Resulta challenge

Coding challenge for Resulta Web Developer position

Assignment details

Deployed application link - https://acmesport.herokuapp.com/

GitHub Repo links - https://github.com/shahaadesh5/resulta-challenge

https://github.com/shahaadesh5/ressulta-challenge-php

Re-routed API link - https://acme-sport-php.herokuapp.com/

- The application is developed in Angular 7 as the frontend framework, PHP to reroute the API (due to CORS issue) and Express and Node.js to deploy the application build on Heroku servers.
- The project code is put on separate GitHub repositories because the GitHub master branch for both projects is serving as a pipeline to deploy application on Heroku servers.
- The deployed application can be accessed at - https://acmesport.herokuapp.com/
- Angular is used as it provides 2-way binding (which has been implemented for sorting data), allows to access API services easily which can be used on multiple components (if required) and also for code maintainability.

Angular Folder structure:

The folder structure for the app on Angular side is pretty simple. The main core files and components for the application is as shown in figure 1 below:

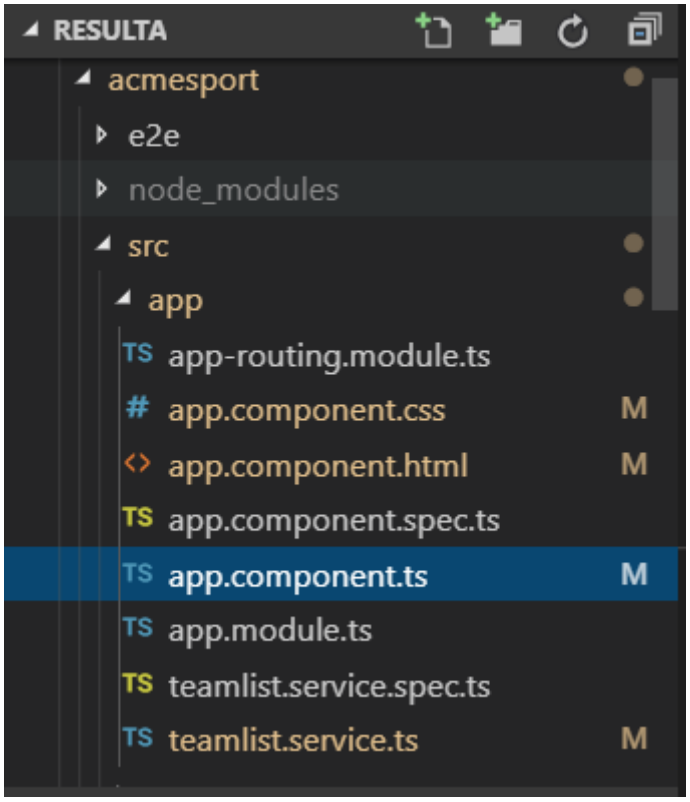


Figure 1: Angular app folder structure.

The teamlist.service.ts is a Angular service which sends an HTTP GET request to the PHP API and returns the response (JSON data).

App.component.ts subscribes to the service and gets data to be binded onto view, app.component.html

The files outside of acmesport folder are node.js and express.js files to create a server and run the angular build on Heroku servers.

Note: If you want to run the app locally, download the zip for the project and make sure that node.js is installed on the system. Traverse inside the resulta-challenge folder and then run the command 'npm install' to install the required dependencies. Then run 'node server.js' in the terminal and it should launch the app on port 3000. On the browser open http://localhost:3000/ to view the application same as hosted on Heroku.

Application details:

The UI of the application is as shown in the figure below:

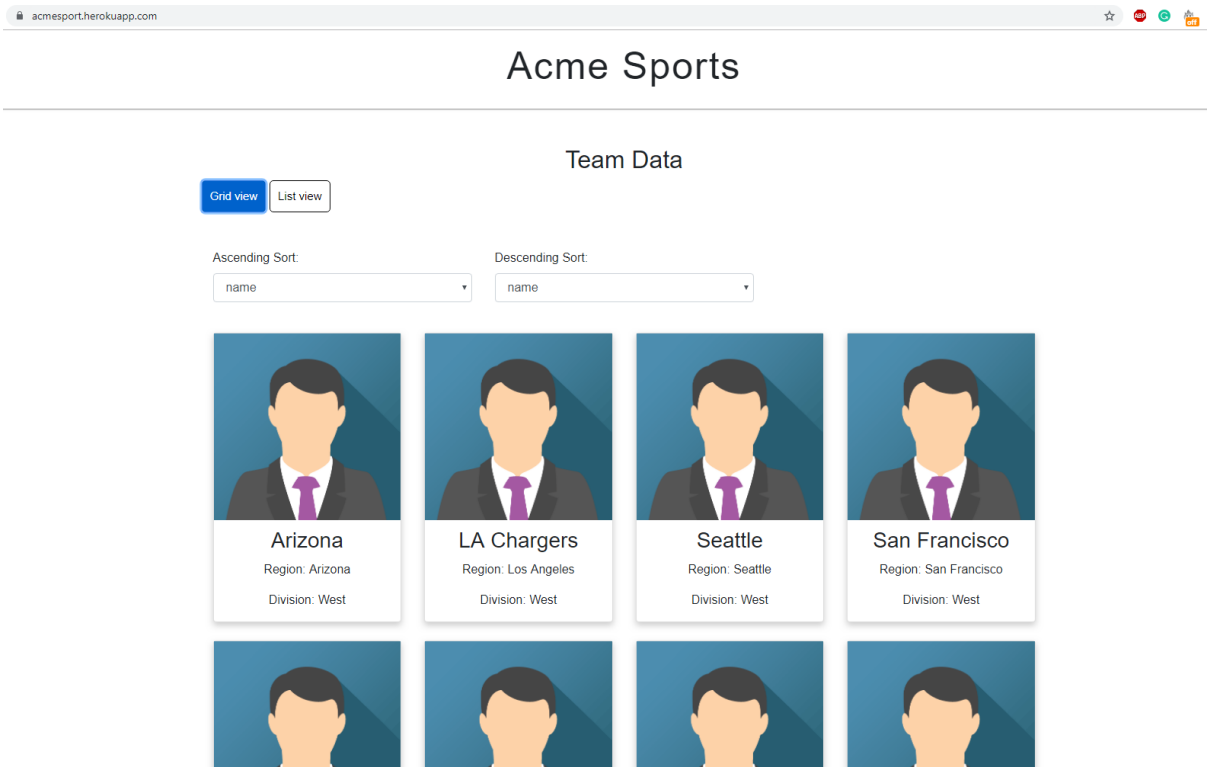
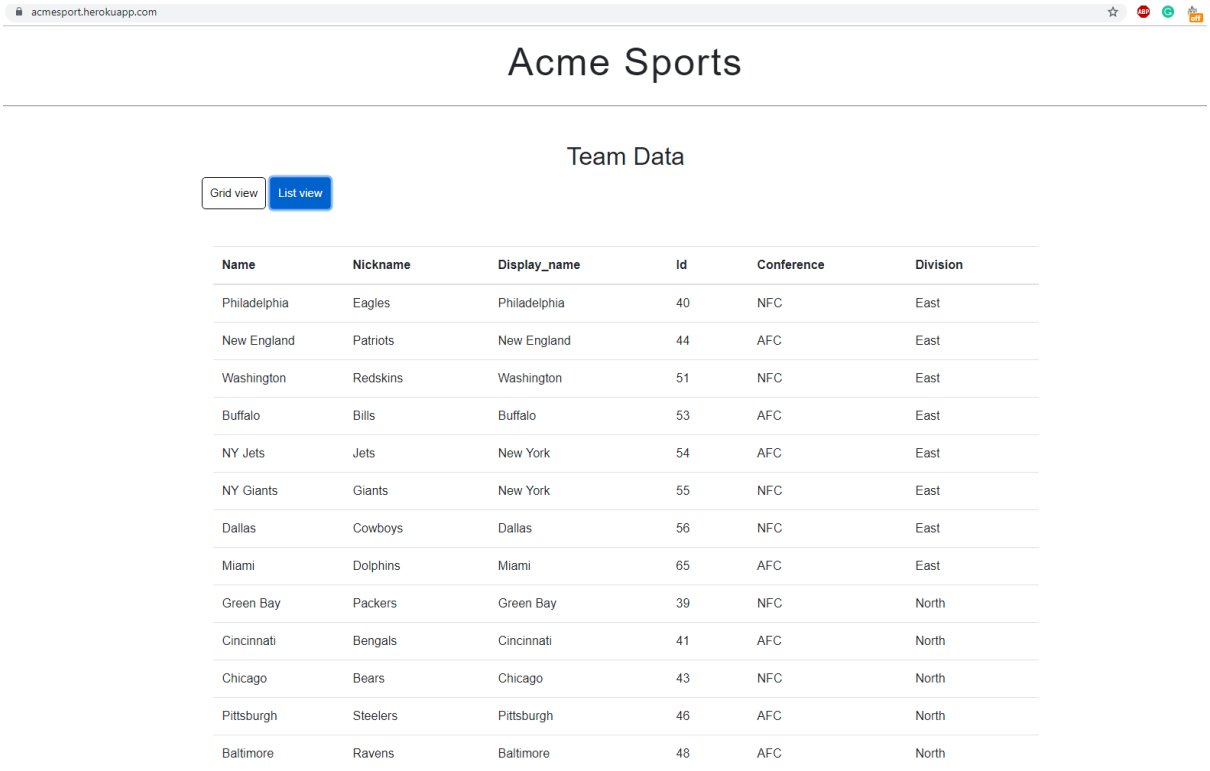


Figure 2: UI for the application.

The UI has two options to view the data, one is grid view and the other is list view. Grid view provides an overview of the team names and region and on hover, it provides some more details. The grid view also has a pagination system where on each page a maximum of 8 data cards are visible. Whereas the list view provides all the data in tabular fashion, everything at one place as shown in figure 3:



Name	Nickname	Display_name	Id	Conference	Division
Philadelphia	Eagles	Philadelphia	40	NFC	East
New England	Patriots	New England	44	AFC	East
Washington	Redskins	Washington	51	NFC	East
Buffalo	Bills	Buffalo	53	AFC	East
NY Jets	Jets	New York	54	AFC	East
NY Giants	Giants	New York	55	NFC	East
Dallas	Cowboys	Dallas	56	NFC	East
Miami	Dolphins	Miami	65	AFC	East
Green Bay	Packers	Green Bay	39	NFC	North
Cincinnati	Bengals	Cincinnati	41	AFC	North
Chicago	Bears	Chicago	43	NFC	North
Pittsburgh	Steelers	Pittsburgh	46	AFC	North
Baltimore	Ravens	Baltimore	48	AFC	North

Figure 3: List view for the data.

Both the views provide the user to sort the data ascending and descending. In grid view, there are dropdowns for it, whereas in list view there are up and down icons (hover on side of column name) for sorting the data column wise.

Issue faced:

The API (http://delivery.chalk247.com/team_list/NFL.JSON?api_key=74db8efa2a6db279393b433d97c2bc843f8e32b0) provided for the problem statement had CORS issues to access it from within the Angular app. I also tried sending an AJAX request to the API but faced the same issue. Please check the Screenshots for it.

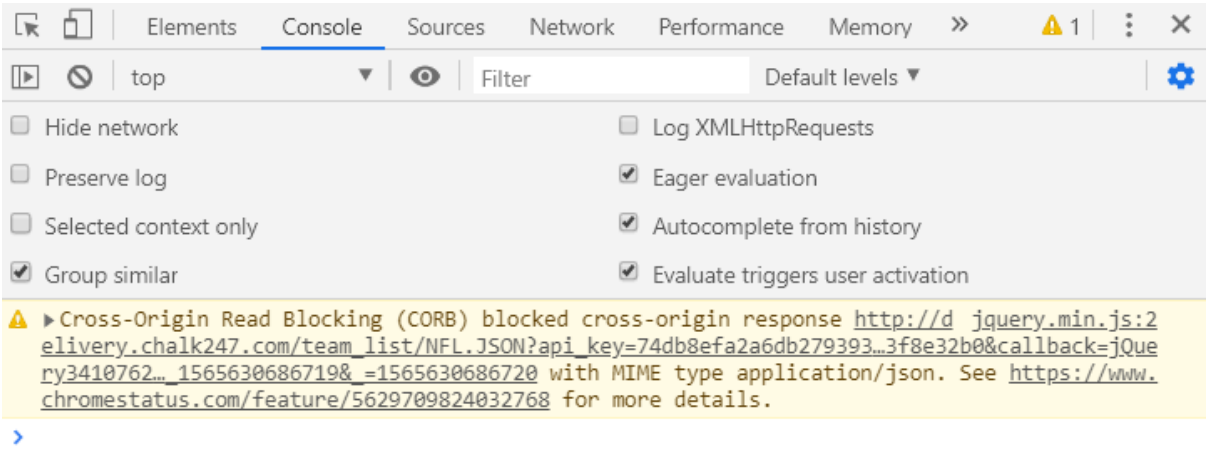


Figure 4: CORS issue using AJAX request.

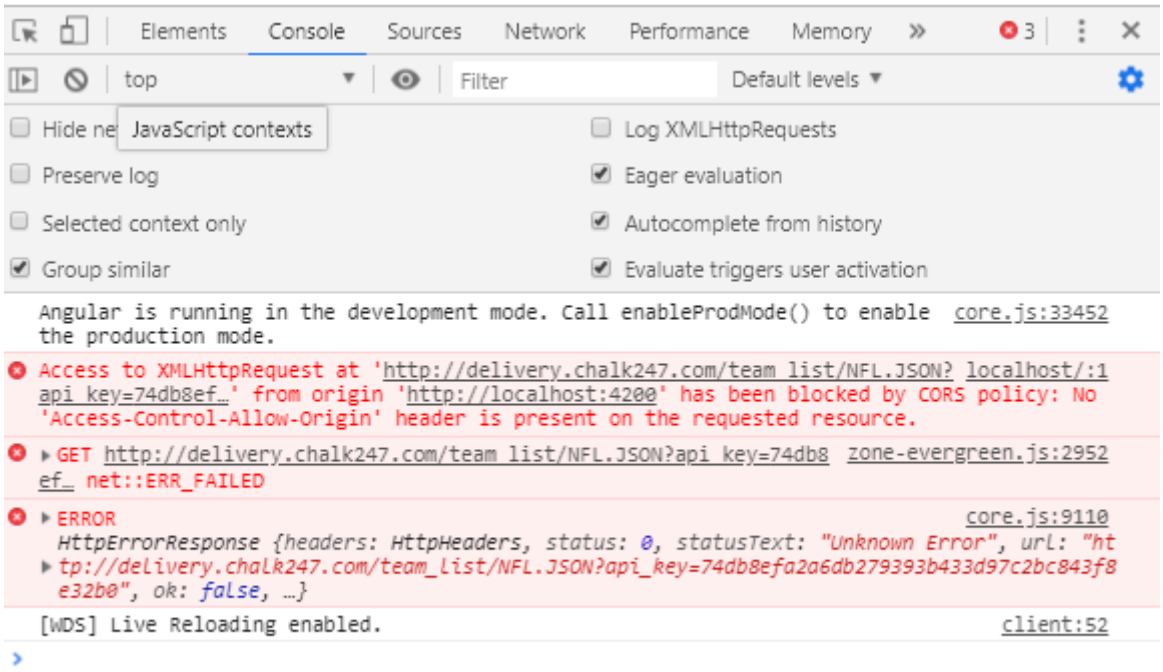


Figure 5: CORS issue using Angular service.

The Solution:

Since the CORS headers have issue with client application requests, I thought of sending request from a server-side scripting language PHP. PHP was able to send the request and retrieve data from it and forward it as another API to the client-side application with headers allowed for all. So basically, PHP reroutes the provided API to the client side (Angular) application. This is done by another application hosted on Heroku and the API for it is - https://acme-sport-php.herokuapp.com/ and the repository link for the code is available at - https://github.com/shahaadesh5/ressulta-challenge-php

There is just one php file on the server side to reroute the API called index.php and has the following code:

Figure 6: php file code