# Structure:

## HTML

### Index.html

the main dashboard with all the JavaScript includes.

Uses **Materialize CSS** (library like bootstrap).

### Modal.html

Included in the index.html, it contains the hidden popup messages (“modals”).

## Javascript

Most classes build upon the **Three.js** library for 3D rendering.  
The project uses an object oriented style (ES6) with different classes in their separate “.JS” file.

Gauges depend on the **Highcharts** library (mainly the “Polar chart”).

## Javascript classes:

### ConvenienceMethods.js

Simple methods, for example extension methods on the Javascript Math object for degree to radian conversion and back

### Cube.js

A test object to create 3D cubes in the scene (Handy to understand the more complex Turbine.js object).

### demo.js

The main 3D class. It contains the scene (= 3D world), the lights, the camera and the render loop to render all objects in it. Other objects (like for example the Turbine.js) are created and imported in this class to be included in the world.

### jquery.fullscreen.js

jquey class to handle some window properties (no need to touch this class)

### JSONModel.js

This class can import a 3D model from a JSON file. It is used to build other models. (Read the 3D coordinates of all the vertices, import the textures and materials etc.)  
The Turbine.js for example is made from 2 JSON models (the wicks at the top and the base).

### main.js

Entry point, it creates the window class for window events and the demo class for all the 3D stuff.

### ModalManager.js

Creates Popup messages and their content based on user interaction. Also creates the highchart gauges at runtime. (when the modal is opened)

### RayCastManager.js

Handles the Raycaster functions. The raycaster converts the mouse coordinates in 3D coordinates so users can interact (click, hover etc) on objects in the 3D world.

### Skybox.js

Creates the sky.

### Turbine.js

A 3D turbine.

It contains the following static variables to keep track of all the turbines that where created.

Turbine.**count**   
Turbine.TurbineList

### Water.js

Creates the water.

### window.js

Handles what happens with the 3D world on window resize. (update camera field of view etc.)

### Chart.js

Contains the different chart classes to be created with highcharts.

# Data flow:

(work in process)

JSON data from server 🡪 parser 🡪 place data in the corresponding 3D model (three.js objects have a parameter “UserData” inside), the JSON data will be stored inside the userData of each model.