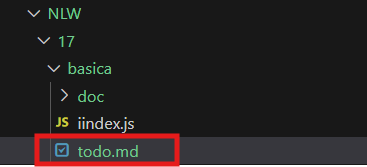
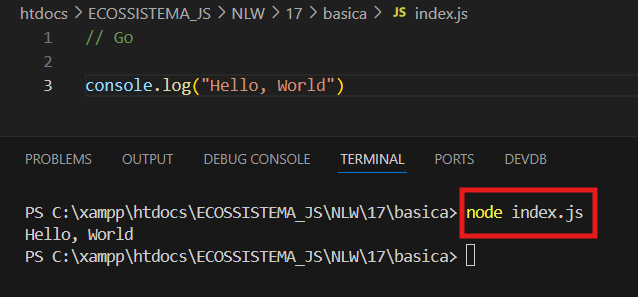
**ANOTAÇÕES NLW – TRILHA BÁSICA**

**Criamos um arquivo com extensão .md:**

Um arquivo com extensão .md é um arquivo de **Markdown**. Markdown é uma linguagem de marcação leve usada principalmente para formatar texto de maneira simples e legível tanto para humanos quanto para máquinas. Esses arquivos são amplamente utilizados em documentação, README files em projetos de software, blogs, e outros conteúdos onde a simplicidade e a legibilidade são importantes.



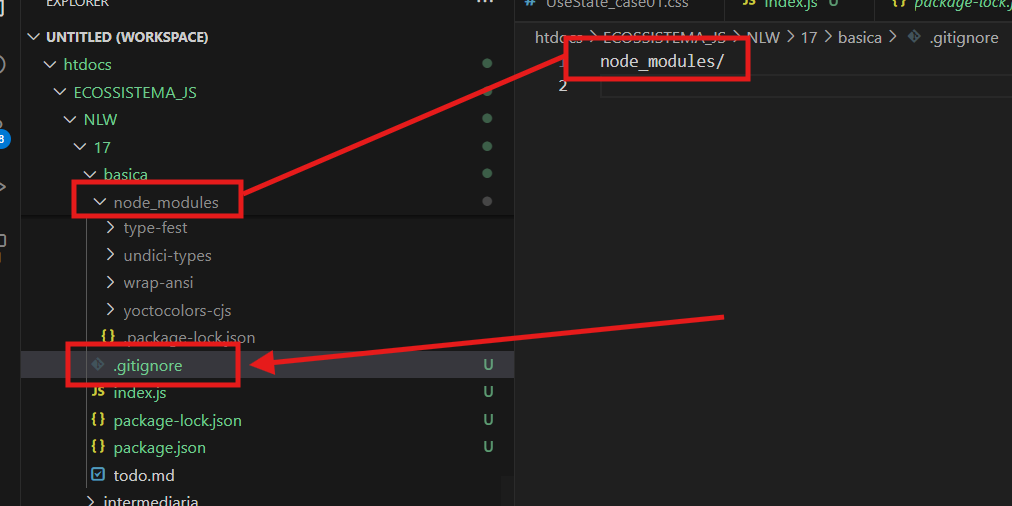
Note que, para executar um arquivo no terminal basta digitar node e o nome do arquivo, caso estejamos no diretório que armazena este arquivo.



Instala os o pacote “inquirer”



Criamos o .gitignore e configuramos para que a pasta node\_modules não seja considerada na criação dos commits porque é onde fica a maior parte do código e que só se altera quando instalamos ou desinstalamos alguma dependência e por isso, com o package.json íntegro, bastará rodar ‘npm i’ para que, caso seja necessário, o aplicativo restabeleça todos os pacotes da node\_modules:

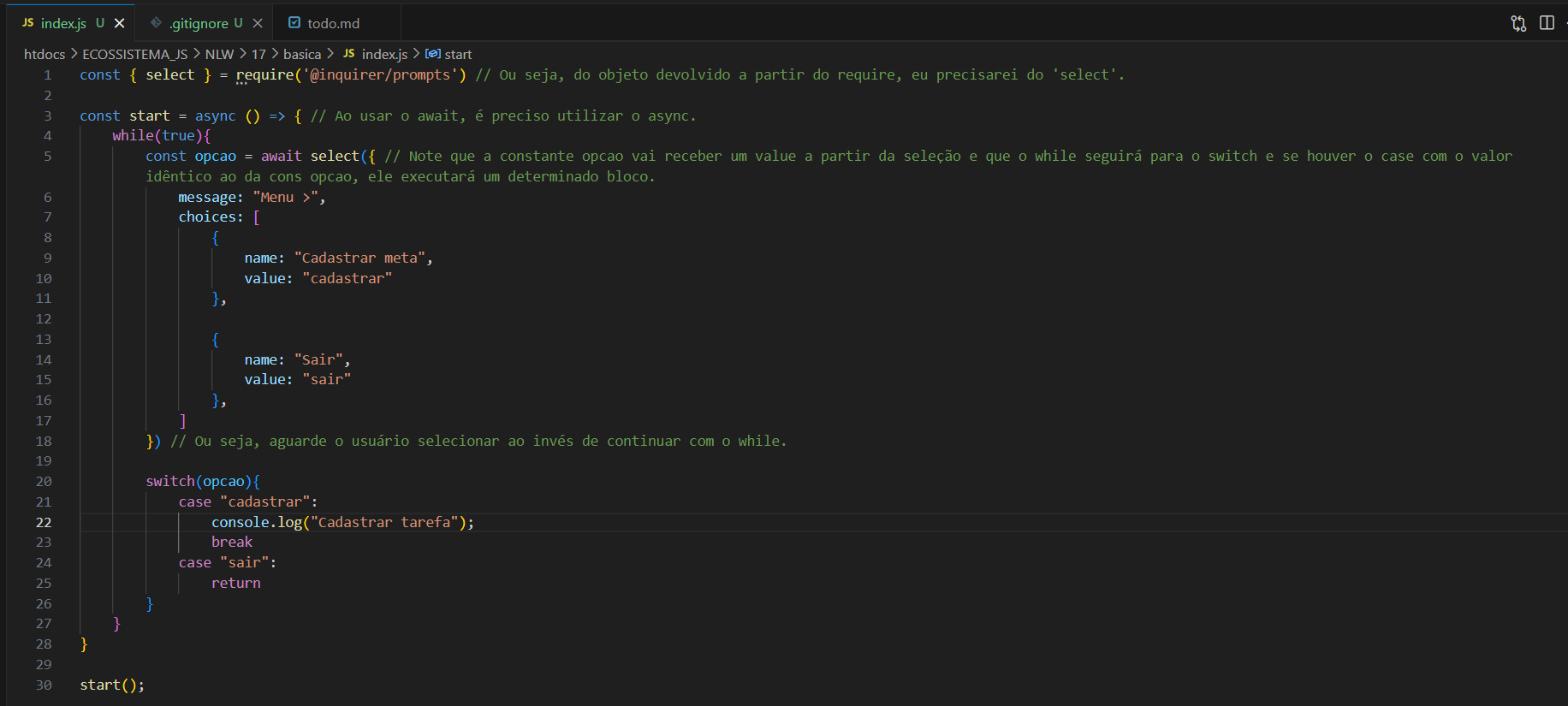


De seguida, importamos o pacote ‘inquirer’ e deixamos disponível o select para utilização na produção do mecanismo de seleção da opção desejada pelo usuário:



Poderíamos ter importado desta maneira: (const prompts = require('@inquirer/prompts')). Contudo, o uso de { select } se trata de tornar mais claro o que está sendo utilizado, embora a importação disponibilize na memória todo o pacote importado. Usar isso { select } significa desestruturação.

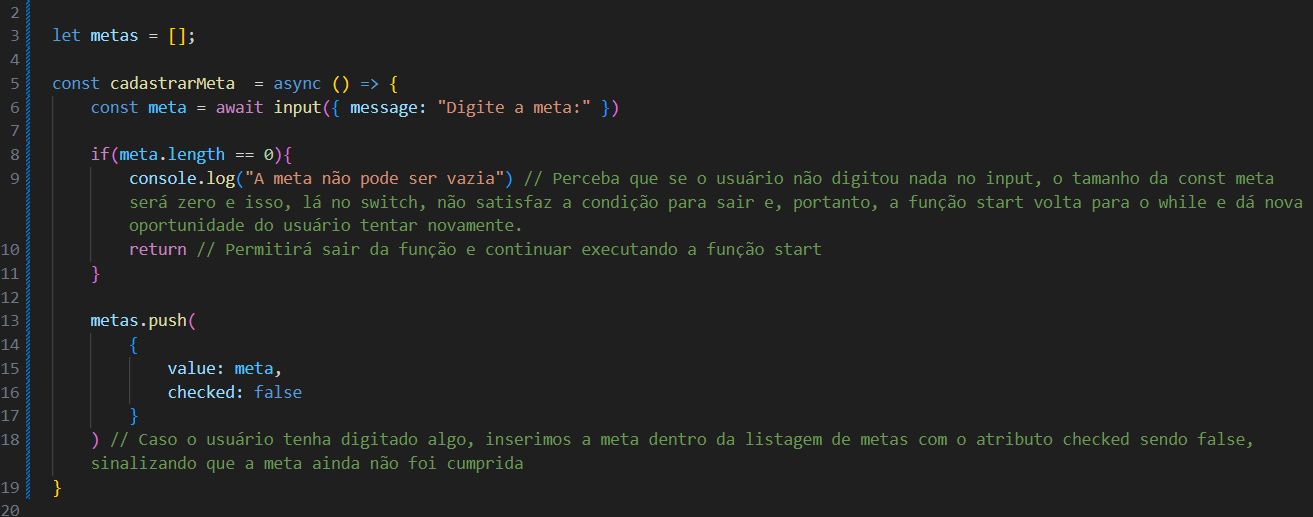
Neste ponto já temos o funcionamento inicial da lógica do menu do programa. Veja:



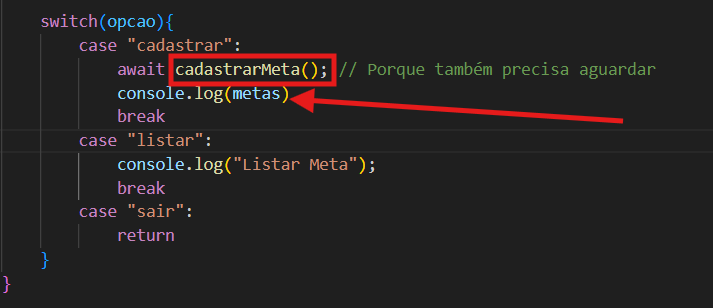
Dentro do while criamos a cons opção, que recebe a função select do pacote requirer (ATENÇÃO: é necessário usar o async no início de onde a função está sendo utilizada e o await logo antes do uso da função, para indicar que é necessário aguardar alguma coisa, neste caso, a escolha da opção antes de prosseguir). Daí, o value da choice selecionada corresponderá ao valor da constante opção e, desta forma, opera o switch e, caso não tenha sido ‘sair’, o while continuará valendo true e voltará a apresentar a lógica. Caso a escolha seja true, ele faz um return e para o uso do while.

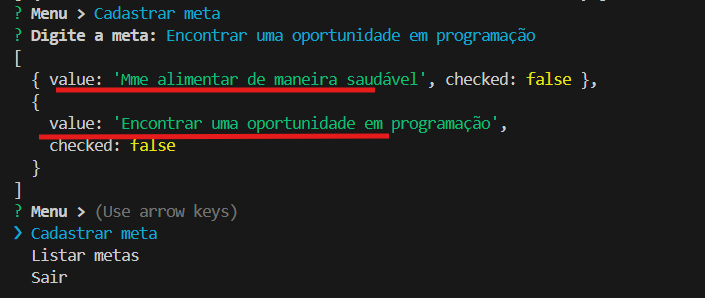
O uso do await é um caso de promise, em que há uma promessa de realização de algo para retornar uma resposta. O async informa que a função está sendo assíncrona.

Agora criamos o mecanismo para cadastro de metas. Ao escolher a opção de menu cadastrar meta, executaremos a função cadastrarMeta, que também é async e utiliza a função input do pacote requirer. Se nada for digitado, retorna uma mensagem sinalizando que a meta não pode ser vazia e faz return. Ao fazer return, o switch não recebe o sair e volta para o início do while. Caso o usuário tenha digitado algo, realizamos um push do value e do checked para a constante metas. Veja:



Para visualizar o resultado, ao cadastrar uma meta e antes do break, exibimos, ainda de maneira provisória, a lista de metas cadastradas:





Retomar do 45 minuto da aula 2.