 - Finalizar **commit** e enviar para o **github:** clicar em **controle de código**, clicar no **maiszinho**, renomear arquivo (Criado lógica do Game), clicar **ctrl enter** e depois **Publish branch** ou digitar **git push origin main:**



65 – **Vídeo 07 – Verificando o campeão –** importar **Game.jsx** o **useEffect**. Usamos o **useEffect** quando queremos que o sistema execute uma ação sem que o usuário faça alguma ação. São as chamadas funções react rooks, que executam funções especificas. Atentar para o modelo da função: **useEffect(( ) => { }, [ ])**

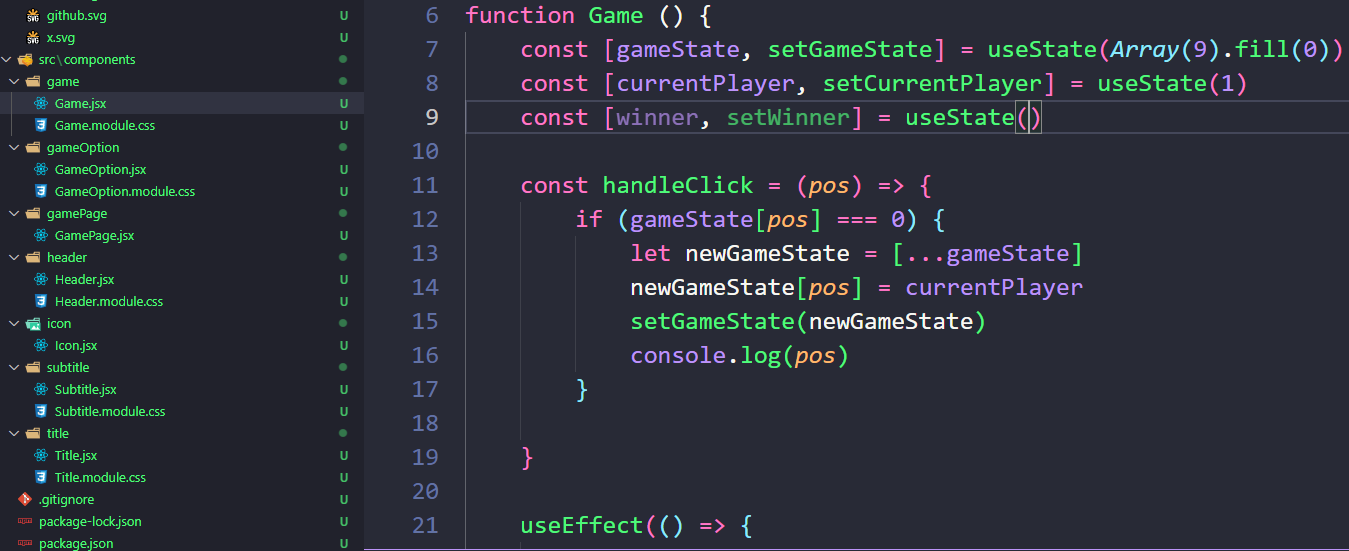




66 – Colocamos o **setCurrentPlayer** dentro do **useEffect**:



67 – Próximo passo é criar uma função que verifique se há um campeão:



68 – Criar um gabarito com todas as posições em que um jogador pode ser campeão:



69 – Verificar se houve algum ganhador. Criar o **verifyGame.** Fazer um **arrow function** com o **forEach** no **winnerTable**:



70 – **Vídeo 08 – Declarando o campeão**

Criar pasta com nome de gameInfo. Depois criar o arquivo GameInfo.jsx e o arquivo GameInfo.module.css

Tirar de dentro do Game e colocar dentro do GameInfo.jsx:

GameInfo.jsx

<https://jogo-da-velha-02-hiff.vercel.app/>

https://jogo-da-velha-02-hiff.vercel.app/