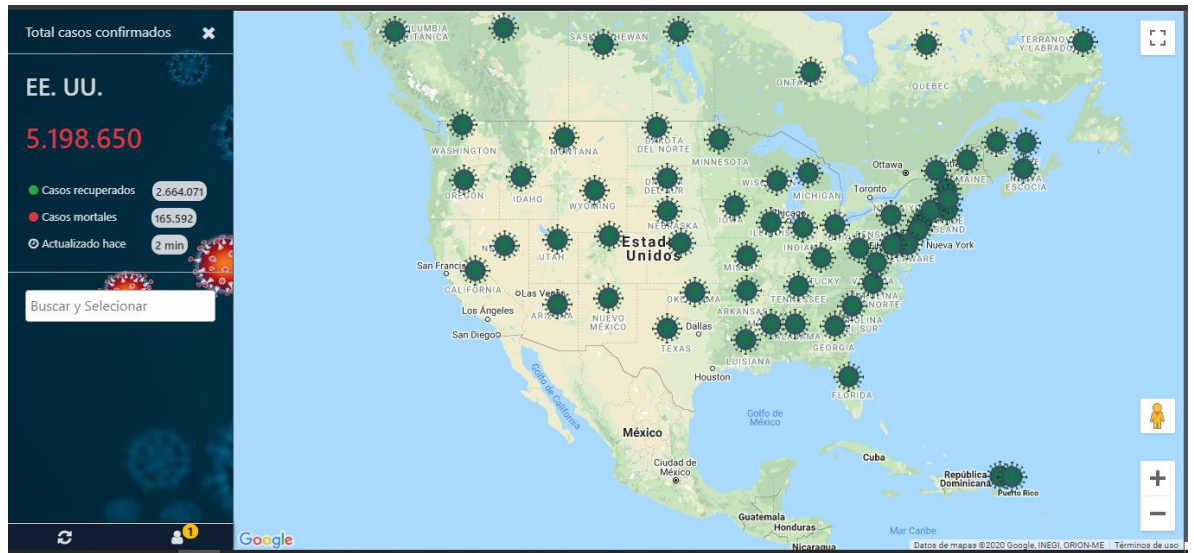


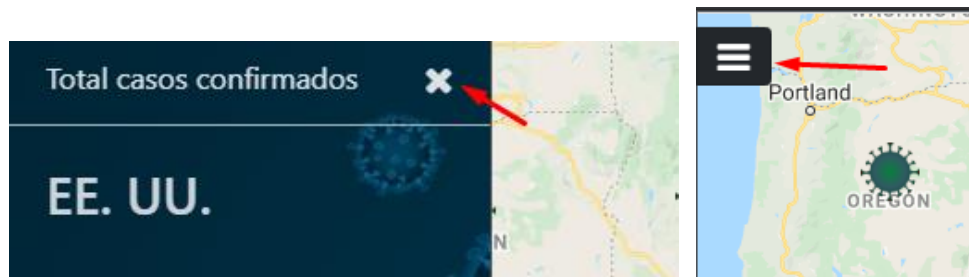
GUIA

1. Pantalla principal:

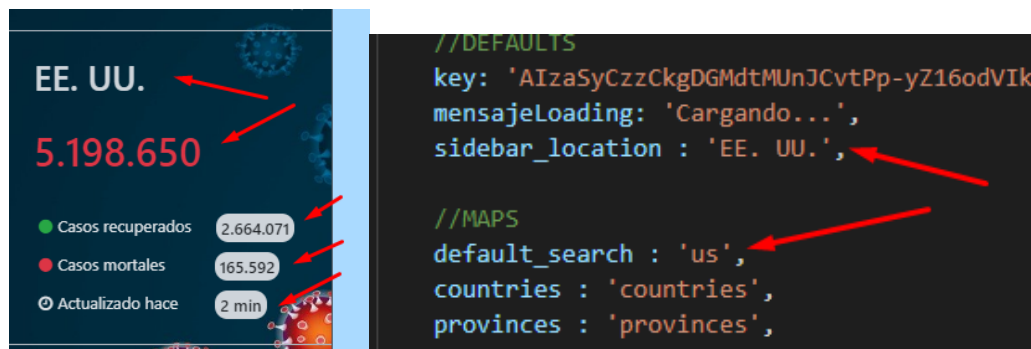


2. Sidebar:

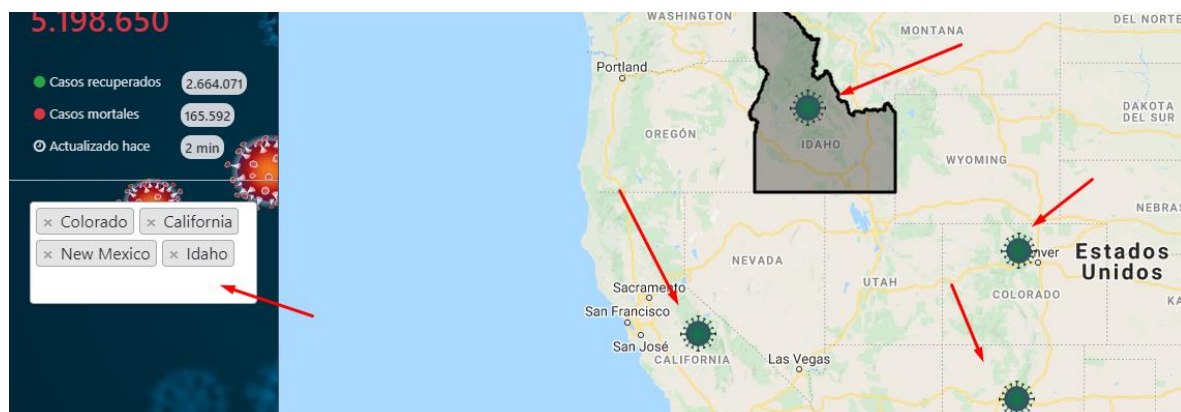
- Abrir o Cerrar menú lateral:



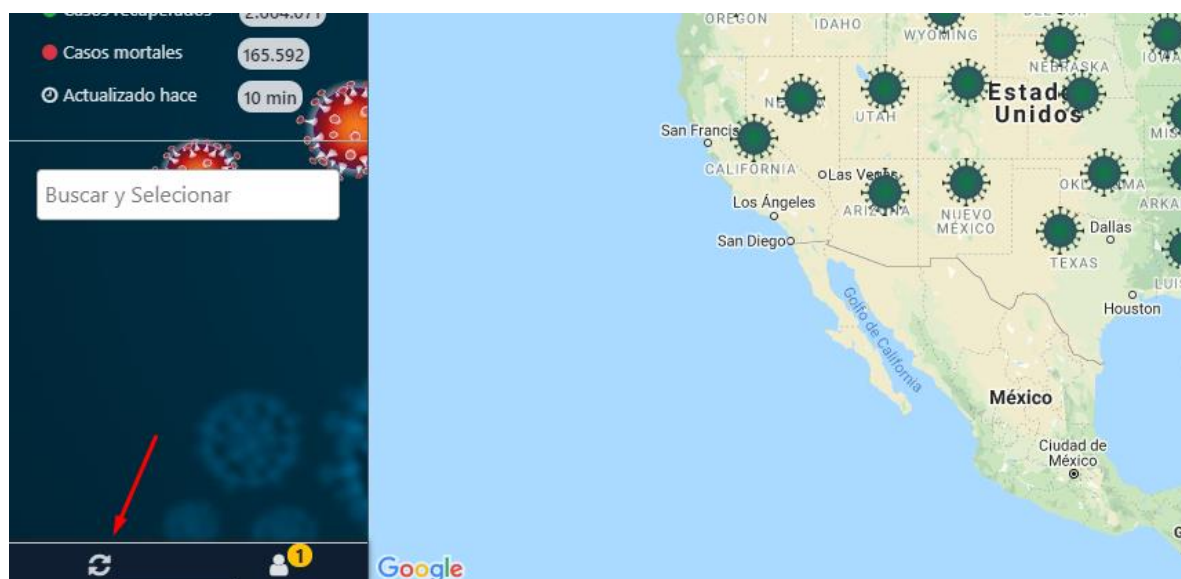
- Total, de casos relacionados a EE.UU., esta configuración es parametrizable en **environments/config.ts**



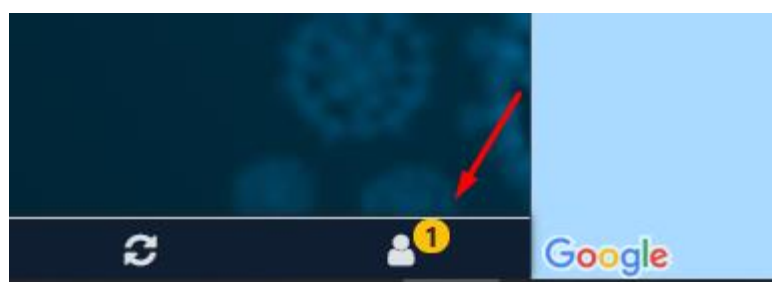
- Combo múltiple para buscar estados



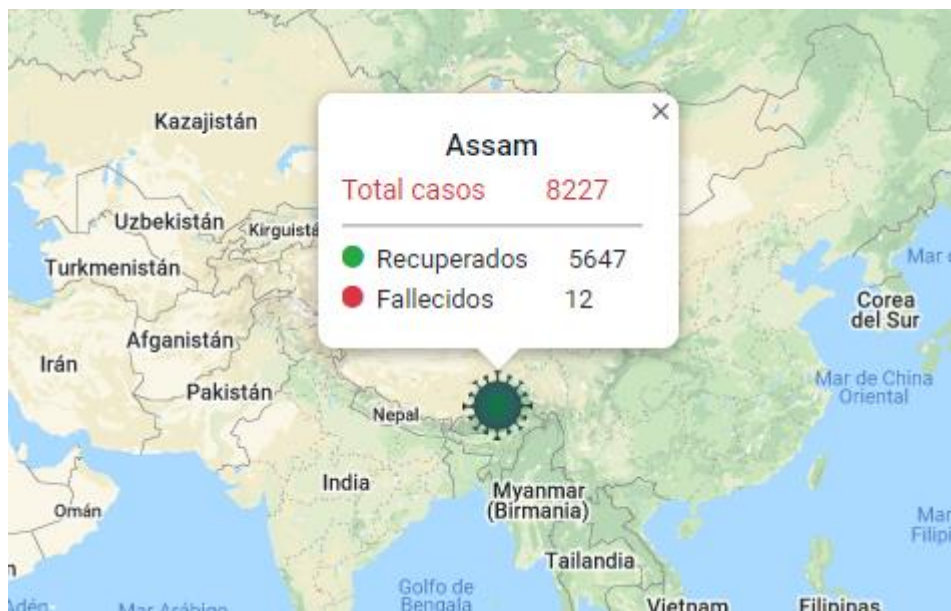
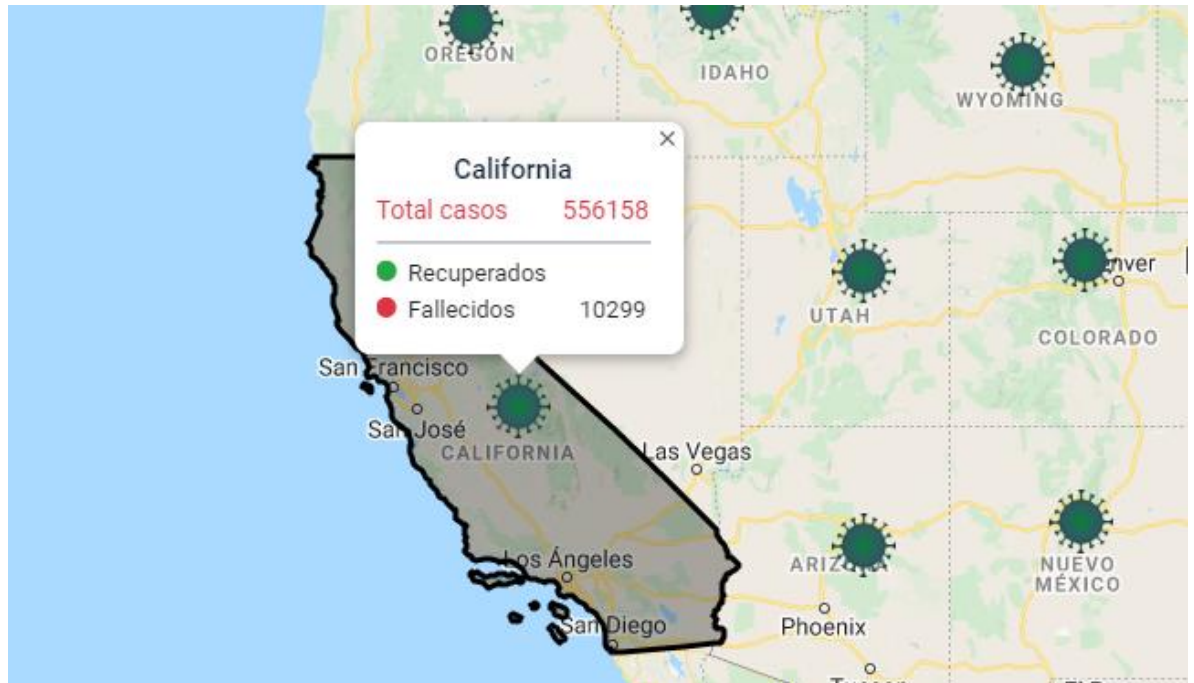
- Reiniciar filtros, en la parte inferior izquierda:



- ver mi perfil: <https://gallerydinamic.web.app/hv>

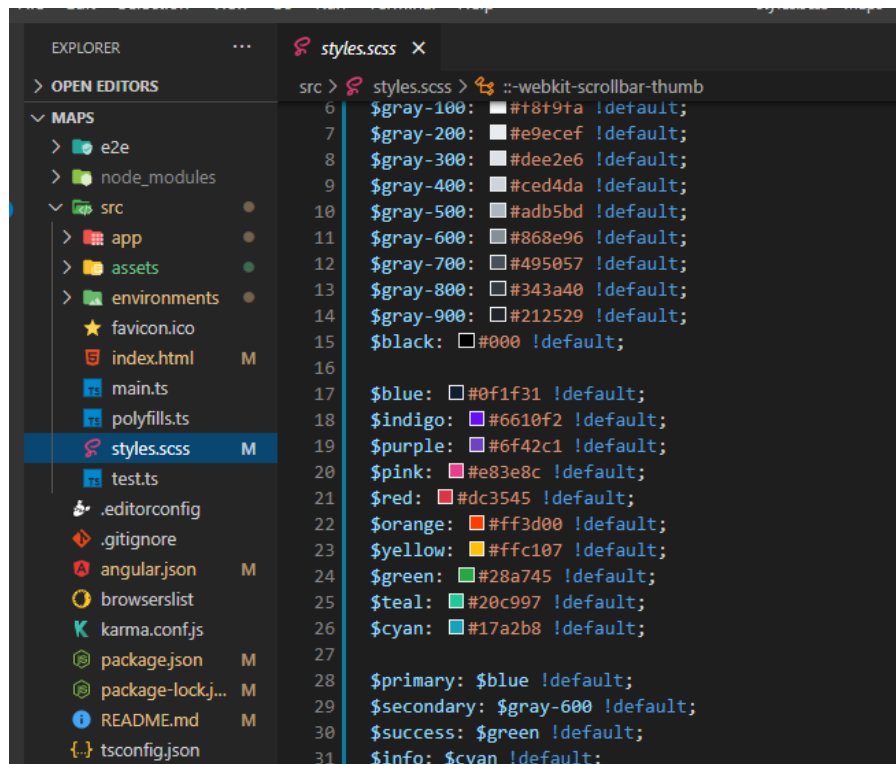
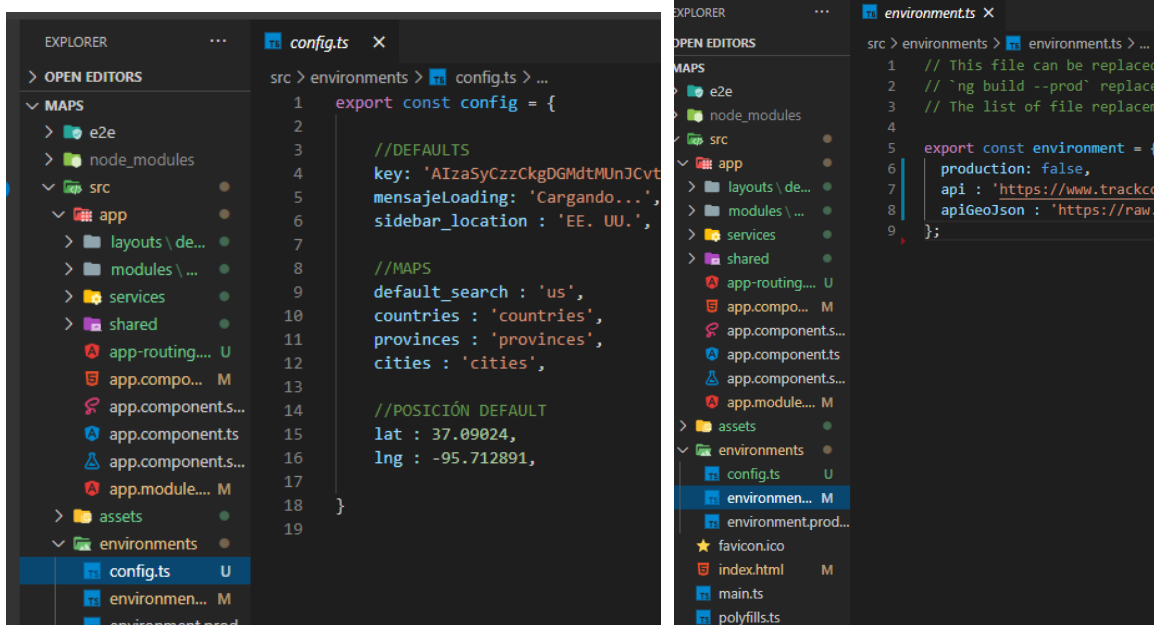


3. Mapas: Al pasar el cursor indica total casos, recuperados (en algunos no hay cantidad de recuperados), fallecidos y en caso de tener posición geoJson se visualiza el polígono.

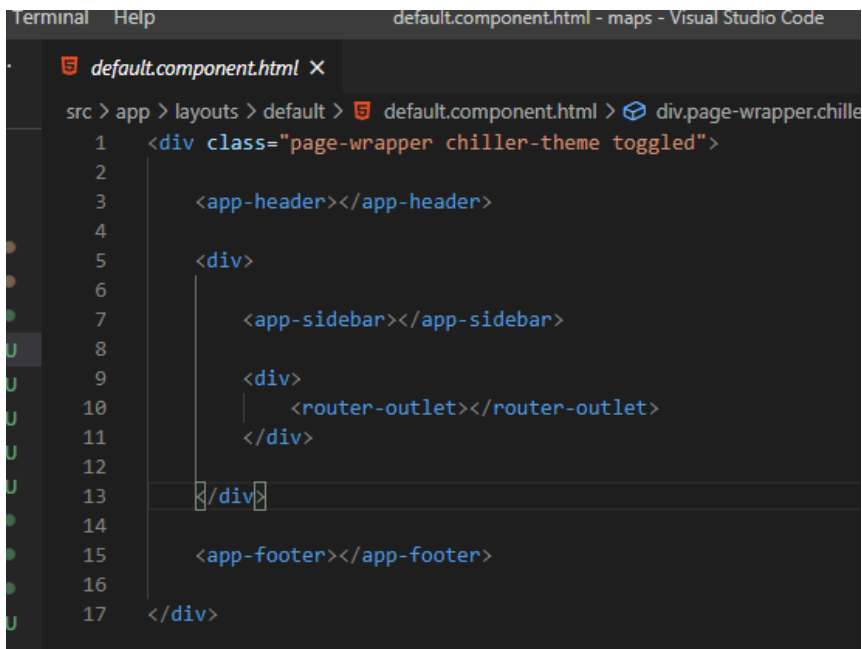


DESARROLLO

Configuraciones:



Definición de módulos: **layouts/default**



```
Terminal  Help  default.component.html - maps - Visual Studio Code

default.component.html X
src > app > layouts > default > default.component.html > div.page-wrapper.chiller
1  <div class="page-wrapper chiller-theme toggled">
2
3      <app-header></app-header>
4
5      <div>
6
7          <app-sidebar></app-sidebar>
8
9          <div>
10             <router-outlet></router-outlet>
11          </div>
12      </div>
13
14      <app-footer></app-footer>
15
16
17 </div>
```

Component maps: **modules/maps**



```
/******
*DEFAULTS
*****

//OBTENER INFO MAPA
getMapsSidebar() {

    this._service.getMetodo(config.countries + '/' + co

    //RESULT DATA MAPAS
    result.data.forEach((val, key) => {

        this._service._total = accounting.formatNumber
        this._service._tl_mortales = accounting.format
        this._service._tl_recuperados = accounting.for

        moment.locale('es');
        var end = moment(new Date());
        var update = moment(end.diff(result.data[key].

        this._service._tl_min = update;

        //POSICIÓN DEFAULT
        this._service._lat = config.lat;
        this._service._lng = config.lng;

    });

//OBTENER INFO MAPA
getMapsEstados() {

    this._service.getMetodo(config.provinces).subscri

    this._service._markers = result.data;
    this._service._options_location = result.data;

    this._spinnerService.hide();

    }, (error) => {

        this._spinnerService.hide();

    });

    //POLYGON GEOJSON
    geoJson(location : any){

        var abreviatura = this._service.getAbreviaturaEs

        if (abreviatura != '') {

            this._service.getGeoJson(abreviatura).subscrib

            if (result != '' && result != null) {

                this._service.polygonJson(result.type, res
```


Service : Métodos estándares y variables estándares. **Services/default**

```
})
export class DefaultService {

  //SIDEBAR
  _total : any;
  _tl_activos : any;
  _tl_recuperados : any;
  _tl_mortales : any;
  _tl_min : any;
  _options_location : any;

  //MAPS
  _markers: any;
  _lat: number;
  _lng: number;
  _polygon : object = {};
  _zoom : number = 4;

  constructor(
    public _httpClient: HttpClient,
    public _http: Http
  ) {

    this.polygonJson("Polygon", []);

  }

  //GET
  getMetodo(params: string) {

    let url = `${environment.api}` + params;

    return this._httpClient.get(url);

  }

  //GET GEOJSON
  getGeoJson(params: string) {

    let url = `${environment.apiGeoJson}` + params + '/';

    return this._httpClient.get(url);

  }

  //POLYGON
  polygonJson(type : any, coordinates : any){

  }
}
```

Sidebar : Activación de eventos. **Shared/components/sidebar**

```
*DEFAULTS
*****

defaultsEvents() {

  //SELECT 2
  var $select = $('select').select2();
  $select.select2({
    placeholder: "Buscar y Seleccionar"
  });

  let _this = this;

  $('select').on('select2:select', function(e) {

    var element = e.params.data.element;
    var $element = $(element);

    $element.detach();
    $(this).append($element);
    $(this).trigger("change");

    _this.changeField();

  });

  //REMOVE ITEM
  $('select').on('select2:unselect', function(e) {

    this.changeField();

  });

}

*EVENTOS
*****

//CHANGE
changeField(){

  this._spinnerService.show();

  var locations = $('#search_locations').val();
  var slash = "/";

  if(!locations){

    locations = [''];
    slash = "";

  }

  this._service._markers = [];

  locations.forEach((val, key) => {

    //SERIVE ESTADOS
    this._service.getMetodo(config.provinces + slash + val);

    result.data.forEach((val_1, key_1) => {

      this._service._markers.push(result.data[key_1]);

      if(slash == ''){

      }

    });

  });

}
```

