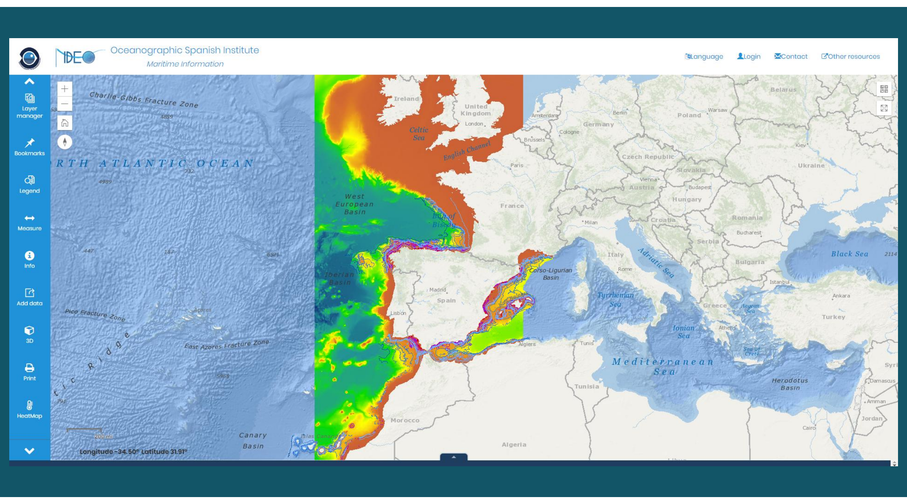
REACT-ESRI
Customizable Application Template

Luis Miguel Agudo Bravo[1], Jon Garrido Martín[2], Olvido Tello Antón[1]

(1) Spanish Institute of Oceanography (www.ieo.es)
(2) Bilbomatica (www.bilbomatica.es)



INTRODUCTION

The **Spanish Institute of Oceanography (IEO)** has developed a new customizable **web application template**, using REACT.JS library in the client side, the API for Javascript 4.6 from ESRI as map library, and NPM/Node.JS as dependency manager. This application template is open source. It is available in a **GITHub repository**, from the QR code posted at the end. Examples of the applications developed with this template are hosted in the IEO Geoportal (**www.geo-ideo.ieo.es/geoportalideo/**)

Once downloaded the user has a large number of configurable widgets available (Measure, Info, Graphs, Data Tables,Legend, 3D, Symbology, Print, Add Data, Heat Maps, etc.). Also is possible develop new components/widgets.

WHY REACT?

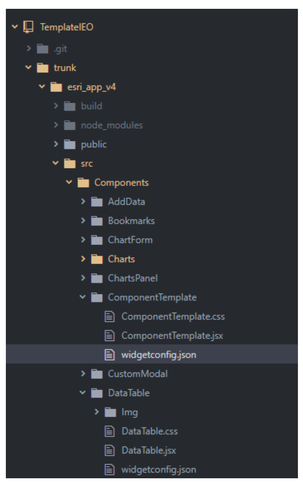
React.js is a JavaScript library for building user interfaces. It is the view layer for web applications.

At the heart of all React applications are components. A component is a self-contained module that renders some output. The user can write interface elements like a button, input field, map or widgets/tools as a React component.

React operates not directly on the browser's Document Object Model (DOM) immediately, but on a virtual DOM. This implies:

- Fast render with Virtual DOM
- Reusable Components

CUSTOMIZING TEMPLATE



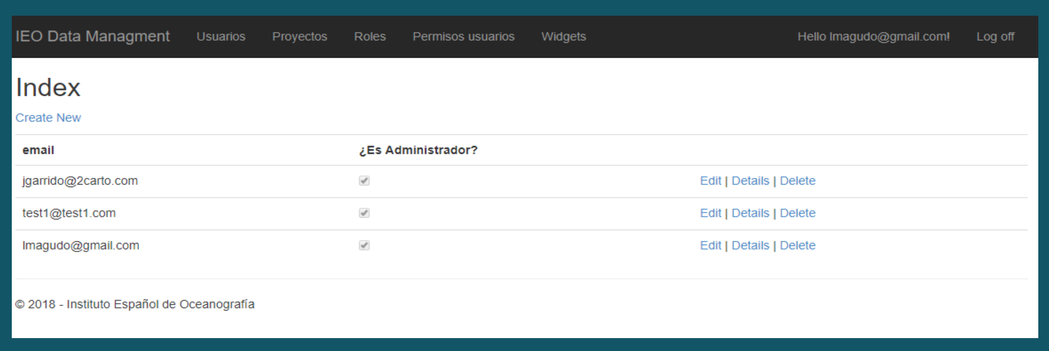
All the application template is **customizable, at the style level and at the functionality level**. In the appconf.JSON file, the user can select the widgets availables, the url of the proxy file (to handle the requests from the final users), and the languages availables (the user can define new languages in the components). Each widget/component have a folder with three files.

The css file have the info about the styles exclusives of this component. JSX file have the logic in the Javascript/REACT code language. Finally **in the JSON file the user have available all the configurable parameters** for this widget. For example, in the widgetconfig.JSON file stored in the EsriMap folder is possible configure the operational layers in the map, in addition to many other styles options map.

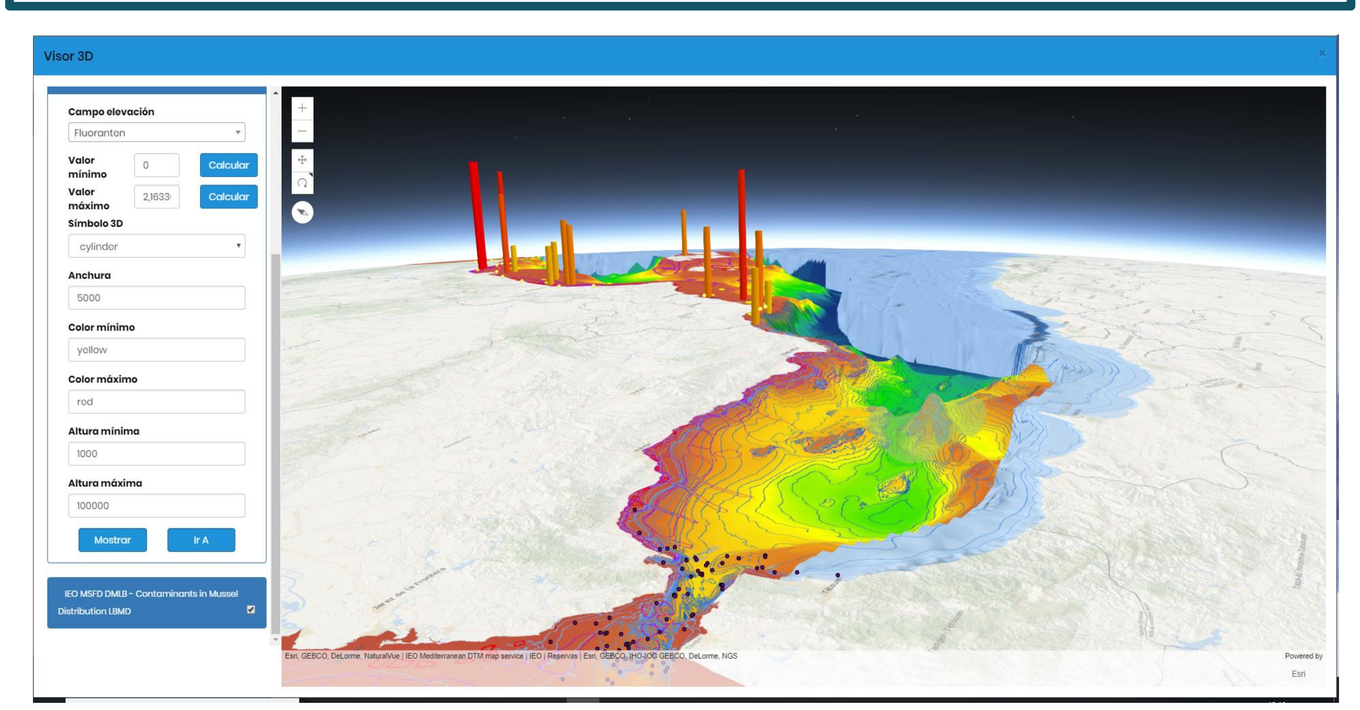
```
{
  "definition":{
    "layers": true,
    "onviewloaded": true,
    "id":"widget_compTemplate",
    "custom":{
      "inPanel": true,
      "icon": "glyphicon glyphicon-pushpin",
      "mountingnode":"",
      "title":""
    },
    "comments": ""
  },
  "labels":[{
    "id": 0,
    "es": "Etiqueta",
    "en": "Label"
  }]
}
```

MANAGER APPLICATIONS

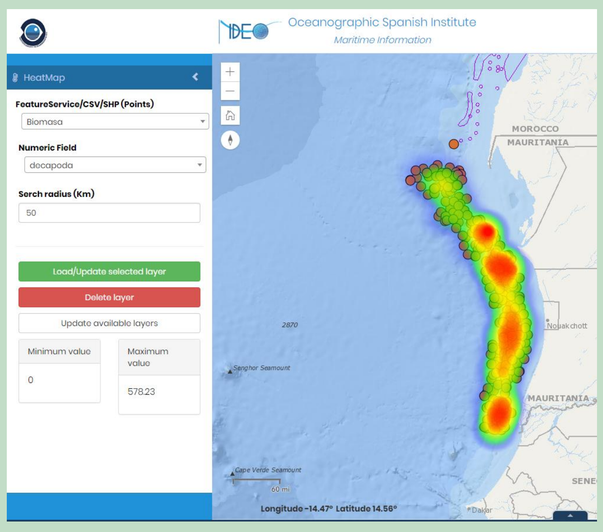
In order to manage users, roles, projects, access permits and widgets/components in the server side, another web application was developed using .**NET framework**. This application is also available in the GITHub repository downloadable from the QR code posted below.



To deploy this app only is required a server with Windows as OS and with the .NET libraries availables. Also a **SQL Server database is neccesary** in order to store the information about users, roles, projects and widgets. All the information to deploy and to configure this application is described in a document in the GITHub repository.



DEVELOPING NEW WIDGETS/COMPONENTS



The user can create new widgets/components with new funcionality and reuse these components in new applications.

The architecture of one component is a folder with three files (CSS, JSX and JSON) described in the “customizing template” section. The name of the CSS and JSX files is the same that the folder’s name. The json file is called “widgetconfig.json”.

In the JSON file the user can define if the component is loaded in view or if is loaded in the left toolbar or if is necessary that has access to the layers, and all requirements in order to deploy in the application. Also it is possible to define the configurable parameters for this new component.

All the new necessary dependencies in this new component are added in the package.json file, and used by NPM/Node.JS in order to install them. The logic of the component is developed in the JSX file (is a preprocessor step that adds XML syntax to JavaScript. JSX makes React more elegant).

A tutorial to generate new components is available in the GITHub repository.

```
import h337 from 'heatmapjs/heatmap.js'
import $ from 'jquery';
import 'select2/dist/css/select2.min.css'
import 'select2/dist/js/select2.min.js'

const appconf = require('../../appconf.json');

const conf = require('./widgetconfig.json');

export default class HeatMapLayers extends React.Component {
  constructor(props) {
    super(props);

    //let token = window.PubSub.subscribe('updatelabels', StaticDevTools._updateLabels.bind(this, conf));

    window.PubSub.subscribe('updatelabels', StaticDevTools._updateLabels.bind(this, conf));

    this.state = {
      layersOptions: null,
      pointLayers: [],
      labels: conf.labels,
      fieldsoptions: null,
      maxvalue: null,
      minvalue: null
    };
    //this.currentZoom = this.props.view.zoom;
    this._createHeatMapAndUpdateSelector();
    this.layerLoaded = false;
```

DOWNLOAD APPLICATION - GITHUB

https://github.com/IEOGit/IEOTemplate.git

IEO: http://www.ieo.es/es/
REACT: https://reactjs.org/
Api for javascirpt 4 (ESRI): https://developers.arcgis.com/javascript/