difference between inout and change event listener

The **input** and **change** event listeners are both used to track changes in form input elements, but they have slight differences in behavior.

1. **input** Event Listener:
    - The **input** event is triggered whenever the value of an input element changes, including when the user types or pastes text, or when the input value is programmatically changed.
    - This event is more responsive and detects changes as they happen in real-time.
    - It is commonly used for live validation or instant feedback as the user interacts with the input.
2. **change** Event Listener:
    - The **change** event is triggered when the value of an input element is committed or finalized, typically when the input loses focus or when the user presses the Enter key.
    - This event is triggered after the user has finished making changes to the input and indicates that the input value has been confirmed.
    - It is commonly used to capture the final value after the user has completed their input.

In summary, the **input** event provides real-time tracking of input changes, while the **change** event captures the final value after the user has finished editing the input. The choice between them depends on the specific use case and desired behavior in your application.

---

i dont want to w[400%] do be static, how can i adjust it to dynamic so that it adjusts with how many li

```
<div class="overflow-x-auto overflow-y-hidden md:overflow-x-hidden md:overflow-y-scroll">
      <ol class="flex md:flex-col gap-2 md:h-80 w-[400%] ">
        @foreach ($agents as $agent)
          <li class="flex gap-1 receiver-el cursor-pointer shadow-md" data-id="{{ $agent->user->id }}">
            <img
              src="https://images.pexels.com/photos/7841717/pexels-photo-7841717.jpeg?auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1"
              alt="user id photo"
              class="h-8 w-8 mb-2 receiver-chat-head-click"
```

```
                    style="border-radius: 50%"
                    data-id="{{ $agent->user->id }}"
                    data-username="{{ $agent->user->username }}"
                  >

                  <span class="md:block hidden receiver-chat-heads">{{ Auth::user()->user_type == 3 ?
$agent->user->username :  (Auth::user()->user_type == 2 ? $agent->availedBy->username : '') }}</span>
                </li>
              @endforeach
            </ol>
          </div>
```

SOLUTION:

```
<div class="overflow-x-auto overflow-y-hidden md:overflow-x-hidden md:overflow-y-scroll">
    <ol class="flex md:flex-col gap-2 md:h-80" id="agent-list">
        <!-- Your <li> items here -->
    </ol>
</div>

#agent-list {
   display: flex;
   flex-wrap: nowrap;
}
#agent-list li {
   flex: 0 0 auto;
}
```

CODE FOR DYNAMIC STAR RATINGS

```
@for ($i = 0; $i < $review->level; $i++)
<svg aria-hidden="true" class="w-5 h-5 text-yellow-400" fill="currentColor" viewBox="0 0 20 20"
xmlns="http://www.w3.org/2000/svg"><title>First star</title><path d="M9.049 2.927c.3-.921
1.603-.921 1.902 0l1.07 3.292a1 1 0 00.95.69h3.462c.969 0 1.371 1.24.588 1.81l-2.8 2.034a1 1 0 00-
.364 1.118l1.07 3.292c.3.921-.755 1.688-1.54 1.118l-2.8-2.034a1 1 0 00-1.175 0l-2.8 2.034c-.784.57-
1.838-.197-1.539-1.118l1.07-3.292a1 1 0 00-.364-1.118L2.98 8.72c-.783-.57-.38-1.81.588-
1.81h3.461a1 1 0 00.951-.69l1.07-3.292z"></path></svg>
@endfor

@for ($i = $review->level; $i < 5; $i++)
<svg aria-hidden="true" class="w-5 h-5 text-slate-400" fill="currentColor" viewBox="0 0 20 20"
xmlns="http://www.w3.org/2000/svg"><title>First star</title><path d="M9.049 2.927c.3-.921
1.603-.921 1.902 0l1.07 3.292a1 1 0 00.95.69h3.462c.969 0 1.371 1.24.588 1.81l-2.8 2.034a1 1 0 00-
.364 1.118l1.07 3.292c.3.921-.755 1.688-1.54 1.118l-2.8-2.034a1 1 0 00-1.175 0l-2.8 2.034c-.784.57-
1.838-.197-1.539-1.118l1.07-3.292a1 1 0 00-.364-1.118L2.98 8.72c-.783-.57-.38-1.81.588-
1.81h3.461a1 1 0 00.951-.69l1.07-3.292z"></path></svg>
@endfor
```

Accessing the PARENT and CHILDREN of parent using JQUERY.

- **eq() is used to get elements of parent, 0 is the first element.**

```
const confirmRejectParentEl = $(e.target).parent()
   confirmRejectParentEl.hide()

   confirmRejectParentEl.parent().children().eq(0).text('Accepted').show()
```

.closest() = To find the ancestors

.find() = To find the descendants

FIND div where data is === chatroomId

```
 const chatHeadEl = $('.receiver-el').filter(function(){
          return $(this).data('chat-id') === chatRoomId
```

```
    })
```

How to implement Laravel task manager

1. **Make new command:**
- **php artisan make:command DisablePlanCommand**

2. **Adjust it on kernel**

Show latest value depending on userId

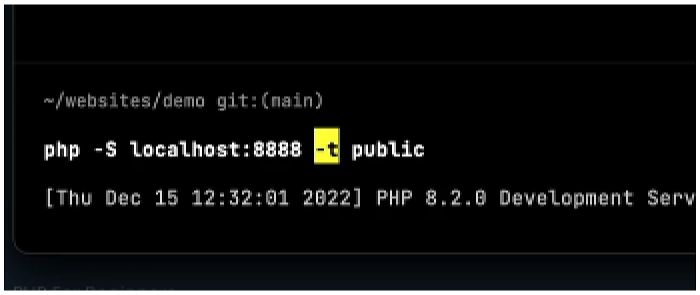
```
->orderByRaw('CASE WHEN request_by = ' .$user->id .' THEN 0 ELSE 1 END')
```

Find specific div

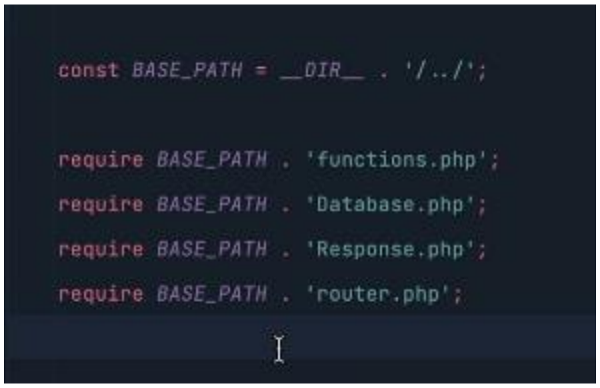
```
const chatRoomId = e.chatRoomId
    const chatHeadEl = $('.receiver-el').filter(function(){
        return $(this).data('chat-id') == chatRoomId
    })

    chatHeadEl.append(`<span class="chat-badge-message bg-red-400 rounded-full p-1 text-xs text-white">new</span>`)
```

PHP only run the public folder



```
~/websites/demo git:(main)
php -S localhost:8888 -t public
[Thu Dec 15 12:32:01 2022] PHP 8.2.0 Development Serv
```



```
const BASE_PATH = __DIR__ . '/../';

require BASE_PATH . 'functions.php';
require BASE_PATH . 'Database.php';
require BASE_PATH . 'Response.php';
require BASE_PATH . 'router.php';
```

VUE in APP.vue

```
<script>
 export default{
   name: 'App',
   data(){
     return{
       title: 'My first vue app.'
     }
   }
 }
</script>
```

Fetch data from Json VUE Js

```
mounted(){
      fetch('api.com')
      .then(response => response.json()).
      then(data => this.job = data).
      catch(err => console.error(err))
   }
```

```
import { ref } from "vue"
```

```
const getPost = (id) => {

   const post = ref(null)
   const error = ref(null)

   const data = async () => {
      try {
         const data = await fetch(`http://localhost:3000/posts/${id}`)
         post.value = await data.json()
         if(! data.ok)
         {
            throw Error('Error fetching post');
         }
      } catch (error) {
         error.value = error.message
      }
   }
   return { post, error, data}
}

export default getPost
```

Install Vue Js on Laravel Application. Run this on terminal/CMD

- npm i vue@next
- npm i @vitejs/plugin-vue
- npm i vue-router

vite.config.js should look like this:

```
import { defineConfig } from 'vite';
import laravel from 'laravel-vite-plugin';
import vue from '@vitejs/plugin-vue'

export default defineConfig({
    plugins: [
        laravel({
            input: ['resources/css/app.css', 'resources/js/app.js'],
            refresh: true,
        }),
        vue({
            template: {
                transformAssetUrls : {
                    base: null,
                    includeAbsolutes: false
                }
            }
        })
    ],
});
```

On resources/js, create ‘components’ folder and create ‘Welcome.vue’

On app.js add:

```
import { createApp } from 'vue';
import './bootstrap';
import router from './router';
import Welcome from '../js/components/Welcome.vue'

createApp(Welcome).use(router).mount('#app')

OR

import { createApp } from 'vue';
import './bootstrap';
import router from './router/index.js';
import App from '../js/App.vue';
import vue3GoogleLogin from 'vue3-google-login';

const app = createApp(App);

// Use the vue3GoogleLogin plugin
app.use(vue3GoogleLogin, {
    clientId: '990638987823-p1qmei763ssev42saugvonf2luq7ls06.apps.googleusercontent.com'
});

app.use(router).mount('#app');
```

On welcome.blade, remove all then replace as:

```
<!DOCTYPE html>
<html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1">

        <title>Laravel</title>
        @vite(['resources/js/app.js', 'resources/css/app.css'])
    </head>
    <body>
        <div id="app"></div>
    </body>
</html>
```

Create router in vue

```
import { createRouter, createWebHistory } from 'vue-router'
import Index from '../vue/Index.vue'
import Register from '../vue/Register.vue'

const router = createRouter({
    history: createWebHistory(),
    routes: [
        {
            path: '/',
            name: 'index',
            component: Index
        },
        {
            path: '/register',
            name: 'register',
            component: Register
        }
    ]
})

export default router
```

Web.php

```
Route::get('/{any?}', function () {
    return view('welcome');
})->where('any', '.*');
```

setToLocalStorage in vue js

```
import { watch, ref } from 'vue'

export function useStorage(key)
{
    let storedVal = localStorage.getItem(key) || ''
    let valRef = ref(storedVal)

    watch(valRef, () => { valRef.value !== '' ? localStorage.setItem(key, valRef.value) :
localStorage.removeItem(key)})

    return valRef
}
```

VUE JS if pic not showing:

php artisan storage:link

Fix CORS problem:

Uncomment this on bootstrap window.axios.defaults.headers.common['X-Requested-With'] = 'XMLHttpRequest';

CONFIGURE PUSHER

1. install on terminal: composer require pusher/pusher-php-server
2. npm install –save pusher-js
3. npm install laravel-echo
4. Check broadcasting.php and check for pusher
5. Go to .env and PUSHER_APP_CLUSTER from mt1 to ap1.
6. Enter the app id from the pusher website
7. In .env, change the BROADCAST_DRIVER from ‘log’ to ‘pusher’
8. Add <meta name="csrf-token" content="width=device-width, initial-scale=1"> in layout.blade header section
9. Open bootstrap.js and uncomment the comments about pusher and change cluster: import.meta.env.VITE_PUSHER_APP_CLUSTER ?? 'mt1', to cluster: import.meta.env.VITE_PUSHER_APP_CLUSTER ?? 'ap1',
10. Run npm run dev again to compile the files
11. Add implements ShouldBroadcast to the event
12. Tweak it
13. On app.php on config, uncomment the App\Providers\BroadcastServiceProvider::class,
14. Edit channels.php for private channels

GitHub

Set up SSH Key:

1. Open git bash
2. ssh-keygen -t rsa -b 4096 -C “email address here”
3. enter
4. enter passphrase
5. confirm passphrase
6. eval $(ssh-agent -s)
7. ssh-add ~/.ssh/id_rsa
8. enter passphrase
9. clip < ~/.ssh/id_rsa.pub
10. open GitHub, add SSH, add title like ‘ralph pc’, paste SSH key
11. DONE

REACT

How to install react?

1. npx create-react-app project name