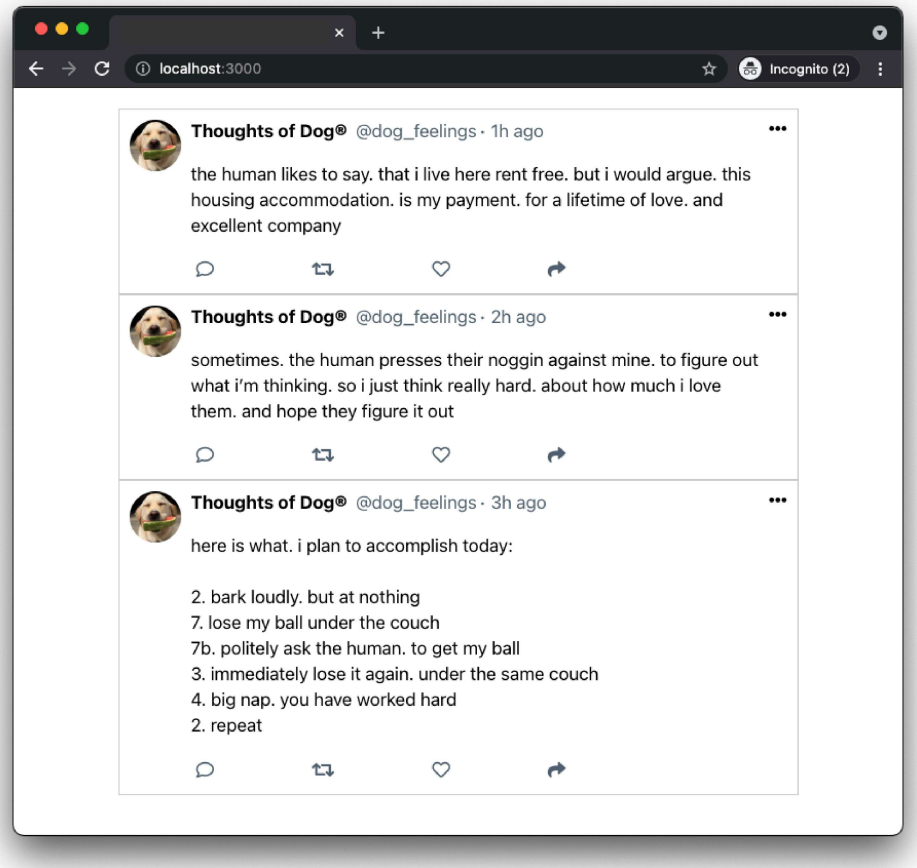
Vue.js Tweets

LAB **Obligatorio**

Introducción

Pasar datos a través de props es un concepto importante de Vue.js que se comprende mejor a través de la práctica. Utilizaremos este ejercicio para ayudarte a solidificar tu comprensión de los props.

Estaremos clonando una pieza de interfaz de usuario existente de una aplicación popular, Twitter. ¡Comencemos!



EN

Vue.js Tweets

Setup

- Descarga el código base proporcionado por el profesor
- Abre el LAB y comienza:

```
$ cd lab-vue-c-tweets-es
$ npm install
$ npm run dev
```

Empezando

1. Usaremos Font Awesome para los iconos en nuestra aplicación. Agrega la siguiente hoja de estilo en el `head` de la página `index.html` :

```
<link
  rel="stylesheet"
  href="https://pro.fontawesome.com/releases/v5.10.0/css/all.css"
  integrity="sha384-
AYmEC3Yw5cVb3ZcuHtOA93w35dYTsvhLPVnYs9eStHfGJvOvKxVfELGroGkvsg+p"
  crossorigin="anonymous"
/>
```

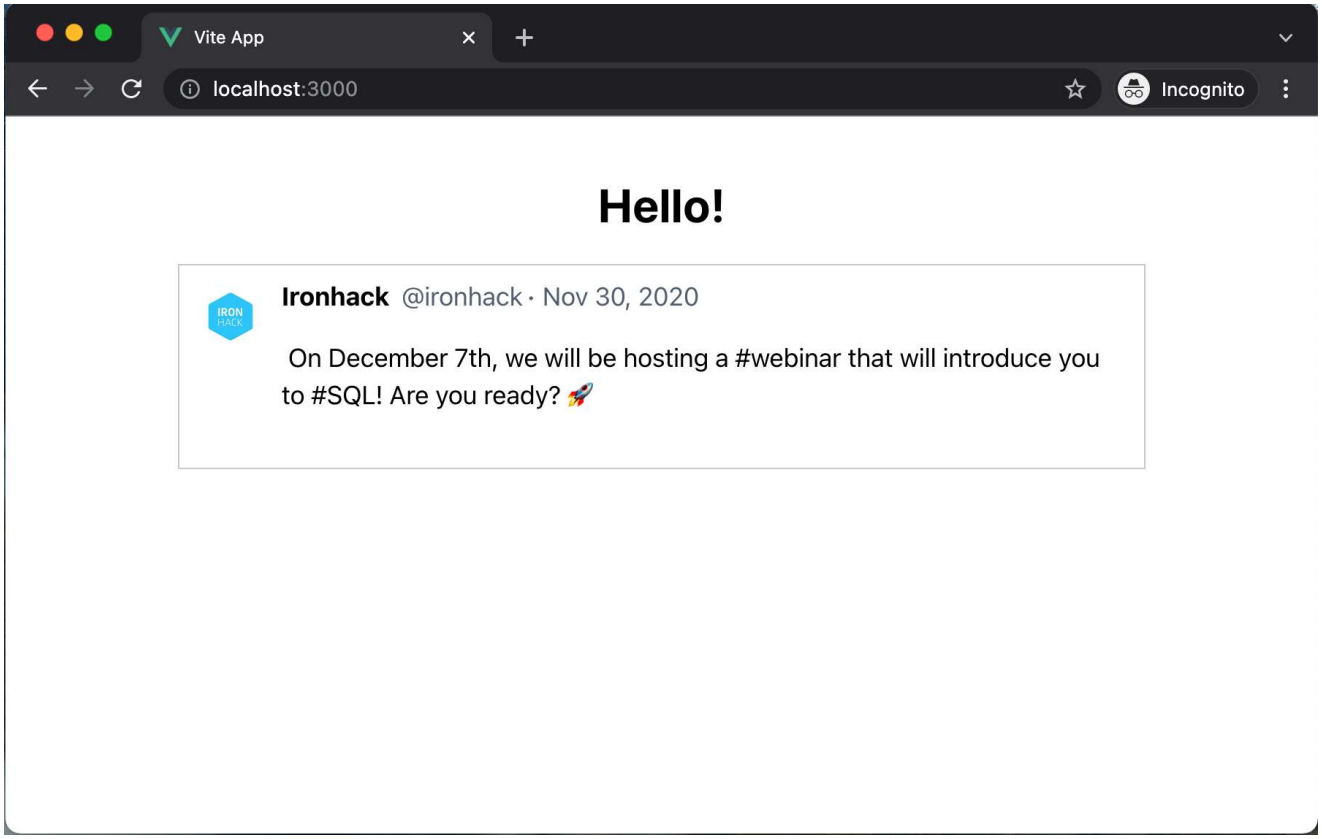
Instrucciones

Iteración 1 | Contenido Inicial

Para permitirte enfocarte en Vue.js sin tener que preocuparte por el estilo, te hemos proporcionado los estilos CSS. Todo el CSS está incluido en el código de inicio en el `).vue` dentro de la etiqueta `<style>` .

momento para revisar estos dos archivos.

Una vez que inicialmente ejecutes la aplicación, deberías ver lo siguiente:



El componente `Tweet` está renderizando contenido estático en este momento. Lo cambiaremos en la próxima iteración. Actualizaremos el componente `Tweet` para mostrar el contenido que viene de `props`.

Iteración 2 | Pasar el tweet como prop

En `App.vue`, tenemos una matriz llamada `tweets` que contiene objetos que representan tweets. Usaremos el componente `Tweet` para mostrar estos objetos de *tweet*. En el `Tweet`, mostraremos el nombre del usuario (`name`), la imagen del usuario (`image`), el `handle` de usuario, la marca de tiempo del tweet (`timestamp`) y el mensaje (`message`).

Pasar el tweet como prop

Pasar el primer objeto de datos de `tweets` como una prop al componente `Tweet` :

```
<!-- src/App.vue -->
```

EN

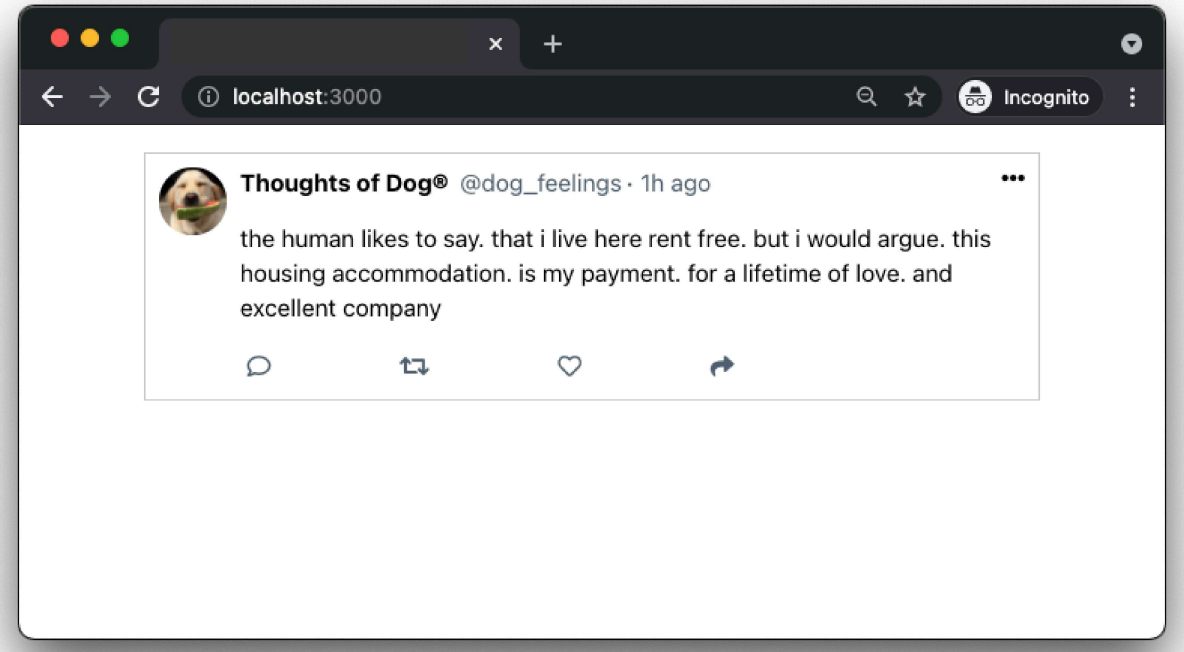
Vue.js Tweets

Mostrar el contenido del tweet en el componente `Tweet`

Actualice el componente `Tweet` para mostrar los valores que vienen de la propiedad `tweet`. ¡Recuerde que el valor que pasamos es un objeto!

Resultado esperado

Una vez hecho esto, su componente `Tweet` debería mostrar el siguiente contenido:



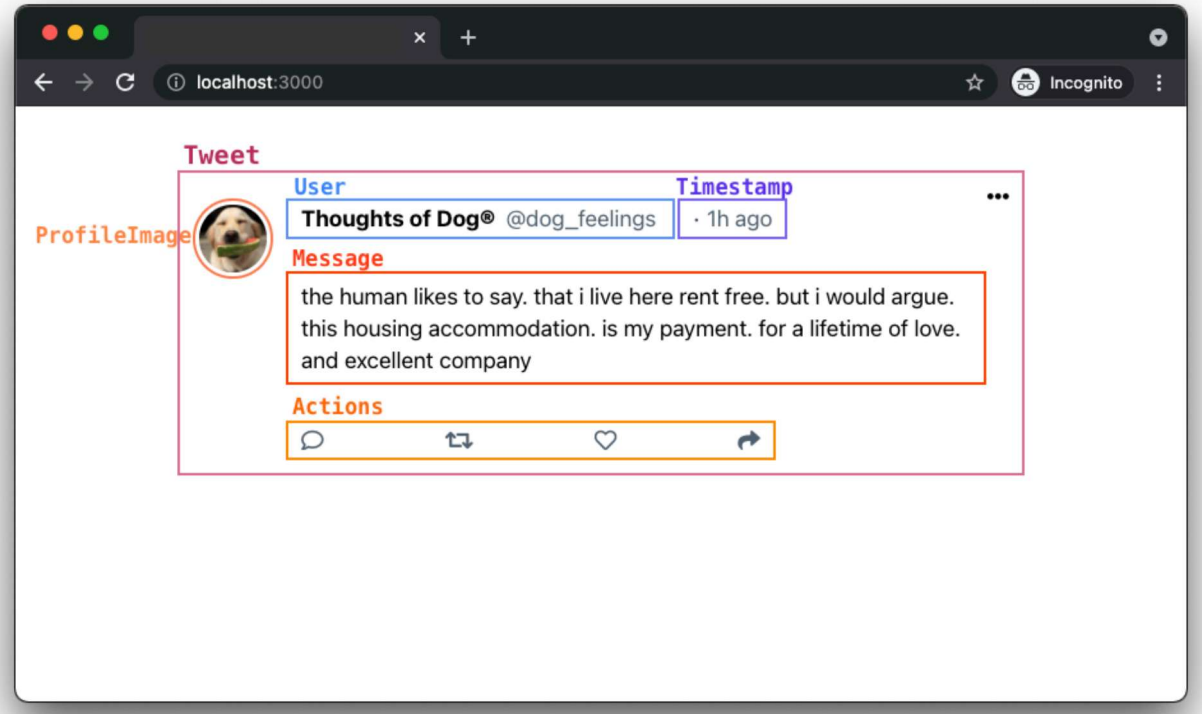
Iteración 3 | Crear los Componentes

Ahora crearemos nuevos archivos para los componentes que haremos en las siguientes iteraciones. Dentro de la carpeta `src/components/` crea los siguientes archivos nuevos:

- `src/components/ProfileImage.vue` ,
- `src/components/User.vue` ,
- `src/components/Timestamp.vue` ,
- `src/components/Message.vue` and
- `src/components/Actions.vue` .

EN

Vue.js Tweets



Cuando se complete todas las iteraciones, la versión final de tu componente `Tweet` se verá así:

▶ **Haz clic para ver el código**

Iteración 4 | Componente de imagen de perfil (ProfileImage)

Extraer HTML

Extraiga la etiqueta `img` existente y renderícela a través del componente `ProfileImage` :

```
<img src="IMAGE_URL" class="profile" alt="profile"/>
```

Renderizar el componente

Una vez hecho esto, importe el componente `ProfileImage` a `Tweet.js` . Después de importarlo, renderice el componente dentro de `Tweet` de la siguiente manera:

EN

```
<div class="tweet">
```

Vue.js Tweets

Acceder a las props

`ProfileImage` recibe una prop `image`. Configure este valor como el `src` de la etiqueta `<img />`.

Iteración 5 | Componente de Usuario (User)

Extraer HTML

Extraiga las etiquetas `span` existentes que muestran la información del usuario y renderícelas a través del componente `User`:

```
<span class="user">
  <span class="name"> USER_NAME </span>
  <span class="handle">@ USER_HANDLE</span>
</span>
```

Renderizar el componente

Importe el componente `User` en `Tweet.js`. Después de importarlo, renderice el componente dentro de `Tweet` de la siguiente manera:

```
<!-- ... -->

<template>
  <div class="tweet">
    <ProfileImage image="user.image" />

    <div class="body">
      <div class="top">
        <User userData="user" />

<!-- ... -->
```

Acceder a las Props

Pasamos el objeto con la información del usuario a través de la prop `userData`. Acceda y muestre el nombre del usuario (*name*) y su manejo de Twitter (*handle*).

EN

Vue.js Tweets

Iteración 6 | Componente de marca de tiempo (Timestamp)

Extraer HTML

Extrae la etiqueta `span` existente que muestra la información de marca de tiempo y haz que se muestre a través del componente `Timestamp` :

```
<span class="timestamp"> TWEET_TIMESTAMP </span>
```

Renderizar el componente

Importa el componente `Timestamp` a `Tweet.js` . Después de importarlo, renderiza el componente dentro de `Tweet` de la siguiente manera:

```
<!-- ... -->

<template>
  <div class="tweet">
    <ProfileImage image="user.image" />

    <div class="body">
      <div class="top">
        <User userData="user" />
        <Timestamp time="timestamp" />

<!-- ... -->
```

Accede a las props

`Timestamp` recibe una prop `time` . Muestra este valor como el contenido de la etiqueta `span` .

Iteración 7 | Componente de mensaje

Extraer HTML

Extrae la etiqueta `p` existente y haz que se muestre a través del componente `Message` :

EN

```
sage"> TWEET_MESSAGE </p>
```

Vue.js Tweets

Una vez hecho esto, importa el componente `Message` y renderízalo en `Tweet.js` de la siguiente manera:

```
<!-- ... -->

<template>
  <div class="tweet">
    <ProfileImage image="user.image" />

    <div class="body">
      <div class="top">
        <User userData="user" />
        <Timestamp time="timestamp" />
      </div>

      <Message message="message" />
<!-- ... -->
```

Accede a las props

`Message` recibe una prop `message`. Muestra este valor en la etiqueta `p`.

Iteración 8 | Componente de Acciones (Actions)

Extraer HTML

Extraer la etiqueta `div.actions` del mensaje existente y renderizarla a través del componente `Actions`:

```
    <div class="actions">
      <i class="far fa-comment"></i>
      <i class="fas fa-retweet"></i>
      <i class="far fa-heart"></i>
      <i class="fas fa-share"></i>
    </div>
```

Renderizar el componente

Cuando se haya hecho esto, importar el componente `Actions` y renderizarlo en `Tweet.js` de esta manera:

EN

```
      <div class="body">
        <div class="top">
          <User userData="user" />
          <Timestamp time="timestamp" />
        </div>

        <Message message="message" />
        <Actions />

  <!-- ... -->
```

El componente `Actions` no recibe ninguna propiedad.

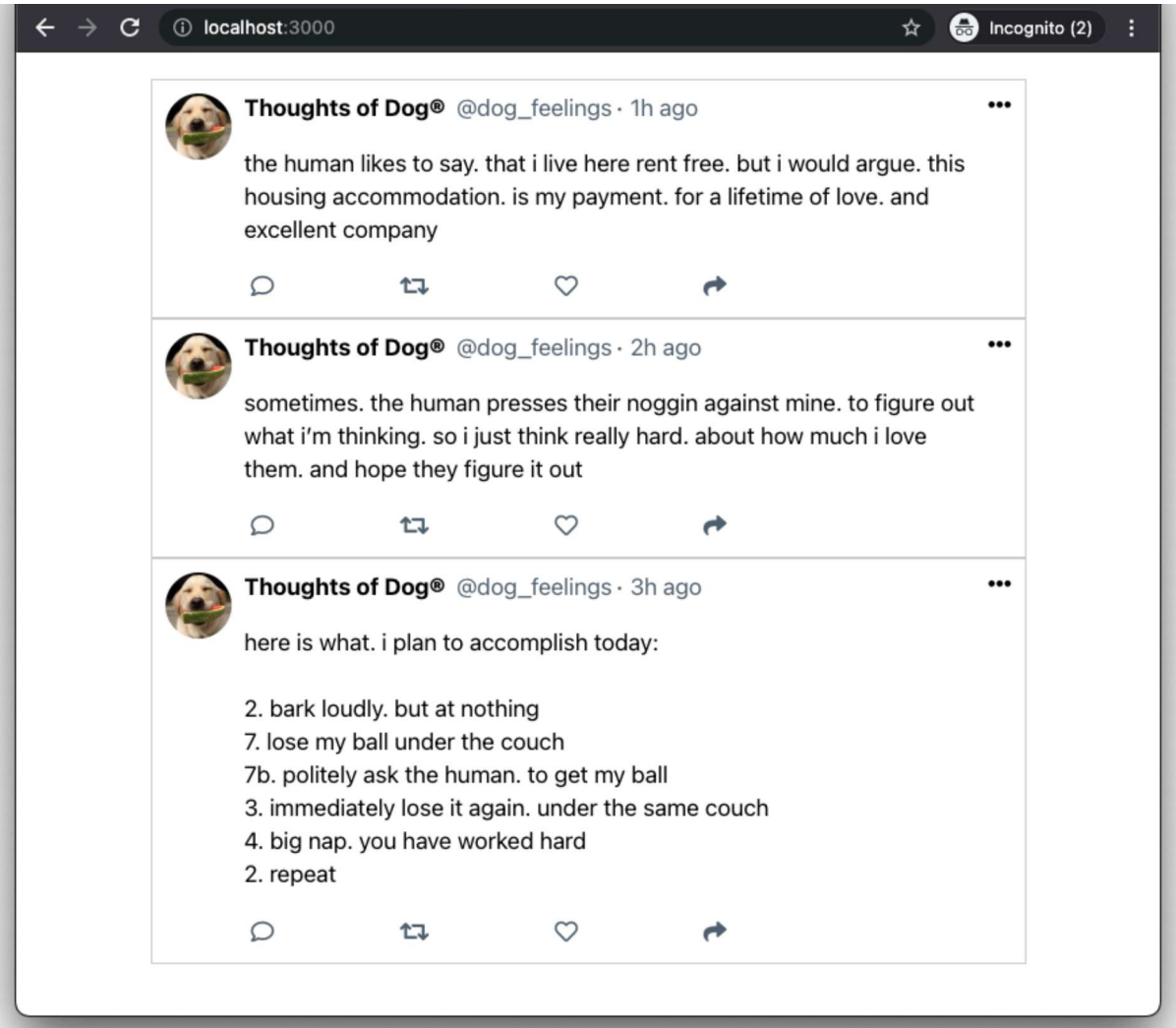
Iteración 9 | Renderiza múltiples `Tweet`s

Una vez que hayas terminado de refactorizar el componente `Tweet`, actualiza `App.vue` para mostrar tres componentes `<Tweet />`. Cada `<Tweet />` debe recibir un objeto de tweet separado del `tweetsArray`.

Una vez finalizado, tu aplicación debería mostrar el siguiente contenido:

▼ **Haz clic para ver la imagen**

Vue.js Tweets



Happy coding!

vue-tweets

Por favor, envía la URL de tu solución aquí:

URL

EN

ANTERIOR
← Watchers (Observadores) en Vue.js

SIGUIENTE
Router en Vue.js