O novo `@Controller` de `@RequestMapping('/oferta")`, que recebe essa requisição para `/oferta` e ele só tem um método para processar essa requisição, que é processar o GET dessa requisição oferta.

E a única coisa que ele faz é redirecionar direto para a Vue, que vai gerar o HTML e apresentar o HTML para o usuário. E a VUE é `return -'oferta/home";` - que está aqui dentro de 'templates". Vamos abrir o arquivo "home.html".

Então, o que vai acontecer? Lembrando que a tecnologia é Thymeleaf ainda, no back-end. O Thymeleaf vai processar esse HTML, vai colocar o título, vai colocar a logo, aquela *tag* `<head>`, vai gerar todo esse HTML - inclusive com essa função JavaScript - e vai jogar esse HTML de volta para o usuário.

O que acontece? Quando o navegador baixar esse HTML, todo o conteúdo dele, nós configuramos esse atributo `onload` de `body` e dizemos para ele chamar uma função JavaScript, ou seja, estamos dizendo para o navegador, na hora em que ele receber esse HTML e carregar todo o conteúdo dele, chamar a função `onLoad()` do JavaScript. Então esse processamento está acontecendo no navegador.

E a função `onLoad()` aqui no final da página vai executar a criação dessa aplicação `Vue`. Lembrando que como isso vai acontecer só depois de todo o carregamento da página, esses *scripts* aqui já terão sido carregados também. Um é o do Vue e o outro do Axios, para fazer requisições Ajax.

O app do Vue que nós criamos tem como objetivo, através do `mounted ()`, fazer uma requisição com Axios para o back-end, para `/api/pedidos/aguardando`; pegando a resposta dessa requisição e colocando nesse atributo `pedidos`.

Esse `pedidos : []`, esse *array* então é percorrido pelo Vue.js nesse `v-for`, e ele vai gerar o HTML; ou seja, toda essa geração de HTML está sendo feita no navegador. Diferentemente do Thymeleaf. O Thymeleaf gerou só esse HTML, mas o processamento mesmo de gerar os formulários, de percorrer pedidos e tudo mais, é feito pelo Vue no navegador.

Continuando o processamento. Depois de o navegador renderizar essa página e chamar a função `onLoad()`, que foi configurada com essa aplicação `Vue`; que no `mounted ()`, que é um método chamado automaticamente, chama o `axios` - ele está chamando o quê? `/api/pedidos/aguardando`. O que responde essa requisição é esse `PedidosRest` que criamos aqui.

Então `/api/pedidos` e é um *get*, para `/aguardando`. Vai fazer o quê? Ele vai no `pedidoRepository`, vai buscar os dados da lista de pedidos e já vai retornar essa própria lista de pedidos.

Então esse não é mais um `Controller` normal, ele é um `RestController`. Ele não vai retornar qual Vue tem que ser renderizada. Como aqui, por exemplo, `PedidoController`, ele retorna `return`

`"pedido/formulario";`, ele retorna uma `String`,
as *actions* retornam *strings*.

No `RestController` ele já retorna os dados mesmo que vão ser convertidos em JSON, ou seja, na resposta dessa requisição agora é que o Axios pega essa resposta e nós colocamos aqui nos pedidos. Aí sim ele vai processar essa lista de pedidos e vai renderizar essa `div`, usando esse `v-for`.

Agora, de outra forma, como as coisas estão acontecendo? Temos o usuário, que seria o navegador do usuário, fazendo uma requisição para `/oferta`. Lembrando que esse `/oferta` é esse, `@RequestMapping("/oferta")`.

E o que esse `/oferta` faz? Ele apenas redireciona o processamento para o Thymeleaf, para o `oferta/home.HTML`, que gera o HTML e devolve essa página HTML com o Vue.js para o usuário.

Logo em seguida, depois que o navegador do usuário, esse "Usuário" representa o navegador dele, quando o navegador faz o carregamento desse HTML - e estamos falando desse `onload` - ao carregar ele processa essa função JavaScript. O que ele faz? Ele faz uma nova requisição que é `/api/pedidos/aguardando`, que vai bater no `PedidosRest`. Estamos falando então dessa requisição feita com Axios.

O que acontece após essa requisição? Lá no PedidosRest, que é esse aqui, ele vai no `pedidoRepository` e busca os pedidos pelo status, dá um `.findByStatus`. O `pedidoRepository` vai no banco de dados e retorna a lista de pedidos. Veja que ele não vai para nenhuma Vue, ele não vai para nenhum Thymeleaf processar, ele já retorna os pedidos no formato JSON.

E esses pedidos do formato JSON vão ser processados pelo Vue, que vai gerar o HTML. Veja que a geração do HTML usando o Thymeleaf é feita no servidor, agora a geração do HTML é feita usando o Vue.js no navegador do usuário. Então temos o processamento feito no navegador do usuário e o processamento feito no servidor.

Essa primeira requisição em resposta aqui é usando o Thymeleaf, então é legal que nós temos um contraste entre essas duas.
Essa `/api/pedidos/aguardando` é usando o Vue.js e essa `/oferta`, é usando o Thymeleaf.

No Thymeleaf, a requisição que bate no `Controller` vai sempre para uma Vue que vai gerar o HTML e retornar para o usuário. Com o Vue usando Ajax, com Vue.js, com React, com Angular etc., o *framework* de JavaScript faz uma requisição REST que vai buscar os dados e retornar JSON. E a geração do HTML fica a cargo do navegador, é um processamento feito em JavaScript no navegador do usuário.