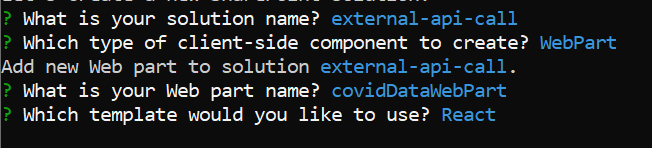
## **Authorized REST API Call from SPFx WebPart**

## Solution: ExternalAPICall

In this article, we will learn how to call an External API and get non-SharePoint based data to display in SPFx web part. We have to understand that SPFx provides us with some **inbuilt library** to access any web service, so the possibilities are endless, as we can **interact/integrate SharePoint with any product/technologies** which expose data via a web service or API.

We are going to call one simple API web service to get COVID 19 stats data, which is publicly available via the RAPID API platform.

**Step 1:** Open Node js command prompt. Go to directory where you want to create web part using cd ‘location’ command. Create web part using yo @microsoft/sharepoint command. Give the details like solution name, web part name etc. Select React framework.



**Step 2:** Use **code . command** to open solution in VS Code. Pass WebPart context to React component. Open **interface** src\webparts\controls\**components\ICovidDataWebPartProps.ts.**

import { WebPartContext } from "@microsoft/sp-webpart-base";

export interface ICovidDataWebPartProps {

  context: WebPartContext;

}

**Step 3: Context** property of react component is set from web part context. Open **CovidDataWebPartWebPart.ts**. Modify **render**() method to pass the context.

 public render(): void {

    const element: React.ReactElement<ICovidDataWebPartProps> = React.createElement(

      CovidDataWebPart,

      {

        context: this.context

      }

    );

**Step 4**: Create a Service Provider class where we will write a method to get data from External API and that will be called inside our component.

End Point: <https://covid-193.p.rapidapi.com/statistics>

API Key - We will get an API key after subscribing to this API.

import { HttpClient, IHttpClientOptions, HttpClientResponse } from '@microsoft/sp-http';

import { WebPartContext } from '@microsoft/sp-webpart-base';

export class ServiceProvider {

    private wpcontext:WebPartContext;

    public constructor(context: WebPartContext) {

       this.wpcontext= context;

      }

      private httpClientOptionsForGlobal: IHttpClientOptions = {

        headers: new Headers({

            "x-rapidapi-host": "covid-193.p.rapidapi.com",

            "x-rapidapi-key": "96a051ffaemshc889cdc82489aadp105867jsn597cfdd0d94a"

        }),

        method: "GET",

        mode: "cors"

  };

  public async getTotals() {

   var response = await this.wpcontext.httpClient

  .get("https://covid-193.p.rapidapi.com/statistics", HttpClient.configurations.v1,this.httpClientOptionsForGlobal);

  console.log(response);

  var responeJson : any = await response.json();

  return responeJson.response[0];

  }

}

**Step 5**: Create a custom react component and call it from our yeoman generated component (entry point to the webpart). Create a new file at src\webparts\covidDataWebPart\**components\Overview.tsx**

import \* as React from 'react';

import styles from './CovidDataWebPart.module.scss';

import { ICovidDataWebPartProps } from './ICovidDataWebPartProps';

import { escape } from '@microsoft/sp-lodash-subset';

import { WebPartContext } from '@microsoft/sp-webpart-base';

import { ServiceProvider } from '../ServiceProvider';

export interface IOverViewProps {

    context: WebPartContext;

}

export interface IOverViewState {

    data: any;

}

export default class OverViewStats extends React.Component<IOverViewProps, IOverViewState> {

    private serviceProvider;

    public constructor(props: IOverViewProps, state: IOverViewState) {

        super(props);

        this.serviceProvider = new ServiceProvider(this.props.context);

        this.state = {

            data: {}

        };

    }

    public render(): React.ReactElement<IOverViewProps> {

        return (

            <React.Fragment>

                <h1>Country Population Overview:</h1>

                <h2>Continent : {this.state.data.continent}</h2>

                <h2>Country: {this.state.data.country}</h2>

                <h2>Population: {this.state.data.population}</h2>

            </React.Fragment>

        );

    }

    public async componentDidMount() {

        this.getData();

    }

    private getData() {

        this.serviceProvider.

            getTotals()

            .then(

                (result: any): void => {

                    console.log('Result: ' + result);

                    this.setState({ data: result });

                }

            )

            .catch(error => {

                console.log(error);

            });

    }

}

**Step 6:** Call custom component. Open **CovidDataWebPart.tsx** component file. Import custom component and update render method.

import Overview from '../components/Overview';

export default class CovidDataWebPart extends React.Component<ICovidDataWebPartProps, {}> {

  public render(): React.ReactElement<ICovidDataWebPartProps> {

    return (

      <React.Fragment>

        <Overview context={this.props.context}>

        </Overview>

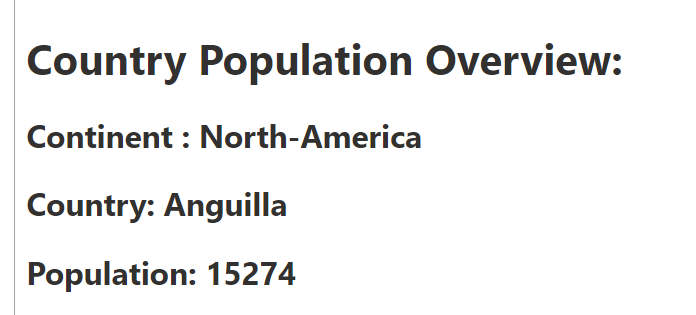
      </React.Fragment>

    );

  }

}

You can see data in web part coming from Rest API.



**Step 6: Debugging**. You may face issue is fetching data in correct format. You can debug your web part in SharePoint Workbench by using chrome debugger using **F12**. Place debug points in the methods which are calling REST API and returning the **response** in JSON format. Data in your components state should be called correctly otherwise you can’t see data in UI.

Functions to check while debugging:

**Overview.getData()** – Check what you are getting in **result**. This **result** is set as state data which you are showing in UI. All proper logs to check the data in console. this.setState({ data: result });

**ServiceProvider.getTotals()** – Check what you are returning. Here we are returning only first response from array. **responeJson.response[0]**