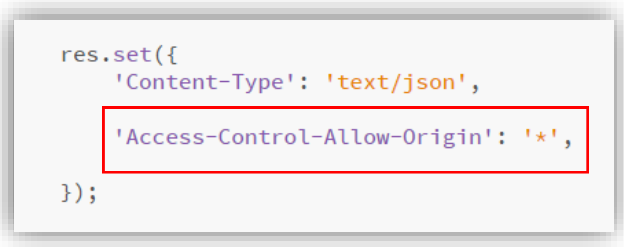
Node and Angular Task

- Create **Online Purchase Order Single Page Application** that meets the following
  - Application should be responsive using **BootStrap 4**
  - You must have a navigation bar as your application header.
  - For any open tab/pill of the navigation links, use breadcrumb or pagination if needed.
  - Application should make all its requests (e.g. getting, updating, sending and deleting data) from node server.
  - There must be a page to displays all of the products where client can select what to purchase.
  - The client can purchase more than item from each product and cannot exceed the available quantity, e.g. if a product has 3 items, the client can purchase up to 3 items only.
  - When client proceeds to checkout there must be a way to get client's info e.g. name, mobile, email and address. You can use **modal** for getting these data.
  - **Validate** entered info before sending back to the server with all selected products to be purchased.
  - Data sent to server must be stored in .json file(s).
  - Allow the client to find out his orders history in a separate page.
  - The client can reorder any of his old orders.
  - In reordering, the client may delete any of the listed items and/or update quantity of any of them, then place his new order.
  - Create a page for the website owner where he can add new products.
  - The product must have name, id, description, quantity, price
  - You can add any other page needed from your point of view to serve your application.

- Set your **node server** using **express** to handle all required **GET** and **POST** requests from client and any other requests if needed to meet the clients different scenarios
  - Let your server listen on **port 7000**
  - Implement all required **middlewares**
  - Use **body-parser** to parse incoming request

- Hints:
  - Create your own classes or interfaces needed.
  - Import and implement any module that serves your application.
  - Reordering is considered as new order.
  - Your data is read, stored, updated in .json file(s).
  - All client pages navigation and routing are implemented in Angular.

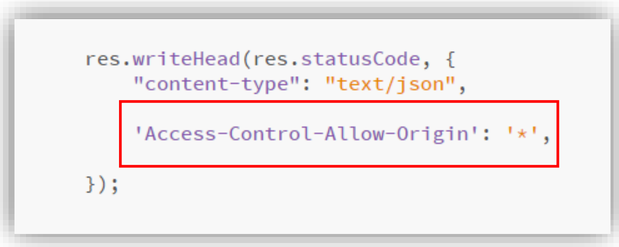
- Node is only for handling any ajax request to read, store, update, delete in the proper .json file(s).
- No authentication or authorization is required.
- If you get a cross origin policy error ensure that **‘Access-Control-Allow-Origin’** property value is set to ‘*****’ so that it can handle any request coming from any other server.
    - when using **express**, this property is added to **set()** passed object as shown in figure below



```
res.set({
     'Content-Type': 'text/json',

     'Access-Control-Allow-Origin': '*',
});
```

    - when using **http module**, this property is added to **writeHead()** passed object as shown in figure below



```
res.writeHead(res.statusCode, {
    "content-type": "text/json",

    'Access-Control-Allow-Origin': '*',
});
```

- What should be delivered:
    - Complete node **application folder excluding node_module folder**.
    - Only angular application **src folder** and its **package.json file**.
    - Angular **dist folder**
    - Read me file that describe the following:
        - Your name and track.
        - Brief description about your idea.
        - In case of using any 3rd party module other than those mentioned above, you have to state your reasons and module’s reference.

- Note:
    - This is an individual task.
    - You can implement the same idea of your MVC project Only with the previous mentioned requirements.
        - Deadline is 15/4/2018.